

HALT: Hallucination Assessment via Log-probs as Time series

Ahmad Shapiro^{1,2*}, Karan Taneja¹, Ashok Goel¹

¹Georgia Institute of Technology ²Microsoft

Project Page: <https://halt-hallucination-detection.github.io>

Abstract

Detecting hallucinations in Large Language Model (LLM) outputs is critical for safe deployment, yet existing methods require white-box access to internals, depend on multiple generations or auxiliary judge models, or collapse token-level uncertainty into scalar summaries that discard temporal structure. We introduce **HALT**, a lightweight grey-box detector that treats the top-20 log-probabilities of an LLM generation as a multivariate time series and processes them with a small bidirectional GRU (5M parameters), in a single forward pass and using no model internals. We evaluate on **HUB**, a unified protocol reorganizing existing hallucination datasets into ten capability clusters spanning factual and logical failure modes. HALT matches or surpasses Lettuce, a fine-tuned ModernBERT-base detector, on macro- F_1 while using $\sim 30\times$ fewer parameters; a controlled comparison against a text classifier trained on HALT’s own data confirms this advantage comes from the log-probability signal itself rather than from differences in training data, and the same recipe applies across eight LLM families from 360M to 70B parameters.

1 Introduction

Large Language Models (LLMs) have achieved remarkable progress in generating fluent and coherent text, yet they remain prone to *hallucinations*: outputs that are verifiably false or unsupported by evidence. Li et al. (2023) report that GPT-3.5 introduces unverifiable details in nearly 19.5% of user queries, and similar failure rates have been documented across model families and tasks (Ji et al., 2023). As LLMs are deployed in increasingly safety-critical settings, reliable post-hoc detection of hallucinations has become essential.

Existing approaches fall into three camps, each with deployment costs. **White-box** methods in-

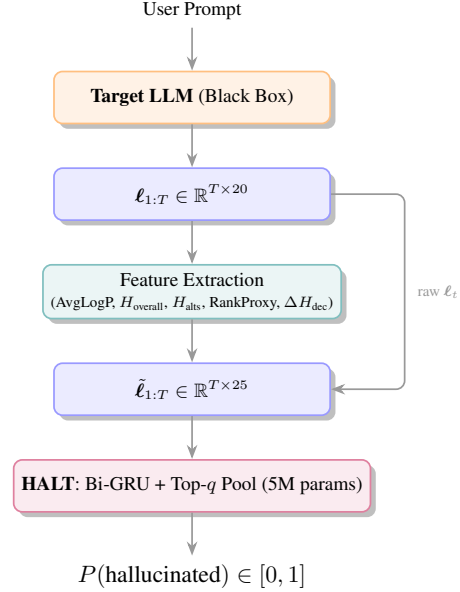


Figure 1: **HALT architecture.** Top-20 log-probabilities from the target LLM are augmented with five engineered features and processed by a 5M-parameter Bi-GRU with Top- q pooling to produce a hallucination probability.

spect internal states such as hidden activations or attention maps (Sriramanan et al., 2024; Chen et al., 2024; Azaria and Mitchell, 2023), requiring model access and per-dataset layer search or training. **Black-box and multi-sample** methods rely on multiple generations (Manakul et al., 2023), auxiliary judges (Lan et al., 2024), or external retrieval (Mishra et al., 2024; Friel et al., 2025), introducing latency or monetary cost. **Aggregate-statistics** methods compress token-level uncertainty into scalars such as perplexity (Varshney et al., 2023; Quevedo et al., 2024), which is cheap but discards temporal information about *how* uncertainty evolves.

This leaves an underexplored point in the design space: *the sequence of top- K log-probabilities returned by most modern APIs and serving frameworks* (e.g., vLLM, Kwon et al., 2023). Top- K

* Work done prior to joining Microsoft.

Correspondence: ahmad.shapiro@gatech.edu

log-probs (typically $K \leq 20$) are richer than a single scalar but far cheaper than full distributions or repeated generations, and require no access to model internals. We argue that the *temporal shape* of this signal carries information about hallucinations that aggregate statistics discard.

We introduce **HALT** (Hallucination Assessment via Log-probs as Time series), a lightweight grey-box detector that treats the top-20 log-probability vectors of an LLM generation as a multivariate time series and processes them with a small bidirectional GRU (Cho et al., 2014). HALT relies on only a single forward pass, no model internals, and no auxiliary models. We evaluate two trained variants—HALT-L and HALT-Q, on Llama 3.1-8B and Qwen 2.5-7B respectively—and show that the same architecture and training procedure transfer across eight LLM families from 360M to 70B parameters, with a separately trained detector per LLM. To enable broad evaluation, we assemble **HUB** (Hallucination detection Unified Benchmark), which reorganizes existing labeled data (FAVA, RAGTruth, HaluEval, CriticBench) into ten LLM capability clusters spanning factual and logical (reasoning-induced) failure modes under a unified evaluation protocol. HUB introduces no new annotations.

On HUB, HALT matches or surpasses Lettuce (Ádám Kovács and Recski, 2025), a fine-tuned ModernBERT-base detector, on macro- F_1 while using approximately $30\times$ fewer parameters. The advantage is most pronounced on reasoning-heavy clusters where surface-form cues are scarce; Lettuce remains stronger on RAGTruth, where it is trained in-domain. HALT also remains competitive against multi-sample (SelfCheckGPT, Manakul et al., 2023) and high- K (LOS-Net, Bar-Shalom et al., 2025b) methods while operating in a strictly cheaper regime.

Contributions. **HALT**, a 5M-parameter bidirectional GRU to detect hallucinations from top-20 log-probability sequences. HALT operates in a single forward pass with no access to internals, surface text, or external references, and we release trained variants for two LLM families. **HUB**, a unified evaluation protocol that organizes existing hallucination datasets (FAVA, RAGTruth, HaluEval, CriticBench) into ten capability clusters spanning factual and logical hallucinations, enabling consistent cross-capability comparison of detection methods. **A controlled empirical comparison** against single-pass alternatives (Manakul et al., 2023; Bar-

Shalom et al., 2025b) showing that low- K temporal modeling can match or exceed methods using multiple generations or much larger K . **Architecture, feature, and top- K ablations** isolating which design choices drive performance, together with cross-scale evaluation across eight LLM families (360M–70B parameters).

2 HUB: A Unified Benchmark

Existing hallucination detection benchmarks cover narrow slices of LLM capability: RAGTruth (Niu et al., 2024) targets three reference-based tasks (Data-to-Text, QA, Summarization), HaluEval (Li et al., 2023) covers QA, Dialogue, and Summarization, and FAVA (Mishra et al., 2024) focuses on knowledge-intensive queries. None systematically include reasoning tasks, where LLMs increasingly fail despite producing fluent output. We assemble HUB to address this gap not by collecting new annotations but by reorganizing existing labeled data into a unified evaluation protocol.

Factual versus logical hallucinations. Following the faithfulness–factuality distinction of Ji et al. (2023), we treat a hallucination as any output unfaithful to its task specification. *Factual hallucinations* violate factuality: they contradict established reality even when correct context is provided, as in RAG settings where the model ignores grounding. *Logical hallucinations* are a faithfulness failure: the model correctly identifies the premises but produces inferences unfaithful to the task’s logical constraints—arithmetic mistakes, invalid steps, mis-classifications, or inconsistent chains. Figure 2 shows a logical hallucination where the model both misclassifies a non-fruit (*carrot*) and errs in the summation. In Ji et al. (2023)’s framework, faithfulness means the output does not introduce content unsupported by, or inconsistent with, the given input—typically applied to source-grounded settings where an unfaithful output contradicts retrieved context. We extend this to a different kind of “given”: in a math or logic problem, the premises and the task’s inferential rules play the same role retrieved context plays in RAG, and an invalid inference step is a conclusion not entailed by those premises—the same structural failure, applied to task constraints rather than a document.

Composition. HUB organizes existing datasets into ten capability clusters: five *reasoning-oriented* (Algorithmic, Commonsense, Mathematical, Sym-

User Prompt

I have a peach, a strawberry, a banana, a carrot, two apples, and four grapes. How many fruits do I have?

Model Response

Let’s think step by step. I will identify the fruits and their quantities: peach (1), strawberry (1), banana (1), **carrot (1)**, apples (2) and grapes (4). Adding the quantities: **$1 + 1 + 1 + 1 + 2 + 4 = 11$** . So the answer is **11**.

Figure 2: **Example of a logical hallucination.** The model commits two distinct reasoning errors: (i) a *classification error*, incorrectly including a carrot (a vegetable) in the fruit count, and (ii) an *arithmetic error*, reporting the sum $1+1+1+1+2+4$ as 11 rather than the correct value 10. The correct answer is 9 (the carrot should be excluded). Both errors are highlighted in red. The sample originates from the *ObjectCounting* shard of CriticBench (Lan et al., 2024) in the Algorithmic Reasoning cluster.

bolic, Code Generation) and five *general-purpose* (Chat, Data-to-Text, Question Answering, Summarization, World Knowledge). Reasoning clusters are built from CriticBench (Lan et al., 2024), whose annotations cover both reasoning-trace errors and final-answer errors. General-purpose clusters draw from FAVA (Mishra et al., 2024), RAGTruth (Niu et al., 2024), and HaluEval (Li et al., 2023). Each cluster aggregates multiple source datasets where available, supporting cross-dataset generalization within a capability. Table 9 reports cluster-level statistics; per-dataset details appear in Appendix A.

Splits and generalization protocol. To stress-test generalization, training is restricted to four general-purpose clusters (Chat, Data-to-Text, Summarization, QA), providing approximately 60K labeled responses while leaving six clusters—all five reasoning clusters plus World Knowledge—held out from training. We avoid the term “zero-shot” here: model selection (checkpointing, hyperparameters, thresholds) uses a pooled validation metric spanning every cluster, including the six held-out ones, so training and selection exposure are not identical—though a weaker form of leakage than training on them directly, and mitigated by construction, since validation and test within each

held-out cluster draw from disjoint source datasets rather than disjoint examples of one dataset (e.g. for Mathematical, validation is TabMWP while test is GSM8K, MATH, and AQuA; full correspondences in Appendix A.3). A single global threshold is tuned once on the pooled validation set and applied uniformly, not per-cluster. For FAVA Annotations and RAGTruth we additionally retain the original test splits for direct comparison with prior baselines.

Hallucination ratios vary sharply across clusters, from $\sim 40\%$ (Chat, Summarization) to $\sim 95\%$ (World Knowledge). This motivates **macro- F_1** as our primary metric, detailed in Section 4.1.

Annotation reliability. HUB introduces no new annotations. Labels are inherited from the source datasets: FAVA, RAGTruth, and HaluEval contribute human and crowd-sourced annotations, and CriticBench contributes a hybrid pipeline combining rule-based heuristics, GPT-4 judging, and human adjudication on disagreements. To verify that CriticBench’s failure-mode taxonomy aligns with our definition of logical hallucination, we manually inspected samples from each reasoning cluster and found approximately 90% agreement with the original labels. We discuss annotation provenance as a known limitation in Section 6 and inherit the licensing terms of the source datasets.

3 Method

3.1 Setup

Let $\mathcal{M}_\theta$ be an autoregressive LLM. At each generation step t , $\mathcal{M}_\theta$ produces a distribution over its vocabulary, of which we observe only the top-20 log-probabilities:

$$\ell_t = (\log p_t^{(0)}, \log p_t^{(1)}, \dots, \log p_t^{(19)}) \in \mathbb{R}^{20}, \quad (1)$$

where $\ell_t^{(0)}$ is the *selected* token (not necessarily greedy under sampling) and indices 1–19 are the top-19 alternatives sorted by probability in descending order, regardless of the selected token’s rank (used in the RankProxy definition, Section 3.2). The serving framework always returns the selected token’s log-probability; when it falls outside the top-19, we take the union of both sets. A response of length T yields $\ell_{1:T} \in \mathbb{R}^{T \times 20}$, obtained by teacher-forcing the labeled response into $\mathcal{M}_\theta$ via vLLM (Kwon et al., 2023). The choice $K = 20$ is justified in Section 3.6.

3.2 Token-Level Features

For each timestep, we augment ℓ_t with five engineered features capturing local calibration behavior:

- **Average log-probability** (AvgLogP) – mean over the top-20 entries; surrogate for distributional sharpness.
- **Rank proxy** (RankProxy) – the position of the selected token among the top-20 alternatives, indicating non-greedy selections.
- **Top- K entropy** (H_{overall}) – entropy of the renormalized top-20 distribution; high values indicate indecision among the leading candidates.
- **Alternatives entropy** (H_{alts}) – entropy over the top-20 distribution *excluding* the selected token, isolating disagreement among competitors.
- **Decision-entropy delta** (ΔH_{dec}) – temporal change in binary entropy between the selected token and its strongest alternative; flags rapid certainty–uncertainty transitions.

These five were not selected by exhaustive search; they span four complementary families: (i) *magnitude* (AvgLogP), (ii) *rank* of the selected token among alternatives (RankProxy), (iii) *distributional-shape entropy* over two supports— H_{overall} (full top- K) and H_{alts} (alternatives only, isolating competitor disagreement from the selected token’s own confidence)—and (iv) *temporal dynamics* (ΔH_{dec}), capturing change rather than level. Each targets a distinct aspect of token-level uncertainty, consistent with the largely non-overlapping, additive gains in the incremental ablation (Table 2).

Concatenating these with the raw ℓ_t yields a 25-dimensional feature vector per timestep:

$$\tilde{\ell}_t = [\phi(\ell_t) \parallel \ell_t] \in \mathbb{R}^{25}. \quad (2)$$

Full derivations and the truncated-softmax normalization used for H_{overall} and H_{alts} are provided in Appendix B.3.

3.3 Architecture

Figure 1 summarizes the architecture. The 25-dimensional feature sequence $\tilde{\ell}_{1:T}$ is first normalized (LayerNorm) and projected to a 128-dimensional embedding via a two-layer GELU

MLP. The projected sequence is encoded by a bidirectional GRU (Choi et al., 2014) with 5 layers and hidden size 256 (dropout 0.4 between recurrent layers). Variable-length responses are handled with packed sequences and a boolean attention mask, ensuring padded positions do not influence hidden dynamics.

We aggregate the bidirectional GRU outputs ($H \in \mathbb{R}^{T \times 512}$) using **Top- q pooling**: timesteps are scored by their hidden-state ℓ_2 norm, padding positions are masked out, and the top $q = 15\%$ of timesteps are averaged. This concentrates the classifier on the most informative regions of the sequence (e.g., bursts of uncertainty around hallucination onsets) rather than diluting the signal across all tokens. We compare Top- q pooling against mean, max, and attention-based pooling in Appendix E.

The pooled vector passes through a linear head to a single logit, trained with BCEWithLogitsLoss. The full model has $\sim 5\text{M}$ parameters; training details are in Appendix C.

3.4 Why a Temporal Model? (Architecture Ablation)

A natural question is whether the temporal structure of $\tilde{\ell}_{1:T}$ is necessary at all, or whether aggregate statistics over the sequence would suffice. We compare HALT’s Bi-GRU encoder against both non-temporal baselines and alternative temporal architectures using identical features and training data. Table 1 reports macro- F_1 averaged over HUB clusters and overall.

We add two attention-based architectures: a parameter-matched Transformer encoder (self-attention over $\tilde{\ell}_{1:T}$, same features, Top- q pooling, and training data, 5.47M params vs. HALT-L’s 5.34M) and a parameter-matched reimplementa-tion of LOS-Net (Bar-Shalom et al., 2025b) (Transformer with rank encoding and CLS pooling, faithful to its published design) evaluated under HUB’s own splits and $K = 20$ rather than its original top-50/top-1000 regime, both retuned with standard Transformer defaults after an initial run with HALT-L’s Bi-GRU-tuned optimizer settings underperformed.

Three patterns emerge. Temporal modeling provides a substantial gain: the best non-temporal baseline reaches only 0.430 Overall F_1 , while every temporal architecture exceeds 0.550. This 12–24 point gap empirically refutes the implicit assumption of aggregate-statistics methods (Varshney et al., 2023; Quevedo et al., 2024) that collapsing

Table 1: Architecture comparison on HUB using Llama 3.1-8B log-probabilities. Non-temporal models operate on per-sequence aggregate statistics; temporal models operate on the full $\ell_{1:T}$ sequence. LOS-Net-faithful and Transformer-encoder are parameter-matched to HALT-L (~ 5.4 M params) and trained under identical data and $K=20$ with tuned hyperparameters.

Type	Model	Avg. F1	Overall F1
Non-temporal	LR	0.385	0.430
Non-temporal	XGBoost	0.372	0.418
Temporal	1D-CNN	0.510	0.565
Temporal	Bi-RNN	0.507	0.551
Temporal	Uni-GRU	0.543	0.600
Temporal	LOS-Net	0.540	0.613
Temporal	Transformer	0.560	0.618
Temporal	Bi-LSTM	0.591	0.655
Temporal	Bi-GRU	0.630	0.670

the log-probability sequence loses no useful information. Gating matters (Bi-RNN trails Bi-LSTM and Bi-GRU by 5–10 points), and bidirectionality contributes another 4–7 points, consistent with the intuition that hallucination signatures often become recognizable in retrospect. Under an identical parameter budget, data, and K , both attention-based alternatives trail Bi-GRU by 7–9 Avg. F_1 points (0.540 and 0.560 vs. 0.630); retuning narrowed but did not close this gap, consistent with a genuine generalization ceiling on HUB’s training scale rather than an optimization artifact.

3.5 What Signal Do the Features Carry?

Within HALT’s input, we further isolate the contribution of each engineered feature by training incremental variants that begin from raw log-probabilities and progressively add features (Table 2). We also include a variant that drops the raw log-probability channels and retains only the five engineered features, still processed by the full Bi-GRU and Top- q pooling—this isolates the marginal contribution of the raw channels given a temporal encoder already in place, and should not be read as a non-temporal baseline (that comparison is Table 1, Section 3.4).

Raw top-20 log-probabilities alone already provide a strong signal (0.616 Overall F_1), exceeding the best non-temporal baseline by 19 points and confirming that the bulk of the signal lies in the *shape* of the top- K distribution. Engineered features contribute an additional ~ 5 points, with the largest gains from H_{alts} (+3.1) and RankProxy (+1.0). Dropping the raw log-probability channels while keeping the full temporal encoder costs 3.2

Table 2: Feature ablation on HUB. Incremental rows add one engineered feature at a time on top of raw top-20 log-probabilities. The last row drops the raw log-probability channels but keeps the full temporal encoder; it isolates the raw channels’ marginal contribution, not the contribution of temporal modeling (see Table 1 for that comparison).

Variant	Avg. F1	Overall F1
Raw log-probs only	0.557	0.616
+ AvgLogP	0.562	0.618
+ H_{alts}	0.575	0.647
+ RankProxy	0.600	0.657
+ H_{overall}	0.614	0.660
Full (+ ΔH_{dec})	0.630	0.670
Full, no raw log-probs (temporal)	0.567	0.638

Table 3: Effect of K on HALT-L (LLaMA 3.1-8B log-probabilities). Performance saturates near $K = 20$.

K	Avg. F1	Overall F1	Top- K mass (12 LLMs)
1	0.546	0.593	57–65%
5	0.556	0.635	88–98%
10	0.604	0.658	93–99%
15	0.612	0.661	94–99%
20	0.630	0.670	95–99.7%

Overall F_1 points, showing the raw channels and the engineered features are complementary rather than redundant; this is a separate quantity from the 12–24 point non-temporal-vs-temporal gap in Table 1, not a competing estimate of it. Per-feature attribution is in Appendix D.

Ruling out a length confound. A natural alternative explanation is that HALT is really picking up on response length, which correlates with both uncertainty accumulation and task difficulty. Predicting the hallucination label from length alone gives an AUROC of 0.56 (barely above chance) on the HUB test set, vs. 0.72 from HALT-L’s predicted probability, and the two classes separate along the probability axis at every length scale examined—evidence against length as the dominant driver.

3.6 Why $K = 20$?

The choice of K determines how much of the predictive distribution HALT observes. Increasing K provides richer temporal signals but also increases extraction cost and storage. We study this trade-off in two ways: by varying K during HALT-L training, and by measuring the cumulative probability mass captured by top- K across diverse LLMs.

Performance increases monotonically with K from 0.593 to 0.670, tapering off beyond $K=10$.

Top-20 captures 95–99.7% of the predictive mass across 12 LLMs, so entropy- and rank-based features computed at $K=20$ closely approximate their full-vocabulary counterparts; beyond this point additional log-probabilities contribute negligible mass at increased serving cost. We adopt $K=20$ as the principled operating point, which also matches the limits exposed by most APIs.

4 Experiments

4.1 Setup

We evaluate HALT on the HUB benchmark (Section 2) and on the FAVA Annotations and RAGTruth subsets retained for comparison with prior published baselines. We compare against four families of methods: (i) *white-box* detectors that inspect internal model states (Sriramanan et al., 2024), (ii) *aggregate-statistics* baselines that reduce token-level uncertainty to scalar summaries (perplexity, H_{overall} , H_{alts} , ΔH_{dec}), (iii) the *span-based* Lettuce detector (Ádám Kovács and Recski, 2025), a fine-tuned ModernBERT-base model trained to predict hallucinated spans, and (iv) *single-pass alternatives* including SelfCheckGPT (Manakul et al., 2023) and LOS-Net (Bar-Shalom et al., 2025b).

Aggregate-statistics baselines reduce the per-token sequence to a scalar (mean, or maximum for RankProxy); Lettuce span predictions are converted to a sentence-level label by flagging any span with probability ≥ 0.5 . All thresholds are tuned on the HUB validation set and fixed for test.

Main-results tables report the mean over three seeds; per-cluster 95% CIs are ± 0.04 – 0.07 macro- F_1 on larger clusters and ± 0.05 – 0.12 on the smallest held-out clusters, where rankings should be read with corresponding caution.

Why macro- F_1 . HUB clusters exhibit hallucination ratios from $\sim 30\%$ to $\sim 95\%$. Under this skew, micro-averaging is dominated by the large general-purpose clusters and masks failures on smaller, more imbalanced reasoning clusters where detection matters most; a detector predicting the majority class on World Knowledge (80% hallucinated) achieves high accuracy while contributing nothing. Macro- F_1 weights each cluster equally and penalizes errors from both classes, making it the appropriate primary metric for cross-capability generalization. Throughout, *per-cluster* scores are macro- F_1 within a cluster, *Overall* is macro- F_1

over the pooled test set, and Avg. is the unweighted mean of per-cluster scores.

Evaluation regime. HUB results (Tables 4–5) use *teacher-forced* log-probabilities: each labeled response is fed through the target LLM and its top-20 log-probabilities recorded. Because HUB’s responses originate from heterogeneous sources, teacher-forcing is what makes a controlled, comparable log-probability signal obtainable across all clusters under a single target LLM. HALT thus detects whether a response is inconsistent with the target LLM’s distribution over faithful text—a detection target directly relevant to the common setting of scoring outputs from a model one does not control. The single-pass alternatives in Section 4.6 instead use each LLM’s own free-running generations; that HALT remains effective there provides evidence that the signal is not an artifact of the teacher-forced regime. We discuss the teacher-forced/self-generated distinction further in Limitations.

4.2 Main Results on HUB

Table 4 reports macro- F_1 across HUB’s ten capability clusters. HALT-L (trained on Llama 3.1-8B log-probabilities) achieves the highest overall macro- F_1 (67.0) and average-across-clusters macro- F_1 (63.0), winning 7 of 10 clusters. HALT-Q (trained on Qwen 2.5-7B log-probabilities) is consistently second across most clusters; it received only a light hyperparameter sweep (+3 Overall F_1 over direct transfer), a likely conservative estimate of its potential.

HALT’s wins are concentrated on the five reasoning clusters (Algorithmic, Symbolic, Mathematical, Commonsense, World Knowledge: +15 to +53 over Lettuce), held out from training for *both* methods—a symmetric comparison, not an in-domain advantage. On the general-purpose clusters the comparison is fair in the other direction: RAGTruth, Lettuce’s training set, is a subset of HALT’s training data, so both are in-domain on Data-to-Text, QA, and Summarization, where Lettuce benefits from ModernBERT’s pretraining on trillions of text tokens against HALT’s random initialization on log-probabilities alone. On the reasoning clusters the aggregate and span baselines collapse to near-constant-class prediction (e.g., five methods at 44.5 on World Knowledge): the comparison there is categorical rather than graded—HALT extracts usable signal where the others fail entirely.

Table 4: Macro- F_1 on HUB test clusters (teacher-forced log-probabilities). Aggregate baselines use Llama 3.1-8B; **bold/underline** mark best/second-best per cluster. HALT’s training covers the capabilities Lettuce and Text-ModernBERT were trained on (Data-to-Text, QA, Summarization, Chat); all five reasoning clusters (including Code Generation) and World Knowledge are held out from training for all methods. Text-ModernBERT is a ModernBERT-base classifier fine-tuned end-to-end on HUB’s own training split—same architecture family as Lettuce, but trained on HALT’s own training distribution rather than RAGTruth.

Cluster	PPL	H_{overall}	ΔH_{dec}	H_{alts}	Lettuce	Text-ModernBERT	HALT-L	HALT-Q
Algorithmic	24.2	24.2	26.4	24.5	24.2	43.1	76.8	32.7
Chat	35.1	34.6	37.4	36.5	41.5	<u>60.0</u>	60.2	58.6
Code Generation	43.1	38.1	<u>62.0</u>	66.7	38.1	46.2	47.7	39.7
Commonsense	33.3	32.0	35.2	34.3	41.1	<u>53.4</u>	56.7	41.3
Data-to-Text	39.2	39.2	42.2	39.2	83.4	60.1	72.9	<u>73.0</u>
Mathematical	44.1	42.0	<u>67.0</u>	61.6	41.8	54.1	72.7	62.9
QA	28.3	25.7	49.4	43.4	77.3	71.0	<u>74.1</u>	68.8
Summarization	24.5	24.5	45.0	33.0	59.7	59.6	<u>66.9</u>	70.8
Symbolic	24.6	24.6	24.9	24.6	33.4	<u>58.6</u>	65.4	49.8
World Knowledge	44.5	44.5	44.5	44.5	44.5	52.8	76.9	<u>58.5</u>
Overall	34.0	32.9	48.0	42.9	<u>64.0</u>	61.9	67.0	62.7
Avg. across clusters	34.1	32.9	43.4	40.8	48.5	<u>55.9</u>	63.0	55.6

Isolating modality from training distribution.

The comparison against Lettuce leaves one variable confounded: modality (text vs. log-probabilities) *and* training distribution (RAGTruth vs. HUB). To isolate modality alone, we add **Text-ModernBERT**, a ModernBERT-base classifier fine-tuned end-to-end on HUB’s own training split—the same four clusters HALT trains on. It is closely matched to HALT-L on Chat (60.0 vs. 60.2) but trails on Data-to-Text, QA, and Summarization (Table 4), and the gap widens further on every held-out cluster (+1.5 to +33.7 macro- F_1). With modality now the only variable, the log-probability signal’s advantage over surface text holds at every capability level and is largest on the clusters neither model trained on.

4.3 Comparison on FAVA and RAGTruth

We additionally report results on the FAVA Annotations and RAGTruth test sets in isolation (Table 5).

On FAVA, HALT-L leads macro- F_1 (61.6) by 17 points over the next-best aggregate baseline; the class imbalance (67% positives) collapses Lettuce’s vanilla F_1 (80.7) to 40.3 macro- F_1 , reflecting a bias toward predicting hallucinations. HALT-L remains balanced despite being $1,400\times$ smaller than FAVA Model and $30\times$ smaller than Lettuce. On RAGTruth, Lettuce wins across all metrics—unsurprising, since it is trained on RAGTruth only—while HALT-L ranks second on every metric without dedicated fine-tuning.

Table 5: Performance on FAVA Annotations and RAGTruth test sets. Macro- F_1 is omitted (–) where not reported by prior baselines. Lettuce is trained on RAGTruth; the FAVA Model is a 7B fine-tuned Llama paired with a retriever (approximately $1,400\times$ HALT’s parameter count).

Method	FAVA Annotations			RAGTruth		
	AUROC	F_1	Mac- F_1	AUROC	F_1	Mac- F_1
<i>White-Box</i>						
LLM-Check (Attn)	68.2	70.5	–	58.3	57.2	–
<i>Aggregate Statistics</i>						
H_{overall}	53.9	80.0	40.3	37.0	51.8	25.9
H_{alts}	48.9	80.0	44.0	66.3	54.0	37.2
ΔH_{dec}	50.9	80.0	<u>44.2</u>	65.7	56.2	47.3
PPL	46.1	80.0	41.8	63.0	51.8	26.0
<i>Black/Grey-Box</i>						
FAVA Model (7B)	53.3	<u>79.9</u>	–	–	–	–
Lettuce	45.8	80.7	40.3	82.6	74.5	80.0
LLM-Check (Hidden)	57.1	65.4	–	57.2	47.5	–
HALT-L (ours)	<u>61.3</u>	77.9	61.6	<u>70.7</u>	<u>59.0</u>	<u>65.7</u>

4.4 Cross-Scale Generalization

A natural question is whether HALT’s architecture and training procedure depend on a specific LLM size or family. We train independent HALT detectors on log-probabilities from eight LLM families spanning 360M to 70B parameters, using identical architecture and (with one exception noted below) identical hyperparameters. Table 6 summarizes the results.

HALT achieves stable performance (Overall F_1 between 0.59 and 0.67) across an order of magnitude of scale and three architectural families (SmolLM (Allal et al., 2024), Llama (Grattafiori et al., 2024), Qwen (Yang et al., 2025)) using a single hyperparameter set tuned only on HALT-L; the seven other LLMs inherit these directly, a likely conservative estimate given that a brief

Table 6: HALT trained on log-probabilities from eight LLMs ranging from 360M to 70B parameters. All models use HALT-L’s hyperparameters; only HALT-Q received a light hyperparameter sweep (+3 Overall F_1).

LLM	Params	Overall F_1	Avg. F_1
SmolLM	360M	0.593	0.527
SmolLM	1.7B	0.609	0.539
Llama 3.2	3B	0.628	0.560
HALT-Q (Qwen 2.5)	7B	0.627	0.556
HALT-L (Llama 3.1)	8B	0.670	0.630
Qwen 3	14B	0.625	0.526
Qwen 3	32B	0.641	0.555
Llama 3.1	70B	0.659	0.595

HALT-Q sweep alone yielded +3 Overall F_1 . The empirical claim is modest and important: *the HALT recipe transfers across scales without per-LLM re-engineering*.

4.5 Cross-Model Transfer is Limited

A separate question is whether a HALT detector trained on one LLM’s log-probabilities can be applied directly to a different LLM’s log-probabilities at inference. We test this by evaluating HALT-L on Qwen 2.5-7B log-probabilities and HALT-Q on Llama 3.1-8B log-probabilities. HALT-L’s Overall F_1 drops from 0.670 (on Llama log-probs) to 0.540 (on Qwen log-probs), and HALT-Q drops from 0.627 to 0.51. Both remain above the constant-class baseline (0.327) and the weighted-random baseline (0.507) but substantially below in-distribution performance.

This reflects that calibration profiles—entropy ranges, uncertainty distributions, and temporal patterns around hallucination boundaries—are not directly shared across LLM families. In deployment, a separate HALT detector must be trained per target LLM, a one-time cost of minutes of compute on a single GPU; we list this in Limitations (Section 6).

4.6 Comparison with Single-Pass Alternatives

Two recent methods target a similar deployment niche to HALT: SelfCheckGPT (Manakul et al., 2023), based on multi-sample self-consistency, and LOS-Net (Bar-Shalom et al., 2025b), which uses richer top- K distributions. Unlike all preceding experiments, which use teacher-forced log-probabilities over labeled responses, these two comparisons use each target LLM’s (Llama 3.1 8B) own free-running generations, following each method’s published evaluation setup.

Against SelfCheckGPT-NLI (Table 7), HALT-

Table 7: HALT-L (Llama 3.1-8B, free generation) against a wider set of black-box, white-box, and multi-sample detectors on CoQA, SQuAD, and XSUM (AUROC). *Single gen.* marks methods that require only one generation pass. Non-HALT numbers are as reported in Bazarova et al. (2026); HALT-L was not trained on any of these three datasets.

Method	Access	Single gen.	CoQA	SQuAD	XSUM
SelfCheckGPT-NLI	Black-box	×	0.91	0.65	0.68
Semantic entropy	Black-box	×	0.78	0.55	0.47
EigenScore	White-box	×	0.80	0.45	0.50
HaloScope	White-box	✓	0.67	0.84	0.55
LLM-Check	White-box	✓	0.54	0.49	0.52
ReDEEP	White-box	✓	0.58	0.39	0.62
TOHA	White-box	✓	<u>0.84</u>	<u>0.87</u>	<u>0.65</u>
HALT-L (ours)	Grey-box	✓	0.65	0.90	0.59

Table 8: HALT vs. LOS-Net (AUROC). HALT-L was not trained on HotPotQA or MOVIES specifically. LOS-Net in-domain is trained on the evaluation dataset using top-50 log-probabilities; LOS-Net cross-domain is trained on MOVIES/IMDB and tested on HotPotQA using top-1000 log-probabilities.

Dataset	HALT-L (top-20)	LOS-Net (in-dom.) (top-50)	LOS-Net (cross) (top-1000)
HotPotQA	0.65	0.69	0.635–0.67
MOVIES	0.66	0.77	0.67–0.75

L outperforms by 25 AUROC points on SQuAD using $20\times$ fewer generations and no auxiliary NLI model, but loses on XSUM by 9 points—consistent with multi-sample consistency capturing summarization-style hallucinations that single-pass methods miss, while single-pass temporal modeling wins where forced-choice grounding matters more than stylistic consistency. *This is a deliberate cost-accuracy trade-off, not a uniform claim of dominance.* Against the wider set of reference numbers from Bazarova et al. (2026), HALT-L’s SQuAD AUROC (0.90) exceeds every method shown, including TOHA (0.87), using only a single forward pass and top-20 log-probabilities; on CoQA and XSUM it is not strongest, consistent with the same trade-off—trading some peak accuracy for requiring neither internals nor resampling.

Against LOS-Net (Table 8), LOS-Net achieves stronger in-domain performance using top-50 log-probabilities and dataset-specific training, while HALT-L remains within range of LOS-Net’s cross-domain numbers on HotPotQA using $50\times$ fewer log-probability values and no dataset-specific training. HALT thus occupies a distinct design point: lower K , single-pass, at some cost in peak in-domain accuracy.

4.7 Efficiency

A 5M-parameter detector matches or exceeds a 150M-parameter alternative: HALT is approximately $30\times$ smaller than Lettuce, and strictly cheaper than multi-sample methods such as SelfCheckGPT-NLI, which requires 20 additional generations through the target LLM *plus* pairwise inference through an auxiliary BERT-style NLI model. HALT instead operates on log-probabilities emitted as a byproduct of the original generation, paid once per response. We extend discussion about efficiency in Appendix G.

5 Related Work

White-box methods. A first line of work inspects internal model states: LLM-Check (Sriraman et al., 2024) aggregates attention/hidden-activation statistics, INSIDE (Chen et al., 2024) uses the eigenvalue spectrum of hidden-state covariance, ACT-ViT (Bar-Shalom et al., 2025a) probes activation tensors directly, and Azaria and Mitchell (2023) train classifiers on activations to detect statements the model “knows” are false. Message-passing over attention graphs (Frasca et al., 2026), hidden-state trajectory tracking (Zhang et al., 2025), and unsupervised internal-state signals (Su et al., 2024) extend this line further. These methods require white-box access and typically per-dataset layer tuning.

Black-box and multi-sample methods. A second line operates without internal access by leveraging multiple generations or auxiliary models. SelfCheckGPT (Manakul et al., 2023) samples 10–20 responses and flags hallucinations via cross-sample inconsistency; semantic entropy (Farquhar et al., 2024) clusters generations by meaning-equivalence and uses cluster-level entropy, an idea Vipulanandan et al. (2026) extend with a quantum-tensor-network uncertainty estimate at finer granularity, at the cost of the same external clustering model. Fact-level decomposition (Sawczyn et al., 2026), consistency-under-uncertain-expression checks (Joo et al., 2025), and cross-model verification (Xue et al., 2026) refine self-consistency further. Detection and correction (Mishra et al., 2024) and retrieval-based methods (Friel et al., 2025; Niu et al., 2024) compare against external evidence instead. All introduce additional inference cost, monetary cost, or external dependencies.

Aggregate-statistics methods. A third line of work, closer to HALT in input modality, derives scalar uncertainty summaries from the LLM’s output distribution: Varshney et al. (2023) use minimum token-level log-probability as a confidence score, Quevedo et al. (2024) aggregate per-token probabilities through hand-crafted reductions, and standard perplexity-based detection averages log-probabilities across the sequence. These are cheap but collapse the response to a single scalar, discarding the temporal structure that Section 3.4’s ablation shows carries 12–24 macro- F_1 points of additional signal.

LOS-Net. Closest to HALT is LOS-Net (Bar-Shalom et al., 2025b), which also identifies the top- K output distribution as a useful hallucination signal, arrived at independently since LOS-Net pre-dates this paper. LOS-Net differs in three respects: much larger K (top-50 in-domain, top-1000 cross-domain), an attention-based static encoder rather than an explicit temporal sequence, and in-domain dataset-specific training, versus HALT’s $K=20$, recurrent temporal encoder, and cross-capability evaluation without per-dataset training. The two findings are complementary: both show output distributions carry hallucination signal beyond aggregate statistics, and HALT shows this signal is recoverable at much lower K when temporal structure is modeled explicitly.

6 Conclusion

We introduced HALT, a grey-box hallucination detector that treats the top-20 log-probabilities of an LLM generation as a multivariate time series and processes them with a small bidirectional GRU. On HUB—ten capability clusters spanning factual and logical failure modes—HALT matches or surpasses a $30\times$ larger ModernBERT detector on macro- F_1 , wins on reasoning clusters where surface-form cues are sparse, and stays competitive against multi-sample and high- K alternatives at lower cost. Architecture and feature ablations isolate the contributions of temporal modeling, and the cross-scale evaluation shows the recipe transfers across eight LLM families.

These results suggest that the temporal shape of top- K log-probabilities is a richer hallucination signal than the aggregate-statistics literature has assumed. Promising directions include a online detection during generation, and span-level localization.

Limitations

While HALT achieves strong performance across diverse LLMs and capability clusters, several limitations should be acknowledged.

Model-specific detectors. HALT’s signal depends on the calibration profile of a specific LLM. Cross-model transfer experiments (Section 4.5) show substantial performance drops when applying a detector trained on one LLM to log-probabilities from another. In deployment this means a separate detector must be trained per target LLM, a one-time cost on the order of minutes of single-GPU compute, but a real engineering overhead.

Annotation provenance. HUB inherits labels from its source datasets and introduces no new annotations. FAVA, RAGTruth, and HaluEval contribute human or crowd-sourced labels, while CriticBench labels come from a hybrid pipeline combining rule-based heuristics, GPT-4 judging, and human adjudication on disagreements. Our manual inspection of CriticBench samples found approximately 90% agreement with the original labels, but the underlying reliance on LLM-assisted labeling for the reasoning clusters remains a constraint on benchmark fidelity.

Validation exposure and small held-out clusters. Model selection uses a pooled validation metric spanning every cluster, including the six held out from training, so we avoid calling them strictly zero-shot. This is sharpest for World Knowledge: only 12 faithful validation examples, and a hallucination rate shifting from 95% (validation) to 80% (test). Since thresholds are tuned globally rather than per cluster (Section 4.1), these 12 examples do not independently fix its cutoff—but rankings on this cluster should still be read with the wide confidence intervals from Section 4.1 in mind.

English-only evaluation. All datasets used in HUB are English. HALT’s performance on multilingual or code-switched text remains unevaluated, and the temporal calibration patterns we exploit may differ qualitatively in non-English settings, particularly for languages with different tokenization characteristics.

Hyperparameter tuning provenance. The HALT architecture and training hyperparameters were tuned on HALT-L (Llama 3.1-8B). HALT-Q received a light additional sweep that yielded +3

Overall F_1 ; the remaining six LLMs in the cross-scale evaluation use HALT-L’s hyperparameters without modification. Their reported numbers are likely conservative estimates of achievable performance.

Teacher-forced evaluation. HUB results use teacher-forced log-probabilities, whereas deployment scores a model’s own free-running generations, where the distribution differs. Forced tokens are not systematically out-of-distribution: mean rank 2.13, top-20 capturing 99.0% of predictive mass, only 2.07% falling outside the top-19. Teacher-forcing also decouples the scored text from the scoring model, enabling the cross-scale comparison in Section 4.4 and matching a real deployment case (scoring text the model did not generate). Section 4.6 reports free-generation results as further evidence of transfer; a systematic free-generation HUB evaluation—requiring re-annotation at scale rather than reusing existing labels—is left to future work.

No closed-source judge comparison. We do not compare against LLM-as-judge baselines using closed-source frontier models (GPT-4, Claude). Such comparisons would require API access at scale and were considered outside the scope of a deployment-oriented detector study, but they represent a relevant point of reference.

Ethical Considerations

Intended use. HALT is designed as an assistive signal to support human verification of LLM outputs, not as a replacement for it. A detector operating below 100% accuracy will produce both false positives (flagging faithful outputs) and false negatives (missing hallucinations). Deployment in safety-critical settings should retain human review of flagged and unflagged outputs alike, and consumer-facing applications should communicate detector uncertainty rather than presenting hallucination scores as ground truth.

Risk of over-reliance. Strong empirical performance on benchmarks does not guarantee strong real-world performance, especially under distribution shift. There is a documented tendency for users to over-trust automated quality signals once they are introduced into a workflow. We caution against deploying HALT—or any current hallucination detector—in settings where users may interpret a low hallucination score as a guarantee of

correctness.

Data licensing and reproducibility. HUB inherits the licensing terms of its source datasets (FAVA, RAGTruth, HaluEval, CriticBench). We will release HUB’s processing scripts and split meta-data, and trained HALT checkpoints, but cannot redistribute source-dataset text where licensing restricts it. Replication requires obtaining the source datasets through their original distribution channels.

References

- Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Leandro von Werra, and Thomas Wolf. 2024. Smollm - blazingly fast and remarkably powerful.
- Amos Azaria and Tom Mitchell. 2023. The internal state of an llm knows when it’s lying. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 967–976.
- Guy Bar-Shalom, Fabrizio Frasca, Yaniv Galron, Yftah Ziser, and Haggai Maron. 2025a. [Beyond token probes: Hallucination detection via activation tensors with act-vit](#). *Preprint*, arXiv:2510.00296.
- Guy Bar-Shalom, Fabrizio Frasca, Derek Lim, Yoav Gelberg, Yftah Ziser, Ran El-Yaniv, Gal Chechik, and Haggai Maron. 2025b. [Beyond next token probabilities: Learnable, fast detection of hallucinations and data contamination on llm output distributions](#). *Preprint*, arXiv:2503.14043.
- Alexandra Bazarova, Andrei Volodichev, Aleksandr Yugay, Andrey Shulga, Alina Ermilova, Konstantin Poley, Julia Belikova, Rauf Parchiev, Dmitry Simakov, Maxim Savchenko, Andrey Savchenko, Serguei Barannikov, and Alexey Zaytsev. 2026. [Hallucination detection in llms with topological divergence on attention graphs](#). *Preprint*, arXiv:2504.10063.
- Chao Chen, Kai Liu, Ze Chen, Yi Gu, Yue Wu, Mingyuan Tao, Zhihang Fu, and Jieping Ye. 2024. [Inside: LLMs’ internal states retain the power of hallucination detection](#). *Preprint*, arXiv:2402.03744.
- Kyunghyun Cho, Bart van Merriënboer, Çağlar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. [Learning phrase representations using RNN encoder-decoder for statistical machine translation](#). *CoRR*, abs/1406.1078.
- Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. 2024. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630:625–630.
- Fabrizio Frasca, Guy Bar-Shalom, Yftah Ziser, and Haggai Maron. 2026. [Neural message-passing on attention graphs for hallucination detection](#). *Preprint*, arXiv:2509.24770.
- Robert Friel, Masha Belyi, and Atindriyo Sanyal. 2025. [Ragbench: Explainable benchmark for retrieval-augmented generation systems](#). *Preprint*, arXiv:2407.11005.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. [The llama 3 herd of models](#). *Preprint*, arXiv:2407.21783.
- Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Delong Chen, Wenliang Dai, Ho Shu Chan, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):236:1–236:38.
- Seongho Joo, Kyungmin Min, Jahyun Koo, and Kyomin Jung. 2025. [Black-box hallucination detection via consistency under the uncertain expression](#). *Preprint*, arXiv:2509.21999.
- Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.
- Tian Lan, Wenwei Zhang, Chen Xu, Heyan Huang, Dahua Lin, Kai Chen, and Xian-Ling Mao. 2024. CriticBench: Evaluating large language models as critic. *arXiv preprint arXiv:2402.13764*.
- Junyi Li, Xiaoxue Cheng, Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2023. HaluEval: A large-scale hallucination evaluation benchmark for large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6449–6464.
- Potsawee Manakul, Adian Liusie, and Mark J. F. Gales. 2023. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Abhika Mishra, Akari Asai, Vidhisha Balachandran, Yizhong Wang, Graham Neubig, Yulia Tsvetkov, and Hannaneh Hajishirzi. 2024. Fine-grained hallucination detection and editing for language models. *arXiv preprint arXiv:2401.06855*.
- Cheng Niu, Yuanhao Wu, Juno Zhu, Siliang Xu, Kashun Shum, Randy Zhong, Juntong Song, and Tong Zhang. 2024. RAGTruth: A hallucination corpus for developing trustworthy retrieval-augmented language models. *arXiv preprint arXiv:2401.00396*.

Ernesto Quevedo, Jorge Yero, Rachel Koerner, Pablo Rivas, and Tomas Cerny. 2024. Detecting hallucinations in large language model generation: A token probability approach. *arXiv preprint arXiv:2405.19648*.

Albert Sawczyn, Jakub Binkowski, Denis Janiak, Bogdan Gabrys, and Tomasz Kajdanowicz. 2026. [Fact-selfcheck: Fact-level black-box hallucination detection for llms](#). *Preprint*, arXiv:2503.17229.

Gaurang Sriramanan, Siddhant Bharti, Vinu Sankar Sadasivan, Shoumik Saha, Priyatham Kattakinda, and Soheil Feizi. 2024. [Llm-check: Investigating detection of hallucinations in large language models](#). In *Advances in Neural Information Processing Systems*, volume 37, pages 34188–34216. Curran Associates, Inc.

Weihang Su, Changyue Wang, Qingyao Ai, Yiran HU, Zhijing Wu, Yujia Zhou, and Yiqun Liu. 2024. [Unsupervised real-time hallucination detection based on the internal states of large language models](#). *Preprint*, arXiv:2403.06448.

Neeraj Varshney, Weixin Yao, Hao Zhang, Jinghui Chen, and Dong Yu. 2023. A stitch in time saves nine: Detecting and mitigating hallucinations of llms by validating low-confidence generation. *arXiv preprint arXiv:2307.XXXXX*.

Pragatheeswaran Vipulanandan, Kamal Premaratne, and Dilip Sarkar. 2026. [Semantic uncertainty quantification of hallucinations in llms: A quantum tensor network based method](#). *Preprint*, arXiv:2601.20026.

Yihao Xue, Kristjan Greenewald, Youssef Mroueh, and Baharan Mirzasoleiman. 2026. [Verify when uncertain: Beyond self-consistency in black box hallucination detection](#). *Preprint*, arXiv:2502.15845.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. [Qwen3 technical report](#). *Preprint*, arXiv:2505.09388.

Zhenliang Zhang, Xinyu Hu, Huixuan Zhang, Junzhe Zhang, and Xiaojun Wan. 2025. [Icr probe: Tracking hidden state dynamics for reliable hallucination detection in llms](#). *Preprint*, arXiv:2507.16488.

Ádám Kovács and Gábor Recski. 2025. [Lettucedetect: A hallucination detection framework for rag applications](#). *Preprint*, arXiv:2502.17125.

A HUB: Extended Details

This appendix provides extended documentation of the HUB benchmark referenced in Section 2.

A.1 Per-Dataset Composition

Table 10 reports the per-source-dataset distribution within each HUB cluster. Asterisks (*) mark clusters held out from training. The breakdown clarifies which source datasets contribute to which clusters and at what scale.

A.2 Response Length Distribution

Response lengths in HUB vary substantially across clusters, reflecting the diversity of underlying tasks. Algorithmic and Commonsense reasoning clusters skew short (often single-sentence answers to multiple-choice or short-form questions), while World Knowledge and Data-to-Text responses are markedly longer (up to 246-word averages). This variability is methodologically relevant: a hallucination detector that exploits token-level uncertainty patterns must be robust across at least an order of magnitude in response length. Table 9 in the main text reports cluster-level mean response lengths; full per-cluster length distributions (quartiles, maxima) are computed identically across train, validation, and test splits and made available alongside HUB’s release.

A.3 Taxonomy Alignment with CriticBench

CriticBench (Lan et al., 2024) originally organizes samples by failure-mode rather than by LLM capability. To map CriticBench to HUB’s capability-based clusters, we apply the following correspondences:

- **Algorithmic** $\leftarrow$ *ObjectCounting, Tracking-Shuffled-Objects, Reasoning-About-Colored-Objects* (numerical and stateful reasoning over discrete elements).
- **Mathematical** $\leftarrow$ *GSM8K-Trace, MultiArith-Trace, Algebra* (arithmetic and symbolic algebra reasoning chains).
- **Symbolic** $\leftarrow$ *Logical-Deduction, Boolean-Expressions, Causal-Judgment* (formal logical inference and chained deduction).
- **Commonsense** $\leftarrow$ *StrategyQA-Trace, Sports-Understanding, Date-Understanding* (everyday reasoning grounded in world knowledge).
- **Code Generation** $\leftarrow$ CriticBench code-execution subset (programs whose execution traces are verifiably correct or incorrect).

Table 9: HUB cluster-level statistics. *Size* is the number of responses; *Ratio* is the proportion labeled hallucinated; *Len* is the average response length in words. Asterisks (*) mark clusters held out from training. A dash (–) indicates the cluster contributes no data to that split.

Task Cluster	Train		Validation			Test		
	Size	Ratio	Size	Ratio	Len	Size	Ratio	Len
Algorithmic*	–	–	32	50.00%	29.97	250	32.00%	55.35
Chat	11,278	39.97%	1,991	39.98%	35.28	1,278	52.03%	86.69
Code Generation*	–	–	164	65.24%	158.64	300	61.67%	39.82
Commonsense*	–	–	229	32.31%	36.57	900	47.00%	28.11
Data-to-Text	2,759	50.02%	487	49.90%	157.83	900	64.33%	156.70
Mathematical*	–	–	300	46.67%	42.03	1,004	72.41%	73.45
Question Answering	35,377	53.20%	1,885	50.34%	35.19	1,400	29.29%	72.16
Summarization	10,594	50.08%	1,870	50.05%	73.78	1,400	32.43%	92.14
Symbolic*	–	–	146	41.10%	66.86	500	32.60%	52.99
World Knowledge*	–	–	238	94.96%	139.03	182	80.22%	246.02
Overall	60,008	50.00%	7,342	48.31%	–	8,114	47.27%	–

Table 10: Per-dataset distribution within each HUB cluster, broken down by source dataset and split. *Hallu* indicates samples labeled hallucinated; *Correct* indicates faithful samples. Asterisks (*) mark clusters not used for training.

Cluster	Source	Train			Validation			Test		
		Hallu	Correct	Total	Hallu	Correct	Total	Hallu	Correct	Total
Algorithmic*	CriticBench	–	–	–	16	16	32	80	170	250
Chat	FAVA (OpenAssistant)	850	1,278	2,128	150	226	376	145	209	354
	FAVA (NoRobots)	1,264	1,898	3,162	224	336	560	217	309	526
	HaluEval (Dialog)	2,393	3,595	5,988	422	633	1,055	303	95	398
Code Generation*	CriticBench	–	–	–	107	57	164	185	115	300
Commonsense*	CriticBench	–	–	–	74	155	229	423	477	900
Data-to-Text	RAGTruth (D2T)	1,380	1,379	2,759	243	244	487	579	321	900
Mathematical*	CriticBench	–	–	–	140	160	300	727	277	1,004
QA	FAVA (Knowledge)	480	720	1,200	85	127	212	82	117	199
	RAGTruth (QA)	1,380	1,379	2,759	243	244	487	132	367	499
	HaluEval (QA)	12,591	18,827	31,418	619	567	1,186	196	506	702
Summarization	RAGTruth (Sum)	1,380	1,379	2,759	244	243	487	132	367	499
	HaluEval (Sum)	3,927	3,908	7,835	692	691	1,383	322	579	901
Symbolic*	CriticBench	–	–	–	60	86	146	163	337	500
World Knowledge*	CriticBench	–	–	–	226	12	238	146	36	182
Overall		29,967	34,363	60,008	3,545	3,797	7,342	3,832	4,282	8,114

World Knowledge is not part of this mapping: unlike the five reasoning clusters above, it draws from FAVA Annotations and WikiBio-WebNLG rather than CriticBench (Table 10).

This mapping preserves CriticBench’s distinction between reasoning errors and final-answer errors: a sample is labeled hallucinated if *either* the reasoning trace contains an error *or* the final answer is incorrect. We use CriticBench solely for validation and testing. To prevent leakage, source datasets are partitioned across shards such that no dataset contributing to a validation cluster also contributes to the corresponding test cluster.

A.4 Manual Review Protocol

To verify alignment between CriticBench’s hybrid labeling pipeline (rule-based heuristics + GPT-4

judging + human adjudication) and our operational definition of logical hallucination, we conducted an informal review of CriticBench samples drawn from each reasoning cluster.

Procedure. For each reasoning cluster, we drew samples uniformly from the validation set. For each sample, we recorded (i) the prompt, (ii) the model response, (iii) the CriticBench label (hallucinated / faithful), and (iv) the reviewer’s independent judgment under HUB’s definition (a response is hallucinated if any reasoning step or the final answer is incorrect with respect to the prompt’s task specification).

Findings. Agreement between reviewer judgment and CriticBench labels was approximately 90% across the reviewed samples. Disagreements

Table 11: CriticBench sub-datasets underlying HUB’s five reasoning clusters. Train is omitted since CriticBench contributes only to Validation and Test (Section 2); these totals match the corresponding rows of Table 9.

Cluster	CriticBench Sub-dataset	Validation			Test		
		Hallu.	Corr.	Total	Hallu.	Corr.	Total
Algorithmic	object_counting	–	–	–	80	170	250
	repeat_copy	16	16	32	–	–	–
Code Generation	humaneval	107	57	164	–	–	–
	mbpp	–	–	–	185	115	300
Commonsense	ambigqa	–	–	–	155	145	300
	csqa	–	–	–	94	206	300
	hotpotqa	–	–	–	174	126	300
	strategyqa	74	155	229	–	–	–
Mathematical	aqua	–	–	–	180	74	254
	gsmlrk	–	–	–	134	116	250
	math	–	–	–	413	87	500
	tabmwp	140	160	300	–	–	–
Symbolic	colored_object	–	–	–	87	163	250
	date	–	–	–	76	174	250
	penguins	60	86	146	–	–	–
Total		397	474	871	1,578	1,376	2,954

clustered in two patterns: (i) samples where the final answer is correct but the reasoning trace contains a transient error later self-corrected, and (ii) samples involving ambiguous prompts where multiple plausible interpretations exist. Both cases are documented in the source datasets as edge cases of automated labeling. We retain CriticBench’s original labels rather than relabeling, since the reviewer-CriticBench agreement is high and full relabeling would lose comparability with prior work on CriticBench.

Limitations of this review. The review was conducted by the authors rather than independent annotators, and inter-annotator agreement statistics in the formal sense (e.g., Cohen’s κ) are not reported. We acknowledge this in Section 6 and frame HUB’s labels as inherited from source datasets rather than as new contributions.

B Method: Extended Formalization

This appendix provides extended formalization and derivations supporting Section 3.

B.1 Design Intuitions

The HALT architecture was motivated by four design intuitions about how hallucinations interact with LLM output distributions. These intuitions guided architectural choices but are *not* directly measured as formal claims; we present them here for transparency and as starting points for future theoretical work.

Intuition 1: Calibration profiles are model-specific. For a given prompt context c and target token y_t , the relationship between an LLM’s confi-

dence and its empirical accuracy is model-specific. Formally, the probability a model $\mathcal{M}_\theta$ assigns to token y_t in context c relates to the “true” generative probability through some model-specific calibration function:

$$p_\theta(y_t | c) = b_\theta(p_*(y_t | c)), \quad (3)$$

where $b_\theta : [0, 1] \rightarrow [0, 1]$ captures the LLM-specific calibration profile (over- or under-confidence patterns, sharpness of the predictive distribution, mass distribution among alternatives). This motivates training a separate HALT detector per LLM rather than expecting cross-model transfer, and is consistent with the empirical drop in cross-model transfer reported in Section 4.5.

Intuition 2: Temporal context disambiguates calibration. Aggregate statistics over $\ell_{1:T}$ (mean log-probability, perplexity) discard information about *when* uncertainty occurs in the response. Two responses with identical mean log-probability can have qualitatively different temporal profiles: one with uniformly moderate uncertainty, another with sharp uncertainty spikes localized around specific tokens. The latter is plausibly more diagnostic of hallucination. This motivates the use of a sequential encoder (Bi-GRU) over a pooling-based aggregator, and is supported empirically by the architecture ablation in Section 3.4.

Intuition 3: Bidirectionality captures retrospective context. A token that looks confident in isolation may be retrospectively recognizable as hallucinated when subsequent tokens reveal that an entity is fabricated or an argument is invalid. Bidirectional encoding gives the detector access to this retrospective signal. This is supported empirically by the Bi-GRU > Uni-GRU gap (Table 1).

Intuition 4: Hallucination signal is concentrated, not diffuse. Hallucinations are localized phenomena: a particular fabricated entity, a particular invalid reasoning step. Averaging across all timesteps dilutes this localized signal. This motivates Top- q pooling (averaging over only the most informative timesteps) over mean pooling, with the pooling ablation in Appendix E providing supporting evidence.

B.2 Truncated Top- K Distribution

Since we observe only the top- K log-probabilities, computing entropy-based features requires constructing a renormalized truncated distribution. Let

$\ell_t = (\log p_t^{(0)}, \dots, \log p_t^{(K-1)})$ be the observed log-probabilities at timestep t . The truncated distribution is

$$\tilde{p}_t^{(k)} = \frac{\exp(\log p_t^{(k)})}{\sum_{j=0}^{K-1} \exp(\log p_t^{(j)})}, \quad k = 0, \dots, K-1. \quad (4)$$

For numerical stability, we apply the standard log-sum-exp trick:

$$\tilde{p}_t^{(k)} = \exp(\log p_t^{(k)} - \text{LSE}(\ell_t)), \quad (5)$$

where $\text{LSE}(\ell_t) = \log \sum_j \exp(\log p_t^{(j)})$. All entropy-based features in Section 3.2 are computed on $\tilde{p}_t$ rather than on the un-normalized top- K probabilities, with empirical justification in Section 3.6 (top-20 captures 95–99.7% of full-vocabulary probability mass across 12 LLMs, so $\tilde{p}_t$ closely approximates the full-vocabulary distribution).

B.3 Engineered Features: Full Definitions

We provide the full definitions of the five engineered features summarized in Section 3.2.

Average log-probability (AvgLogP).

$$\text{AvgLogP}(\ell_t) = \frac{1}{K} \sum_{k=0}^{K-1} \log p_t^{(k)}. \quad (6)$$

This captures the overall sharpness of the top- K region: when AvgLogP is high, the top- K tokens are collectively concentrated; when low, the model is hedging mass across many alternatives.

Rank proxy (RankProxy). Defined as the position of the selected token among the top- K alternatives. When the selected token is the greedy choice, RankProxy = 0. Non-zero values indicate that the LLM selected a non-greedy alternative (under stochastic decoding) or that the labeled token was not the model’s preferred choice (under teacher-forcing).

Top- K entropy (H_{overall}).

$$H_{\text{overall}}(\ell_t) = - \sum_{k=0}^{K-1} \tilde{p}_t^{(k)} \log \tilde{p}_t^{(k)}. \quad (7)$$

This is the entropy of the renormalized top- K distribution, capturing aggregate indecision among the leading candidates.

Alternatives entropy (H_{alts}). Entropy computed over the top- K distribution *excluding* the selected token, renormalized over the remaining $K - 1$ tokens:

$$H_{\text{alts}}(\ell_t) = - \sum_{k \neq k^*} \tilde{q}_t^{(k)} \log \tilde{q}_t^{(k)}, \quad (8)$$

where k^* is the index of the selected token and $\tilde{q}_t$ is the renormalized distribution over the $K - 1$ alternatives. This isolates disagreement among competitors and is empirically the most informative single engineered feature (Table 2).

Decision-entropy delta (ΔH_{dec}). The temporal change in binary decision-entropy between the selected token’s probability and its strongest alternative:

$$\Delta H_{\text{dec}}(\ell_t) = H_{\text{bin}}(\tilde{p}_t^{(k^*)}) - H_{\text{bin}}(\tilde{p}_{t-1}^{(k_{t-1}^*)}), \quad (9)$$

where $H_{\text{bin}}(p) = -p \log p - (1 - p) \log(1 - p)$. This flags rapid certainty-to-uncertainty transitions across adjacent tokens.

B.4 Top- q Pooling

Given the bidirectional GRU output sequence $H \in \mathbb{R}^{T \times 512}$, Top- q pooling proceeds as follows. We compute the ℓ_2 norm of each timestep’s hidden state, $s_t = \|H_t\|_2$, and mask out padded positions. We select the top $\lceil q \cdot T_{\text{valid}} \rceil$ timesteps by score (with $q = 0.15$ and T_{valid} the number of non-padded positions) and average their hidden states to obtain a single pooled representation $h_{\text{pool}} \in \mathbb{R}^{512}$. This concentrates the classifier on the most informative regions of the sequence rather than diluting the signal across all tokens. A comparison against mean, max, and attention-based pooling is provided in Appendix E.

C Training Details

Optimization. HALT is trained with the Adam optimizer (learning rate 4.41×10^{-4} , weight decay 2.34×10^{-6}) using BCEWithLogitsLoss on the binary hallucination label. Batch size is 512. Training runs for up to 100 epochs with early stopping (patience 15 epochs on validation macro- F_1). Learning rate is reduced by factor 0.5 on plateau (patience 3, threshold 10^{-4}).

Architecture hyperparameters. The architectural hyperparameters were tuned on HALT-L using a small grid search on the HUB validation set: GRU hidden size $\in \{128, 256, 512\}$, GRU

depth $\in \{3, 5, 7\}$, dropout $\in \{0.2, 0.3, 0.4\}$, projection dimension $\in \{64, 128, 256\}$, Top- q value $\in \{0.05, 0.10, 0.15, 0.25\}$. The selected configuration (hidden 256, depth 5, dropout 0.4, projection 128, $q = 0.15$) is used for all cross-scale experiments without re-tuning, with the exception of HALT-Q, which received a light additional sweep that yielded +3 Overall F_1 relative to direct hyperparameter transfer.

Sequence handling. Variable-length log-probability sequences are processed using PyTorch’s packed sequences with a boolean attention mask. Padding tokens do not contribute to GRU hidden updates or to the Top- q pooling score computation.

Hardware and runtime. All training was performed on a single NVIDIA A100 40GB GPU. End-to-end training for a single HALT detector (60K training samples, 100 epoch budget with early stopping) completes in approximately 10-25 minutes depending on early-stopping triggers and average sequence length.

D Feature Importance: Extended Analysis

To probe which input features drive HALT’s predictions, we compute gradient $\times$ input attribution scores per feature dimension, averaged across the HUB validation set.

Table 12: Top-10 input features ranked by mean absolute gradient $\times$ input attribution (HALT-L, HUB validation). Engineered features and raw log-probability ranks both appear among the most important inputs.

Feature	Mean $ g \cdot x $
H_{alts}	0.0421
logprob ₁₅	0.0387
logprob ₄	0.0362
RankProxy	0.0349
logprob ₁₇	0.0331
H_{overall}	0.0318
logprob ₈	0.0304
AvgLogP	0.0289
logprob ₀	0.0276
ΔH_{dec}	0.0258

Three patterns are notable. First, the most informative engineered features (H_{alts} , RankProxy, H_{overall}) are consistent with the incremental feature ablation in Table 2, where these same features produced the largest individual gains. Second, mid-rank log-probabilities (logprob₁₅,

logprob₄, logprob₁₇) rank higher than the top-1 log-probability (logprob₀), suggesting that the *shape* of the alternative distribution carries more diagnostic signal than the selected token’s confidence alone. Third, the gap between the top feature and the 10th-ranked feature is modest ($\sim 1.6\times$), indicating that HALT’s predictions rely on a distributed combination of inputs rather than any single dominant signal.

E Pooling Ablation

We compare Top- q pooling against alternatives by replacing the pooling layer in HALT-L while keeping all other architectural choices and training hyperparameters fixed. Results (HUB validation macro- F_1) are:

- Mean pooling: -1.8 relative to Top- q ($q=0.15$).
- Max pooling: -2.4 relative to Top- q .
- Attention-based pooling (single-head): -0.6 relative to Top- q .
- Top- q pooling, $q \in \{0.05, 0.10, 0.25, 0.50\}$: all within ± 0.8 of $q=0.15$; performance degrades sharply at $q=1.0$ (equivalent to mean pooling).

Top- q pooling outperforms mean and max pooling, supporting the design intuition that hallucination signal is localized rather than diffuse (Appendix B.1, Intuition 4). The narrow gap with attention-based pooling suggests that the specific aggregation mechanism matters less than the choice to weight informative timesteps more heavily than uninformative ones; we adopt Top- q for its lower parameter count and absence of additional learned weights.

F Baseline Implementation Details

Aggregate-statistics baselines. For each aggregate-statistics baseline (H_{overall} , H_{alts} , ΔH_{dec} , PPL), we compute the relevant per-token feature across the full response and reduce to a single scalar per response. We use mean reduction for entropy-based features and perplexity, and maximum reduction for RankProxy, motivated by the intuition that a single high-rank selection event is informative even if surrounding tokens are confident. Decision thresholds are tuned on the HUB validation set to maximize macro- F_1 and held fixed for test evaluation.

Lettuce. We use the public LettuceDetect (Ádám Kovács and Recski, 2025) ModernBERT-base checkpoint trained on RAGTruth. Lettuce outputs token-level span predictions; we convert these to a sentence-level binary label by flagging a response as hallucinated if any predicted span has probability ≥ 0.5 . We verified that this threshold yields competitive performance on RAGTruth itself (consistent with reported numbers) before applying it to HUB.

LLM-Check. We use the published LLM-Check implementation (Sriramanan et al., 2024) on Llama 3.1-8B with default layer-selection and statistic-computation procedures. We report Attention and Hidden-state variants separately, consistent with the original paper.

SelfCheckGPT-NLI. We use SelfCheckGPT’s NLI-based variant (Manakul et al., 2023), which compares the original generation against 20 stochastically resampled responses using a fine-tuned NLI model (DeBERTa-large-mnli). The detection score is the mean per-sentence inconsistency probability across the 20 comparisons.

LOS-Net. In the Free-Generation experiments, we report numbers from the LOS-Net paper (Bar-Shalom et al., 2025b) as published rather than reproducing the method. The in-domain numbers correspond to LOS-Net trained on the evaluation dataset with top-50 log-probabilities; cross-domain numbers correspond to LOS-Net trained on MOVIES/IMDB and evaluated on HotPotQA with top-1000 log-probabilities. As for evaluation on HUB we re-implemented a faithful method to the LOS-Net manuscript.

FAVA Model. The FAVA hallucination detector (Mishra et al., 2024) is a Llama-7B model fine-tuned for hallucination span prediction in retrieval-grounded settings. We report results from the public FAVA evaluation as provided in the original paper.

G Computational Cost: Extended

HALT pipeline cost decomposition. HALT’s inference pipeline consists of three stages: (i) log-probability extraction from the target LLM during teacher-forcing (or as a byproduct of online generation), (ii) feature engineering ($\phi(\ell_t)$ for each timestep), and (iii) the Bi-GRU forward pass plus Top- q pooling and the linear head. Stage (i) is

shared with any output-distribution-based method and is the dominant cost; stages (ii) and (iii) together comprise the marginal detector cost.

Marginal detector cost. Stage (ii) is implemented as vectorized tensor operations over the $T \times K$ log-probability matrix and is dominated by elementwise arithmetic and softmax normalizations. Stage (iii) is a single Bi-GRU forward pass plus a small projection head; with 5M parameters and a typical response length of 100–200 tokens, the marginal cost is small relative to the LLM forward pass that produced the log-probabilities in the first place.

Comparison with Lettuce. Lettuce processes the full response through a ModernBERT-base encoder (150M parameters), which is roughly $30\times$ HALT’s parameter count and requires its own forward pass over the response after generation completes. HALT’s marginal cost is correspondingly smaller. We do not report wall-clock latency numbers in this paper to avoid potentially misleading comparisons across hardware setups, batch sizes, and serving stacks; we focus instead on the structural cost difference (parameter count and pipeline structure) that holds invariant across deployment configurations.

Comparison with multi-sample methods. SelfCheckGPT-NLI requires 20 additional generation passes through the target LLM plus pairwise NLI inference using a fine-tuned BERT-style NLI model (DeBERTa-large-mnli, ~ 400 M parameters). This is structurally more expensive than HALT regardless of the specific hardware: HALT’s per-detection LLM cost is amortized into the original generation, while SelfCheckGPT’s 20 additional generations are required before any detection signal is available.

Memory footprint. HALT’s 5M parameters and small intermediate activations make it deployable alongside the target LLM on a single GPU with negligible memory overhead, in contrast to Lettuce (which adds a separate 150M-parameter model in memory) and SelfCheckGPT (which requires hosting an additional NLI model).