RAGognizer: Hallucination-Aware Fine-Tuning via Detection Head Integration

Fabian Ridder, Laurin Lessel, and Malte Schilling
Computer Science Department
University of Münster
Münster, Germany
{fridder, llessel, malte.schilling}@uni-muenster.de

***Abstract*—Retrieval-Augmented Generation (RAG) is widely used to augment the input to Large Language Models (LLMs) with external information, such as recent or domain-specific knowledge. Nonetheless, current models still produce closed-domain hallucinations and generate content that is unsupported by the retrieved context. Current detection approaches typically treat hallucination as a post-hoc problem, relying on black-box consistency checks or probes over frozen internal representations. In this work, we demonstrate that hallucination detection based on internal state representation can also serve as a direct training signal. We introduce RAGognize, a dataset of naturally occurring closed-domain hallucinations with token-level annotations, and RAGognizer, a hallucination-aware fine-tuning approach that integrates a lightweight detection head into an LLM, allowing for the joint optimization of language modeling and hallucination detection. This joint objective forces the model to improve the separability of its internal states regarding hallucinations while simultaneously learning to generate well-formed and meaningful responses. Across multiple benchmarks, RAGognizer achieves state-of-the-art token-level hallucination detection while substantially reducing hallucination rates during generation, without degrading language quality or relevance.**

***Index Terms*—Large Language Models, Token-Level Hallucination Detection, Retrieval-Augmented Generation, Internal States, LoRA Fine-Tuning.**

I. Introduction

Large Language Models (LLMs) have achieved impressive performance in natural language understanding and generation [1], [2]. Despite this progress, LLMs remain prone to *hallucinations*: the generation of content that is unsupported by, or contradicts, available evidence [3]. This phenomenon fundamentally limits their reliability, particularly in high-stakes or knowledge-intensive applications.

A central difficulty in defining and detecting hallucinations lies in the dual nature of knowledge in LLMs. During pre-training, models encode vast amounts as implicit *parametric knowledge* that is stored in their weights [4], while at inference time, this may be complemented by explicit information added into the model's context window as *contextual knowledge*. These sources differ substantially in accessibility and verifiability, yet are often conflated when hallucinations are treated simply as factual errors [5]. Retrieval-Augmented Generation (RAG) aims at guiding generation by explicitly providing LLMs with access to external, dynamic information—such as company-specific data or breaking news—that the model was not exposed to during pre-training [4], [6]. But RAG does not inherently solve the problem of reliability. Even when provided with correct context, models frequently exhibit *closed-domain hallucinations*: generating plausible but incorrect information that is not grounded in the retrieved context [7], [8]. This disconnect between the provided evidence (contextual knowledge) and the generated output undermines the trust required for high-stakes applications.

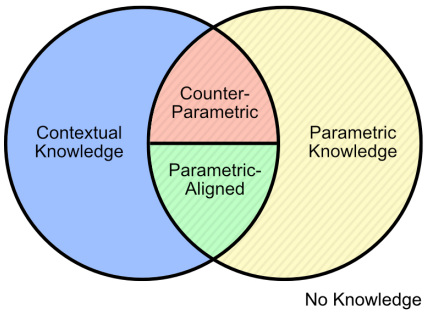

Fig. 1. Distinction of Contextual and Parametric Knowledge: The Venn diagram illustrates possible knowledge scenarios in LLM generation. Prompts may rely solely on contextual knowledge (left), solely on parametric knowledge (right), or on their intersection where the two sources may either align (Parametric-Aligned) or contradict (Counter-Parametric). The *No Knowledge* region corresponds to unanswerable prompts. Regions marked with stripes indicate scenarios not covered by the RAGognize dataset, which focuses exclusively on closed-domain settings where hallucinations are verifiable.

We argue that hallucinations cannot be meaningfully defined or detected without distinguishing between the different knowledge sources. As illustrated in Fig. 1, contextual and parametric knowledge may appear in isolation or in combination. To obtain a decidable notion, we focus on closed-domain settings using exclusively recent information to prevent reliance on parametric knowledge. In this setting—where prompts fall within the *Contextual Knowledge* area if *answerable* or the *No Knowledge* area if *unanswerable*—hallucinations can be unambiguously identified as generations introducing unsupported content.

Focusing on this closed-domain setting, we make three contributions: First, we introduce **RAGognize**, a comprehensive dataset of naturally occurring closed-domain hallucinations with granular token-level annotations. Second, we propose **RAGognizer**, a hallucination-aware model architecture that integrates a simple detection head into an LLM, enabling token-

level hallucination prediction from internal representations and achieving state-of-the-art detection performance on closed-domain benchmarks. Third, we show that jointly optimizing language modeling and hallucination detection objectives using LoRA-based fine-tuning improves the separability of internal states with respect to hallucination, leading to both stronger detection performance and substantially reduced hallucination rates during generation, while preserving language quality.

Our experiments demonstrate that RAGognizer achieves state-of-the-art token-level hallucination detection with a compact Qwen3-4B generation model [9], while significantly improving generation faithfulness in closed-domain RAG settings. Further, we show that this also generalizes to other settings as well, when evaluated on other datasets. Together, these findings indicate that hallucination detection is closely tied to representation learning and that integrating detection signals during training can improve model reliability. The dataset, models, and code can be found online.[1]

II. Related Work

Hallucinations in LLMs have been studied from different perspectives, including detection, mitigation, and dataset construction. In this section, we first review prior work on hallucination detection methods, focusing on how they differ in model access and granularity, and secondly, discuss existing hallucination datasets.

A. Hallucination Detection

Detection methods are commonly categorized by their required access: white-box methods exploit internal activations or attention patterns, while black-box methods operate on outputs alone. Further practical distinctions are the granularity at which hallucinations are identified and whether a method requires stochastic sampling (multiple generations) to estimate consistency, or can run in a single forward pass (see Table I).

White-box approaches include uncertainty proxies such as perplexity [10] and entropy-based scores [11], representation-statistic methods such as INSIDE (EigenScore) [12], attention-based detectors such as Lookback Lens [13], and probe/classifier approaches that train on hidden activations (e.g., SAPLMA [14]). HallucinationProbes train a linear, token-level classifier on hidden states and further explores adapter training via Low-Rank Adaptation (LoRA) alongside the probe head to improve detection while minimally altering base model behavior [15], [16], closely aligning with our approach. Other white-box methods include unsupervised internal-state detectors (MIND [17]), relevance-propagation applied to RAG (LRP4RAG [18]), and cross-layer dynamics probes (ICR Probe [19]).

Black-box approaches include sampling-based consistency checks such as SelfCheckGPT [20], and external evaluator or judge models fine-tuned for factuality (e.g., NLI/entailment models built on DeBERTa-style encoders [21] and specialized evaluators such as MiniCheck, Lynx, and Granite-Guardian [22]–[24]). Community and benchmark models (e.g., HHEM-2.1) provide readily usable open evaluators [25]. Methods tailored to RAG include faithfulness scoring that combines entailment with retrieval evidence (RAGAS) [26] and joint context/knowledge verification models such as HDM-2 [27]. Other work (e.g., LUMINA) examines the balance between reliance on retrieved context and internal parametric knowledge when detecting hallucinations in RAG outputs [28].

TABLE I
Comparison of hallucination detectors. Lvl.: detection granularity (token/response); Samp.: requires multiple samples; Backbone: underlying model.

Approach	Lvl.	Samp.	Backbone
Model-Agnostic (Black-box)			
LettuceDetect [29]	Tok	No	ModernBERT
HDM-2 [27]	Tok	No	LLM
DeBERTa-based NLI [21]	Resp	No	DeBERTa
MiniCheck [22]	Resp	No	Llama3
Lynx [23]	Resp	No	Llama3
Granite Guardian 3.3 [24]	Resp	No	Granite 3.3
HHEM-2.1-Open [25]	Resp	No	FLAN-T5-base
RAGAS Faithfulness [26]	Resp	No	LLM
SelfCheckGPT [20]	Resp	Yes	Generator
Model-Intrinisic (White-box)			
Semantic Entropy (SE) [11]	Resp	Yes	Generator
INSIDE (EigenScore) [12]	Resp	Yes	Generator
ICR Probe [19]	Resp	No	Generator
HaloScope [30]	Resp	No	Generator
LRP4RAG [18]	Resp	No	Generator
Perplexity (PPL) [10]	Tok	No	Generator
LUMINA [28]	Tok	No	Generator
SAPLMA [14]	Tok	No	Generator
MIND [17]	Tok	No	Generator
Lookback Lens [13]	Tok	No	Generator
HallucinationProbes [15]	Tok	No	Generator

B. Datasets

Existing hallucination datasets differ in their annotation granularity, underlying knowledge assumptions, and the nature of hallucinations. A primary distinction concerns the level at which hallucinations are labeled. While most benchmarks provide supervision only at the level of complete responses, a small number of recent datasets offer token-level annotations, which makes these particularly relevant for studying internal model representations and token-level detection (e.g., RAGTruth [8]).

We believe it is important to take the assumed knowledge regime into account. A common issue in many RAG and context-based QA datasets is that they do not strictly ensure questions require the provided context to be answered. This blurs the line between contextual and parametric knowledge; for instance, HaluEval [31] contains questions that LLMs can answer using pre-trained memory. This contrasts with strictly closed-domain settings where valid generations must be supported exclusively by the given context. Finally, datasets differ in how hallucinations are produced: while HaluEval relies on synthetically induced response-level hallucinations,

[1] https://github.com/F4biian/RAGognizer

others like HDM-Bench [27] focus on natural response-level hallucinations that arise during standard model generation.

III. Methods

We first introduce the RAGognize dataset, then present the RAGognizer architectural approach for hallucination-aware LLM fine-tuning, followed by the joint training setup.

A. The RAGognize Dataset

Most existing hallucination benchmarks operate at the response level, rely on synthetic perturbations, or do not preclude open-domain settings, which limits fine-grained detection of hallucination or deviations from given evidence. To address this gap, we introduce the **RAGognize** dataset designed for natural, token-level hallucination detection in closed-domain RAG scenarios. It is constructed in multiple steps and extends the HalluRAG approach [32] with increased prompt diversity and token-level annotations. As illustrated in Fig. 2, the pipeline consists of (i) sourcing of recent factual statements from Wikipedia, (ii) generation of diverse question–answer pairs, (iii) controlled assembly of answerable and unanswerable RAG prompts, (iv) response generation by multiple LLMs, and (v) automated token-level hallucination annotation.

As we want to keep relevant information restricted to the provided context, we adopt a strict recency constraint and extract factual statements from Wikipedia whose associated reference is time-stamped with a date later than May 23, 2024. This ensures that facts were not available for training and cannot be represented in the parametric knowledge of the evaluated models (we used Llama-2-7B-Chat (Llama2-7B) [33], Llama-3.1-8B-Instruct (Llama3-8B) [34], Mistral-7B-Instruct-v0.1 (Mistral-7B-v0.1), and Mistral-7B-Instruct-v0.3 (Mistral-7B-v0.3) [35]). Therefore, RAGognize only deals with *Contextual Knowledge* or *No Knowledge* scenarios (Fig. 1), which establishes a well-defined distinction between answerable and unanswerable queries.

For each factual statement, we use Gemini 2.5 Pro [36] to generate diverse user queries and corresponding reference answers under stylistic variations (e.g., typographical errors, subjective framing, or adding misleading cues) that should encourage linguistic diversity. Answerable and unanswerable RAG prompts are then constructed by applying a modular template strategy by selectively inserting or withholding the context part that contains the crucial evidence, while semantically similar distractor passages are retrieved using BGE-M3 [37]. This procedure yields paired prompts that differ only in the availability of relevant contextual evidence. Answerable prompts are, in this way, formed by replacing one distractor with the ground-truth chunk containing the necessary evidence. All prompts in both the training and test splits are afterwards passed to Llama2-7B, Llama3-8B, Mistral-7B-v0.1, and Mistral-7B-v0.3 using greedy decoding (temperature 0.0), yielding model-generated responses for subsequent annotation.

For the token-level hallucination annotation, the model responses are annotated using Gemini 2.5 Flash [36] as an oracle evaluator. A chain-of-thought prompting strategy [38] identifies hallucinated spans by comparing generated outputs against the retrieved context. Thus, annotation is performed at the token level. Formally, for a generated sequence of length T, we derive a binary label sequence $\mathbf{y}_{\text{det}} \in \{0,1\}^T$, where $y_{\text{det},t} = 1$ indicates that the token generated at time step t is hallucinated. A manual response-level validation on 100 samples yielded an F1 score of 95.4%, suggesting that the annotation approach is broadly consistent with human judgment (this validation was conducted by a single annotator and should be interpreted as a preliminary consistency check rather than a definitive evaluation). Importantly, this process enables the capture of natural hallucinations, which prior work has shown to differ from synthetic hallucinations [3], [39]. The resulting RAGognize dataset exhibits a balanced distribution of answerable $(2,315)$ and unanswerable $(2,308)$ queries across different domains and is divided into training (40%) and test (60%) sets, comprising a total of $18,492$ annotated responses.

B. The RAGognizer Architecture

RAGognizer consists of two interacting components: a base language model responsible for text generation and a detection head that predicts token-level hallucinations from internal representations. Unlike prior probing-based approaches that used the internal representation of a fixed language model, RAGognizer jointly trains both components at the same time. In this way, the emerging hallucination signal in the hallucination classifier directly shapes the model's internal representations.

1) Base Language Model: Let Θ^* denote the pre-trained parameters of the LLM, and let θ_{LoRA} denote the trainable low-rank adapters for the model. Given an input prefix $x_{<t}$, the hidden state at layer ℓ and time step t is

$$\mathbf{h}_t^{(\ell)} = \text{LLM}(x_{<t}; \Theta^*, \theta_{\text{LoRA}}). \tag{1}$$

The model computes the next-token probability distribution based on the final layer's hidden state, $\mathbf{h}_t^{(-1)}$, by passing it through a linear head and a softmax function.

2) Hallucination Detection Head: We attach a detection head f_ϕ, parameterized by ϕ and implemented as an MLP, to an intermediate hidden state of the LLM. The head outputs the probability that the current token is hallucinated:

$$\hat{p}_t = \sigma\left(f_\phi\left(\mathbf{h}_t^{(\ell)}\right)\right), \tag{2}$$

where $\sigma(\cdot)$ denotes the sigmoid function. Unless stated otherwise, we select ℓ as the middle layer of the network, as intermediate layers show the highest separability between hallucinated and grounded tokens [14], [27], [39], [40].

3) Joint Optimization of the LLM and Detection Head: Most prior work trained hallucination detectors post-hoc on fixed LLM representations, effectively treating the model as a static feature extractor. In contrast, RAGognizer uses gradients from the detection head that propagate through the hidden states into the LoRA adapters of the earlier layers of the LLM (Fig. 3). This aims at learning internal representations that are simultaneously predictive for next-token generation and separable with respect to hallucinated content.

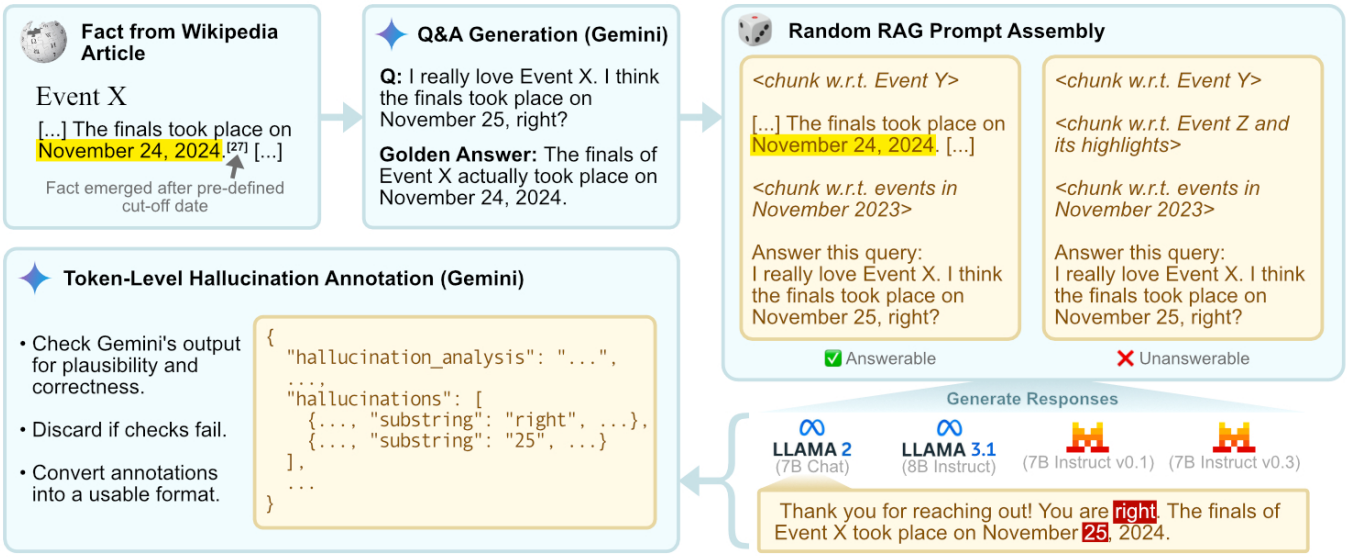

Fig. 2. Automatic Data Generation and Annotation Pipeline for the RAGognize dataset: Wikipedia facts post-dating the training cut-off date (May 23, 2024) are extracted which ensures that this information was not used for training of the considered LLMs. Secondly, we generate Q&A pairs using Gemini 2.5 Pro and assemble randomly two different RAG configurations: Answerable (containing the relevant chunk) and Unanswerable (containing irrelevant but similar chunks) querries. We collect natural responses from four target LLMs (Llama-2/3.1, Mistral-v0.1/v0.3). Finally, Gemini 2.5 Flash is used with a structured chain-of-thought prompt for substring verification to compare responses with the provided context, which returns granular, token-level hallucination annotations.

We jointly optimize LoRA adapters and the detection head using a multi-task objective. Let x denote the input sequence consisting of a prompt of length L and a response of length K, such that the total length is $T = L + K$. We compute losses only on the response tokens ($t > L$). The causal language modeling loss is

$$\mathcal{L}_{\text{CE}} = -\sum_{t=L+1}^{T} \log P(x_t \mid x_{<t}; \Theta^*, \theta_{\text{LoRA}}),$$

and the token-level hallucination detection loss is

$$\mathcal{L}_{\text{BCE}} = \sum_{t=L+1}^{T} \text{BCE}(\hat{p}_t, y_{\text{det},t}),$$

where $\hat{p}_t$ is the probability predicted by the detection head f_ϕ. Let $\mathcal{D}$ denote the training distribution. The joint training objective is

$$\min_{\theta_{\text{LoRA}},\phi} \mathbb{E}_{(x,\mathbf{y}_{\text{det}})\sim\mathcal{D}} \left[\mathcal{L}_{\text{CE}} + \lambda\, \mathcal{L}_{\text{BCE}}\right], \tag{3}$$

where λ controls the trade-off between generation quality and hallucination detection performance.By integrating hallucination supervision into training, RAGognizer steers the LLM toward representations that enhance the separability of faithful and hallucinatory states, while preserving generation quality.

C. Training Setup

We fine-tune the base LLMs using LoRA to ensure parameter efficiency while keeping the pre-trained backbone frozen. We target all transformer modules with a rank $r = 32$ and a scaling factor $\alpha = 16$. No dropout is applied to the LoRA adapters. Models are trained for a maximum of five epochs using a cosine learning rate scheduler with a warmup ratio of 0.1 and a peak learning rate of 4×10^{-5}.

The integrated detection head is implemented as a three-layer Multi-Layer Perceptron (MLP) with a hidden size of 1024. To focus the learning signal exclusively on the model's generations, we employ a masked detection loss that ignores user prompt tokens and only considers generated tokens. Regarding the loss weighting in Eq. (3), we adopt a balanced optimization strategy by setting $\lambda = 1.0$. This enforces an equal prioritization of language modeling and hallucination detection objectives.

IV. Results

A. Effects of Joint Training on Representations & Generation

A key question is whether hallucination detection through an internal classifier can be used as an effective training signal rather than only as a post-hoc diagnostic. We, therefore, compare three different models: First, the **Base** model is compared to **Det. Head FT**, which jointly optimizes language modeling and token-level hallucination detection via Eq. (3). Lastly, we check if fine-tuning on a hallucination dataset itself might lead to improvements and therefore compare to a text-based fine-tuning (**Text FT**), which applies standard LoRA fine-tuning on golden answers without an explicit hallucination objective. We highlight three aspects: first, representation separability of hallucinated vs. nonhallucinated tokens in hidden states; second, how this separability varies across different language models; and third, the effect of training on generation.

1) Joint Training Increases Hallucination Separability: Fig. 4 shows that for the joint training (Det. Head FT), AUROC shows a consistent and substantial improvement for

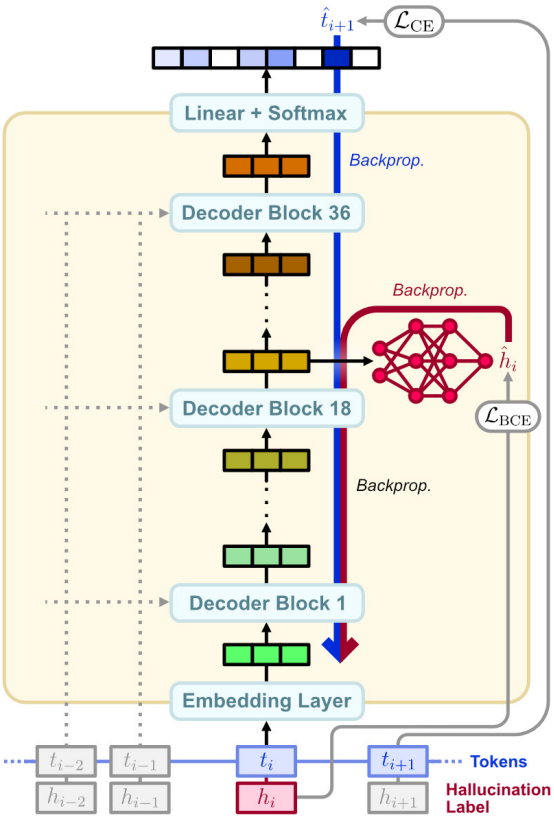

Fig. 3. RAGognizer Architecture: An MLP detection head is integrated at an intermediate layer (e.g., Block 18 for Qwen3-4B-Instruct-2507) to predict the hallucination probability $\hat{h}_i$ of the current token t_i. The model is optimized using a joint objective function combining the next-token prediction loss ($\mathcal{L}_{\text{CE}}$) and the hallucination detection loss ($\mathcal{L}_{\text{BCE}}$). Gradients from both tasks (blue and red arrows) are backpropagated to update LoRA adapters and the newly integrated MLP.

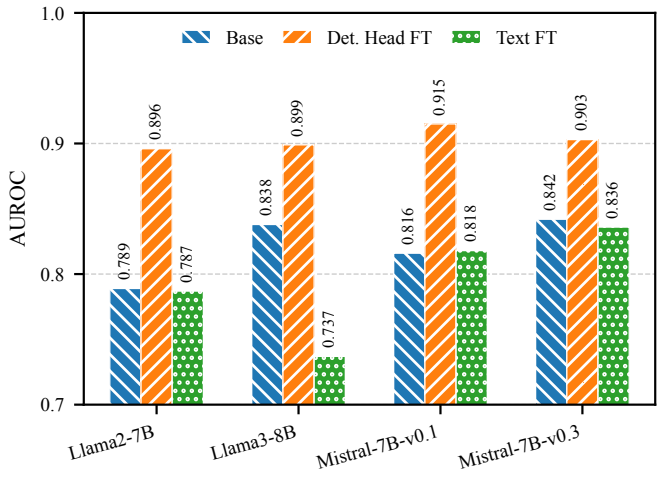

Fig. 4. Impact of Fine-Tuning Strategies on Hallucination Detection: We evaluate the separability of internal states at the middle layer across all response tokens. Det. Head FT results in the highest AUROC, outperforming the Base model. In contrast, standard Text FT often diminishes separability.

TABLE II
TOKEN-LEVEL AUROC OF SELECTED LANGUAGE MODELS AFTER JOINT OPTIMIZATION ON RAGOGNIZE

Model	AUROC (%)
Gemma-3-270M [41]	78.34
Qwen3-0.6B [9]	86.14
Gemma-3-1B [41]	87.33
LFM2-1.2B-RAG [42]	83.33
LLaMA-3.2-1B [43]	88.46
Granite-4.0-micro [44]	86.24
Qwen3-4B [9]	92.69

hallucination detection when separating hallucinated from nonhallucinated tokens based on middle-layer hidden states. For example, on Llama2-7B separability increases from 78.9% to 89.6%. In contrast, Text FT fails to improve separability and in one case shows even a negative influence, with Llama3-8B dropping to 73.7%. This indicates that training with an explicit hallucination objective encourages internal representations that better distinguish grounded from unsupported content.

2) Comparison of Different Base Models: For RAGognizer, we considered different language models for further evaluations. Table II compares a range of small and medium-scale language models for which we applied the described RAGognizer approach. Overall, the evaluation indicates that RAGognizer's performance is robust across different language

TABLE III
COMPARISON OF FINE-TUNING APPROACHES ON RAGOGNIZE USING RESPONSE-LEVEL METRICS (%), HIGHLIGHTING THEIR IMPACT ON LANGUAGE QUALITY AND HALLUCINATION BEHAVIOR.

Metric	Base	Det. Head FT	Text FT
LLaMA-2-7B-Chat			
Answerability Acc. (↑)	63.11	**91.47**	58.21
Answerability F1 (↑)	70.94	**91.86**	49.87
Language Quality (↑)	**99.93**	98.41	99.86
Relevance Rate (↑)	**98.59**	98.03	98.48
Rejection Rate (Ideal: 50%)	23.04	45.21	66.63
Hallucination Rate (↓)	56.98	**13.29**	43.83
Answerable (↓)	43.02	**15.53**	60.89
Unanswerable (↓)	70.94	**11.06**	26.78
Qwen3-4B-Instruct-2507			
Answerability Acc. (↑)	92.28	**95.53**	–
Answerability F1 (↑)	92.64	**95.66**	–
Language Quality (↑)	**99.96**	99.93	–
Relevance Rate (↑)	100.00	100.00	–
Rejection Rate (Ideal: 50%)	45.17	46.79	–
Hallucination Rate (↓)	5.57	**3.59**	–
Answerable (↓)	8.25	**5.50**	–
Unanswerable (↓)	2.89	**1.69**	–

Det. Head FT: LoRA with integrated MLP detection head; **Text FT**: LoRA fine-tuning on golden answers; **Rejection Rate**: percentage of prompts rejected. The ideal value is 50%, as the evaluation dataset is balanced (50% answerable / 50% unanswerable); therefore, an optimal model rejects exactly half of the prompts. **Language Quality**: percentage of non-malformed responses; Text FT has not been conducted on Qwen3-4B; **Bold**: best; Underline: second-best.

TABLE IV
TOKEN-LEVEL AUROC (%) OF HALLUCINATION DETECTORS ON QA DATASETS.

Approach	RT	RG	HD	Avg.
Black-box detectors				
HDM-2-3B	90.61†	68.72	74.99†	78.11
LettuceDetect-L	**95.60**†	61.04	69.00	75.21
White-box detectors				
HaloScope	55.93†	48.41†	-	-
LookbackLens	85.42†	77.79†	-	-
Perplexity	64.59	58.40	59.60	60.86
HallucinationProbes	70.13	69.95	76.38	72.15
RAGognizer (Ours)	90.25	**93.33**†	**77.03**	**86.87**

RT = RAGTruth (QA), **RG** = RAGognize, **HD** = HDM-Bench; HDM-Bench contains no training split; Perplexity, HaloScope, LookbackLens, and RAGognizer use Qwen3-4B-Instruct-2507 as proxy generator; HallucinationProbes uses a LoRA-KL probe on LLaMA-3.1-8B ($\lambda_{\text{KL}} = 0.05$); **Bold**: best, Underline: second-best; † same-dataset evaluation (split-disjoint).

TABLE V
RESPONSE-LEVEL AUROC (%) OF DETECTORS ON QA DATASETS.

Approach	RT	RG	HE	HD	Avg.
Black-box detectors					
MiniCheck-7B	63.10	83.48	81.38	71.70	74.91
HDM-2-3B	87.95†	75.41	83.85	69.62†	79.21
DeBERTa-v3 (Con.)	56.69	64.31	68.10	60.90	62.50
DeBERTa-v3 (Ent.)	53.25	68.21	78.32	62.60	65.59
SelfCheckGPT (NLI)	61.13	74.22	83.43	68.44	71.81
LettuceDetect-L	**91.89**†	65.75	77.02	60.85	73.88
HHEM-2.1-Open	71.70	61.49	77.00	59.97	67.54
White-box detectors					
HaloScope	53.54†	71.38†	-	-	-
LookbackLens	87.54†	94.36†	-	-	-
Semantic Entropy	67.98	61.48	74.07	55.81	64.84
INSIDE (EigenScore)	60.96	64.49	47.46	56.35	57.32
HallucinationProbes	78.08	83.25	**94.93**	**76.30**	83.14
RAGognizer (Ours)	87.43	**94.91**†	90.38	72.72	**86.36**

RT = RAGTruth (QA), **RG** = RAGognize, **HE** = HaluEval (QA, 5,000 samples), **HD** = HDM-Bench; HaluEval and HDM-Bench contain no training split; HaloScope, LookbackLens, INSIDE, Perplexity, and RAGognizer use Qwen3-4B-Instruct-2507 as proxy generator; HallucinationProbes uses a LoRA-KL probe on LLaMA-3.1-8B ($\lambda_{\text{KL}} = 0.05$); **Bold**: best, Underline: second-best; † same-dataset evaluation (split-disjoint).

models and not tied to a specific model. Among the evaluated candidates, Qwen3-4B achieves the highest token-level AUROC (92.69%).

3) Effect on Language Generation: The increased separability is accompanied by substantial improvements for generations themselves for RAGognize (Table III).For Llama2-7B, Det. Head FT reduced hallucinations ($56.98\% \rightarrow 13.29\%$) and raised Answerability F1 ($70.94\% \rightarrow 91.86\%$). This was achieved without further manipulation or explicit instructions, solely through increased internal state separation at the middle layer. This improved refusal logic is also reflected in the rejection rate (45.21%), which nears the 50% ideal for this balanced dataset. The reduction of hallucinations holds for both answerable and unanswerable prompts, indicating improved grounding when evidence is present and more reliable refusal when it is absent. By contrast, Text FT yields a worse Answerability F1 and higher hallucination rates for answerable prompts.

As fine-tuning affects the generated answers, we further check if the produced answers are still of high language quality and if the model provides relevant answers. Despite the strong shift in hallucination behavior, language quality and relevance remain high after Det. Head FT (Table III). This suggests that incorporating token-level hallucination supervision as an auxiliary objective can improve faithfulness without degrading response well-formedness or relevance.

B. Comparative Performance in Closed-Domain Detection

We evaluate how RAGognizer performs as a hallucination detector compared to state-of-the-art black-box and white-box approaches in closed-domain RAG settings.

1) Token-Level Detection Performance: Table IV reports token-level AUROC across closed-domain QA benchmarks. We compare the performance of different hallucination detectors on three datasets using their respective test splits. We experimentally report HaloScope at the token level despite its original response-level design. Overall, RAGognizer achieves the highest average performance, outperforming both black-box detectors, such as LettuceDetect and HDM2-3B, and white-box approaches, such as HallucinationProbes.

For each method, it is important to distinguish the dataset used during training or calibration (indicated by †) as the results reveal a dependency on the dataset for some baselines, which indicates a limited transferability. In contrast, RAGognizer maintains strong performance for both the native RAGognize test dataset as well as on other datasets, such as RAGTruth and HDM-Bench. This suggests that token-level supervision learned in RAGognize generalizes better compared to transfer in the case of other approaches. Several of these methods exhibit a pronounced dataset sensitivity. For instance, black-box detectors such as LettuceDetect achieve strong performance on the specific type of dataset used in training, but degrade on others. Other approaches, e.g., HallucinationProbes, show weaker token-level performance in closed-domain settings. These results suggest that integrating hallucination supervision into the language model during training leads to more robust token-level detection signals than post-hoc probing or purely logit-based heuristics.

2) Response-Level Aggregation: While RAGognizer operates natively on the token level, many applications and detection approaches use simpler response-level hallucination scores. We therefore use max-pooling to aggregate token-level predictions and report response-level AUROC in Table V. With this aggregation, RAGognizer performs quite strongly across QA benchmarks with the highest average AUROC compared to all state-of-the-art methods. In the case of sampling-based detectors, we generate five additional responses per prompt and evaluate 100 randomly sampled prompts per dataset

TABLE VI
RESPONSE-LEVEL AUROC (%) ON CONFLICTQA (POPQA) UNDER CONTEXTUAL–PARAMETRIC ALIGNMENT SETTINGS.

Approach	Ctr-P	P-Alg	NoCtx	All
Black-box detectors				
MiniCheck-7B	**95.41**	**98.14**	68.49	**91.65**
HDM2-3B	<u>94.62</u>	<u>96.72</u>	58.91	87.63
DeBERTa-v3 (Con.)	91.05	96.26	61.01	72.95
DeBERTa-v3 (Ent.)	93.36	97.37	63.54	83.73
SelfCheckGPT (NLI)	84.73	93.72	66.75	82.18
LettuceDetect-L	93.50	96.56	30.85	74.01
HHEM-2.1-Open	93.01	97.37	48.98	85.55
White-box detectors				
Semantic Entropy	70.20	65.56	61.58	65.12
INSIDE (EigenScore)	62.52	64.78	58.83	61.20
Perplexity	82.71	81.28	61.84	74.53
HallucinationProbes	85.13	97.35	**72.29**	85.20
RAGognizer (Ours)	93.81	96.11	<u>69.26</u>	<u>89.33</u>

Ctr-P: Counter-Parametric; **P-Alg**: Parametric-Aligned; **NoCtx**: No Context; INSIDE, Perplexity, and RAGognizer use Qwen3-4B-Instruct-2507 as proxy generator; HallucinationProbes uses a LoRA-KL probe on LLaMA-3.1-8B ($\lambda_{\mathrm{KL}} = 0.05$); **Bold**: best; <u>Underline</u>: second-best.

category (for RAGognize, all samples are used). RAGognizer consistently outperforms general NLI-based baselines such as DeBERTa-v3 and even compares favorably with larger or more specialized fact-checking models. HallucinationProbes, as a similar approach, shows good performance as well, in two cases even outperforming RAGognizer, which further strengthens the case for such joint training of LoRA adapters and an integrated detection probe.

C. Generalization to Detection in Open-Domain Scenarios

RAGognizer has been trained in a rigorous closed-domain setting for detecting hallucinations with respect to a provided context. More generally in real-world use cases, requests to LLMs often involve both contextual and parametric knowledge. Therefore, we are interested in how hallucination signals learned in the closed-domain setup transfer to the more general case in which contextual evidence may align or contradict the model's parametric knowledge. We, therefore, evaluate RAGognizer on the PopQA subset of ConflictQA [45], assuming that information from the most popular Wikipedia articles is contained in the models' parametric knowledge.

Table VI evaluates AUROC where context aligns with parametric knowledge (P-Alg), contradicts it (Ctr-P), or is absent (NoCtx). Most detectors, including RAGognizer, perform well in P-Alg. In Ctr-P, RAGognizer is competitive (93.81%), nearing MiniCheck-7B. NoCtx is the hardest scenario, with performance degrading for all; here, HallucinationProbes achieves the highest AUROC (72.29%), consistent with its training objective that treats parametric knowledge as the primary reference signal. RAGognizer ranks second (69.26%), indicating that hallucination signals learned from closed-domain supervision partially transfer to settings without contextual grounding, but do not fully subsume parametric knowledge-based detection. Overall, in this case, RAGognizer generalizes surprisingly well to open-domain and mixed scenarios. It is only outperformed by MiniCheck-7B as a heavy-weight and nonintegrated detector.

In summary, closed-domain hallucination signals partially overlap with open-domain truthfulness signals. While RAGognizer generalizes beyond its training regime, performance variations across scenarios indicate that hallucination detection remains knowledge source-dependent to a certain degree. This validates our central premise: distinguishing contextual from parametric knowledge is a practical necessity for understanding and modeling hallucinations in LLMs.

V. DISCUSSION & CONCLUSION

Our results show that hallucination detection is closely tied to internal representations. Jointly optimizing language modeling and token-level hallucination detection consistently increases the separability of hallucinated and grounded tokens, while substantially reducing hallucination rates during generation without degrading language quality or relevance. In contrast to post-hoc probing or purely logit-based heuristics, integrating hallucination supervision during training resulted in better and transferable detection signals. While trained exclusively in closed-domain settings, RAGognizer generalizes well to mixed and open-domain scenarios, indicating that hallucination-related signals learned with respect to contextual grounding partially transfer to more ambiguous cases. Crucially, this transfer relies on first establishing a clear-cut definition of which information is available to the model when judging hallucinations. The introduced RAGognize dataset provides such a well-defined closed-domain dataset, enabling principled learning before we extended this to scenarios where contextual and parametric knowledge interact.

Beyond accuracy, RAGognizer, in addition, offers practical advantages for deployment as it provides real-time token-level hallucination scores with only 3.7M additional parameters, avoiding the computational overhead of external verification models, but also being bound to the host LLM and to the fine-tuning of it. As a limitation of the current study, we focused on single-turn interactions and on one task (Q&A). We relied on automated annotation using Gemini 2.5 Flash, employed a fixed weighting between the interacting loss signals, and did not monitor performance on other non-hallucination benchmarks; these aspects should be further analyzed in future work.

Overall, our findings support the view that hallucination detection is fundamentally a representation learning problem and that integrating detection signals into the training process provides a principled path toward more reliable language models in retrieval-augmented generation.

VI. ETHICAL CONSIDERATIONS

The LLM-generated, Wikipedia-derived RAGognize dataset may contain biases or inaccuracies. It is intended for research only and should not be used as ground truth or in high-stakes applications.

REFERENCES

[1] T. B. Brown, B. Mann, N. Ryder, and et al., "Language models are few-shot learners," 2020. [Online]. Available: https://arxiv.org/abs/2005.14165

[2] J. Wei, Y. Tay, R. Bommasani, and et al., "Emergent abilities of large language models," 2022. [Online]. Available: https://arxiv.org/abs/2206.07682

[3] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, and T. Liu, "A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions," *ACM Transactions on Information Systems*, vol. 43, no. 2, pp. 1–55, Jan. 2025. [Online]. Available: http://dx.doi.org/10.1145/3703155

[4] F. Petroni, T. Rocktäschel, P. Lewis, A. Bakhtin, Y. Wu, A. H. Miller, and S. Riedel, "Language models as knowledge bases?" 2019. [Online]. Available: https://arxiv.org/abs/1909.01066

[5] R. Xu, Z. Qi, Z. Guo, C. Wang, H. Wang, Y. Zhang, and W. Xu, "Knowledge conflicts for LLMs: A survey," 2024. [Online]. Available: https://arxiv.org/abs/2403.08319

[6] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. tau Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive NLP tasks," 2021. [Online]. Available: https://arxiv.org/abs/2005.11401

[7] A. Agrawal, M. Suzgun, L. Mackey, and A. T. Kalai, "Do language models know when they're hallucinating references?" 2024. [Online]. Available: https://arxiv.org/abs/2305.18248

[8] C. Niu, Y. Wu, J. Zhu, S. Xu, K. Shum, R. Zhong, J. Song, and T. Zhang, "Ragtruth: A hallucination corpus for developing trustworthy retrieval-augmented language models," 2024. [Online]. Available: https://arxiv.org/abs/2401.00396

[9] A. Yang, A. Li, B. Yang, and et al., "Qwen3 technical report," 2025. [Online]. Available: https://arxiv.org/abs/2505.09388

[10] F. Jelinek, R. L. Mercer, L. R. Bahl, and J. M. Baker, "Perplexity—a measure of the difficulty of speech recognition tasks," *Journal of the Acoustical Society of America*, vol. 62, 1977. [Online]. Available: https://api.semanticscholar.org/CorpusID:121680873

[11] S. Farquhar, J. Kossen, L. Kuhn, others, and Y. Gal, "Detecting hallucinations in large language models using semantic entropy," *Nature*, vol. 630, pp. 625–630, 2024. [Online]. Available: https://www.nature.com/articles/s41586-024-07421-0

[12] C. Chen, K. Liu, Z. Chen, Y. Gu, Y. Wu, M. Tao, Z. Fu, and J. Ye, "INSIDE: LLMs' internal states retain the power of hallucination detection," 2024. [Online]. Available: https://arxiv.org/abs/2402.03744

[13] Y.-S. Chuang, L. Qiu, C.-Y. Hsieh, R. Krishna, Y. Kim, and J. Glass, "Lookback lens: Detecting and mitigating contextual hallucinations in large language models using only attention maps," 2024. [Online]. Available: https://arxiv.org/abs/2407.07071

[14] A. Azaria and T. Mitchell, "The internal state of an LLM knows when it's lying," 2023. [Online]. Available: https://arxiv.org/abs/2304.13734

[15] O. Obeso, A. Arditi, J. Ferrando, J. Freeman, C. Holmes, and N. Nanda, "Real-time detection of hallucinated entities in long-form generation," 2025. [Online]. Available: https://arxiv.org/abs/2509.03531

[16] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-rank adaptation of large language models," 2021. [Online]. Available: https://arxiv.org/abs/2106.09685

[17] W. Su, C. Wang, Q. Ai, Y. HU, Z. Wu, Y. Zhou, and Y. Liu, "Unsupervised real-time hallucination detection based on the internal states of large language models," 2024. [Online]. Available: https://arxiv.org/abs/2403.06448

[18] H. Hu, C. He, X. Xie, and Q. Zhang, "LRP4RAG: Detecting hallucinations in retrieval-augmented generation via layer-wise relevance propagation," *arXiv preprint 2408.15533*, 2025.

[19] Z. Zhang, X. Hu, H. Zhang, J. Zhang, and X. Wan, "ICR Probe: Tracking hidden state dynamics for reliable hallucination detection in llms: Tracking hidden state dynamics for reliable hallucination detection in LLMs," 2025. [Online]. Available: https://arxiv.org/abs/2507.16488

[20] P. Manakul, A. Liusie, and M. J. F. Gales, "SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models: Zero-resource black-box hallucination detection for generative large language models," 2023. [Online]. Available: https://arxiv.org/abs/2303.08896

[21] P. He, J. Gao, and W. Chen, "DeBERTaV3: Improving DeBERTa using ELECTRA-Style pre-training with gradient-disentangled embedding sharing," 2023. [Online]. Available: https://arxiv.org/abs/2111.09543

[22] L. Tang, P. Laban, and G. Durrett, "MiniCheck: Efficient fact-checking of LLMs on grounding documents," 2024. [Online]. Available: https://arxiv.org/abs/2404.10774

[23] S. S. Ravi, B. Mielczarek, A. Kannappan, D. Kiela, and R. Qian, "Lynx: An open source hallucination evaluation model," 2024. [Online]. Available: https://arxiv.org/abs/2407.08488

[24] I. Padhi, M. Nagireddy, G. Cornacchia, and et al., "Granite guardian," 2024. [Online]. Available: https://arxiv.org/abs/2412.07724

[25] O. Mendelevitch, F. Bao, M. Li, and R. Luo, "HHEM 2.1: A better hallucination detection model and a new leaderboard," Vectara blog, Aug 2024. [Online]. Available: https://www.vectara.com/blog/hhem-2-1-a-better-hallucination-detection-model

[26] S. Es, J. James, L. Espinosa-Anke, and S. Schockaert, "Ragas: Automated evaluation of retrieval augmented generation," 2025. [Online]. Available: https://arxiv.org/abs/2309.15217

[27] B. Paudel, A. Lyzhov, P. Joshi, and P. Anand, "HalluciNot: Hallucination detection through context and common knowledge verification," 2025. [Online]. Available: https://arxiv.org/abs/2504.07069

[28] S. Yeh, S. Li, and T. Mallick, "LUMINA: Detecting hallucinations in RAG system with context-knowledge signals," 2025. [Online]. Available: https://arxiv.org/abs/2509.21875

[29] Á. Kovács and G. Recski, "Lettucedetect: A hallucination detection framework for RAG applications," *arXiv preprint arXiv:2502.17125*, 2025.

[30] X. Du, C. Xiao, and Y. Li, "HaloScope: Harnessing unlabeled LLM generations for hallucination detection," 2024. [Online]. Available: https://arxiv.org/abs/2409.17504

[31] 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," 2023. [Online]. Available: https://arxiv.org/abs/2305.11747

[32] F. Ridder and M. Schilling, "The HalluRAG dataset: Detecting closed-domain hallucinations in rag applications using an LLM's internal states," 2025. [Online]. Available: https://arxiv.org/abs/2412.17056

[33] H. Touvron, T. Lavril, G. Izacard, and et al., "LLaMA: Open and efficient foundation language models," 2023. [Online]. Available: https://arxiv.org/abs/2302.13971

[34] A. Grattafiori, A. Dubey, A. Jauhri, and et al., "The Llama 3 herd of models," 2024. [Online]. Available: https://arxiv.org/abs/2407.21783

[35] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed, "Mistral 7b," 2023. [Online]. Available: https://arxiv.org/abs/2310.06825

[36] G. Comanici, E. Bieber, M. Schaekermann, and et al., "Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities," 2025. [Online]. Available: https://arxiv.org/abs/2507.06261

[37] J. Chen, S. Xiao, P. Zhang, K. Luo, D. Lian, and Z. Liu, "BGE M3-Embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation," 2023.

[38] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," 2023. [Online]. Available: https://arxiv.org/abs/2201.11903

[39] S. CH-Wang, B. V. Durme, J. Eisner, and C. Kedzie, "Do androids know they're only dreaming of electric sheep?" 2024. [Online]. Available: https://arxiv.org/abs/2312.17249

[40] H. Duan, Y. Yang, and K. Y. Tam, "Do LLMs know about hallucination? an empirical investigation of LLM's hidden states," 2024. [Online]. Available: https://arxiv.org/abs/2402.09733

[41] G. Team, A. Kamath, and J. F. et al., "Gemma 3 technical report," 2025. [Online]. Available: https://arxiv.org/abs/2503.19786

[42] A. Amini, A. Banaszak, H. Benoit, and et al., "LFM2 technical report," 2025. [Online]. Available: https://arxiv.org/abs/2511.23404

[43] Meta AI, "LLaMA 3.2 1B language model," Hugging Face model card, https://huggingface.co/meta-llama/Llama-3.2-1B, 2024, [Online; accessed 5-Nov-2025].

[44] IBM Research, "Granite 4.0 language models," GitHub, https://github.com/ibm-granite/granite-4.0-language-models, 2025, [Online; accessed 5-Nov-2025].

[45] J. Xie, K. Zhang, J. Chen, R. Lou, and Y. Su, "Adaptive chameleon or stubborn sloth: Revealing the behavior of large language models in knowledge conflicts," 2024. [Online]. Available: https://arxiv.org/abs/2305.13300