Learned Hallucination Detection in Black-Box LLMs using Token-level Entropy Production Rate

Tuesday, March 31st 2026

Charles Moslonka,
Artefact Research Center.
MICS, Université Paris-Saclay, France.

Hicham Randrianarivo,
Artefact Research Center.
MICS, Université Paris-Saclay, France.

Arthur Garnier,
Ardian,
Paris France

Emmanuel Malherbe,
Artefact Research Center
Paris France

48th European Conference on Information Retrieval
ECIR 2026
29 March – 2 April
Delft, The Netherlands

Large Language Models Hallucinations

- False/unfaithful facts given by LLMs, ofter with high confidence and fluency.
- High stakes for AI adoption.
- May happen whatever the domain :
  - Copilot hallucinates API/functions.
  - LLM lawyer hallucinates a ruling.
  - LLM doctor hallucinates a treatment for your disease.

- Our goal is to detect these hallucinations and mitigate them (abstention, RAG, etc.), using the **entropy** of the generated outputs.
- In Generative Retrieval (e.g. DSI, NCI) : Uncertainty Quantification of generated DocIDs.
- Agentic Retrieval: Indicator when to call a retriever agent.

Hallucination Detection Methods

Multi-shot methods

- The easiest method to determine when a LLM hallucinates is to... **ask it again** (Monte-Carlo approach). Multiple calls and compare the results.
  - **Black-box** set-up: SelfCheckGPT for automatic comparison.
  - Semantic Entropy with **log-probs access.**
- **Costly** in both **time** and **money.**

Single-shot methods

- Realistic use-case: quantify the uncertainty of the produced output.
  - **White-box** set-up (weights and internal states): Semantic entropy probes.
  - In production : **Black-box** APIs with only limited top-token log-probs.

➡ Our work : **single-shot black-box.**

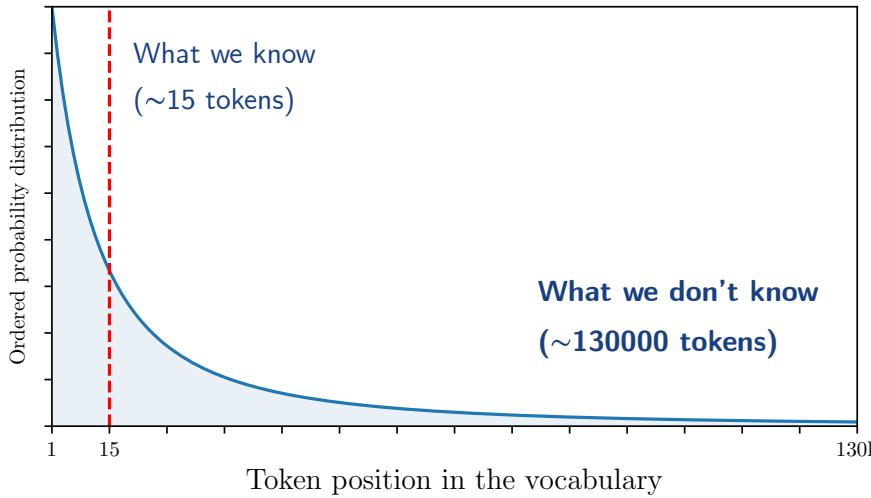
Entropy from top-K tokens

Modern LLM APIs (OpenAI, Gemini) give us access to the top $K \leq 20$ top-tokens log-probabilities. Approximate the entropy H:

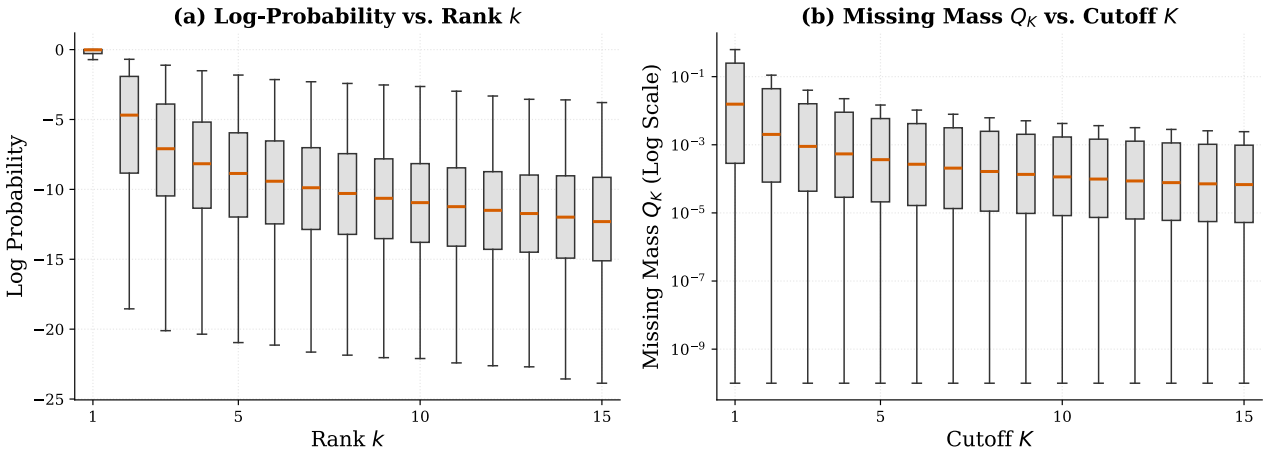
$$H(q, t_{<j}) \simeq \tilde{H}_K(q, t_{<j}) = -\sum_{k=1}^{K} p_{r(k),j} \log_2 \left(p_{r(k),j} \right)$$

For one generated token t_j, K ranked probabilities.

Is K enough ?



Missing Probability mass



The log-probabilities tend to stack quickly at « low » probabilities :

The vast majority of the probability mass is located in the first tokens.

Moreover, we have an upper bound of how much information we lack in the tail.

$$\Delta H_{K,\max}(q, t_{<j}) = -Q_K(q, t_{<j}) \left[\log_2(Q_K(q, t_{<j})) - \log_2(|\mathscr{V}| - K) \right]$$

Q_K is the missing mass at rank K, $|\mathscr{V}|$ the vocabulary size.

Baseline Method: Entropy Production Rate (EPR)

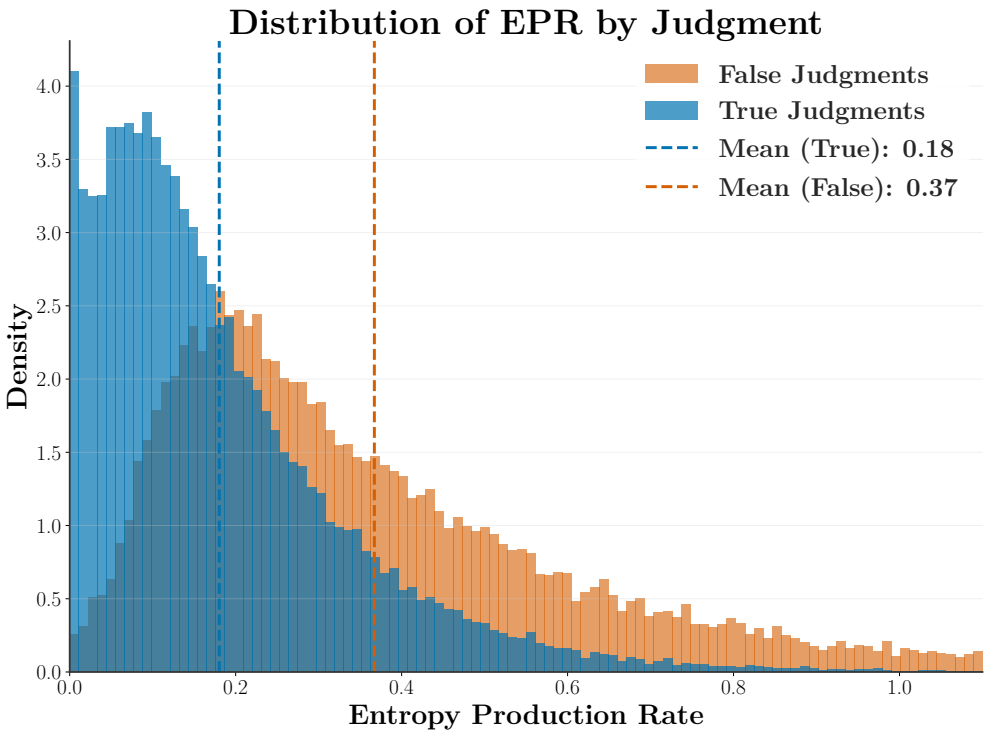
- **The Idea:** If a model is uncertain at many steps while generating an answer, the answer is more likely to be a hallucination.

- We derive the **Entropy Production Rate (EPR)** as the average of the estimated entropy across all tokens $\{t_{<j}\}$ in the generated answer (length L_q) to a query q.

$$\mathrm{EPR}_K(q) = \frac{1}{L_q} \sum_{j=1}^{L_q} \tilde{H}_K(q, t_{<j})$$

Single, one-shot, unsupervised "hesitation score" for the entire sequence. **Temperature independent** (logprobs are returned **before** the sampling softmax).

Baseline Method: Entropy Production Rate (EPR)



On **TriviaQA** dataset

Dataset of general knowledge and trivia question answering.

Annotated True/False answer distribution by the LLM (model : `Falcon-10B`).

False (hallucinated) answer tend to have higher EPR.

Our method : Weighted EPR (WEPR)

A Learned Estimator

- Use annotated answers to learn a better estimator.
- Instead of a simple sum, we learn a **weighted sum** of the entropic contributions from **each rank**

$$s_{k,j} = -\,p_{r(k),j}\log_2(p_{r(k),j})$$

- The model learns weights coefficients β_k for each rank's contribution. Same β_k across the sequence.

$$S_\beta(q, t_{<j}) = \beta_0 + \sum_{k=1}^{K} \beta_k s_{k,j} \qquad \text{and} \qquad \mathrm{WEPR}_K(q) = \frac{1}{L_q}\sum_{j=1}^{L_q} S_\beta(q, t_{<j}) + \sum_{k=1}^{K} \gamma_k \max_j s_{k,j}$$

$S_\beta(q, t_{<j})$ are token-level scores.

Training: Logistic regression model trained on a dataset of answers labeled as "True" or « False" (Extracted from TriviaQA).

$\sigma(\mathrm{WEPR}_K(q)) \in [0,1]$ is a normalized hallucination score.

Experimental set-up

- 4 different models :
  - `Mistral-Small-3.1-24B-Instruct-2503`
  - `Falcon3-10B-Instruct`
  - `Phi-4`
  - `Ministral-8B-Instruct-2410`

- 3 different datasets: TriviaQA (training), WebQuestions (generalization) and in-house out-of-domain « ArGiMi-Ardian » on financial data.

- Annotation process: **LLM-as-a-Comparator** to ground truth with `Gemma-3-12B-it`.

95.7 % agreement with human annotators (1300+ samples).

WEPR and EPR Performance (black-box)

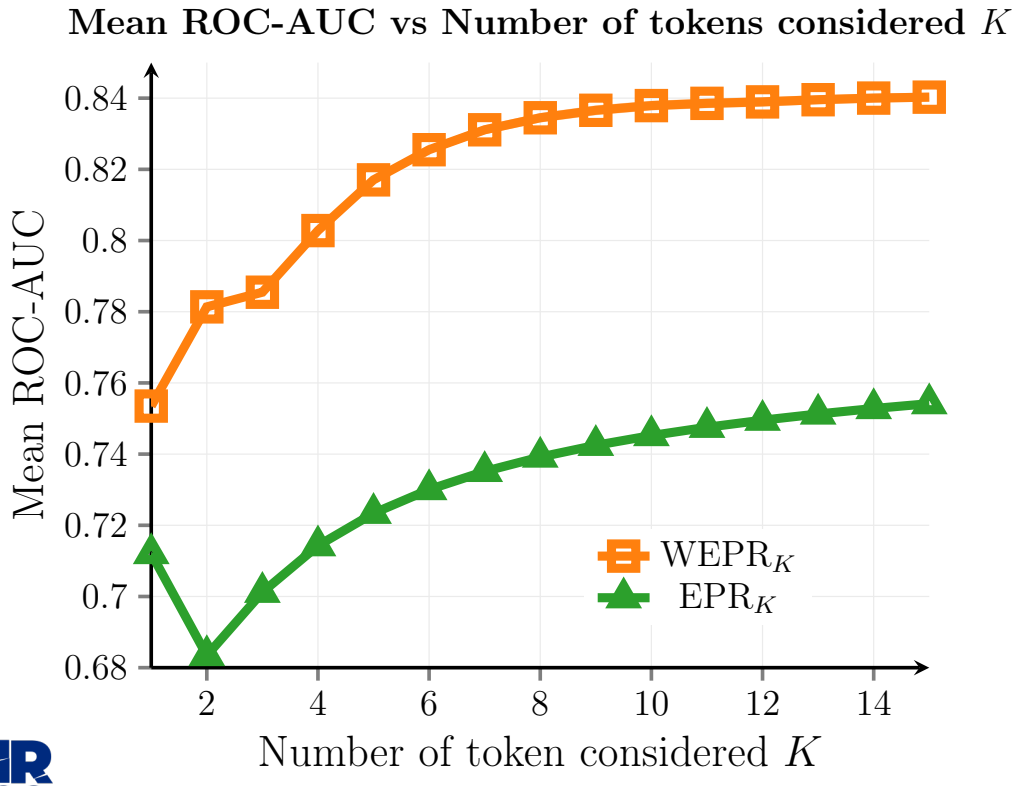
Dataset	Model	*Unsupervised*		*Supervised*	
		SCGPT	EPR (ours)	HalluDetect	WEPR (ours)
TriviaQA (Hallucination Detection)	`Mistral-Small-24B`	79.0	74.6	78.7	**82.0**
	`Falcon-3-10B`	70.1	75.4	79.0	**84.1**
	`Phi-4 (14.7B)`	71.4	78.2	83.8	**85.4**
	`Ministral-8B-2410`	81.1	81.4	**86.1**	85.8

Across all **models** and **datasets** (TriviaQA, WebQuestions), WEPR ROC-AUCs are consistently higher compared to baseline EPR and other single (HalluDetect) and multi-shot (SelfCheckGPT) black-box methods.

Robust to domain change.

RAG-framework: detect when you need to retrieve more context.

How many top-K do we need ?



- Performance improves with K but starts to plateau around $K \simeq 8 - 10$
- The method is effective even with the limited data from standard APIs.
- In practice, for $K = 15$, we need around 2000 annotated samples.

Token-level entropy indicator

- Each token contribution $S_\beta(q, t_{<j})$ can be used to compute a score $\sigma\left(S_\beta\right) \in [0,1]$, that can be interpreted as the probability for the j-th token to be unreliable.

What is Emerson's net revenue in 2023?

Emerson's reported net sales for the fiscal year 2023 were $18.5 billion.

- Token-level flag for unreliable data (say, $\sigma\left(S_\beta\left(q, t_{<j}\right)\right) \geq 1/2$)

Limitations

- Tested on « small » to « medium » models. **Not tested** on models larger than 24B.
- **High-Certainty Hallucinations:** Our method can't catch confidently-stated falsehoods learned from bad training data.
- Automatic annotation can have false positives.
- Trained on short QA answers. Aggregated EPR/WEPR signal might get diluted in very long text. Thought/reasoning tokens may also contribute to this effect.
- **Multiple choice QA** is also a problem : Hesitation is normal and expected (ex : « name a red fruit »). Might be handled with semantic regrouping of tokens / cluster of embeddings.

Conclusion and perspectives

- **WEPR is a lightweight, one-shot, black-box** hallucination detection via learned entropic contributions.
- Works best with 8+ top-logprobs per generated token.

  Current work :
  - Entity grouping,
  - « Distillation » of annotated answers
  - Reasoning models
- **These methods (with demos) are implemented in our Python Package `artefactual`**

  `https://github.com/artefactory/artefactual` (or `~$pip install artefactual`)



Package repo





Paper

Thank you !





