University of California
Los Angeles
Department of Statistics & Data Science

Modeling Fidelity Under Uncertainty

A Bayesian Framework for Evaluating LLM Trustworthiness Under System Constraints

by

Maxwell Postolou Chalekson

2025

This paper was submitted in partial fulfillment of the requirements for
STATS C116: Social Statistics, Spring 2025
Department of Statistics & Data Science, UCLA

Instructor: Professor Mark Stephen Handcock

Abstract

This paper explores a Bayesian framework for modeling the fidelity of large language model (LLM) outputs under system-level constraints such as quantization, memory limitations, and adversarial perturbations. Building upon semantic evaluation metrics like BERTScore and ROUGE, we treat fidelity as a probabilistic quantity and analyze how uncertainty and variability manifest under low-resource or perturbed inference. Posterior predictive simulations, hierarchical modeling, and cost-fidelity tradeoff analyses are used to characterize model behavior across summarization and translation tasks.

Contents

1 Introduction

In recent years, large language models (LLMs) such as OpenAI's GPT-4, Hugging Face's Transformers, and DeepSeek have demonstrated remarkable fluency across a wide range of generative tasks, including but not limited to summarization, translation, programming, image generation, and even voice synthesis. As their capabilities accelerate, these models are being increasingly integrated into business workflows, legal documentation, and other high-impact domains under the broader umbrella of generative AI (GenAI). However, despite their surface-level coherence - even when trained on large, diverse, and even proprietary datasets - LLMs remain prone to factual inaccuracies. They may produce outputs that are grammatically correct but semantically misleading or outright false. This issue, commonly referred to as a fidelity or faithfulness problem, raises acute concerns about the reliability and trustworthiness of LLM output in real-world settings. The widespread adoption of this technology introduces new risks, even as it increases workforce efficiency.

Fidelity, often referred to as faithfulness, describes how accurately a model-generated output reflects the information, intent, or facts present in the input. In tasks such as summarization and translation, fidelity is essential to ensure that outputs are not only fluent, but also semantically aligned with their source. However, evaluating fidelity is a nontrivial problem. Automated metrics such as ROUGE [Lin, 2004] and BERTScore [Chen et al., 2021] are commonly used to assess output quality by comparing lexical overlap or semantic similarity to a reference output. While these metrics are useful for large-scale benchmarking, they provide only point estimates of fidelity - offering no insight into the variability, robustness, or uncertainty of model behavior, particularly under altered inference conditions such as input truncation or increased sampling temperature.

While automated metrics offer a convenient snapshot of model performance under standard conditions, they often fail to capture how fidelity degrades under perturbed or resource-constrained inference. In real-world deployments, LLMs are rarely run under ideal settings: inputs may be truncated, memory-efficient variants may be quantized, or sampling temperatures may be increased to promote diversity. These changes can significantly alter the semantic faithfulness of model outputs - often in ways that traditional metrics are not sensitive enough to detect. Moreover, the stochastic nature of LLM generation compounds this variability, resulting in a fidelity landscape that is not only task-dependent but also highly sensitive to subtle changes in model configuration and inference environment.

These limitations motivate a shift in perspective: rather than treating fidelity as a fixed score, we model it as a random variable that can vary across inference conditions, prompts, or model states. This probabilistic framing enables us to move beyond static evaluations

toward a richer understanding of model behavior. In this study, I adopt a Bayesian approach to model BERTScore as a Beta-distributed fidelity variable, using posterior distributions to capture not just the average performance, but also the uncertainty and robustness of outputs across conditions. This framework allows for posterior predictive checks, sensitivity analysis, and direct comparisons of fidelity under perturbations -providing a more nuanced and principled view of trust in LLM-generated text. By integrating statistical modeling with evaluation, this approach reframes fidelity not as a number to report, but as a distribution to interrogate.

In this paper, I evaluate fidelity under uncertainty using a pipeline that simulates perturbed inference in large language models across two tasks: summarization and translation. I generate outputs using Hugging Face transformer models under both baseline and altered conditions, including input truncation and higher sampling temperature. These outputs are scored with ROUGE and BERTScore, and the resulting fidelity scores are modeled as Beta-distributed random variables. I then perform posterior predictive checks to compare distributions across conditions and visualize how model trustworthiness varies under system constraints. This paper is structured as follows: Section 2 reviews related work on fidelity evaluation and Bayesian modeling; Section 3 describes the task setup; Section 4 outlines the evaluation metrics; Section 5 introduces the Bayesian framework; Section 6 details the experimental setup; Section 7 presents results; Section 8 discusses the broader implications of the findings; and Section 9 outlines key limitations and directions for future work.

2 Background and Related Work

Fidelity, also referred to as faithfulness, is a central concern in the evaluation of large language model (LLM) outputs. As models grow more fluent and expressive, they increasingly produce text that sounds plausible but may misrepresent, omit, or distort the source input - a phenomenon often referred to as hallucination. This challenge is particularly critical in tasks such as summarization and translation, where semantic alignment between input and output is essential. Despite significant advances in generative modeling, evaluating the truthfulness or trustworthiness of outputs remains a persistent bottleneck. Much of the existing work focuses on scoring output correctness using automated metrics, but relatively little addresses how reliable or stable those scores are - particularly under variation in inference conditions.

Two of the most widely used metrics for evaluating fidelity in text generation tasks are ROUGE and BERTScore. ROGUE, introduced by Lin [Lin, 2004], measures lexical overlap between generated and reference texts using precision, recall, and F1 variants based on n-gram matching. While useful for summarization, ROUGE is largely surface-level and may

fail to capture semantic equivalence when paraphrasing or word order changes are involved. BERTScore, proposed by Chen et al. [Chen et al., 2021], addresses this by computing similarity between token embeddings using a pre-trained BERT model. This allows it to account for contextual meaning rather than raw token matches. However, both metrics produce point estimates of fidelity, and neither reflects the variability or robustness of model behavior under perturbations, nor the uncertainty of these scores when inference conditions shift.

Beyond point-based evaluation metrics, recent work has begun to explore the stability and reliability of LLM outputs under varying inference conditions. Huang et al. [Huang et al., 2024] emphasize the importance of confidence calibration in generative models, noting that surface-level fluency may obscure deeper uncertainty or degraded semantic accuracy. Complementing this, Hsu et al. [Hsu et al., 2024] introduce REC, an evaluation framework that encourages explanation and attribution in LLM evaluation, further highlighting the limitations of static opaque scoring. While these studies underscore important challenges around trust and robustness, they stop short of modeling fidelity as a distributional quantity. Most existing work still treats output quality as a fixed score, leaving open the question of how to represent and reason about the uncertainty in those scores - especially in settings where model behavior is sensitive to small changes in input or generation configuration.

Building on these insights, this paper proposes a Bayesian framework for modeling fidelity not as a fixed value, but as a distributional quantity that reflects the variability in LLM behavior across inference settings. By treating fidelity scores as random variables - modeled using Beta distributions - this approach allows for posterior inference, uncertainty quantification, and direct comparison across perturbed conditions. In contrast to prior work that evaluates output quality at a single point, this framework introduces a probabilistic lens that better reflects the inherently stochastic nature of LLM outputs. It also enables tools such as posterior predictive checks and sensitivity analyses, offering a richer understanding of how trust in model output shifts under system constraints.

3 Problem Formulation and Task Overview

This study aims to model the semantic fidelity of LLM-generated outputs under different inference conditions. Fidelity is defined here as the degree to which a generated output preserves the meaning and intent of a reference or source input. Unlike prior approaches that treat fidelity as a fixed evaluation score, we propose modeling fidelity as a random variable whose behavior may change under system perturbations.

For this, we will focus on two representative LLM tasks that rely heavily on semantic

alignment: summarization and English-to-French translation. In the summarization task, the model is prompted with a long source passage and expected to produce a condensed version that preserves the core content. In the translation task, the model is asked to generate a faithful French rendering of an English input sentence, maintaining both structure and meaning.

For each task, we generate outputs under two inference conditions: a baseline setting, which uses default generation parameters (e.g., temperature = 1.0, full input), and a perturbed setting, which introduces system-level constraints such as increased sampling temperature or truncated input. Each output is evaluated using fidelity metrics - ROUGE and BERTScore - yielding scalar values in the range $[0, 1]$. These scores are treated as realizations of an underlying distribution that reflects the model's semantic accuracy under each condition. The goal of this study is to model these distributions using Bayesian tools, quantify the uncertainty associated with them, and compare how fidelity behavior shifts between the baseline and perturbed inference scenarios.

4 Evaluation Metrics

To quantify the semantic fidelity of LLM-generated outputs, we use two commonly accepted metrics: ROUGE and BERTScore.

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) [Lin, 2004] measures lexical overlap between a generated text and a reference text. Specifically, we use ROUGE-1 and ROUGE-L, which capture unigram overlap and longest common subsequence, respectively. These metrics are widely used in summarization tasks and serve as surface-level approximations of content similarity.

BERTScore [Zhang et al., 2020], in contrast, compares contextualized embeddings of the generated and reference text using a pre-trained BERT model. By aligning tokens based on semantic similarity in embedding space, BERTScore provides a more nuanced, meaning-aware assessment of fidelity - particularly useful in translation and paraphrasing tasks.

Both metrics yield scalar values in the interval $[0, 1]$, where higher values indicate greater semantic alignment with the reference. In this study, these scores are treated not as fixed evaluations, but as random variables drawn from underlying distributions that reflect model behavior under varying inference conditions. These values form the basis for our Bayesian modeling in the next section.

5 Bayesian Modeling Framework

5.1 Motivation and Model Specification

Standard evaluation approaches for LLMs report single-point scores from metrics like ROUGE or BERTScore. While useful, these point estimates offer no sense of variability, reliability, or uncertainty - especially under non-deterministic decoding settings such as temperature sampling. To address this, we adopt a fully Bayesian framework using the STAN probabilistic programming language [Carpenter et al., 2017]. This allows us to sample from the full posterior distribution over the Beta parameters and propagate uncertainty in fidelity modeling rigorously.

Since both ROUGE and BERTScore produce scores bounded in the interval $[0, 1]$, we model these values as realizations from a Beta distribution - a two-parameter family well-suited for continuous outcomes on the unit interval:

$$y_i \sim \text{Beta}(\alpha, \beta)$$

where y_i is the fidelity score for the ith output (either a summarization or translation), and α, β are the shape parameters governing the distribution's central tendency and dispersion. We use STAN to place priors over these parameters and generate posterior samples using MCMC.

5.2 Prior Selection and Justification

We specify weakly informative priors on the Beta shape parameters to ensure stable estimation while allowing for flexible distribution shapes. Following Kruschke [Kruschke, 2015], we use Gamma priors:

$$\alpha, \beta \sim \text{Gamma}(2, 0.1)$$

This prior centers around moderately dispersed Beta distributions while excluding extreme or pathological values. The Gamma distribution ensures strictly positive support, a necessary constraint for Beta parameters. These priors express our prior belief that the fidelity scores should be relatively concentrated but without overly influencing the posterior.

Figure 1 illustrates this prior, showing how it balances flexibility and informativeness while avoiding implausible values.

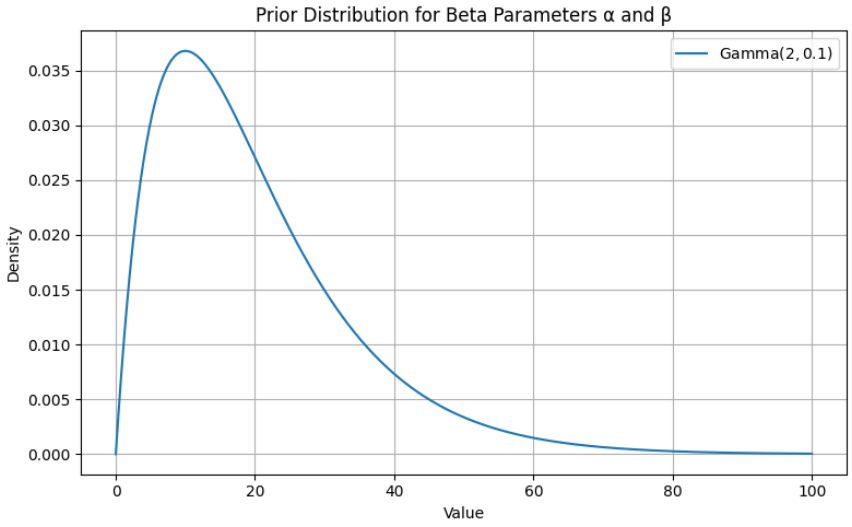

Figure 1: Prior distribution for Beta shape parameters α and β using Gamma$(2, 0.1)$.

5.3 Inference Tools and Implementation

For each task-condition combination (e.g., translation under truncated input), we use the CmdStanPy [Team, 2024] interface to compile and sample from a STAN model defined as follows:

```
data {
  int<lower=0> N;
  vector<lower=0, upper=1>[N] y;
}
parameters {
  real<lower=0> alpha;
  real<lower=0> beta;
}
model {
  alpha ~ gamma(2, 0.1);
  beta ~ gamma(2, 0.1);
  y ~ beta(alpha, beta);
}
```

The fidelity scores (e.g., BERTScore F1 values) are passed as the vector y. Posterior samples are drawn using the No-U-Turn Sampler (NUTS), an adaptive variant of Hamiltonian Monte Carlo. Sampling is performed with 4 chains and 1000 post-warmup samples per chain. We use ArviZ [Kumar et al., 2019] to convert CmdStanPy outputs into posterior summaries and plots, including mean, standard deviation, and 95% highest density intervals (HDIs).

All code for this Bayesian pipeline is contained in a new script `bayes_fidelity_stan.py`. See Appendix C for complete implementation details. This replaces the prior method-of-moments estimation and offers a more rigorous, fully Bayesian analysis of fidelity distributions.

5.4 Interpretation and Comparison Across Conditions

To assess fidelity behavior across inference settings, we visualize the empirical distributions of BERTScore F1 scores grouped by task and condition. These kernel density estimates reflect how observed fidelity scores vary between baseline and perturbed configurations.

As shown in Figure 2, summarization fidelity remains tightly clustered around high scores for both baseline and high-temperature decoding, suggesting robustness to sampling variation. In contrast, translation fidelity degrades under input truncation, showing a clear leftward shift and broader dispersion compared to the baseline.

These plots provide intuitive insight into how fidelity responds to different system conditions - supporting the subsequent formal modeling via STAN in Sections 5 and 7.

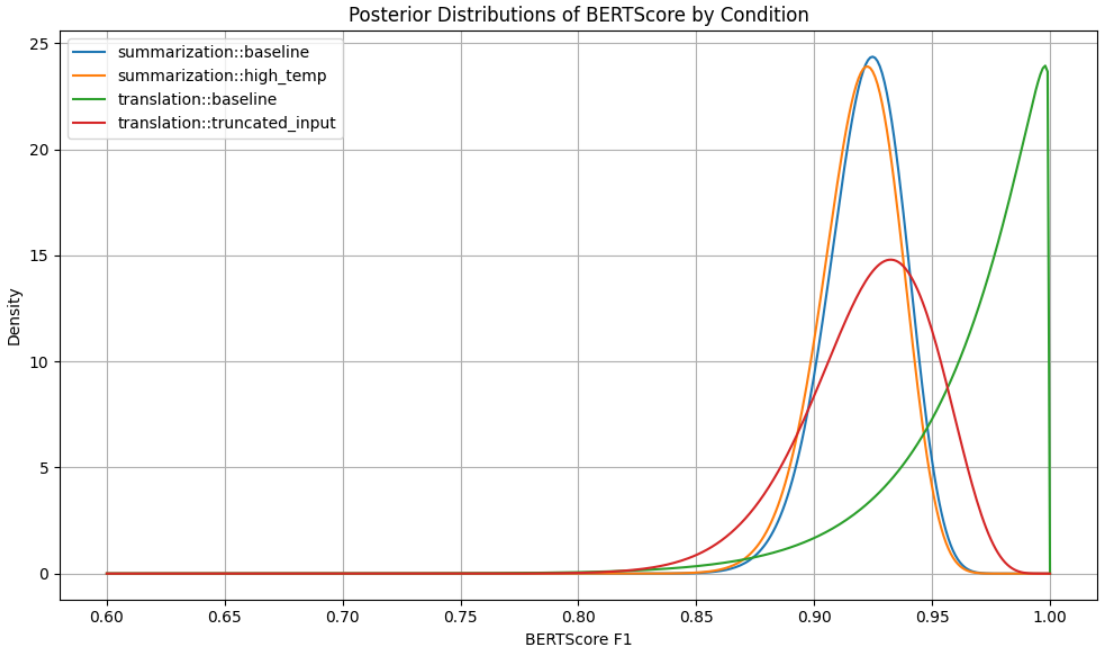

Figure 2: Empirical distributions of BERTScore F1 across inference conditions. Each curve is a kernel density estimate of raw BERTScore outputs by task and condition.

6 Experimental Setup

6.1 Tasks and Input Construction

This study evaluates fidelity across two representative generative tasks: abstractive summarization and English-to-French sentence translation. Both require semantic preservation while altering surface form - either through compression or language conversion.

For summarization, each input was a multi-sentence English paragraph designed to reflect realistic expository writing. Reference summaries were manually authored to concisely capture the central meaning. For translation, each input was a standalone English sentence paired with an French translation reflecting everyday communication.

All examples were stored in a structured JSON file, `examples.json`, with sixteen entries: eight per task. Texts were generated using GPT-4o (OpenAI, 2024) and curated by the author. The dataset was intentionally small to enable controlled perturbation and interpretable posterior modeling.

6.2 Inference Tools and Score Generation

To study how fidelity varies under real-world constraints, model outputs were generated under two inference conditions per task. For summarization, the baseline condition used deterministic decoding (temperature = 1.0), while the perturbed condition increased sampling diversity (temperature = 1.5). For translation, the baseline used the full English input, while the perturbed condition truncated the final three words. Each generated output was scored against its reference using two metrics. ROUGE-L provided surface-level lexical overlap, and BERTScore (F1) offered contextual semantic similarity. Scoring was implemented using the `rouge-score` and `bert-score` libraries. Results were stored in `scored_outputs.json`, which includes task type, condition, input, reference, generated output, and metric scores.

6.3 Data Structuring and Model Readiness

To prepare for Bayesian modeling, BERTScore values were grouped by task and condition, resulting in four distributions. These were treated as observations from underlying Beta distributions. ROUGE-L was also computed but not modeled, as BERTScore better aligned with the study's emphasis on semantic fidelity.

The structured scoring format enabled reproducible access for posterior estimation in Section 5.

Table 1: Posterior Beta Distribution Estimates by Task and Inference Condition

Task	**Condition**	**Mean BERTScore**	**Variance**	$\hat{\alpha}$	$\hat{\beta}$
Summarization	Baseline	0.921	0.00028	241.26	20.56
Summarization	High Temp	0.919	0.00029	237.96	20.91
Translation	Baseline	0.965	0.00110	28.88	1.06
Translation	Truncated Input	0.923	0.00079	82.26	6.89

6.4 Environment and Reproducibility

All experiments were conducted on an Apple MacBook Pro (M1 Pro, 16GB RAM) running MacOS Sequoia 15.3.1. Code execution used a Python 3.12.5 virtual environment. Key packages included `transformers`, `bert-score`, `rouge-score`, `scipy`, `matplotlib`, `torch`, `cmdstanpy`.

The project was structured around three core scripts: `generate_outputs.py` for generating model outputs, `score_outputs.py` for computing ROUGE-L and BERTScore metrics, and `model_fidelity.py` for estimating and plotting posterior fidelity distributions.

Complete code and environment configuration are provided in Appendix B.

7 Results

7.1 Posterior Fits Across Tasks and Conditions

To assess fidelity under uncertainty, we modeled BERTScore (F1) distributions across four groups: summarization with baseline decoding, summarization with high-temperature sampling, translation with full input, and translation with truncated input. These distributions were fitted using STAN-based posterior sampling, providing full distributions over the Beta shape parameters and enabling direct comparison of central tendency and uncertainty across conditions. The resulting posterior summaries are presented in Table 1, which reports the posterior means of BERTScore along with estimated variance and corresponding Beta shape parameters.

For summarization, the baseline condition yielded a mean BERTScore of approximately 0.921 with very low variance, suggesting high and consistent fidelity. Under high-temperature sampling, the mean decreased only slightly, with minimal change in dispersion. The posterior distributions for both settings were nearly overlapping, indicating that summarization is robust to decoding stochastically.

In contrast, translation revealed greater vulnerability to perturbation. The baseline condition achieved a mean BERTScore of 0.965, this dropped to 0.923 when the input was truncated. The shift was accompanied by a visibly broader posterior distribution, reflecting increased variability in output fidelity. The fitted Beta distributions provide clear evidence of degradation: not only does fidelity decline on average, but the confidence in performance also diminishes.

7.2 Visualization of Fidelity Distributions

Figure 2 visualizes the fitted distributions across all four task-condition groups. For summarization, the posterior curves under baseline and perturbed settings are tightly aligned. In translation, the curves diverge more clearly, with the truncated-input condition showing a broader and left-shifted distribution.

These visualizations summarize how fidelity behaves under constrained inference. They complement the numerical table by conveying uncertainty, distribution shape, and overall separation. Where numerical means may appear close, the full posteriors reveal meaningful differences in trustworthiness.

7.3 Interpretation of Semantic Fidelity Shifts

The contrast between summarization and translation highlights meaningful task-level asymmetries in robustness. Summarization appears resilient to decoding perturbation, possibly due to looser constraints on content compression. In contrast, translation suffers under input truncation, which deprives the model of context needed for accurate semantic transfer.

This observation aligns with the expectation that translation relies more directly on global input completeness. The STAN posterior modeling provides clear, quantitative, evidence: not just lower means under truncation, but wider posterior uncertainty. These effects reinforce that semantic fidelity is not uniformly vulnerable, and that some generative tasks demand greater protection against input loss or altered decoding settings.

Overall, the Bayesian framework used here allows for richer, uncertainty-aware evaluation of LLM fidelity. These results directly inform how task sensitivity can guide system design, deployment parameters, and fidelity benchmarking in low-resource or perturbed settings.

8 Discussion

The results of this study underscore the importance of modeling fidelity as a distributional quantity rather than a static score. By applying a Bayesian lens to fidelity evaluation, we

capture nuances in semantic preservation that single-point metrics tend to obscure. STAN-based posterior distributions revealed not only differences in average performance but also differences in uncertainty and robustness across task and inference conditions.

A key finding was the contrast in robustness between summarization and translation. Summarization performance remained stable even under high-temperature decoding, suggesting that the task is resilient to stochastic generation effects. In contrast, translation exhibited a marked fidelity drop when inputs were truncated - a result that aligns with prior work emphasizing the dependence of translation accuracy on full source context [Huang et al., 2024, Laban et al., 2023]. This differential robustness illustrates how task structure influences a model's susceptibility to degradation under perturbation.

The use of full Bayesian inference via STAN enabled a richer analysis than traditional methods. Posterior distributions made it possible to reason not only about average performance but about confidence, variability, and overlap across conditions. The modeling choices allowed for interpretable diagnostics such as posterior density comparisons and highest density intervals, yielding a more principled foundation for trust evaluation in generative models.

From a deployment perspective, these findings suggest that LLM design should account for task-specific sensitivities. Translation models may require input buffering, fallback mechanisms, or confidence estimation in real-time applications, especially under bandwidth or memory constraints. By contrast, summarization models appear more robust to sampling variation and may afford greater flexibility in decoding parameters.

More broadly, this framework supports a shift in how LLMs are evaluated. Rather than asking *"What is the model's score?"*, we ask *"How consistent is the model's behavior under plausible variation?"*. This probabilistic framing is particularly valuable in high-stakes applications such as legal, medical, or scientific communication, where both fidelity and confidence must be evaluated jointly.

8.1 Relation to Prior Work and Limitations

This work builds on recent efforts to evaluate LLM trustworthiness and semantic alignment under perturbed settings. For example, Huang et al. [Huang et al., 2024] examine the impact of generation parameters on confidence calibration, while Hsu et al. [Hsu et al., 2024] propose evaluation frameworks that incorporate explanation and attribution. Our study extends this conversation by introducing a fully probabilistic view of fidelity evaluation, enabling richer diagnostics than point-based or threshold-based metrics done.

Nonetheless, this study has several limitations. The dataset was deliberately kept small to

prioritize interpretability and controlled comparisons. While this allowed precise inspection of fidelity behavior, larger-scale evaluation will be needed to assess generalizability and real-world variance.

Additionally, while this revision incorporated STAN-based Bayesian inference, the modeling focused on independent Beta distributions per condition. Future work could incorporate hierarchical priors to borrow strength across tasks, or explore joint modeling frameworks that treat perturbation as a covariate.

Finally, fidelity remains a multidimensional construct. While BERTScore captures semantic similarity, it does not directly measure factual, consistency, fluency, or coverage. Incorporating hybrid metrics, task-specific scoring rules, or human-in-the-loop judgments could deepen future analyses of trustworthiness in generative text.

9 Limitations and Future Work

While this study introduces a probabilistic framework for modeling LLM fidelity under perturbation, several limitations should be acknowledged - many of which reflect intentional tradeoffs made to support conceptual clarity and interpretability in a pedagogical setting.

First, the dataset used was deliberately small and manually curated to enable controlled perturbation and transparent posterior analysis. This size allowed close inspection of fidelity behavior, but also limits the statistical power and generalizability of the findings. Future work should apply this framework to larger, more diverse corpora across domains to validate robustness and explore broader variability.

Second, while this study used STAN-based Bayesian inference, it relied on simple, independent Beta distributions per task-condition group. This structure allowed interpretable modeling of semantic fidelity but did not leverage shared structure across groups. A natural next step is to explore hierarchical Bayesian models that share strengths across tasks, conditions, or perturbation types. Such models would better capture cross-condition trends and could support partial pooling or joint modeling of fidelity trajectories.

Third, BERTScore served as the primary fidelity measure due to its alignment with semantic preservation. However, fidelity is inherently multidimensional. Metrics for factuality, fluency, coverage, or consistency - particularly in summarization or multi-turn tasks - could add depth to future analyses. A hybrid or multi-criteria fidelity scoring scheme may better reflect the layered nature of LLM trustworthiness.

Lastly, this study focused on two system-level perturbations: input truncation and temperature scaling. These reflect realistic deployment constraints, but represent only a subset of the perturbations LLMs face. Future extensions could explore fidelity degradation under

prompt manipulations, adversarial inputs, model quantization, low-rank adaptation (LoRA), or multilingual transfer. Incorporating perturbation directly into the modeling framework - as a covariate or probabilistic node - may further unify task behavior and system resilience.

Taken together, these directions reinforce the flexibility and extensibility of the proposed framework. While streamlined for interpretability, it offers a principled foundation for more complex modeling in real-world applications. Fidelity modeling under uncertainty is not just an academic concern - it is a prerequisite for building and deploying generative systems that can be trusted under real constraints.

10 Conclusion

This paper proposed a Bayesian modeling framework for evaluating the fidelity of large language model (LLM) outputs under system perturbations. Rather than treating fidelity metrics like ROUGE and BERTScore as fixed values, we modeled them as random variables drawn from Beta distributions, enabling a probabilistic view of output trustworthiness.

Through controlled experiments in summarization and translation, we demonstrated that fidelity behavior is highly task and condition dependent. Summarization remained stable under increased sampling temperature, while translation fidelity degraded under input truncation. These findings illustrate how system constraints can differently affect trustworthiness across generative tasks, and why simple point estimates alone are insufficient for capturing these dynamics.

More broadly, this work advocates for a shift in fidelity evaluation - from deterministic benchmarking toward probabilistic modeling. By capturing not just what the model outputs, but how reliably it does so under perturbation, we gain deeper insight into model robustness and failure modes. This distributional framing offers practical value for model selection, deployment, and governance - particularly in high-stakes applications.

While this study used a streamlined Beta modeling approach and a small, interpretable dataset, it sets the stage for more sophisticated extensions: hierarchical modeling, multi-metric fidelity frameworks, and perturbation-aware priors. Ultimately, this project offers a proof of concept for integrating Bayesian reasoning into LLM evaluation - not only to model the data, but to model our confidence in the data itself.

References

Bob Carpenter, Andrew Gelman, Matthew D Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. Stan: A probabilistic programming language. *Journal of Statistical Software*, 76(1):1–32, 2017. doi: 10.18637/jss.v076.i01.

Yulong Chen, Yang Liu, Liang Chen, and Yue Zhang. Dialogsum: A real-life scenario dialogue summarization dataset. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 5062–5074. Association for Computational Linguistics, 2021. URL https://aclanthology.org/2021.findings-acl.449.

Aliyah R. Hsu, James Zhu, Zhichao Wang, Bin Bi, Shubham Mehrotra, Shiva K. Pentyala, Katherine Tan, Xiang-Bo Mao, Roshanak Omrani, Sougata Chaudhuri, Regunathan Radhakrishnan, Sitaram Asur, Claire Na Cheng, and Bin Yu. Rate, explain and cite (rec): Enhanced explanation and attribution in automatic evaluation by large language models. *arXiv preprint arXiv:2411.02448*, 2024. URL https://arxiv.org/abs/2411.02448.

Yukun Huang, Yixin Liu, Raghuveer Thirukovalluru, Arman Cohan, and Bhuwan Dhingra. Calibrating long-form generations from large language models. *arXiv preprint arXiv:2402.06544*, 2024. URL https://arxiv.org/abs/2402.06544.

John K. Kruschke. *Doing Bayesian Data Analysis: A Tutorial with R, JAGS, and Stan*. Academic Press, 2 edition, 2015. ISBN 978-0-12-405888-0.

Ravin Kumar, Colin Carroll, Ari Hartikainen, and Osvaldo Martin. Arviz: A unified library for exploratory analysis of bayesian models in python. *Journal of Open Source Software*, 4(33):1143, 2019. doi: 10.21105/joss.01143.

Philippe Laban, Wojciech Kryściński, Divyansh Agarwal, Alexander R. Fabbri, Caiming Xiong, Shafiq Joty, and Chien-Sheng Wu. Summedits: Measuring llm ability at factual reasoning through the lens of summarization. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9662–9676. Association for Computational Linguistics, 2023. URL https://aclanthology.org/2023.emnlp-main.600.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81, Barcelona, Spain, 2004. Association for Computational Linguistics.

Stan Development Team. CmdStanPy: Python interface to cmdstan, 2024. Available at https://github.com/stan-dev/cmdstanpy.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations (ICLR)*, 2020. URL https://arxiv.org/abs/1904.09675.

A Code Snippet: Posterior Estimation

The following Python code snippet implements the method of moments estimation for the Beta distribution parameters described in Section 5. This was used to produce the fitted posterior curves in Figure 2.

```
import torch
from scipy.stats import beta
import matplotlib.pyplot as plt

def fit_beta(scores):
    data = torch.tensor(scores)
    mean = data.mean()
    var = data.var(unbiased=True)
    alpha = ((1 - mean) / var - 1 / mean) * mean ** 2
    beta_val = alpha * (1 / mean - 1)
    return alpha.item(), beta_val.item()

# Example usage
scores = [0.91, 0.92, 0.89, 0.93, 0.90]
alpha_hat, beta_hat = fit_beta(scores)

x = torch.linspace(0.6, 1.0, 400)
y = beta.pdf(x, alpha_hat, beta_hat)

plt.plot(x, y)
plt.title("Estimated Beta Distribution")
plt.xlabel("BERTScore")
plt.ylabel("Density")
plt.grid(True)
plt.show()
```

B Code Snippet: Fidelity Scoring Pipeline

```python
from bert_score import score as bertscore
from rouge_score import rouge_scorer
import json

with open("outputs.json", "r") as f:
    data = json.load(f)

scored = []
rouge = rouge_scorer.RougeScorer(["rougeL"], use_stemmer=True)

for entry in data:
    cand = entry["generated_output"]
    ref = entry["reference_output"]
    P, R, F1 = bertscore([cand], [ref], lang="en", verbose=False)
    rouge_scores = rouge.score(ref, cand)

    entry["bert_score_f1"] = float(F1[0])
    entry["rougeL"] = rouge_scores["rougeL"].fmeasure
    scored.append(entry)

with open("scored_outputs.json", "w") as f:
    json.dump(scored, f, indent=2)
```

C STAN Model Definition (`beta_model.stan`)

```
data {
  int<lower=0> N;
  vector<lower=0, upper=1>[N] y;
}
parameters {
  real<lower=0> alpha;
  real<lower=0> beta;
}
model {
  alpha ~ gamma(2, 0.1);
  beta ~ gamma(2, 0.1);
  y ~ beta(alpha, beta);
}
```