Inv-Entropy: A Fully Probabilistic Framework for Uncertainty Quantification in Language Models

Haoyi Song
University of Michigan
haoyiso@umich.edu

Ruihan Ji
University of Minnesota
ji000234@umn.edu

Naichen Shi
Northwestern University
naichen.shi@northwestern.edu

Fan Lai
University of Illinois Urbana-Champaign
fanlai@illinois.edu

Raed Al Kontar*
University of Michigan
alkontar@umich.edu

Abstract

Large language models (LLMs) have transformed natural language processing, but their reliable deployment requires effective uncertainty quantification (UQ). Existing UQ methods are often heuristic and lack a probabilistic interpretation. This paper begins by providing a theoretical justification for the role of perturbations in UQ for LLMs. We then introduce a dual random walk perspective, modeling input–output pairs as two Markov chains with transition probabilities defined by semantic similarity. Building on this, we propose a fully probabilistic framework based on an inverse model, which quantifies uncertainty by evaluating the diversity of the input space conditioned on a given output through systematic perturbations. Within this framework, we define a new uncertainty measure, Inv-Entropy. A key strength of our framework is its flexibility: it supports various definitions of uncertainty measures, embeddings, perturbation strategies, and similarity metrics. We also propose GAAP, a perturbation algorithm based on genetic algorithms, which enhances the diversity of sampled inputs. In addition, we introduce a new evaluation metric, Temperature Sensitivity of Uncertainty (TSU), which directly assesses uncertainty without relying on correctness as a proxy. Extensive experiments demonstrate that Inv-Entropy outperforms existing semantic UQ methods. The code to reproduce the results can be found at `https://github.com/UMDataScienceLab/Uncertainty-Quantification-for-LLMs`.

1 Introduction

Large language models (LLMs) have demonstrated remarkable success in various natural language processing tasks, such as text generation, question answering, and summarization [Brown et al., 2020, Chowdhery et al., 2023, Touvron et al., 2023]. These models have pushed the boundaries of what is achievable in language understanding and generation [Chung et al., 2024, OpenAI, 2023]. However, despite their impressive capabilities, a significant challenge remains: LLMs tend to hallucinate, or generate confidently wrong predictions [Maynez et al., 2020, Zhang et al., 2024b]. This is a serious concern in applications where reliability is paramount, such as healthcare, autonomous systems, and legal domains, where incorrect outputs can have dire consequences [Ji et al., 2023]. Addressing these challenges is essential for ensuring the reliable and responsible deployment of LLMs at scale, ultimately unlocking their full potential in real-world applications.

*Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

 https://doi.org/10.52202/085713-3598

A central step in addressing these limitations is developing effective measures for UQ, enabling LLMs to acknowledge their confidence in a generated output. Existing UQ approaches rely predominantly on heuristic consistency checks. Most commonly, they use the model's own generation likelihood or perplexity as a proxy for confidence, or they measure dispersion across multiple sampled continuations (e.g. via n-gram overlap or embedding-space variance) [Mudumbai and Bell, 2024]. Such likelihood-based and sampling-based measures, however, lack a grounded probabilistic justification. Also, token-level probabilities are known to dramatically under-estimate uncertainty as models can be "confidently wrong" [Jiang et al., 2021], and they are often inaccessible in black-box LLMs.

To probe deeper LLM brittleness, recent work has turned to input-perturbation UQ methods. Variant prompts are created through paraphrasing, adversarial token insertion, or temperature shifts. Output sensitivity is then quantified as an uncertainty signal [Gao et al., 2024, Tuna et al., 2022, Seeböck et al., 2019]. For instance, Gao et al. [2024] randomly perturb both prompt wording and sampling temperature, then aggregate variation across outputs to flag unstable predictions. Tuna et al. [2022] applies adversarial paraphrases to uncover "blind spots" where small semantic-preserving edits trigger large output changes, while Seeböck et al. [2019] uses systematic character-level and word-level corruptions to map regions of high model vulnerability.

Despite these advances, an *ab initio* probabilistic framework has not been established, as existing methods mainly rely on heuristic score functions for UQ. To address this, we introduce Inv-Entropy, a fully probabilistic framework built on random-walk theory that offers a new perspective: it learns the statistical connections between LLM inputs and outputs. This is accomplished through structured perturbations, which simultaneously capture input variability and the influence of different inputs on model predictions. Our contributions are summarized as follows:

1. We present the first work in UQ for LLMs that adopts a **fully probabilistic framework**, grounded in random walk theory. This framework is highly flexible, and its probabilistic nature allows for the use of various UQ measures. It is also intrinsically capable of handling black-box models, as it does not rely on token probabilities.
2. We introduce an **inverse perspective** that quantifies input diversity given an output, inspired by "Asymmetry in Semantic Variability" (defined later). Extensive simulations highlight its advantages, especially for short-form answers where traditional methods often struggle.
3. Theoretically, our work provides a **principled foundation** for using perturbation-based methods for UQ, justifying the recent momentum behind such approaches.
4. We propose GAAP, a novel perturbation algorithm that enhances **input sampling diversity**. Empirical results show that GAAP significantly improves perturbation-based UQ.
5. Finally, we introduce a new evaluation metric, **Temperature Sensitivity of Uncertainty (TSU)** capable of evaluating uncertainty *without relying on correctness as a proxy*. This enables evaluation of UQ on any dataset, even when labels are unavailable.

For the remainder of the paper, we use $\mathbb{E}[\cdot]$ to denote expectation and $\mathbb{V}[\cdot]$ to denote variance. We note that a more detailed related work section can be found in Appendix A.

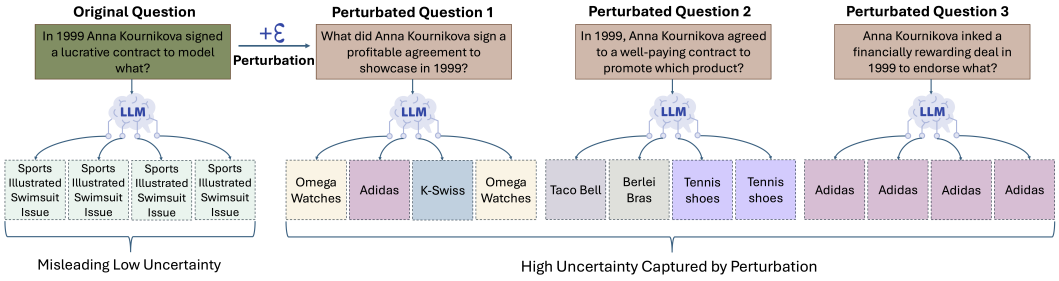

Figure 1: Toy example highlighting the importance of perturbations. The original question is from TriviaQA [Joshi et al., 2017], and the correct answer is "bras." The responses are generated by ChatGPT-3.5-Turbo. Input perturbations reveal hidden variability that multiple sampling (i.e., replications) alone fails to capture, as replication alone can be confidently wrong.

2 Perturb-then-Quantify

2.1 Why perturb the input?

We begin with a simple illustrative example in Fig. 1 to highlight the importance of input perturbations. In this example, when the input question is simply replicated (as is common in much of the existing literature [Lin et al., 2024]) all generated responses are identical. This misleadingly suggests low uncertainty even though all answers are incorrect, thus providing false confidence. Replication alone is therefore insufficient to capture the underlying variability. In contrast, applying input perturbations yields a diverse set of responses across semantically equivalent prompts, enabling a more faithful characterization of uncertainty. The importance of such perturbations is not only evident from this example but also supported by simplified theoretical argument in what follows.

To build theoretical insight into the importance of perturbations, we leverage a key property specific to language: semantic equivalence. An *ideal* LLM should respond consistently to semantically equivalent inputs. We use this property, in a simplified setting, to construct input perturbations within equivalence classes, which allows us to expose inconsistencies in model behavior and formally reason about uncertainty. We start from a proof-of-concept example where the target function $f^\star$ is a single output function $f^\star : \mathbb{R}^d \to \mathbb{R}$. We lack access to $f^\star$, but have access to an approximation $\hat{f} : \mathbb{R}^d \to \mathbb{R}$, such as a pre-trained model. At first sight, it seems hopeless to quantify the alignment between $\hat{f}$ and $f^\star$ without knowing $f^\star$. However, the semantic equivalence class relative to an input x_0 provides a set $\mathcal{I}(x_0) \subseteq \mathbb{R}^d$, such that $f^\star(x') = f^\star(x_0), \forall x' \in \mathcal{I}(x_0)$. The equivalence class provides valuable information for UQ as shown in the subsequent argument and Figure 2.

The shape of $\mathcal{I}(x_0)$ could be complex for general $f^\star$. However, in a small neighborhood of x_0 where $\nabla f^\star(x_0) \neq 0$, $\mathcal{I}(x_0)$ should be locally close to the tangent space of $f^\star$. More specifically, we define a tangent invariance set as $\mathcal{I}_{\text{tangent}}(x_0) = \{x_0 + (I - \eta_{\nabla f^\star})\, z; z \in \mathbb{R}^d\}$, where $\eta_{\nabla f^\star} = \frac{\nabla f^\star(x_0) \nabla f^\star(x_0)^\top}{\|\nabla f^\star(x_0)\|^2}$. Intuitively, $I - \eta_{\nabla f^\star}$ acts as the orthogonal projector onto the subspace orthogonal to $\nabla f^\star(x_0)$. Taylor expansion shows $\mathcal{I}_{\text{tangent}}(x_0) \approx \mathcal{I}(x_0)$ when restricted to the small neighborhood of x_0. To generate algorithmic insight, we further define a probability density on $\mathcal{I}_{\text{tangent}}(x_0)$: we use $P(x'; x_0, \sigma)$ to denote the probability density function of $x' = x_0 + (I - \eta_{\nabla f^\star})\, z$ where z is sampled from a d-dimensional isotropic Gaussian distribution $\mathcal{N}(0, \sigma^2)$. It is easy to verify that $P(\cdot; x_0, \sigma)$ is a Gaussian distribution supported on $\mathcal{I}_{\text{tangent}}(x_0)$ whose mean is x_0.

The following Lemma shows that we can estimate the angle between $\nabla \hat{f}(x)$ and $\nabla f^\star(x)$ by examining the variance of $\hat{f}(x')$, where x' is sampled from $P(x'; x, \sigma)$.

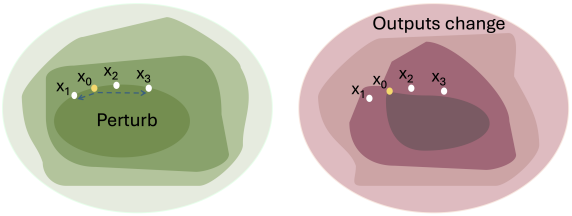

Figure 2: **Left**: Conceptual illustration of level sets of the ground truth $f^\star$. We perturb the input x_0 to $x_1, x_2, \ldots$ along a schematic isocontour such that $f^\star(x_0) = f^\star(x_1) = \cdots$. **Right**: Conceptual illustration of level sets of the model $\hat{f}$. The deviations of $\hat{f}(x_i)$ for $i \geq 1$ from $\hat{f}(x_0)$ reflect the model's uncertainty around x_0.

Lemma 2.1 *Assume (1) $\hat{f}$ is twice differentiable, and both $\|\nabla \hat{f}(x)\|$ and $\|\nabla^2 \hat{f}(x)\|_{op}$ are bounded for all $x \in \mathbb{R}^d$, and (2) $\nabla \hat{f}(x_0) \neq 0$ and $\nabla f^\star(x_0) \neq 0$. Then, for sufficiently small σ, we have*

$$\frac{1}{\sigma^2} \mathbb{V}_{x' \sim \mathcal{P}(\cdot; x_0, \sigma^2)}[\hat{f}(x')] = \|\nabla \hat{f}(x_0)\|^2 \sin^2 \theta \left(\nabla \hat{f}(x_0), \nabla f^\star(x_0)\right) + \mathcal{O}(\sigma), \tag{1}$$

where $\theta(v_1, v_2) = \arccos\left(\frac{v_1^\top v_2}{\|v_1\| \|v_2\|}\right)$ denotes the angle between two vectors v_1 and v_2.

Equation (1) shows that the variance of function values of $\hat{f}$ on the invariance set is indicative of the alignment between $\nabla \hat{f}$ and $\nabla f^\star$. When the variance is larger, $\hat{f}$ respects the invariance of $f^\star$ less,

https://doi.org/10.52202/085713-3598

which in turn implies larger misalignment between $\nabla \hat{f}$ and $\nabla f^\star$ and larger uncertainty. The full proof of Lemma 2.1 is relegated to the Appendix B.

Despite its simplicity, Lemma 2.1 demonstrates two important elements of perturbation-based UQ: input perturbation and output variance evaluation. In what follows, we introduce concrete designs that implement these elements in the LLM context to effectively characterize uncertainty.

2.2 A probabilistic framework via dual random walks

The variance estimate in Lemma 2.1 presents useful insights yet is oversimplified to realistically model LLMs, whose outputs are token sequences rather than a single scalar. Also, it is difficult to sample exactly from a semantic equivalence class. To tackle these challenges, we introduce a probabilistic approach inspired by Markov chains.

We use S_{x_0} to denote a semantic input to a LLM f, and $S_{y_0} \leftarrow f(S_{x_0})$ to denote one of its possible corresponding outputs. Because f is typically stochastic, the output S_{y_0} is a random variable. Next, consider a semantic embedding function ψ that maps both inputs and outputs into a continuous space $\mathcal{X}$, such that $(x_0, y_0) = (\psi(S_{x_0}), \psi(S_{y_0}))$. We also assume a perturbation algorithm $\mathcal{P}\mathrm{er}(S_{x_0})$ that produces a finite set of perturbed inputs:

$$\mathcal{P}\mathrm{er}(S_{x_0}) = \{S_{x_0}, S_{x_1}, \ldots, S_{x_n}\},$$

whose detailed implementation will be described in Sec. 2.6.

Applying the embedding function $\psi(S_{x_i})$, we obtain the embeddings:

$$X_n = \{x_0, x_1, \ldots, x_n\}, \text{ where } x_i = \psi(S_{x_i}).$$

One corresponding output embedding set is hence given as

$$Y_n = \{y_0, y_1, \ldots, y_n\}, \text{ where } y_i = \psi(S_{y_i}) = \psi(f(S_{x_i})).$$

Samples in X_n and Y_n are one-to-one correspondent. We can thus estimate the structural similarity of the samples in the two sets. In the following, we propose a probabilistic approach to characterize the structural similarity through the lens of a random walk.

We define two Markov chains, $\mathcal{M}_x$ and $\mathcal{M}_y$, over the same state set $\mathcal{S} = \{0, 1, \ldots, n\}$, where each state i corresponds to a perturbed instance S_{x_i}. The chains differ in their transition dynamics: $\mathcal{M}_x$ is built from similarities among input embeddings X_n, and $\mathcal{M}_y$ from similarities among output embeddings Y_n. Let $a_{\text{Similarity}}(x, x') : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_0^+$ denote a non-negative similarity function that measures the closeness between embeddings in $\mathcal{X}$. Using this function, we define the transition matrices $\mathrm{P}_x, \mathrm{P}_y \in \mathbb{R}^{(n+1)\times(n+1)}$ for $\mathcal{M}_x$ and $\mathcal{M}_y$ elementwise as:

$$\mathrm{P}_x[i,j] \triangleq \frac{a_{\text{Similarity}}(x_i, x_j)}{\sum_{k=0}^{n} a_{\text{Similarity}}(x_i, x_k)}, \quad \mathrm{P}_y[i,j] \triangleq \frac{a_{\text{Similarity}}(y_i, y_j)}{\sum_{k=0}^{n} a_{\text{Similarity}}(y_i, y_k)}. \tag{2}$$

The two transition probability matrices characterize two random walks in the space of $n+1$ pairs $\{x_i, y_i\}_{i=0}^n$, where the transition probability from i to j is higher if their input or output semantic features are closer. Notice that a variety of well-established methods have been proposed for $a_{\text{Similarity}}$, some of which are deployed in our numerical studies (see Sec. 3).

At its core, (2) defines stochastic dynamics on the set $\mathcal{S}$, capturing the similarity structures in both the input and output spaces. These dual dynamics uncover meaningful semantic patterns that can be leveraged for UQ. There are various ways to characterize uncertainty by examining the alignment between the two induced graphs [Vishwanathan et al., 2010]. In what follows, we construct a framework tailored for discrete input and output spaces to rigorously define uncertainty.

2.3 Constructing the distributions

We use X and Y to denote discrete random variables whose supports are X_n and Y_n, respectively. There are many possible ways to define possible distributions of X and Y. In the following, we will introduce one design of $P(Y)$ and $P(X|Y)$ based on (2). For notational simplicity, we denote the uniform distribution over all states $\mathcal{S}$ by π^{Uniform}, which is given by $\pi^{\text{Uniform}} = \left[\frac{1}{n+1} \quad \frac{1}{n+1} \quad \cdots \quad \frac{1}{n+1}\right]$.

The marginal distribution The random variable Y corresponds to the LLM's response to a question that is perturbed from the original question x_0. We define the marginal distribution of Y as:

$$P(Y = y_j) \triangleq (\pi^{\text{Uniform}} \mathrm{P}_y)[j] = \frac{1}{n+1} \sum_{i=0}^{n} \frac{a_{\text{Similarity}}(y_i, y_j)}{\sum_k a_{\text{Similarity}}(y_i, y_k)}, \tag{3}$$

where notation $[j]$ denotes the j-th element of a vector.

The distribution (3) has an intuitive interpretation: we randomly sample a point uniformly from the state space $\mathcal{S}$, then randomly transit the sample with P_y for one step. After the transit step, nodes whose corresponding output samples are surrounded by many similar outputs are assigned a higher probability, while isolated nodes with fewer similar neighbors are assigned a lower probability. Therefore, the mass is concentrated on regions of high semantic density in the output space.

The conditional distribution Now, we introduce the conditional probability $P(X = x_i \mid Y = y_j)$

$$P(X = x_i | Y = y_j) = (\mathrm{P}_y \mathrm{P}_x)[j, i] = \sum_k \frac{a_{\text{Similarity}}(x_i, x_k)}{\sum_m a_{\text{Similarity}}(x_m, x_k)} \frac{a_{\text{Similarity}}(y_j, y_k)}{\sum_l a_{\text{Similarity}}(y_j, y_l)}, \tag{4}$$

where notation $[j, i]$ denotes the (j, i)-th entry of the matrix $\mathrm{P}_y \mathrm{P}_x \in \mathbb{R}^{(n+1)\times(n+1)}$.

Essentially, (4) defines the conditional probability through composite transition dynamics on the state space $\mathcal{S}$. We design a two-stage random walk: first, states transition under P_y, capturing output similarities; then, they transition under P_x, capturing input similarities. Since both P_x and P_y operate on the same state space $\mathcal{S}$ linked by the LLM, (4) establishes a probabilistic bridge connecting similarity structures in the output and input spaces (see Fig. 3).

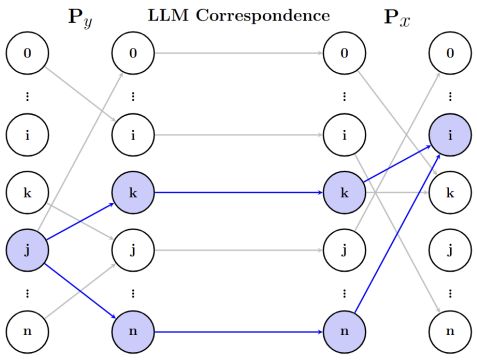

Figure 3: Random-walk transitions underlying $P(X \mid Y) = \mathrm{P}_y \mathrm{P}_x$. Highlighted blue paths show two representative transitions (one through k and one through n), each following $y_j \xrightarrow{\mathrm{P}_y} y_k \xrightarrow{\text{LLM}} x_k \xrightarrow{\mathrm{P}_x} x_i$.

We explicitly model the conditional distribution of inputs X rather than outputs Y for two main reasons. First, the perturbed samples generated by $\mathcal{P}\mathrm{er}(S_{x_0})$ may differ semantically from the original input. Reweighting these samples using both input- and output-space similarities ensures that semantically consistent perturbations are emphasized while spurious ones are down-weighted. Second, LLMs exhibit an inherent *semantic asymmetry* between inputs and outputs: many distinct prompts can lead to similar responses, whereas small input changes typically cause only small to modest output variations. Modeling $P(X \mid Y)$ therefore provides a more stable and informative view of uncertainty by capturing the diversity of possible inputs that could relate to a given output.

Our goal is to use the information encoded in Y to guide the conditional distribution of X. This coupling between input and output similarities allows the model to assign higher probability to inputs supported by multiple semantically consistent input-output pairs, yielding a more faithful representation of uncertainty. Based on the defined $P(Y)$ and $P(X \mid Y)$, we can then derive $P(X) \triangleq \pi^{\text{Uniform}} \mathrm{P}_y \mathrm{P}_x$ and $P(Y \mid X)$ via Bayes' theorem. Such formulation provides a flexible foundation for defining diverse uncertainty measures, including divergence-based metrics (e.g., KL or Wasserstein distances), entropy-based quantities, and other probabilistic constructs. In the next section, we introduce an entropy-based metric as a concrete example.

2.4 Inv-Entropy via bootstrapping and Monte Carlo

We next introduce how to leverage the probabilistic framework described above to define our UQ measure, denoted as Inverse-Entropy (Inv-Entropy).

https://doi.org/10.52202/085713-3598

Algorithm 1 Inv-Entropy overall framework

Input: (S_{x_0}, S_{y_0}), f, ψ, $\mathcal{P}er$, and $a_{\text{Similarity}}$
Perturb: Use $\mathcal{P}er(S_{x_0})$ and ψ to obtain X_n, Compute P_x using (2)
LLM Generation: Input each question in $\mathcal{P}er(S_{x_0})$, r times into f and obtain $\{\mathcal{R}_i\}_{i=0}^n$.
for $b = 1$ to B **do**
    Sample $Y_n^{(b)} = \{y_0^{(b)}, \ldots, y_n^{(b)}\}$
    Compute $\mathrm{P}_y^{(b)}$ using (2)
    Compute $P^{(b)}(x_i|y_i) = (\mathrm{P}_y^{(b)} \cdot \mathrm{P}_x)[i, i]$ (from (4)) for $\forall\, i$
    Compute $H(X_n \mid Y_n^{(b)})$ using (5)
end for
Compute $\widehat{H}(X|Y)$ using (6)

Inv-Entropy A natural measure of uncertainty is the conditional sample entropy $H(X_n \mid Y_n)$, which connects the similarity of the input set X_n to the similarity of the corresponding output set Y_n:

$$H(X_n \mid Y_n) \triangleq -\operatorname{trace}\big(\mathrm{P}_y \mathrm{P}_x \odot \log(\mathrm{P}_y \mathrm{P}_x)\big) = -\sum_{i=0}^{n} P(x_i \mid y_i) \log P(x_i \mid y_i), \tag{5}$$

where $\odot$ denotes the Hadamard product. This quantity captures the degree of alignment between P_y and P_x. High entropy indicates that semantically similar inputs yield divergent outputs, or that semantically distinct outputs correspond to similar inputs, both revealing uncertainty in the model's behavior. Note that Inv-Entropy represents an *unnormalized* entropy measure, designed to capture not only the dispersion of $P(x_i \mid y_i)$ but also its magnitude, thereby jointly modeling these two sources of uncertainty.

That said, it is important to realize that X_n does not map to a unique Y_n due to the inherent stochasticity of f as LLMs can produce different outputs for the same input. Thus, beyond perturbations that capture epistemic uncertainty, replications can help account for aleatoric uncertainty arising from sampling randomness. To this end, each perturbed input $S_{x_i} \in \mathcal{P}\mathrm{er}(S_{x_0})$ can be queried r times, yielding a replicated output set $\mathcal{R}_i = \{y_{i,1}, \ldots, y_{i,r}\}$ for each question x_i, where $y_{i,\cdot} = \psi(f(S_{x_i}))$.

Bootstrapping & Monte Carlo With this setup, we are able to generate B bootstrap samples $Y_n^{(b)} = \{y_0^{(b)}, \ldots, y_n^{(b)}\}$ for $b \in \{1, \ldots, B\}$, each corresponding to X_n, where $y_i^{(b)} \sim \mathcal{R}_i$ is drawn with replacement for $i \in \{0, \ldots, n\}$, yielding a transition matrix $\mathrm{P}_y^{(b)}$. Each bootstrapped output embedding set $Y_n^{(b)}$, together with the input embeddings X_n, defines a probabilistic instance of our uncertainty measure, Inv-Entropy. The final UQ estimate is obtained by averaging Inv-Entropy over the B bootstrap replicates:

$$\widehat{H}(X \mid Y) \triangleq \frac{1}{B} \sum_{b=1}^{B} H(X_n \mid Y_n^{(b)}). \tag{6}$$

Our overall framework in given in Algorithm 1. It is important to emphasize that a key strength of our framework lies in its flexibility: it accommodates arbitrary choices of embedding function ψ, perturbation strategy $\mathcal{P}er$, and similarity metric $a_{\text{Similarity}}$, allowing it to adapt to different tasks and model architectures. Moreover, $\widehat{H}(X \mid Y)$ represents just one possible uncertainty measure; many others can be defined within our probabilistic framework. For instance, our numerical studies evaluate alternatives such as the Wasserstein distance between the marginal distributions $P(X)$ and $P(Y)$.

2.5 Insights into Inv-Entropy

To provide some insights into our UQ measure, we introduce two parameters: $\epsilon_x \in (0, 1]$, which controls the input perturbation level, and $\epsilon_y \in [0, 1]$, which controls the output dispersion level (defined via $a_{\text{Similarity}}$ below). The corresponding transition matrices are then readily derived as:

$$a_{\text{Similarity}}(z_i, z_j) = \begin{cases} 1, & i = j, \\ 1 - \epsilon_z, & i \neq j, \end{cases} \rightarrow \mathrm{P}_z(\epsilon_z)[i, j] = \begin{cases} \dfrac{1}{(n+1) - n\epsilon_z}, & i = j, \\ \dfrac{1 - \epsilon_z}{(n+1) - n\epsilon_z}, & i \neq j, \end{cases}$$

where $z \in \{x, y\}$. With this, the joint conditional probability can be written as: $P(x_i \mid y_i; \epsilon_x, \epsilon_y) = \mathrm{P}_y(\epsilon_y)\mathrm{P}_x(\epsilon_x)[i, i] = \frac{1+n-n\epsilon_x-n\epsilon_y+n\epsilon_x\epsilon_y}{(n+1-n\epsilon_x)(n+1-n\epsilon_y)}$. We can then show that for the same input perturbation level ϵ_x, a larger output dispersion ϵ_y corresponds to higher uncertainty, consistent with the definition of Inv-Entropy. To see this, notice that the derivative of $P(x_i \mid y_i; \epsilon_x, \epsilon_y)$ with respect to ϵ_y is $\frac{\partial P(x_i|y_i;\epsilon_x,\epsilon_y)}{\partial \epsilon_y} = \frac{n\epsilon_x}{(n+1-n\epsilon_x)(n+1-n\epsilon_y)^2} > 0$. Now, if we assume all $P(x_i \mid y_i; \epsilon_x, \epsilon_y)$ are smaller than $\frac{1}{e}$ (a condition generally satisfied for large n), then the Inv-Entropy function H in (5) becomes also increasing in $P(x_i \mid y_i; \epsilon_x, \epsilon_y)$ and hence in ϵ_y.

2.6 GAAP

In this section, we present a genetic algorithm-based adversarial perturbation (GAAP) that progressively modifies the semantic input S_{x_0} to generate controlled perturbations, $\mathcal{P}er(S_{x_0})$. Below we highlight our overarching framework, while algorithmic details are relegated to Appendix C. As shown in Fig. 4, the process consists of an initialization step and multiple iterative procedures. In the initialization step, we construct a population $\mathrm{Pop}_0(S_{x_0})$ consisting of perturbed versions of S_{x_0}. More specifically, each text in $\mathrm{Pop}_0(S_{x_0})$ is derived from S_{x_0} by replacing one keyword with a synonym, hypernym, hyponym from WordNet [Miller, 1995], or a deletion.

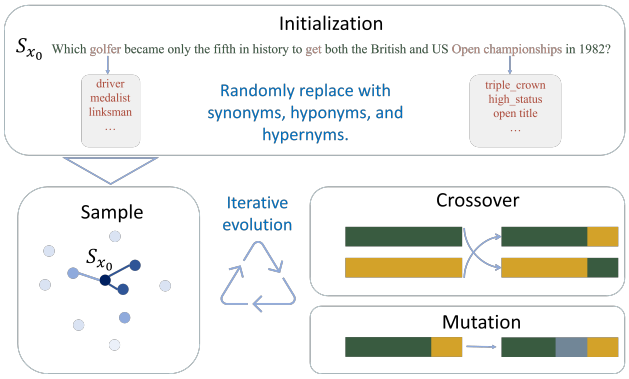

Figure 4: Illustration of GAAP on a TriviaQA [Joshi et al., 2017] question.

With an initial $\mathrm{Pop}_0(S_{x_0})$, GAAP updates the $\mathrm{Pop}_t(S_{x_0})$ through subsequent steps of crossovers and mutations. In the crossover step, we first select a random subset of $\mathrm{Pop}_t(S_{x_0})$ based on $a_{\mathrm{Similarity}(x_0,\cdot)}$, such that samples are chosen with higher probability if they are closer to x_0. Next, we randomly segment each selected sentence. These sentence segments are then randomly concatenated to generate new sentences, as illustrated in Crossover of Fig. 4.

In the mutation step, we perturb the recombined sentences with further key word substitutions and deletions, which introduce additional variations to the cross-pollinated texts. We construct the next generation population $\mathrm{Pop}_{t+1}(S_{x_0})$ as the union of the selected, crossovered, and mutated texts.

GAAP proceeds by iteratively updating $\mathrm{Pop}_t(S_{x_0})$ with crossovers and mutations for T iterations or until all texts in $\mathrm{Pop}_t(S_{x_0})$ have similarity to S_{x_0} smaller than a predefined constant δ. Finally, we construct $\mathcal{P}er(S_{x_0})$ by sampling from populations at different generations $\{\mathrm{Pop}_t(S_{x_0})\}_{t=0,\tau,2\tau,\cdots}$, where $\tau \in \mathbb{Z}$ is a fixed gap. Since texts in $\mathrm{Pop}_t(S_{x_0})$ tend to deviate from S_{x_0} further with larger t, such construction of $\mathcal{P}er(S_{x_0})$ ensures diverse representation of perturbed texts with different levels of similarity to S_{x_0}.

3 Experiments

Models and tasks We conducted experiments using two language models: GPT-3.5-Turbo, a black-box model accessed via API, and LLaMA-3.1-8B-Instruct, a grey-box model. We evaluated our framework on datasets spanning three categories: question answering (TriviaQA [Joshi et al., 2017], SciQ [Welbl et al., 2017], Natural Questions [Kwiatkowski et al., 2019] (NQ, long-answer questions with details in Appendix D.3)), multiple choice (MMLU [Hendrycks et al., 2020]), and mathematical reasoning (GSM8K [Cobbe et al., 2021]).

https://doi.org/10.52202/085713-3598

Table 1: Comparison of AUROC, PRR, and Brier scores across all the 5 datasets. We use GPT-3.5-Turbo with ChatGPT-based paraphrasing, and DeBERTa-v2-xlarge-MNLI embedding function. **Bold** and underline denote the best and second-best performers, respectively.

Metric	Method	Datasets				
		TriviaQA	SciQ	NQ	MMLU	GSM8K
AUROC (↑)	Semantic Entropy	0.579 ± 0.044	0.679 ± 0.045	0.521 ± 0.034	0.518 ± 0.048	0.589 ± 0.052
	Kernel Entropy	0.687 ± 0.062	0.685 ± 0.063	0.556 ± 0.055	0.653 ± 0.059	0.560 ± 0.060
	VU	0.695 ± 0.060	0.480 ± 0.060	0.533 ± 0.056	0.523 ± 0.054	0.557 ± 0.057
	P(True)	0.604 ± 0.050	0.522 ± 0.026	0.519 ± 0.020	0.474 ± 0.027	0.571 ± 0.056
	LexSim	0.649 ± 0.055	0.681 ± 0.046	0.518 ± 0.055	0.643 ± 0.054	0.598 ± 0.060
	DegMat	0.734 ± 0.056	0.672 ± 0.059	0.551 ± 0.052	0.608 ± 0.058	0.678 ± 0.059
	LUQ	0.637 ± 0.067	0.726 ± 0.048	0.627 ± 0.055	0.648 ± 0.057	0.662 ± 0.064
	KLE	0.333 ± 0.054	0.341 ± 0.056	0.410 ± 0.060	0.360 ± 0.064	0.338 ± 0.061
	Inv-Entropy	**0.788** ± 0.054	**0.740** ± 0.050	**0.661** ± 0.052	**0.780** ± 0.041	**0.695** ± 0.051
	NI-Entropy	0.786 ± 0.057	0.681 ± 0.056	0.637 ± 0.053	0.710 ± 0.052	0.650 ± 0.069
	NR-Inv-Entropy	0.743 ± 0.061	0.720 ± 0.049	0.627 ± 0.054	0.604 ± 0.059	0.677 ± 0.064
	WD-px-py	0.518 ± 0.060	0.303 ± 0.060	0.558 ± 0.055	0.573 ± 0.061	0.605 ± 0.069
	MAX-py-x	0.723 ± 0.054	0.674 ± 0.054	0.547 ± 0.051	0.585 ± 0.059	0.618 ± 0.059
PRR (↑)	Semantic Entropy	0.517 ± 0.060	0.763 ± 0.044	0.505 ± 0.049	0.690 ± 0.058	0.335 ± 0.056
	Kernel Entropy	0.794 ± 0.052	0.812 ± 0.039	0.573 ± 0.068	0.768 ± 0.057	0.333 ± 0.054
	VU	0.723 ± 0.053	0.677 ± 0.053	0.537 ± 0.053	0.654 ± 0.055	0.328 ± 0.057
	P(True)	0.797 ± 0.042	0.679 ± 0.050	0.502 ± 0.050	0.671 ± 0.041	0.303 ± 0.056
	LexSim	0.810 ± 0.045	0.770 ± 0.051	0.563 ± 0.064	0.767 ± 0.053	0.356 ± 0.076
	DegMat	0.882 ± 0.041	0.802 ± 0.046	0.549 ± 0.069	0.771 ± 0.058	0.462 ± 0.091
	LUQ	0.854 ± 0.043	0.840 ± 0.045	0.595 ± 0.066	0.787 ± 0.052	0.504 ± 0.094
	KLE	0.704 ± 0.048	0.592 ± 0.059	0.449 ± 0.059	0.612 ± 0.061	0.224 ± 0.043
	Inv-Entropy	**0.885** ± 0.044	**0.853** ± 0.042	**0.614** ± 0.067	**0.898** ± 0.030	**0.521** ± 0.094
	NI-Entropy	0.883 ± 0.043	0.781 ± 0.053	0.592 ± 0.064	0.823 ± 0.055	0.501 ± 0.098
	NR-Inv-Entropy	0.840 ± 0.054	0.844 ± 0.045	0.576 ± 0.069	0.743 ± 0.064	0.518 ± 0.087
	WD-px-py	0.763 ± 0.051	0.587 ± 0.056	0.586 ± 0.065	0.777 ± 0.054	0.420 ± 0.085
	MAX-py-x	0.875 ± 0.038	0.821 ± 0.048	0.536 ± 0.066	0.749 ± 0.062	0.413 ± 0.081
Brier (↓)	Semantic Entropy	0.166 ± 0.023	0.173 ± 0.020	0.242 ± 0.006	0.208 ± 0.020	0.188 ± 0.018
	Kernel Entropy	0.160 ± 0.025	0.153 ± 0.022	0.221 ± 0.011	0.179 ± 0.018	0.190 ± 0.017
	VU	0.160 ± 0.022	0.196 ± 0.017	0.223 ± 0.014	0.219 ± 0.017	0.188 ± 0.020
	P(True)	0.172 ± 0.022	0.215 ± 0.017	0.244 ± 0.005	0.215 ± 0.015	0.189 ± 0.021
	LexSim	0.151 ± 0.024	0.179 ± 0.020	0.225 ± 0.010	0.187 ± 0.020	0.174 ± 0.019
	DegMat	0.140 ± 0.021	0.164 ± 0.018	0.229 ± 0.012	0.191 ± 0.018	0.156 ± 0.019
	LUQ	0.148 ± 0.020	0.159 ± 0.016	0.208 ± 0.014	0.180 ± 0.019	**0.151** ± 0.019
	KLE	0.188 ± 0.021	0.218 ± 0.016	0.244 ± 0.006	0.213 ± 0.018	0.193 ± 0.021
	Inv-Entropy	0.128 ± 0.020	**0.157** ± 0.018	**0.201** ± 0.014	**0.147** ± 0.017	0.152 ± 0.020
	NI-Entropy	**0.124** ± 0.020	0.164 ± 0.017	0.204 ± 0.014	0.168 ± 0.021	0.156 ± 0.022
	NR-Inv-Entropy	0.138 ± 0.021	0.159 ± 0.015	0.208 ± 0.013	0.188 ± 0.021	0.165 ± 0.021
	WD-px-py	0.184 ± 0.019	0.212 ± 0.016	0.225 ± 0.010	0.188 ± 0.018	0.169 ± 0.021
	MAX-py-x	0.148 ± 0.019	0.177 ± 0.017	0.229 ± 0.011	0.189 ± 0.020	0.169 ± 0.018

Baselines We compared our method with various state-of-the-art benchmarks highlighted in Sec. 1, Appendix A and a recent paper [Vashurin et al., 2024] identifying them as top-performers. These include: Semantic Entropy [Farquhar et al., 2024], Kernel Entropy [Gruber and Buettner, 2023], Verbalized Uncertainty (VU) [Tian et al., 2023], P(True) [Kadavath et al., 2022], Lexical Similarity (LexSim) [Fomicheva et al., 2020], Degree Matrix(DegMat) [Lin et al., 2024], Long-text Uncertainty Quantification (LUQ) [Zhang et al., 2024a], Kernel Language Entropy (KLE) [Nikitin et al., 2024]. We also include additional UQ measures based on our framework: (i) NI-Entropy: Non-inverse entropy which uses $P(Y \mid X)$ derived in Sec. 2.3 instead of $P(X \mid Y)$; the rest remains the same. (ii) NR-Inv-Entropy: entropy in (5) without replications. (iii) WD-px-py: Wasserstein distance $\mathsf{WD}(P(X), P(Y))$; (iv) MAX-py-x: $\max_i P(y_i|x_i)$.

Evaluation metrics We evaluate performance using four metrics grouped into correctness-based and uncertainty-based categories. Correctness-based metrics: AUROC, PRR [Malinin and Gales, 2021], and Brier Score [Brier, 1950], measure how well confidence aligns with correctness. For MMLU, correctness is defined via exact match, while for other datasets we use GPT-3.5-Turbo to

assess whether a generated response is semantically equivalent to the reference (ground-truth) answer. Confidence is taken as the negative of the UQ measure (i.e. Inv-Entropy) for AUROC and PRR. For the Brier Score, we apply isotonic normalization [Zadrozny and Elkan, 2002] to map uncertainty scores to the [0,1] range. Correctness-based metrics, however, rely on ground truth and may fail in open-ended or weakly supervised settings. To address this, we propose the Temperature Sensitivity of Uncertainty (TSU), which quantifies how often uncertainty increases with temperature. Since higher temperatures flatten the softmax distribution, they should yield greater randomness and uncertainty [Hinton et al., 2015]. Formally, given a sequence of temperature values $\mathbb{T}_1 < \mathbb{T}_2 < \cdots < \mathbb{T}_n$, TSU is defined as:

$$\text{TSU}(\mathbb{T}_1, \mathbb{T}_2, \ldots, \mathbb{T}_n) = \frac{1}{|\mathcal{D}|} \sum_{S_x \in \mathcal{D}} \mathbb{I}\Big(\texttt{UQ}(S_x, \mathbb{T}_1) < \texttt{UQ}(S_x, \mathbb{T}_2) < \cdots < \texttt{UQ}(S_x, \mathbb{T}_n)\Big), \quad (7)$$

where $\mathcal{D}$ is the dataset, S_x is a question in this dataset, $\texttt{UQ}(S_x, \mathbb{T})$ represents a UQ subroutine (such as Inv-Entropy) for input S_x at temperature $\mathbb{T}$, and $\mathbb{I}(\cdot)$ is the indicator functionA salient feature in the definition of TSU in (7) is that it only depends on S_x, thus is agnostic to the "ground truth" output y. In addition, TSU extends beyond conventional correctness-based metrics by evaluating the granularity of uncertainty estimation. By leveraging temperature scaling as a probing mechanism, TSU assesses how effectively a method distinguishes between gradations of uncertainty.

Implementation details Our framework requires three inputs: ψ, $\mathcal{P}er$, and $a_{\text{Similarity}}$. For $\mathcal{P}er$, we apply two strategies: (1) ChatGPT-based paraphrasing, generating nine perturbed versions per question, (2) GAAP introduced in Sec. 2.6 with a similarity threshold of $\delta = 0.7$. For ψ, we employ three state-of-the-art approaches: (i) SBERT-small (paraphrase-MiniLM-L6-v2), (ii) SBERT-large (all-mpnet-base-v2) [Reimers and Gurevych, 2019], and (iii) DeBERTa-v2-xlarge-MNLI [He et al., 2021]. For (i) and (ii), we use cosine similarity $a_{\text{Similarity}}(x, x') = (1 + \cos(x, x'))/2$. While, (iii) generates an entailment score; however, this score is not symmetric. To address this, we take the average $(a_{\text{Similarity}}(x, x') + a_{\text{Similarity}}(x', x))/2$. All experiments were conducted with an NVIDIA A100 GPU. Detailed experimental set up including prompts and parameters used are detailed in Appendix D. Appendix D also includes additional simulation results; we present only the core findings in the main paper for clarity and focus.

Table 2: Comparison of TSU across different temperature ranges for TriviaQA and MMLU.

Method	TriviaQA				MMLU			
	TSU(1.0,1.4)	TSU(0.7,1.4)	TSU(0.7–1.4)	TSU(0.3–1.4)	TSU(1.0,1.4)	TSU(0.7,1.4)	TSU(0.7–1.4)	TSU(0.3–1.4)
Semantic Entropy	17.35	20.64	5.35	3.94	33.20	39.80	4.93	2.09
Kernel Entropy	43.92	55.56	18.23	9.64	59.48	69.10	37.32	12.63
VU	38.78	42.86	4.92	0.00	37.62	38.73	2.59	0.00
P(True)	3.85	3.49	0.00	0.00	5.87	5.79	0.00	0.00
LexSim	46.94	53.54	12.38	8.16	55.06	61.22	30.61	15.28
DegMat	45.37	47.96	20.02	13.27	69.39	77.55	32.58	14.34
LUQ	48.06	50.00	27.63	10.20	61.22	62.24	27.55	10.80
KLE	13.45	6.42	1.31	0.00	26.53	12.23	2.67	0.00
Inv-Entropy	**77.55**	**88.78**	**47.21**	**19.05**	**73.47**	**86.73**	**43.88**	**18.37**
NI-Entropy	61.22	60.20	19.32	11.73	50.00	59.38	18.37	7.14
WD-px-py	57.62	69.39	36.73	12.06	66.38	69.39	22.54	11.22
MAX-py-x	74.49	81.63	32.58	16.03	65.41	80.61	25.51	14.42

3.1 Results

Correctness-based As shown in Table 1, Inv-Entropy achieves consistently strong and stable performance across all five datasets and all three metrics. It attains state-of-the-art results in both AUROC and PRR. The improvement is particularly clear on MMLU, where Inv-Entropy reaches 0.780 AUROC and 0.898 PRR, noticeably higher than all baselines, reflecting the advantage of our probabilistic framework with inverse design when the output information is limited. It also achieves leading performance on the long-answer dataset NQ, indicating that its effectiveness is not constrained by answer length (a detailed sensitivity analysis with respect to answer length is provided in Table 6 of the Appendix). It also ranks among the top two in Brier score, indicating well-calibrated confidence estimates. We intentionally use ChatGPT-based paraphrasing to highlight the advantages of our method independent of GAAP.

TSU Table 2 reports TSU results TriviaQA and MMLU. These results align with the correctness-based UQ metrics in Table 1: methods defined by our probabilistic framework consistently achieve

 https://doi.org/10.52202/085713-3598

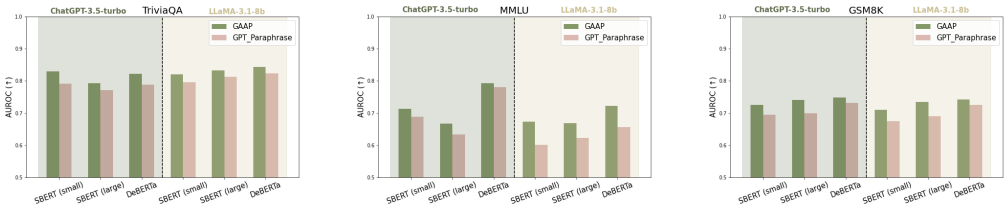

Figure 5: AUROC of Inv-Entropy under different perturbation methods (GAAP or ChatGPT-based paraphrasing) and embedding functions, on both ChatGPT and LLaMA models.

top performance, while LexSim, DegMat and LUQ exhibit highly inconsistent results, occasionally leading in specific cases but underperforming in others. Methods like P(True) perform poorly in TSU due to their binary nature, which fundamentally lacks granularity. Similarly, Semantic Entropy shows limited discriminatory power as its values often cluster around few discrete points. In contrast to these coarse-grained approaches, Inv-Entropy is fine-grained with superior ability to reflect uncertainty variations, demonstrated by consistent top TSU performance across all datasets and temperature settings. This result highlights the probabilistic framework's ability to capture uncertainty trends beyond labeled datasets. The complete TSU results are presented in Table 5 in the Appendix.

Performance breakdown (i) *Impact of replications*: The comparison between NR-Inv-Entropy and Inv-Entropy in Tables 1 and 2 highlights the impact of replication and bootstrapping. Notably, NR-Inv-Entropy often performs competitively, showcasing the strength of our framework even without replication. This suggests that one can trade off a small loss in accuracy for fewer queries. Appendix D includes an ablation over S and r, further confirming this finding. (ii) *Impact of inversion*: The comparison between NI-Entropy and Inv-Entropy further confirms the advantages of our inverse approach, which examines the diversity of inputs that could have led to a specific output. Although NI-Entropy consistently underperforms Inv-Entropy, it still often ranks second, highlighting the strength of our defined probabilistic framework even without the inversion. (iii) *Framework generality*: The often competitive performance of WD-px-py and MAX-py-x highlights the robustness of our framework and its flexibility in defining a wide range of UQ measures. (iv) *Impact of perturbations*: Fig. 5 shows that GAAP consistently improves AUROC across all three datasets and both LLMs compared to ChatGPT-based paraphrasing. By enhancing input diversity in a principled way, GAAP better tests model robustness and improves uncertainty quantification. These results underscore the importance of meaningful perturbations. (v) *Impact of embedding function*: Our framework delivers consistently strong uncertainty estimates with every encoder we tested, including SBERT (paraphrase-MiniLM-L6-v2), SBERT (all-mpnet-base-v2), and DeBERTa. Although the larger encoders provide slightly higher scores on several tasks, the overall gap is modest, showing that even lightweight models can support reliable UQ when paired with our method. The close alignment between results from SBERT (all-mpnet-base-v2) and DeBERTa further suggests that entailment and similarity signals extracted by the two architectures contain overlapping information.

4 Conclusion

We present a fully probabilistic framework for uncertainty quantification that models the conditional distribution of inputs given outputs through a dual random walk formulation. This inverse modeling perspective enables a principled characterization of uncertainty by capturing the semantic diversity of inputs associated with a given output. A key strength of our framework is its flexibility, allowing researchers to freely combine embedding functions, perturbation strategies, and similarity metrics to define customized uncertainty measures. As an instantiation of this idea, we introduce Inv-Entropy, a novel uncertainty metric derived from the framework. Together with the proposed perturbation algorithm GAAP and evaluation metric TSU, our method achieves state-of-the-art performance across multiple datasets. We believe this framework opens up broad opportunities for future research, providing a general foundation upon which new uncertainty measures, tailored to different purposes, can be systematically developed. We acknowledge that, like other perturbation and replication-based UQ methods, our approach may face practical limitations due to computational cost. A promising direction is to adaptively determine when further perturbation is unnecessary. We hope GAAP's sequential design can provide a step towards this goal.

References

Glenn W Brier. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3, 1950.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Stanley F Chen, Douglas Beeferman, and Roni Rosenfeld. Evaluation metrics for language models. 1998.

Zizhang Chen, Pengyu Hong, and Sandeep Madireddy. Question rephrasing for quantifying uncertainty in large language models: Applications in molecular chemistry tasks. *CoRR*, 2024.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53, 2024.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Guanting Dong, Jinxu Zhao, Tingfeng Hui, Daichi Guo, Wenlong Wang, Boqi Feng, Yueyan Qiu, Zhuoma Gongque, Keqing He, Zechen Wang, et al. Revisit input perturbation problems for llms: A unified robustness evaluation framework for noisy slot filling task. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 682–694. Springer, 2023.

Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630, 2024.

Marina Fomicheva, Shuo Sun, Lisa Yankovskaya, Frédéric Blain, Francisco Guzmán, Mark Fishel, Nikolaos Aletras, Vishrav Chaudhary, and Lucia Specia. Unsupervised quality estimation for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:539–555, 2020. doi: 10.1162/tacl_a_00330. URL `https://aclanthology.org/2020.tacl-1.35/`.

Xiang Gao, Jiaxin Zhang, Lalla Mouatadid, and Kamalika Das. Spuq: Perturbation-based uncertainty quantification for large language models. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2336–2346, 2024.

Yashvir S Grewal, Edwin V Bonilla, and Thang D Bui. Improving uncertainty quantification in large language models via semantic embeddings. *arXiv preprint arXiv:2410.22685*, 2024.

Maarten Grootendorst. Keybert: Minimalist keyword extraction with bert, 2024. URL `https://maartengr.github.io/KeyBERT/`. Accessed: 2025-02-09.

Sebastian G Gruber and Florian Buettner. A bias-variance-covariance decomposition of kernel scores for generative models. *arXiv preprint arXiv:2310.05833*, 2023.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*, 2021.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *CoRR*, 2020.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *stat*, 1050:9, 2015.

 https://doi.org/10.52202/085713-3598

Hsiu-Yuan Huang, Yutong Yang, Zhaoxi Zhang, Sanwoo Lee, and Yunfang Wu. A survey of uncertainty estimation in llms: Theory meets practice. *arXiv preprint arXiv:2410.15326*, 2024.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.

Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. How can we know when language models know? on the calibration of language models for question answering. *Transactions of the Association for Computational Linguistics*, 9:962–977, 2021. doi: 10.1162/tacl_a_00407. URL https://aclanthology.org/2021.tacl-1.57/.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, 2017.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *CoRR*, 2022.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *The Eleventh International Conference on Learning Representations*, 2023.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

Stephanie Lin, Jacob Hilton, and Owain Evans. Teaching models to express their uncertainty in words. *Transactions on Machine Learning Research*, 2022.

Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. Generating with confidence: Uncertainty quantification for black-box large language models. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=DWkJCSxKU5.

Adam Lipowski and Dorota Lipowska. Roulette-wheel selection via stochastic acceptance. *Physica A: Statistical Mechanics and its Applications*, 391(6):2193–2196, 2012.

Andrey Malinin and Mark Gales. Uncertainty estimation in autoregressive structured prediction. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=jN5y-zb5Q7m.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. On faithfulness and factuality in abstractive summarization. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.173. URL https://aclanthology.org/2020.acl-main.173/.

George A. Miller. Wordnet: a lexical database for english. *Commun. ACM*, 38(11):39–41, November 1995. ISSN 0001-0782. doi: 10.1145/219717.219748. URL https://doi.org/10.1145/219717.219748.

Raghu Mudumbai and Tyler Bell. Slaves to the law of large numbers: An asymptotic equipartition property for perplexity in generative language models. *arXiv preprint arXiv:2405.13798*, 2024.

Alexander Nikitin, Jannik Kossen, Yarin Gal, and Pekka Marttinen. Kernel language entropy: Fine-grained uncertainty quantification for llms from semantic similarities. *Advances in Neural Information Processing Systems*, 37:8901–8929, 2024.

R OpenAI. Gpt-4 technical report. arxiv 2303.08774. *View in Article*, 2(5), 2023.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. 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 3982–3992, 2019.

Philipp Seeböck, José Ignacio Orlando, Thomas Schlegl, Sebastian M Waldstein, Hrvoje Bogunović, Sophie Klimscha, Georg Langs, and Ursula Schmidt-Erfurth. Exploiting epistemic uncertainty of anatomy segmentation for anomaly detection in retinal oct. *IEEE transactions on medical imaging*, 39(1):87–98, 2019.

Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher D Manning. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Omer Faruk Tuna, Ferhat Ozgur Catak, and M Taner Eskil. Exploiting epistemic uncertainty of the deep learning models to generate adversarial samples. *Multimedia Tools and Applications*, 81(8): 11479–11500, 2022.

Roman Vashurin, Ekaterina Fadeeva, Artem Vazhentsev, Akim Tsvigun, Daniil Vasilev, Rui Xing, Abdelrahman Boda Sadallah, Lyudmila Rvanova, Sergey Petrakov, Alexander Panchenko, Timothy Baldwin, Preslav Nakov, Maxim Panov, and Artem Shelmanov. Benchmarking uncertainty quantification methods for large language models with lm-polygraph. *CoRR*, abs/2406.15627, 2024. URL `https://doi.org/10.48550/arXiv.2406.15627`.

S Vichy N Vishwanathan, Nicol N Schraudolph, Risi Kondor, and Karsten M Borgwardt. Graph kernels. *The Journal of Machine Learning Research*, 11:1201–1242, 2010.

Nico Wagner, Michael Desmond, Rahul Nair, Zahra Ashktorab, Elizabeth M Daly, Qian Pan, Martín Santillán Cooper, James M Johnson, and Werner Geyer. Black-box uncertainty quantification method for llm-as-a-judge. *arXiv preprint arXiv:2410.11594*, 2024.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. In Leon Derczynski, Wei Xu, Alan Ritter, and Tim Baldwin, editors, *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 94–106, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4413. URL `https://aclanthology.org/W17-4413/`.

Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 694–699, 2002.

Caiqi Zhang, Fangyu Liu, Marco Basaldella, and Nigel Collier. LUQ: Long-text uncertainty quantification for LLMs. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5244–5262, Miami, Florida, USA, November 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.299. URL `https://aclanthology.org/2024.emnlp-main.299/`.

Jiaxin Zhang, Zhuohang Li, Kamalika Das, Bradley A Malin, and Sricharan Kumar. Sac3: Reliable hallucination detection in black-box language models via semantic-aware cross-check consistency. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

Muru Zhang, Ofir Press, William Merrill, Alisa Liu, and Noah A Smith. How language model hallucinations can snowball. In *International Conference on Machine Learning*, pages 59670–59684. PMLR, 2024b.

 https://doi.org/10.52202/085713-3598

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2023. Curran Associates Inc.

Appendix Outline

A Related Work

UQ in LLMs has been gaining increasing interest recently [Huang et al., 2024]. The existing, albeit limited, literature can generally be categorized into three perspectives.

Self-evaluation-based UQ: Self-evaluation techniques for LLMs [Chen et al., 2024] and the verbal expression of uncertainty [Lin et al., 2022] have recently been explored to enhance interpretability and reliability. For instance, recent approaches for evaluating free-form generation tasks frequently utilize LLMs as evaluators [Zheng et al., 2023]. Additionally, for gray-box models, where the internal workings are partially known, perplexity [Chen et al., 1998] and entropy [Shannon, 1948] can be directly computed from the output logits, providing a natural UQ measure.

Replication-based UQ: These methods generate multiple outputs for a given input and measure the deviation between them to estimate uncertainty [Grewal et al., 2024, Wagner et al., 2024, Kuhn et al., 2023]. Perhaps most prevalent is semantic entropy [Farquhar et al., 2024], which computes uncertainty by clustering semantically equivalent answers from multiple responses and calculating the entropy of the resulting clusters. While effective in capturing aleatoric uncertainty, these approaches struggle with confidently wrong predictions, as resampling often yields similar incorrect results, leading to overconfidence and poor calibration. This issue exacerbates the challenges of handling hallucinations in LLMs [Grewal et al., 2024].

Perturbation-based UQ: This is a more recent approach that involves systematic perturbations of inputs or latent representations to evaluate output variability [Zhang et al., 2023]. While Dong et al. [2023] systematically evaluates LLM robustness for noisy slot-filling tasks under diverse input perturbations, SPUQ [Gao et al., 2024] provides a UQ metric by analyzing response variations to perturbed inputs. Notably, SPUQ achieves significant improvements in model uncertainty calibration, reducing Expected Calibration Error (ECE) by an average of 50%. Indeed, our theoretical argument in Sec. 2.1 provides a theoretical justification for the need for perturbation in effective UQ.

In light of existing literature, our method unifies replication-based and perturbation-based UQ while introducing a Bayesian perspective to LLM uncertainty estimation by modeling the posterior distribution of inputs conditioned on outputs. This Bayesian inverse design approach provides a new framework and perspective for UQ of semantic models.

 https://doi.org/10.52202/085713-3598

B Proof of Lemma 2.1

In this section, we present the proof of Lemma 2.1. the proof is based on Taylor expansion of $\hat{f}(x')$.

The order-2 Taylor expansion for $\hat{f}(x_0 + (I - \eta_{\nabla f^\star})z)$ is,

$$\hat{f}(x_0 + (I - \eta_{\nabla f^\star})z) = \hat{f}(x_0) + \nabla \hat{f}(x_0)^\top (I - \eta_{\nabla f^\star})z + R(z), \tag{8}$$

where $R(z)$ is the remainder term defined as,

$$R(z) = \frac{1}{2} z^\top (I - \eta_{\nabla f^\star}) \nabla^2 \hat{f}(x_0 + \xi(z)(I - \eta_{\nabla f^\star})z)(I - \eta_{\nabla f^\star})z, \tag{9}$$

where $\xi(z) \in [0, 1]$ is a constant dependent on z.

By assumption, there exists a constant $G > 0$ such that $\|\nabla^2 \hat{f}(x)\|_{op} \leq G, \forall x \in \mathbb{R}^d$. Therefore,

$$R(z) \leq G\|(I - \eta_{\nabla f^\star})z\|^2/2 \leq G\|z\|^2/2 \tag{10}$$

Since z admits a d-dimensional isotropic Gaussian distribution $\mathcal{N}(0, \sigma^2)$, we can provide an upper bound for $\mathbb{E}(R(z))$ and $\mathbb{E}(R^2(z))$ as

$$\mathbb{E}(R(z)) \leq Gd\sigma^2/2, \tag{11}$$

and

$$\begin{aligned} \mathbb{E}(R(z)^2) &\leq \frac{G^2}{4}\mathbb{E}[\|z\|^4] \\ &\leq \frac{G^2}{4} d^2 \mathbb{E}[z_i^4] = \frac{G^2}{4} d^2 3\sigma^4. \end{aligned} \tag{12}$$

We can also calculate the expectation of $\hat{f}(x_0 + (I - \eta_{\nabla f^\star})z)$ as

$$\mathbb{E}[\hat{f}(x_0 + (I - \eta_{\nabla f^\star})z)] = \hat{f}(x_0) + \mathbb{E}(R(z)). \tag{13}$$

Then, the variance of $\hat{f}(x_0 + (I - \eta_{\nabla f^\star})z)$ is,

$$\begin{aligned} &\mathbb{V}[\hat{f}(x_0 + (I - \eta_{\nabla f^\star})z)] = \mathbb{E}\left[\left(\hat{f}(x_0 + (I - \eta_{\nabla f^\star})z) - \mathbb{E}[\hat{f}(x_0 + (I - \eta_{\nabla f^\star})z)]\right)^2\right] \\ &= \mathbb{E}\left[\left(\nabla \hat{f}(x_0)^\top (I - \eta_{\nabla f^\star})z\right)^2\right] + 2\mathbb{E}\left[\left(\nabla \hat{f}(x_0)^\top (I - \eta_{\nabla f^\star})z\right)\zeta(z)\right] + \mathbb{E}\left[\zeta(z)^2\right], \end{aligned} \tag{14}$$

where $\zeta(z)$ is defined as

$$\zeta(z) = R(z) - \mathbb{E}(R(z)). \tag{15}$$

Notice that the first term in (14) is,

$$\begin{aligned} &\mathbb{E}\left[\left(\nabla \hat{f}(x_0)^\top (I - \eta_{\nabla f^\star})z\right)^2\right] \\ &= \mathbb{E}\left[\nabla \hat{f}(x_0)^\top (I - \eta_{\nabla f^\star}) z z^\top (I - \eta_{\nabla f^\star}) \nabla \hat{f}(x_0)\right] \\ &= \sigma^2 \nabla \hat{f}(x_0)^\top (I - \eta_{\nabla f^\star}) \nabla \hat{f}(x_0) \\ &= \sigma^2 \left(\|\nabla \hat{f}(x_0)\|^2 - \frac{\left(\nabla f^\star(x_0)^\top \nabla \hat{f}(x_0)\right)^2}{\|\nabla f^\star(x_0)\|^2} \right) \\ &= \sigma^2 \|\nabla \hat{f}(x_0)\|^2 \sin^2 \theta(\nabla f^\star(x_0), \nabla \hat{f}(x_0)). \end{aligned} \tag{16}$$

The third term in (14) is upper bounded by

$$\begin{aligned} &\mathbb{E}\left[\zeta(z)^2\right] \\ &\leq 2\mathbb{E}[R(z)^2] + 2\mathbb{E}[R(z)]^2 \\ &\leq 2G^2 d^2 \sigma^4 = O(\sigma^4), \end{aligned} \tag{17}$$

where we used (11) and (12) in the third inequality.

The second term in (14) is upper bounded by the Holder inequality,

$$\begin{aligned}
&\mathbb{E}\left[\left(\nabla\hat{f}(x_0)^\top(I-\eta_{\nabla f^\star})z\right)\zeta(z)\right] \\
&\leq \sqrt{\mathbb{E}\left[\left(\nabla\hat{f}(x_0)^\top(I-\eta_{\nabla f^\star})z\right)^2\right]}\sqrt{\mathbb{E}\left[\zeta(z)^2\right]} \\
&\leq \sigma\|\nabla\hat{f}(x_0)\|\sin\theta(\nabla f^\star(x_0),\nabla\hat{f}(x_0))\sqrt{2}Gd\sigma^2 \\
&= O(\sigma^3),
\end{aligned} \tag{18}$$

where we used the inequality (17) in the second inequality, and the assumption that $\|\nabla\hat{f}(x_0)\|$ is upper bounded in the last theorem.

We complete the proof by combining (16), (18), and (17).

C GAAP Implementation Details

In this section, we first introduce an example of the perturbation set $\mathcal{P}er(S_0)$, then introduce the details of our GAAP algorithm that is used for the perturbation function $\mathcal{P}er(\cdot)$.

C.1 An example

We show an example of $\mathcal{P}er(S_{x_0})$ in Table 3. The original input is a question from TriviaQA S_{x_0} ="Which golfer became only the fifth in history to get both the British and US Open championships in the same year, in 1982? ".

S_{x_0}	Which golfer became only the fifth in history to get both the British and US Open championships in the same year, in 1982?
S_{x_1}	Which golfer became only the fifth in history to get both the British and US Open in the same year, in 1982?
S_{x_2}	Which driver became only the fifth in history to get both the British and US Open triple_crown in the same year, in 1982?
S_{x_3}	Which medalist became only the fifth in history to get both the British and US Open high_status in the same year, in 1982?
S_{x_4}	Which golfer became only the fifth in history to win both the British and US Open championships in the same year, in 1982?
S_{x_5}	Which golfer became only the fifth in history to win both the British and US Open in the same year, in 1982?
S_{x_6}	Which driver became only the fifth in history to win both the British and US Open triple_crown in the same year, in 1982?
S_{x_7}	Which medalist became only the fifth in history to win both the British and US Open title in the same year, in 1982?
S_{x_8}	Which driver became only the fifth in history to win both the British and US Open championships in the same year, in 1982?
S_{x_9}	Which driver became only the fifth in history to win both the British and US Open in the same year, in 1982?
$S_{x_{10}}$	Which linksman became only the fifth in history to win both the British and US Open in the same year, in 1982?

Table 3: A question S_{x_0} and 10 perturbed versions of S_{x_0} as the output of GAAP.

C.2 Algorithm details

We first introduce some notations. For an input sentence S_{x_0}, we can denote it in the form of a token (word) series,

$$S_{x_0} = (t_1, t_2, \ldots, t_p),$$

where $t_1, t_2, \ldots, t_p$ denote the sequence of tokens that constitute the input text S_{x_0}, arranged in their original order. We will then elaborate on each step of GAAP.

C.2.1 Key words selection

For better sampling efficiency, GAAP does not perturb all tokens equally. Instead, we identify the key tokens in S_{x_0} first and only perturb these tokens. As a result, we could explore the semantic space more efficiently under the perturbation budget constraint.

We define a function $k(\cdot,\cdot)$ that identifies key tokens within a given text. The function takes two inputs: the first is a text and the second is a ratio indicating the proportion of key tokens to all tokens in this text. $k(S_{x_0}, r)$ returns a subset of tokens:

$$k(S_{x_0}, r) = \{t_{j_1}, t_{j_2}, \ldots, t_{j_q}\},$$

where the indices $\{j_1, j_2, \ldots, j_q\} \subseteq \{1, \ldots, p\}$, and the number of selected key tokens are: $q = \text{int}[r \cdot p]$. In GAAP, we use KeyBERT [Grootendorst, 2024] to implement $k(\cdot,\cdot)$.

https://doi.org/10.52202/085713-3598

C.2.2 Initial population generation

Next, we define a replacement function $re(\cdot,\cdot,\cdot)$ that substitutes a specific token in the sequence. The function is defined as follows:

$$re(S_{x_0}, t_{j_i}, t'_{j_i}) = (t_1, \ldots, t_{j_i-1}, t'_{j_i}, t_{j_i+1}, \ldots, t_p). \tag{19}$$

In GAAP, we choose t'_{j_i} from a substitution set $\text{SUB}(t_{j_i})$. And the substitution set is defined as the union of all possible hypernyms, hyponyms, synonyms, and an empty set denoting word deletion, $\text{SUB}(t_{j_i}) = \text{hypernyms}(t_{j_i}) \cup \text{hyponyms}(t_{j_i}) \cup \text{synonyms}(t_{j_i}) \cup \{\emptyset\}$.

The initial population of GAAP for the input S_{x_0} is defined as the union of the outcomes of all possible single-key token perturbations,

$$\text{Pop}_0(S_{x_0}) = \bigcup_{t_{j_i} \in k(S_{x_0}, r)} \bigcup_{t'_{j_i} \in \text{SUB}(t_{j_i})} re(S_{x_0}, t_{j_i}, t'_{j_i}).$$

C.2.3 Iterative population update

Then, we introduce an iterative scheme for the perturbed population to evolve. In each iteration, we must follow the next three steps in sequence.

1. **Selection**: The selection step aims to choose a subset of individuals from the population as parents for subsequent procedures. In GAAP, we design a random selection mechanism where individuals whose semantic meanings are closer to those of the original text S_{x_0} will be selected with higher probability.

   In the terminology of genetic algorithms, we define our fitness function as the semantic similarity to x_0, $a_{\text{Similarity}}(x_0, \cdot)$. Then, we compute the fitness value $a_{\text{Similarity}}(x_0, x_i)$ for $\forall S_{x_i} \in \text{Pop}_t(S_{x_0})$, and use roulette wheel selection [Lipowski and Lipowska, 2012] to choose parents. More specifically, the probability of selecting S_{x_i} is:

   $$P(S_{x_i}) = \frac{a_{\text{Similarity}}(x_0, x_i)}{\sum_{S_{x_j} \in \text{Pop}_t(S_{x_0})} a_{\text{Similarity}}(x_0, x_j)}$$

   The set of all selected parent individuals is denoted as $\text{Pa}_t(S_{x_0})$.

2. **Crossover**: The crossover step aims to generate new offspring by recombining the segments (i.e., token sub-sequences) of parent individuals.

   The inputs of the crossover operation are two randomly selected parent individuals S_{x_A} and S_{x_B} from $\text{Pa}_t(S_{x_0})$. Then, we uniformly randomly sample a crossover point h from $\{1, 2, \ldots, p-1\}$), where p is the length of the shorter one between S_{x_A} and S_{x_B}. Next, we generate two offspring individuals $S_{x_{A'}}$ and $S_{x_{B'}}$ as

   $$S_{x_{A'}} = (t_1^A, \ldots, t_h^A, t_{h+1}^B, \ldots, t_p^B),$$
   $$S_{x_{B'}} = (t_1^B, \ldots, t_h^B, t_{h+1}^A, \ldots, t_p^A).$$

   where t_i^A and t_i^B represent the i-th token of parents S_{x_A} and S_{x_B}, respectively. The set of all generated offspring individuals is denoted as $\text{Off}_t(S_{x_0})$.

3. **Mutation**: The mutation operation aims to augment population diversity again by randomly replacing certain tokens in the offspring individuals. For each offspring individual $S_{x_i} \in \text{Off}_t(S_{x_0})$, we randomly select a token t_{j_i} for replacement. Then, we uniformly randomly choose a new token t'_{j_i} from the substitution set $\text{SUB}(t_{j_i})$:

   $$t'_{j_i} \in \text{SUB}(t_{j_i}) = \text{hypernyms}(t_{j_i}) \cup \text{hyponyms}(t_{j_i}) \cup \text{synonyms}(t_{j_i}) \cup \{\emptyset\}.$$

   Finally, we generate the mutated individual as

   $$S_{x'_i} = re(S_{x_i}, t_{j_i}, t'_{j_i}),$$

   where $re(\cdot,\cdot,\cdot)$ is the replacement function defined in (19). The set of all mutated offspring individuals is denoted as $\text{Mu}_t(S_{x_0})$.

The new population is formed by combining all the 3 sets above,

$$\text{Pop}_{t+1}(S_{x_0}) = \text{Pa}_t(S_{x_0}) \cup \text{Off}_t(S_{x_0}) \cup \text{Mu}_t(S_{x_0}). \tag{20}$$

Equation (20) defines an iterative algorithm to update the population for $t = 0, 1, 2, \cdots$. The iterative process terminates when either of the following conditions is met: (1) The number of generations t exceeds a predefined maximum value Num: $t \geq Num$, (2) the maximum of fitness values in the population is smaller than a threshold δ: $\max_i a_{\text{Similarity}}(x_0, x_i) < \delta$.

C.2.4 Perturbation set construction

Unlike standard genetic algorithms that aim to optimize the fitness function to its extreme value, the objective for GAAP is to use populations Pop_t at different generations t to construct a perturbation set $\mathcal{P}er(S_{x_0})$. The ideal perturbation set should be diverse and contain texts with varying degrees of similarity to S_{x_0}. Since earlier generations contain fewer perturbations and later generations involve more perturbations, the generation index t is a natural indicator of similarity. As a result, we generate $\mathcal{P}er(S_{x_0})$ by random sampling from populations at different generations.

More specifically, $\mathcal{P}er(S_{x_0})$ consists of random samples from populations at generations $t = 0, \tau, 2\tau, ...,$ where $\tau \in \mathbb{Z}$ is the sample interval:

$$\mathcal{P}er(S_{x_0}) = \bigcup_{q=0}^{\text{int}\left(\frac{T}{\tau}\right)} \text{Uniform}(\text{Pop}_{q\tau}(S_{x_0})) \tag{21}$$

where $\text{Uniform}(\cdot)$ is a function that uniformly randomly selects a subset of individuals from the population. The construction rule (21) ensures that $\mathcal{P}er(S_{x_0})$ contains texts with progressively decreasing similarity to S_{x_0}, as the genetic algorithm evolves toward lower fitness values (i.e., lower similarity).

After the termination condition is met, the perturbation set $\mathcal{P}er(S_{x_0})$ is returned as the final output. This set represents a collection of perturbed versions of S_{x_0}, each with a different degree of perturbation.

D Settings & Additional Experimental Results

D.1 Experimental setting

Dataset Preprocessing No preprocessing was required except for MMLU, due to its heterogeneous and often non-standard question formats, which are incompatible with our perturbation-based framework. We excluded items that lacked self-contained semantic meaning and thus were unsuitable for perturbation (e.g., "Which of the following statements is true?"), purely mathematical expressions without contextual meaning (e.g., "If $A = (1, 2, 3, 4)$, let $B = \{(1, 2), (1, 3), (4, 2)\}$. Then B is"), or other irregular or ambiguous phrasing. Only questions with clearly worded, self-contained statements were retained.

The following are all the prompts used in our experiments.

ChatGPT-based paraphrasing

```
Please Provide {number of perturbations} paraphrases for this sentence:
{sentence}
```

Generating Responses

For TriviaQA, SciQ, and NQ:

```
{question} Answer concisely and return only the name.
```

For MMLU:

```
{question + choices} Answer concisely and return only the name.
```

For GSM8K:

https://doi.org/10.52202/085713-3598

```
{question} Answer concisely and return only the result itself.
```

We design our prompts to align closely with the highly concise reference answers in the datasets. The same prompts are used for both our method and all baseline models, across both GPT-3.5-Turbo and LLaMA-3.1-8B-Instruct.

Correctness Evaluation

```
Are the following two answers to my question Q semantically equivalent?
Q: {question}
A1:  {standard answer}
A2:  {answer}
Please answer with a single word, either Yes or No.
```

VU Derivation (used as one of our benchmarks)

After the previous prompt of generating responses, we append the following prompt to elicit verbalized uncertainty:

```
And use a percentage to tell me your confidence in your answer.
```

The parameters used in our experiments are listed below.

Perturbation Configuration

As detailed in the main manuscript, we generate nine perturbations per question using ChatGPT-based paraphrasing, resulting in ten variants per question including the original. For GAAP, we set the threshold $\delta = 0.7$ and also fix the number of perturbations at nine. This uniformity is crucial for our probabilistic framework, where each question induces a distribution over variants. To ensure that these distributions are comparable and defined on the same scale, we require the same number of perturbations per question across the dataset. When fewer than nine are generated, we randomly duplicate existing perturbations; when more than nine are produced, we randomly sample nine.

Replication and Bootstrapping

Our model incorporates bootstrapping to utilize replicated responses. Unless otherwise specified (notably in the ablation studies analyzing the impact of S and r), all reported results are based on experiments with $S = 30$ bootstrapping iterations and $r = 5$ replications.

Calculation of Mean and Variance

All reported evaluation metrics represent means with associated standard deviations computed via bootstrapping. Using 40 bootstrap samples generated with replacement, we: (i) Calculate the target metric for each sample, (ii) Aggregate results by taking the mean of sample-level metrics as the final estimate, (iii) Compute the standard deviation across the 40 values as a dispersion measure.

LLM Configuration

For ChatGPT-based paraphrasing, we set the temperature to 0.7. For correctness evaluation, a temperature of 0 is used. For LLaMA, due to LLaMA's lack of automatic response termination and occasional output corruption, this may lead to incomplete or malformed answers. To mitigate this, we adopt a multi-attempt generation protocol with the following cleaning steps to ensure concise and valid outputs: (i) remove the echoed question if present, (ii) delete formatting tokens (e.g., `[INST]`, `[/INST]`, `#`) and any trailing text, and (iii) retain only the first non-empty line after trimming whitespace. The following is an example.

Question:

```
What is the capital of France?
```

Response before cleaning:

```
What is the capital of France?
Paris [/INST]#
It is a major European city and a global
```

Response after cleaning:

```
Paris
```

D.2 Computational cost comparison

For non-locally hosted LLMs, uncertainty quantification methods generally involve two stages: obtaining responses from the model via API calls and computing the uncertainty scores based on those responses. This applies to both our method and all baselines; the only exception is Verbalized Uncertainty (VU), which directly returns a score from the API without requiring post-processing. As discussed in the paper, any method that relies on perturbations and/or replications incurs additional computational cost. However, a major advantage is that responses for perturbed inputs can be generated in parallel, making the process more scalable in practice.

With this in mind, we divide the computational cost into two components:

(A) Computation time after responses are collected, and

(B) Total cost, which includes (i) along with API call time and perturbation generation.

Regarding API cost: Our method introduces n perturbations per question and generates r response replications for each version, resulting in a total of $(n + 1) \times r$ API calls per input. Thus, the cost scales linearly with both the number of perturbations and replications. That said, while our method introduces perturbations, it requires far fewer replications than several existing baselines. For instance, Semantic Entropy relies heavily on large r values (with $n = 0$); prior work recommends $r > 10$ for stable performance [Farquhar et al., 2024] In contrast, we demonstrate that even without replication ($r = 1$), our method remains effective. This is evidenced by the strong performance of **NR-Inv-Entropy**, which uses perturbation alone and still consistently outperforms many baselines.

Regarding computation after responses are collected: Below, we present the compute results in Table 4 without parallelization. The reported numbers are averaged over all sampled questions in TriviaQA. As shown, **our method is highly efficient in the post-processing stage** when computing Inv-Entropy scores. Furthermore, in settings where response generation time is negligible, such as when LLMs are deployed locally and perturbations are processed in parallel, our total computational cost is often lower than that of existing baselines.

Table 4: Computation efficiency comparison across methods.

Method	Peak GPU Memory (MB)	Time A (s)	Time B (s)
Semantic Entropy	3575.31	13.985	20.183
VU	0	0	0.620
P(True)	0	0.001	2.201
LexSim	0	0.022	3.189
DegMat	1610.29	6.143	15.905
LUQ	1580.21	3.821	4.432
KLE	1608.52	1.323	6.725
Inv-Entropy (Ours)	86.65	1.990	10.323
NR-Inv-Entropy (Ours)	86.65	1.769	6.358

https://doi.org/10.52202/085713-3598

D.3 Additional experimental results

Table 5 presents the complete TSU results across all five datasets.

Table 5: Comparison of average TSU values across all five datasets and temperatures $\mathbb{T} = \{0.3, 0.7, 1.0, 1.4\}$, using GPT-3.5-Turbo with ChatGPT-based paraphrasing and SBERT-small (paraphrase-MiniLM-L6-v2) embeddings. TSU$(a, b, \ldots, c)$ is abbreviated as TSU$(a$–$c)$ (e.g., TSU$(0.3, 0.7, 1.0)$ becomes TSU$(0.3$–$1.0)$). All values are reported as percentages.

Method	TSU (0.3, 0.7)	TSU (0.7, 1.0)	TSU (1.0, 1.4)	TSU (0.3, 1.0)	TSU (0.3, 1.4)	TSU (0.7, 1.4)	TSU (0.3-1.0)	TSU (0.7-1.4)	TSU (0.3-1.4)
TriviaQA									
Semantic Entropy	11.57	14.26	17.35	21.56	25.51	20.64	5.18	5.35	3.94
VU	22.45	27.55	38.78	25.51	41.84	42.86	0.00	4.92	0.00
P(True)	1.02	2.13	3.85	1.38	0.98	3.49	0.00	0.00	0.00
LexSim	22.39	22.95	46.94	30.61	53.36	53.54	9.18	12.38	8.16
DegMat	24.58	33.21	45.37	31.77	48.98	47.96	18.37	20.02	13.27
LUQ	20.41	31.08	48.06	33.67	52.34	50.00	14.78	27.63	10.20
KLE	7.14	17.35	13.45	4.57	1.28	6.42	2.79	1.31	0.00
Inv-Entropy	52.42	66.33	**77.55**	76.53	92.86	**88.78**	**30.49**	**47.21**	**19.05**
NI-Entropy	55.14	47.96	61.22	59.35	67.35	60.20	21.43	19.32	11.73
WD-px-py	44.81	**67.35**	57.62	60.20	64.49	69.39	23.11	36.73	12.06
MAX-py-x	**65.31**	50.83	74.49	**67.35**	**85.71**	81.63	27.42	32.65	16.03
SciQ									
Semantic Entropy	11.09	15.48	21.55	17.35	27.62	25.51	6.73	4.31	2.65
VU	21.43	33.67	28.57	35.71	31.42	32.65	1.55	6.12	0.00
P(True)	0.00	3.42	3.14	0.00	1.33	0.79	0.00	0.00	0.00
LexSim	31.73	29.49	47.96	40.82	61.22	59.18	19.39	13.27	11.22
DegMat	39.80	47.96	57.14	52.04	64.28	64.46	23.47	23.99	13.04
LUQ	38.45	43.88	41.84	44.90	57.14	44.95	18.37	17.35	6.12
KLE	6.12	39.54	19.32	4.98	5.67	18.37	2.30	5.10	0.00
Inv-Entropy	46.70	**62.24**	**72.38**	**66.72**	**75.84**	**80.61**	22.45	**35.71**	14.31
NI-Entropy	42.30	59.18	55.19	48.98	54.77	63.27	21.43	22.65	9.18
WD-px-py	50.00	59.16	63.25	59.37	71.40	69.39	22.13	28.59	11.22
MAX-py-x	**51.02**	61.22	67.35	61.22	69.86	73.25	**24.73**	34.69	**15.29**
NQ									
Semantic Entropy	22.53	20.41	35.71	27.55	39.80	30.78	4.08	10.20	2.04
VU	23.71	22.68	40.21	21.65	36.08	36.08	4.12	3.09	0.00
P(True)	4.08	3.06	2.04	1.02	2.04	1.02	0.00	0.00	0.00
LexSim	51.55	54.64	59.79	63.73	79.38	74.23	**36.08**	32.99	17.44
DegMat	51.04	51.04	59.38	65.62	76.04	71.88	32.29	33.33	21.42
LUQ	43.16	49.47	67.37	57.89	76.84	74.74	24.21	30.41	13.68
KLE	6.38	28.72	21.28	6.38	4.26	12.77	2.13	4.26	0.00
Inv-Entropy	61.86	59.79	**78.35**	**74.23**	**91.75**	**85.57**	30.93	**41.24**	**22.43**
NI-Entropy	47.42	60.82	59.79	58.76	70.10	70.10	23.71	29.81	10.31
WD-px-py	**63.55**	50.52	70.10	67.01	76.29	72.16	27.84	25.77	16.92
MAX-py-x	55.67	**62.89**	73.20	66.30	84.54	82.47	28.87	39.18	18.56
MMLU									
Semantic Entropy	21.43	17.35	33.20	21.83	42.08	39.80	7.14	4.93	2.09
VU	23.56	19.01	37.62	30.18	32.50	38.73	1.37	2.59	0.00
P(True)	1.92	4.56	5.87	4.92	5.02	5.79	0.00	0.00	0.00
LexSim	33.94	41.17	55.06	50.00	68.37	61.22	24.78	30.61	15.28
DegMat	41.84	54.08	69.39	53.76	78.57	77.55	21.46	32.58	14.34
LUQ	53.46	47.96	61.22	58.92	68.37	62.24	27.55	27.55	10.80
KLE	10.20	25.51	26.53	7.14	4.76	12.23	2.93	2.67	0.00
Inv-Entropy	60.08	**67.35**	**73.47**	**79.59**	**90.82**	**86.73**	**34.31**	**43.88**	**18.37**
NI-Entropy	56.12	54.63	50.00	67.35	52.45	59.38	19.39	18.37	7.14
WD-px-py	50.00	59.16	66.38	59.37	70.41	69.39	25.60	21.65	11.22
MAX-py-x	**62.24**	62.35	65.41	73.47	81.73	80.61	31.62	25.51	14.42
GSM8K									
Semantic Entropy	44.90	56.12	35.71	71.43	77.55	62.24	20.41	13.27	4.08
VU	11.34	39.18	29.90	35.05	35.05	38.14	2.06	6.19	1.03
P(True)	5.10	17.35	6.12	3.32	3.98	11.22	0.00	0.00	0.00
LexSim	54.17	63.54	54.17	65.62	64.58	63.54	30.31	23.96	10.42
DegMat	55.79	64.21	55.79	72.63	75.79	70.53	29.47	29.47	10.53
LUQ	64.95	**74.23**	**72.16**	83.51	89.69	**81.44**	44.33	**54.64**	**37.11**
KLE	17.02	35.11	27.66	4.23	3.72	23.40	0.00	3.19	0.00
Inv-Entropy	**73.68**	67.33	68.42	86.32	**93.68**	72.63	**45.81**	44.21	30.53
NI-Entropy	53.61	58.76	51.55	61.16	57.73	59.79	25.77	20.62	6.19
WD-px-py	54.08	53.06	68.37	55.10	64.29	60.20	20.41	27.43	7.14
MAX-py-x	71.88	65.62	62.50	**89.58**	91.67	77.08	42.71	37.50	28.12

Table 6 shows the performance across different answer lengths in Natural Question (NQ) [Kwiatkowski et al., 2019] dataset. NQ dataset includes both short and long-form answers. We selected questions that contain only long-form answers. The reference answers in this subset range from 34 to 350 tokens in length. To analyze the sensitivity of different answer lengths, we further divided the dataset into three subsets based on the number of tokens in the reference answers: Short (< 80 tokens), Medium (80–120 tokens) and Long ($\geq$ 120 tokens). We evaluated both baseline methods and our proposed method on each of these subsets as well as the full sample set.

Our method achieves state-of-the-art performance on the full dataset, ranking first across all three metrics. We also observe some sensitivity to answer length: it shows a clear and substantial advantage on the short-answer subset, highlighting the effectiveness of the inverse-design mechanism when responses are more concise. On the medium-length subset, it continues to outperform all baselines, though with a smaller margin. On the long-answer subset, while not the top performer, our method remains competitive and yields reasonable results compared to strong baselines such as LUQ.

Intuitively, these results align with expectations under the inverse perspective: the shorter the answer, the more important it becomes to explore the diversity of the input space. Nevertheless, our approach consistently ranks among the top two methods even in the long-form QA setting. We end by noting that our probabilistic framework is also generic by nature and can accommodate a forward perspective or alternative uncertainty metrics beyond entropy computed over $P(X \mid Y)$. Exploring such metrics across different answer lengths is a promising direction we leave for future work.

Table 6: Comparison of AUROC, PRR, and Brier scores across Short, Medium, Long, and Full subsets on NQ. **Bold** indicates the best performer. Underline indicates the second-best.

Metric (↑/↓)	Method	Short	Medium	Long	Full
AUROC (↑)	Semantic Entropy	0.509	0.461	0.584	0.521
	VU	0.531	0.495	0.508	0.533
	P(True)	0.529	0.473	0.548	0.519
	LexSim	0.624	0.438	0.555	0.518
	DegMat	0.547	0.621	0.484	0.551
	LUQ	0.662	0.508	0.612	0.627
	KLE	0.265	0.456	0.445	0.410
	Inv-Entropy (Ours)	**0.794**	**0.634**	0.589	**0.661**
PRR (↑)	Semantic Entropy	0.420	0.584	0.507	0.505
	VU	0.489	0.615	0.495	0.537
	P(True)	0.427	0.584	0.478	0.502
	LexSim	0.628	0.684	0.544	0.563
	DegMat	0.543	0.682	**0.548**	0.549
	LUQ	0.649	0.655	0.523	0.595
	KLE	0.354	0.649	0.444	0.449
	Inv-Entropy (Ours)	**0.747**	**0.742**	0.508	**0.614**
Brier (↓)	Semantic Entropy	0.227	0.230	0.221	0.242
	VU	0.209	0.218	0.216	0.223
	P(True)	0.225	0.232	0.230	0.244
	LexSim	0.169	0.199	0.207	0.225
	DegMat	0.187	0.181	0.213	0.229
	LUQ	0.164	0.209	**0.179**	0.208
	KLE	0.228	0.203	0.225	0.244
	Inv-Entropy (Ours)	**0.125**	**0.175**	0.193	**0.201**

https://doi.org/10.52202/085713-3598

Table 7 shows how the experimental results vary with the number of bootstrapping iterations S. We observe that increasing S initially enhances the performance of our method, Inv-Entropy. However, beyond a certain threshold, further increases yield diminishing or no significant returns. Therefore, using more than 10 iterations appears to provide limited additional benefit.

Table 7: Comparison of Inv-Entropy performance across all the 5 datasets with varying numbers of bootstrapping iterations S. We use GPT-3.5-Turbo with ChatGPT-based paraphrasing and DeBERTa-v2-xlarge-MNLI embedding function.

Dataset	**Metric**	S=1	S=5	S=10	S=30	S=50	S=100
TriviaQA	AUROC	0.743±0.061	0.781±0.057	0.785±0.056	0.788±0.054	0.780±0.054	0.788±0.054
	PRR	0.840±0.054	0.881±0.043	0.882±0.043	0.885±0.044	0.882±0.044	0.885±0.044
	Brier	0.138±0.021	0.127±0.020	0.125±0.020	0.128±0.020	0.131±0.021	0.128±0.020
SciQ	AUROC	0.724±0.052	0.733±0.050	0.740±0.049	0.740±0.050	0.739±0.050	0.743±0.049
	PRR	0.821±0.052	0.840±0.046	0.844±0.045	0.843±0.042	0.842±0.046	0.845±0.046
	Brier	0.163±0.016	0.160±0.015	0.159±0.015	0.157±0.018	0.160±0.015	0.159±0.015
NQ	AUROC	0.659±0.063	0.686±0.060	0.699±0.057	0.703±0.060	0.702±0.058	0.705±0.059
	PRR	0.709±0.068	0.730±0.071	0.760±0.065	0.764±0.064	0.764±0.063	0.766±0.064
	Brier	0.199±0.019	0.194±0.020	0.184±0.019	0.182±0.020	0.183±0.020	0.182±0.021
MMLU	AUROC	0.604±0.059	0.762±0.042	0.777±0.042	0.780±0.041	0.790±0.041	0.789±0.040
	PRR	0.743±0.064	0.863±0.046	0.897±0.031	0.898±0.030	0.905±0.028	0.902±0.030
	Brier	0.188±0.021	0.152±0.017	0.148±0.017	0.147±0.017	0.142±0.017	0.143±0.017
GSM8K	AUROC	0.674±0.054	0.684±0.056	0.685±0.054	0.695±0.051	0.690±0.055	0.695±0.055
	PRR	0.487±0.090	0.530±0.096	0.527±0.095	0.521±0.094	0.520±0.094	0.527±0.095
	Brier	0.159±0.021	0.151±0.022	0.150±0.021	0.152±0.020	0.152±0.020	0.146±0.021

Table 8 shows the variation of several proposed UQ measures (Inv-Entropy, NI-Entropy, WD-px-py, MAX-py-x) as the number of replications r increases. Performance improves consistently with larger r, as additional replications help better capture aleatoric uncertainty. However, this improvement comes at the expense of increased computational cost.

Table 8: Comparison of UQ methods on the TriviaQA and SciQ datasets with varying numbers of replications r. We use GPT-3.5-Turbo with ChatGPT-based paraphrasing and SBERT-large (all-mpnet-base-v2) embedding function.

Dataset		**TriviaQA**			**SciQ**		
	Metric	$r=2$	$r=4$	$r=6$	$r=2$	$r=4$	$r=6$
Inv-Entropy	AUROC	0.812±0.044	0.815±0.044	0.820±0.044	0.794±0.044	0.801±0.043	0.806±0.042
	PRR	0.916±0.028	0.915±0.029	0.915±0.029	0.888±0.038	0.888±0.039	0.892±0.038
	Brier	0.128±0.020	0.121±0.020	0.123±0.020	0.143±0.019	0.140±0.019	0.136±0.019
NI-Entropy	AUROC	0.702±0.072	0.631±0.081	0.617±0.083	0.729±0.056	0.731±0.058	0.746±0.055
	PRR	0.813±0.054	0.765±0.060	0.751±0.059	0.805±0.055	0.805±0.056	0.809±0.054
	Brier	0.133±0.024	0.152±0.023	0.150±0.024	0.147±0.021	0.146±0.021	0.144±0.021
WD-px-py	AUROC	0.762±0.058	0.771±0.058	0.772±0.057	0.684±0.052	0.688±0.058	0.685±0.057
	PRR	0.866±0.046	0.875±0.044	0.876±0.044	0.825±0.050	0.818±0.054	0.819±0.051
	Brier	0.134±0.021	0.131±0.021	0.130±0.021	0.173±0.017	0.169±0.020	0.172±0.019
MAX-py-x	AUROC	0.782±0.047	0.794±0.045	0.804±0.044	0.754±0.049	0.760±0.051	0.757±0.049
	PRR	0.904±0.028	0.910±0.027	0.915±0.026	0.864±0.047	0.862±0.049	0.861±0.048
	Brier	0.134±0.019	0.130±0.019	0.124±0.019	0.154±0.018	0.155±0.019	0.154±0.017

Table 9 demonstrates the robustness of our framework across different embedding functions. On all three datasets (TriviaQA, SciQ, and MMLU), Inv-Entropy performs strongly with SBERT-small, SBERT-large, and DeBERTa. This consistency allows practitioners to select an encoder based on available computational resources or domain-specific requirements without compromising the quality of uncertainty estimation.

Table 9: Comparison of AUROC, PRR, and Brier scores across the TriviaQA, SciQ, and MMLU datasets using different embedding functions. We use GPT-3.5-Turbo and ChatGPT-based paraphrasing.

Dataset	Metric	Embedding	Inv-Entropy	NI-Entropy	WD-px-py	MAX-py-x
TriviaQA	AUROC	SBERT-small	0.792±0.051	0.666±0.072	0.833±0.050	0.835±0.046
		SBERT-large	0.772±0.057	0.617±0.083	0.804±0.044	0.816±0.044
		DeBERTa	0.788±0.054	0.786±0.057	0.518±0.060	0.723±0.054
	PRR	SBERT-small	0.899±0.037	0.803±0.053	0.920±0.029	0.920±0.029
		SBERT-large	0.876±0.044	0.751±0.059	0.915±0.026	0.915±0.030
		DeBERTa	0.885±0.044	0.883±0.043	0.763±0.051	0.875±0.038
	Brier	SBERT-small	0.127±0.021	0.151±0.022	0.106±0.021	0.114±0.021
		SBERT-large	0.130±0.021	0.150±0.024	0.124±0.019	0.124±0.020
		DeBERTa	0.128±0.020	0.124±0.020	0.184±0.019	0.148±0.019
SciQ	AUROC	SBERT-small	0.774±0.049	0.750±0.054	0.655±0.058	0.742±0.050
		SBERT-large	0.806±0.042	0.746±0.055	0.685±0.057	0.757±0.049
		DeBERTa	0.740±0.050	0.681±0.056	0.303±0.060	0.674±0.054
	PRR	SBERT-small	0.874±0.044	0.820±0.059	0.796±0.054	0.852±0.049
		SBERT-large	0.892±0.038	0.809±0.054	0.819±0.051	0.861±0.048
		DeBERTa	0.843±0.042	0.781±0.053	0.587±0.056	0.821±0.048
	Brier	SBERT-small	0.147±0.021	0.150±0.021	0.181±0.019	0.160±0.018
		SBERT-large	0.136±0.019	0.144±0.021	0.172±0.019	0.154±0.017
		DeBERTa	0.157±0.018	0.164±0.017	0.212±0.016	0.177±0.017
MMLU	AUROC	SBERT-small	0.689±0.060	0.576±0.063	0.723±0.049	0.704±0.056
		SBERT-large	0.634±0.058	0.532±0.057	0.670±0.055	0.676±0.057
		DeBERTa	0.780±0.041	0.710±0.052	0.573±0.061	0.585±0.059
	PRR	SBERT-small	0.812±0.064	0.719±0.069	0.869±0.035	0.832±0.057
		SBERT-large	0.790±0.059	0.695±0.068	0.834±0.049	0.820±0.056
		DeBERTa	0.898±0.030	0.823±0.055	0.777±0.054	0.749±0.062
	Brier	SBERT-small	0.170±0.019	0.192±0.020	0.163±0.018	0.165±0.019
		SBERT-large	0.185±0.020	0.200±0.020	0.177±0.019	0.173±0.020
		DeBERTa	0.147±0.017	0.168±0.021	0.188±0.018	0.189±0.020

https://doi.org/10.52202/085713-3598

Table 10 and Table 11 present how various uncertainty measures respond to temperature changes on the TriviaQA and SciQ datasets, respectively. Our probabilistic methods consistently outperform baseline models across all temperature settings, highlighting their robustness to decoding variability.

Table 10: Comparison of AUROC, PRR, and Brier scores on the TriviaQA dataset under varying temperatures ($\mathbb{T} = 0.3,\ 0.7,\ 1.0,\ 1.4$). We use GPT-3.5-Turbo with ChatGPT-based paraphrasing and the SBERT-small (paraphrase-MiniLM-L6-v2) embedding function. **Bold** and underline indicate the best and second-best performers, respectively.

Metric	Method	Temperature			
		$\mathbb{T}$=0.3	$\mathbb{T}$=0.7	$\mathbb{T}$=1.0	$\mathbb{T}$=1.4
AUROC(↑)	Semantic Entropy	0.579±0.044	0.679±0.052	0.700±0.062	0.732±0.051
	VU	0.695±0.060	0.625±0.058	0.629±0.063	0.578±0.067
	P(True)	0.604±0.050	0.592±0.046	0.571±0.043	0.583±0.038
	LexSim	0.649±0.055	0.715±0.051	0.775±0.055	0.792±0.051
	DegMat	0.734±0.056	0.688±0.067	0.794±0.049	0.730±0.059
	LUQ	0.637±0.067	0.712±0.059	0.748±0.063	0.817±0.044
	KLE	0.333±0.054	0.327±0.059	0.221±0.056	0.155±0.042
	Inv-Entropy	**0.870**±0.044	0.795±0.051	0.827±0.043	0.810±0.042
	NI-Entropy	0.762±0.063	0.666±0.073	0.634±0.083	0.673±0.055
	WD-px-py	0.829±0.044	0.827±0.051	0.816±0.054	0.829±0.039
	MAX-py-x	0.854±0.043	**0.832**±0.046	**0.856**±0.040	**0.846**±0.041
PRR(↑)	Semantic Entropy	0.787±0.044	0.803±0.046	0.834±0.043	0.811±0.045
	VU	0.836±0.044	0.791±0.041	0.798±0.053	0.727±0.051
	P(True)	0.797±0.042	0.760±0.044	0.776±0.042	0.726±0.046
	LexSim	0.810±0.045	0.824±0.040	0.877±0.043	0.854±0.043
	DegMat	0.882±0.041	0.816±0.053	0.895±0.037	0.812±0.054
	LUQ	0.854±0.043	0.856±0.042	0.874±0.048	0.893±0.039
	KLE	0.704±0.048	0.646±0.050	0.623±0.051	0.516±0.049
	Inv-Entropy	**0.939**±0.028	0.900±0.037	0.936±0.022	0.903±0.037
	NI-Entropy	0.862±0.043	0.799±0.054	0.777±0.060	0.765±0.055
	WD-px-py	0.938±0.021	0.918±0.029	0.912±0.043	**0.923**±0.025
	MAX-py-x	0.932±0.029	**0.920**±0.030	**0.946**±0.021	0.913±0.039
Brier(↓)	Semantic Entropy	0.166±0.023	0.150±0.026	0.141±0.023	0.156±0.023
	VU	0.160±0.022	0.178±0.017	0.175±0.023	0.203±0.016
	P(True)	0.172±0.022	0.188±0.020	0.179±0.021	0.198±0.020
	LexSim	0.151±0.024	0.146±0.021	0.128±0.024	0.127±0.021
	DegMat	0.140±0.021	0.149±0.022	0.115±0.018	0.145±0.023
	LUQ	0.148±0.020	0.142±0.021	0.121±0.023	0.121±0.018
	KLE	0.188±0.021	0.199±0.020	0.192±0.021	0.218±0.014
	Inv-Entropy	**0.085**±0.019	0.127±0.021	0.117±0.019	0.132±0.017
	NI-Entropy	0.104±0.021	0.151±0.022	0.142±0.024	0.164±0.018
	WD-px-py	0.115±0.018	**0.102**±0.022	0.117±0.021	0.121±0.017
	MAX-py-x	0.103±0.019	0.116±0.021	**0.108**±0.019	**0.114**±0.016

Table 11: Comparison of AUROC, PRR, and Brier scores on the SciQ dataset under varying temperatures ($\mathbb{T} = 0.3,\ 0.7,\ 1.0,\ 1.4$). We use GPT-3.5-Turbo with ChatGPT-based paraphrasing and the SBERT-large (all-mpnet-base-v2) embedding function. **Bold** and underline indicate the best and second-best performers, respectively.

Metric	Method	Temperature			
		$\mathbb{T}$=0.3	$\mathbb{T}$=0.7	$\mathbb{T}$=1.0	$\mathbb{T}$=1.4
AUROC(↑)	Semantic Entropy	0.548±0.040	0.679±0.045	0.699±0.046	0.773±0.043
	VU	0.625±0.061	0.480±0.060	0.625±0.050	0.607±0.065
	P(True)	0.566±0.037	0.522±0.026	0.531±0.026	0.545±0.026
	LexSim	0.552±0.041	0.681±0.046	0.713±0.049	0.770±0.051
	DegMat	0.569±0.065	0.672±0.059	0.730±0.055	**0.816**±0.040
	LUQ	0.565±0.062	0.740±0.050	0.650±0.060	0.772±0.051
	KLE	0.386±0.054	0.341±0.056	0.277±0.059	0.184±0.051
	Inv-Entropy	**0.767**±0.047	**0.800**±0.044	0.754±0.046	0.795±0.046
	NI-Entropy	0.682±0.062	0.755±0.055	0.596±0.065	0.649±0.063
	WD-px-py	0.711±0.053	0.689±0.056	0.795±0.047	0.771±0.045
	MAX-py-x	0.766±0.048	0.760±0.048	**0.796**±0.046	0.793±0.044
PRR(↑)	Semantic Entropy	0.705±0.045	0.763±0.044	0.780±0.043	0.798±0.046
	VU	0.739±0.062	0.677±0.053	0.736±0.053	0.678±0.061
	P(True)	0.713±0.044	0.679±0.050	0.687±0.045	0.634±0.045
	LexSim	0.701±0.052	0.770±0.051	0.794±0.047	0.796±0.056
	DegMat	0.739±0.060	0.802±0.046	0.834±0.046	0.864±0.040
	LUQ	0.709±0.058	0.843±0.042	0.734±0.061	0.813±0.061
	KLE	0.632±0.048	0.592±0.059	0.554±0.052	0.479±0.051
	Inv-Entropy	**0.900**±0.029	**0.888**±0.039	0.864±0.044	**0.870**±0.035
	NI-Entropy	0.814±0.047	0.820±0.055	0.707±0.059	0.693±0.064
	WD-px-py	0.858±0.040	0.820±0.052	0.868±0.050	0.851±0.037
	MAX-py-x	0.893±0.029	0.862±0.048	**0.887**±0.037	0.867±0.034
Brier(↓)	Semantic Entropy	0.203±0.018	0.173±0.020	0.171±0.021	0.159±0.022
	VU	0.191±0.022	0.196±0.017	0.202±0.018	0.212±0.016
	P(True)	0.204±0.017	0.215±0.017	0.212±0.016	0.225±0.013
	LexSim	0.207±0.020	0.179±0.020	0.169±0.022	0.156±0.019
	DegMat	0.198±0.017	0.164±0.018	0.149±0.020	**0.135**±0.020
	LUQ	0.195±0.020	0.157±0.018	0.173±0.021	0.159±0.020
	KLE	0.216±0.016	0.218±0.016	0.219±0.016	0.233±0.011
	Inv-Entropy	**0.151**±0.018	**0.139**±0.020	0.153±0.020	0.146±0.020
	NI-Entropy	0.164±0.021	0.140±0.022	0.188±0.020	0.186±0.020
	WD-px-py	0.167±0.018	0.175±0.019	**0.140**±0.020	0.156±0.015
	MAX-py-x	0.151±0.019	0.155±0.018	0.143±0.020	0.153±0.018

https://doi.org/10.52202/085713-3598

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims made in the abstract and introduction reflect the results in the main paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss limitations.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The assumptions and theorems are all included in the main paper and Appendix.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in Appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: All settings are includes in the paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

https://doi.org/10.52202/085713-3598

Answer: [Yes]

Justification: Data is publically available and we provide code.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Experimental settings are provided in the paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in Appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We provide confidence intervals for our results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Thiis is discussed in the paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have abided by NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: They are no expected social impact for our work.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to

https://doi.org/10.52202/085713-3598

generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: No such risks in this paper.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All of the existing assets are welly cited and credited.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: Details are documented in the main paper.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: No such issue involved in this paper.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: No such issue involved in this work.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

https://doi.org/10.52202/085713-3598

Justification: LLMs are not used to generate important and original components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.