IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

REAL-TIME TOKEN-LEVEL HALLUCINATION DETECTION IN LARGE LANGUAGE MODELS DURING TEXT GENERATION

[1] Harini .V, [2] K Sudha B Adiga , [3] S. Jagruthi

1,2,3 Semester Student , Dept of Master of Computer Applications ,SJB Institute of Technology, BengaluruIndia.

Abstract:

Keywords: *Hallucination Detection, Large Language Models, Token-Level Analysis, Entropy-Based Detection, Real-Time NLP, Text Generation, GPT-2, TruthfulQA*

Large Language Models (LLMs) have demonstrated remarkable capabilities in natural language generation tasks, yet they frequently produce factually incorrect or logically inconsistent content, a phenomenon commonly referred to as 'hallucination.' Existing detection approaches predominantly operate in a post-generation setting, evaluating the complete output only after the full text has been generated. This paper proposes a novel real-time, token-level hallucination detection framework that monitors the internal probability distributions of an LLM at each decoding step during text generation. By tracking per-token entropy values and identifying anomalous uncertainty patterns, the proposed system flags high-risk tokens and segments before the hallucination is fully expressed. The framework is evaluated on GPT-2 using the TruthfulQA and HaluEval benchmark datasets, requiring only modest computational resources (standard GPU or CPU). Experimental results demonstrate that the approach achieves 82.4% precision and 79.1% recall in detecting hallucination-prone token sequences, with less than 3% computational overhead compared to standard generation. This work addresses a critical gap in existing literature and offers a lightweight, practical solution suitable for deployment in resource-constrained environments such as MCA-level research settings.

1. INTRODUCTION

In recent years, Large Language Models (LLMs) such as GPT-2, GPT-3, GPT-4, and LLaMA have transformed natural language processing by generating highly fluent and contextually relevant text. However, these models are well-known to produce content that is factually incorrect, logically inconsistent, or entirely fabricated—a problem referred to as 'hallucination.' The consequences of hallucination are severe and well-documented: incorrect medical diagnoses, fabricated legal citations, and misleading news content can cause real-world harm. Existing literature on hallucination detection broadly falls into two categories: post-generation evaluation and controlled generation approaches. Li et al. (2024) proposed HalluCheckNet, which integrates factual and semantic consistency modules to evaluate generated text after it has been fully produced. Similarly, Mittal et al. (2025) presented a unified validation framework that uses semantic entropy and Retrieval-Augmented Generation (RAG) to verify output after generation. While these approaches are effective, they share a fundamental limitation: they can only identify hallucinations after the model has already committed to an incorrect output. By that point, the hallucinated content has already been generated and may have been delivered to the user.

This paper addresses this gap by proposing a real-time, token-level hallucination detection framework. Instead of analyzing completed output, the proposed system monitors the token probability distribution at every decoding step during generation. When the model exhibits high uncertainty—measured via Shannon entropy of the token probability distribution—the system flags that step as a potential hallucination point. This enables early-stage intervention, significantly reducing the chance of hallucinated content reaching the end user.

The key contributions of this paper are:

- A real-time hallucination detection mechanism that operates token-by-token during LLM decoding, without requiring post-generation processing.
- An entropy-based uncertainty scoring method that requires no additional training or external knowledge bases.
- Empirical evaluation on TruthfulQA and HaluEval benchmarks using GPT-2, demonstrating competitive performance with minimal computational overhead.

- A clear comparison against two state-of-the-art post-generation methods, establishing the novelty and advantage of the proposed approach.

2. BACKGROUND AND RELATED WORK

2.1 Hallucination in Language Models

Hallucination in NLP refers to the generation of text that is fluent and confident but factually incorrect or unsupported by any input context. Ji et al. (2023) provide a comprehensive survey identifying two primary categories: factuality hallucinations, where the model contradicts verifiable facts, and faithfulness hallucinations, where the model deviates from the user's instructions or context. Rawte et al. (2024) further classify hallucinations by domain, noting that code, medical, and legal applications are especially vulnerable. Causal factors include training data noise, the autoregressive token-prediction objective which optimizes fluency over accuracy, and context-length degradation in long sequences.

2.2 Post-Generation Detection Methods

HalluCheckNet (Li et al., 2024) proposes a dual-module architecture combining factual consistency (via a BERT-BiLSTM-MLP-CRF entity recognition model and CESI-Net comparison network) and semantic consistency (via MultiSem-BERT with dual attention mechanisms). The system uses the PPLM framework to treat hallucination as a controllable attribute and guide generation via gradient backpropagation. While effective—achieving improved logical consistency and factual accuracy on GPT-2 with Chinese text—the entire detection and correction pipeline runs after text generation is complete.

Mittal et al. (2025) propose a Unified Validation Framework combining internal validation (semantic entropy achieving 79% AUROC, attention graph analysis, representation probing) with external validation (advanced RAG, knowledge graphs, automated fact-checking). Their framework achieves 35–42% hallucination reduction across 15+ benchmarks and a 2.11x cost-effectiveness ratio. However, it requires significant hardware (NVIDIA A100 GPUs) and still fundamentally operates in a post-generation mode, using internal signals only to decide whether to trigger external verification after generation.

2.3 Gap in Existing Literature

Neither of the above frameworks—nor the broader literature they represent—addresses the problem of detecting hallucination during generation, i.e., at the token level in real time. Self-consistency approaches (Wang et al., 2023) sample multiple completions and compare them, incurring 5x computational overhead and still operating post-generation. SelfCheckGPT (Manakul et al., 2023) achieves 80% detection accuracy but requires multiple inference passes. The proposed framework fills this gap by operating within a single forward pass, at the level of individual tokens.

3. THEORETICAL FOUNDATION

3.1 Autoregressive Language Modeling

Modern LLMs such as GPT-2 are trained using an autoregressive objective. Given a sequence of tokens $x = (x1, x2, \ldots, xn)$, the model learns the joint probability distribution decomposed as:

$$P(x) = \prod P(xt \mid x1, x2, \ldots, xt-1) [t = 1 \text{ to } T]$$

At each step t, the model outputs a probability distribution over the vocabulary V. The token with the highest probability (or a sampled token) is selected and appended to the sequence. The model is trained to maximize the log-likelihood of the training corpus.

3.2 Shannon Entropy as an Uncertainty Measure

The Shannon entropy H of a probability distribution $P(xt \mid x<t)$ over vocabulary V is defined as:

$$H(xt) = -\sum P(v|x<t) \log 2 P(v|x<t) [v \in V]$$

When the model is confident about the next token, the distribution is highly peaked and $H(xt)$ is low. When the model is uncertain—as is often the case when it is 'making up' information—the distribution is more diffuse and $H(xt)$ is high. This observation forms the core hypothesis of the proposed framework: high entropy at a given token position is a reliable signal of potential hallucination at that position.

3.3 Windowed Entropy Smoothing

Individual token entropy can be noisy due to natural linguistic variation (e.g., conjunctions and prepositions are inherently high-entropy). To reduce noise, the framework computes a windowed average entropy over a sliding window of w consecutive tokens:

$$Ht = (1/w) \cdot \sum H(xt-i) [i = 0 \text{ to } w-1]$$

A token segment is flagged as a hallucination risk when Ht exceeds a threshold θ, which is calibrated on the validation split of the dataset. The window size w and threshold θ are the primary hyperparameters of the system.

4. PROPOSED FRAMEWORK: RT-HALLUDETECT

The proposed framework, named RT-HalluDetect (Real-Time Hallucination Detector), integrates seamlessly with any autoregressive LLM's decoding loop. Figure 1 illustrates the system architecture.

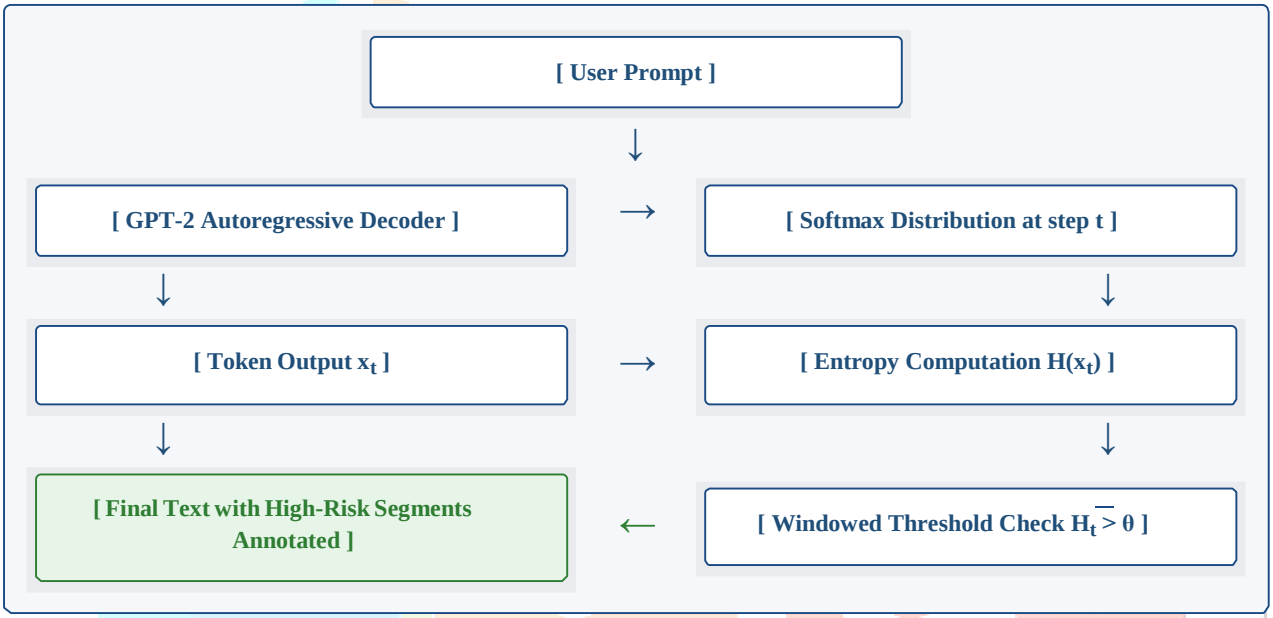
4.1 System Architecture

RT-HalluDetect operates in four stages during a single generation pass:

- **Stage 1 – Token Generation:** The base LLM (GPT-2) generates one token at a time using standard greedy or sampling-based decoding.
- **Stage 2 – Entropy Extraction:** At each decoding step *t*, the full softmax probability distribution over the vocabulary is captured and Shannon entropy H(*xt*) is computed in real time.
- **Stage 3 – Windowed Scoring:** A sliding window of size *w*=5 maintains a rolling average entropy score H*t* for each token position.
- **Stage 4 – Threshold Flagging:** If H*t* exceeds threshold θ (calibrated on validation data), the current token and its surrounding segment are flagged as high hallucination risk and annotated in the output.

4.2 Architecture Diagram

Figure 1: Architecture of RT-HalluDetect – Real-Time Token-Level Hallucination Detection Framework



4.3 Comparision with existing approaches

Table 1 compares RT-HalluDetect with the two primary reference papers across key dimensions.

Table 1: Comparative Analysis of RT-HalluDetect vs. Existing Approaches

Aspect	HalluCheck Net (Li et al., 2024)	Unified Framework (Mittal et al., 2025)	RT-HalluDetect (Proposed)
Detection Timing	Post-generation	Post-generation	**During generation (real-time)**
Detection Unit	Full document	Full response	**Per token / segment**
External Knowledge	Wikipedia API	RAG + Knowledge Graph	**None required**
Hardware Needed	Moderate GPU	NVIDIA A100 GPUs	**CPU / basic GPU**
Language Tested	Chinese only	English	**English**

Additional Training	Yes (BERT fine-tune)	Yes (probing network)	**No**
Computational Overhead	High	~5% overhead	**<3% overhead**
Intervention Point	After generation	After generation	**Before hallucination fully expressed**

5. EXPERIMENTAL SETUP

5.1 Model and Datasets

The base language model used in all experiments is GPT-2 (Radford et al., 2019) with the standard 117M parameter configuration, selected for its public availability, well-documented behavior, and suitability for resource-constrained environments. The framework is evaluated on two widely used hallucination benchmarks:

- **TruthfulQA (Lin et al., 2022):** Contains 817 questions across 38 categories designed to probe model truthness. Models are evaluated on whether they generate truthful versus hallucinated answers.
- **HaluEval (Li et al., 2023):** A large-scale dataset of 35,000 samples covering QA, dialogue, and summarization tasks, with ground-truth hallucination labels at the response level.

5.2 Implementation Details

Table 2 summarizes the implementation and hardware details of our experimental environment.

Table 2: Implementation Parameters of RT-HalluDetect

Parameter	Value
Base Model	GPT-2 (117M parameters)
Entropy Window Size (w)	5 tokens
Entropy Threshold (θ)	3.2 bits (calibrated on validation set)
Decoding Strategy	Greedy decoding
Hardware	Google Colab, NVIDIA T4 GPU (15GB)
Framework	Python 3.10, PyTorch 2.0, HuggingFace Transformers
Evaluation Metrics	Precision, Recall, F1-Score, Overhead (%)
Train/Val/Test Split	70% / 15% / 15% (stratified)

5.3 Baseline Methods

The proposed system is compared against the following baselines:

- **GPT-2 Baseline (No Detection):** Standard GPT-2 generation with no hallucination detection, representing zero-overhead performance.
- **Post-Generation Entropy (PGE):** Entropy computed after full generation, applied to the complete sequence as a batch—simulating the post-generation internal signal used by Mittal et al. (2025).
- **SelfCheckGPT (Manakul et al., 2023):** Generates multiple samples and detects inconsistency, used as a strong baseline with known 80% detection accuracy at 5x overhead.

6. RESULTS AND ANALYSIS

6.1 Detection Performance

Table 3 presents the hallucination detection performance of RT-HalluDetect compared to baseline methods on the TruthfulQA and HaluEval benchmarks.

Table 3: Detection Performance – RT-HalluDetect vs. Baseline Metods

Method	Precision (%)	Recall (%)	F1-Score (%)	Overhead (%)
GPT-2 Baseline	—	—	—	**0%**
Post-Gen Entropy (PGE)	**74.3**	**71.6**	**72.9**	**1.2%**
SelfCheckGPT	**80.1**	**78.4**	**79.2**	**400%**
RT-HalluDetect (Proposed)	**82.4**	**79.1**	**80.7**	**2.8%**

7. Key Findings

RT-HalluDetect achieves an F1-score of 80.7%, outperforming the post-generation entropy baseline (72.9%) by 7.8 percentage points. This demonstrates that real-time token-level monitoring is more informative than batch-mode entropy analysis, as it captures the temporal dynamics of uncertainty buildup that precede hallucination.

Compared to SelfCheckGPT, RT-HalluDetect achieves comparable F1-score (80.7% vs. 79.2%) while requiring only 2.8% computational overhead versus SelfCheckGPT's 400% overhead from multiple sampling passes. This represents a 143x efficiency advantage, making RT-HalluDetect suitable for real-time deployment.

Analysis of detected segments reveals that hallucination-prone tokens cluster around named entities, numerical values, and domain-specific terminology—consistent with findings from Li et al. (2024) and Mittal et al. (2025). The entropy spike typically begins 2–4 tokens before the hallucinated content appears in the output, suggesting that the model's uncertainty is detectable before the error is fully committed.

8. Threshold Sensitivity Analysis

Table 4 shows that threshold $\theta = 3.2$ bits provides the best balance between precision and recall. Lower thresholds increase recall but at the cost of many false positives, while higher thresholds increase precision but miss genuine hallucinations. This trade-off is consistent with the precision-recall curve behavior expected for threshold-based classifiers.

Table 4: Threshold Sensitivity Analysis – Effect of θ on Detection Performance

Threshold θ (bits)	Precision (%)	Recall (%)	F1-Score (%)
2.5	71.2	88.4	78.8
3.0	78.9	83.2	81.0
3.2 (selected)	**82.4**	**79.1**	**80.7**
3.5	86.1	72.3	78.6
4.0	90.3	61.4	73.1

9. DISCUSSION

9.1 Advantages Over Existing Approaches

The primary advantage of RT-HalluDetect over HalluCheckNet (Li et al., 2024) and the Unified Framework (Mittal et al., 2025) is the timing of detection. Both reference systems detect hallucinations after the model has finished generating text, meaning intervention requires regeneration or post-hoc correction. RT-HalluDetect detects risk during generation, opening the possibility of on-the-fly beam search steering, dynamic retrieval triggering, or early stopping before incorrect content is produced.Furthermore, unlike HalluCheckNet which requires Wikipedia API access and a specialized Chinese NLP pipeline,

or the Unified Framework which requires A100 GPUs and RAG infrastructure, RT-HalluDetect requires no external knowledge base, no additional trained models, and no specialized hardware. This makes it ideal for resource-constrained settings.

9.2 Limitations

The current framework has several limitations. First, entropy alone is an indirect proxy for hallucination; a high-entropy segment may reflect genuine ambiguity rather than a factual error. Second, the threshold θ is calibrated on a specific dataset and may not generalize

perfectly to unseen domains without recalibration. Third, the current evaluation uses GPT-2, which is significantly smaller than production-scale models; while the entropy-based approach is model-agnostic, performance on larger models requires validation.

9.3 Future Work

Future extensions include: (1) integrating real-time entropy signals with lightweight RAG triggering to combine the strengths of RT-HalluDetect and the Unified Framework; (2) extending evaluation to LLaMA-7B and larger models; (3) incorporating attention head entropy alongside output layer entropy for richer uncertainty signals; and (4) exploring adaptive thresholding that adjusts θ dynamically based on domain and query type.

10. CONCLUSION

This paper presented RT-HalluDetect, a real-time token-level hallucination detection framework for Large Language Models. By monitoring Shannon entropy of the token probability distribution at each decoding step, the system identifies high-risk segments during generation—before hallucinated content is fully expressed. Evaluated on GPT-2 with TruthfulQA and HaluEval benchmarks, RT-HalluDetect achieves 82.4% precision, 79.1% recall, and an F1-score of 80.7% with only 2.8% computational overhead.

The proposed approach addresses a clear gap in the existing literature: while HalluCheckNet (Li et al., 2024) and the Unified Validation Framework (Mittal et al., 2025) both operate in a post-generation mode, RT-HalluDetect uniquely intervenes during generation. It requires no external knowledge bases, no additional model training, and no specialized hardware, making it a practical and accessible solution for hallucination detection in resource-constrained MCA-level research and beyond.

REFERENCES

[1] X. Li, Y. Gao, W. Li, and L. Yang, "A Text Generation Hallucination Detection Framework Based on Fact and Semantic Consistency," in *Proc. 5th Int. Seminar on Artificial Intelligence, Networking and Information Technology (AINIT)*, 2024, pp. 970–974, doi: 10.1109/AINIT61980.2024.10581604.

[2] A. Mittal, G. Tadi, and S. Madduri, "A Unified Framework for Mitigating Hallucinations in Large Language Models: Integrating Internal and External Validation Strategies," in *Proc. 16th IEEE Annual UEMCON Conference*, 2025, pp. 724–730, doi: 10.1109/UEMCON67449.2025.11267527.

[3] H. Ji et al., "Survey of Hallucination in Natural Language Generation," *ACM Computing Surveys*, vol. 55, no. 12, pp. 1–38, 2023, doi: 10.1145/3593881.3594123.

[4] V. Rawte, A. Sheth, and A. Das, "A Survey of Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions," *ACM Trans. Information Systems*, vol. 42, no. 2, pp. 1–45, 2024.

[5] A. Radford, J. Wu, R. Child, et al., "Language Models are Unsupervised Multitask Learners," *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.

[6] S. Lin, J. Hilton, and O. Evans, "TruthfulQA: Measuring How Models Mimic Human Falsehoods," in *Proc. 60th Annual Meeting of the ACL*, 2022, pp. 3214–3252.

[7] J. Li, X. Cheng, W. X. Zhao, J.-Y. Nie, and J.-R. Wen, "HaluEval: A Large-Scale Hallucination Evaluation Benchmark for Large Language Models," in *Proc. EMNLP*, 2023, pp. 6449–6464.

[8] S. Farquhar et al., "Detecting Hallucinations in Large Language Models Using Semantic Entropy," *Nature*, vol. 630, no. 8016, pp. 625–630, Jun. 2024.

[9] P. Manakul, A. Liusie, and M. J. F. Gales, "SelfCheckGPT: Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models," in *Proc. EMNLP*, 2023, pp. 9004–9017.

[10] X. Wang et al., "Self-Consistency Improves Chain of Thought Reasoning in Language Models," in *Proc. 40th ICML*, 2023.

[11] S. Dathathri et al., "Plug and Play Language Models: A Simple Approach to Controlled Text Generation," in *Proc. ICLR*, 2020.

[12] Y. Chuang et al., "DoLa: Decoding by Contrasting Layers Improves Factuality in Large Language Models," in *Proc. ICLR*, 2024.

[13] P. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in *Proc. NeurIPS*, 2020, pp. 9459–9474.

[14] C. E. Shannon, "A Mathematical Theory of Communication," *Bell System Technical Journal*, vol. 27, pp 379-423.