Detecting Natural Hallucinations in Local RAG: Span Detectors, LLM Judges, and the Limits of Ensembling

Moiz Shahid

Supervisor: Syed Rafee
MSc. Artificial Intelligence
School of Electronic Engineering and Computer Science
Queen Mary University of London

***Abstract*—Retrieval-augmented generation (RAG) grounds large language model (LLM) answers in retrieved documents, but the responses can still contain hallucinations. Detectors usually get tested on the same benchmark distributions they were trained on. However, the detection of naturally occurring hallucinations in well-built RAG systems remains largely untested. We explore this question in a completely local RAG system based on six UK government guidance documents ("coloured books"). We compare five claim-level verification mechanisms: RAGTruth-trained encoder span detector (LettuceDetect), an LLM-as-a-judge method, and three hybrid methods (a max-severity combination termed AND, uncertainty-aware voting, and a cost-saving cascade). We contribute a hand-labelled evaluation set of 346 claims (345 scored) based on 100 questions from government documents, with difficult cases discussed with two additional reviewers. Our results show that naturally occurring hallucinations are relatively rare in our evaluation set (12.2% of claims) and often subtle. They mainly arise in ordinarily answerable questions and defeat every detector we tested. The span detector seems to be practically oblivious to hallucinations; 40 of 42 hallucinated claims show no detected spans at any threshold (AUROC 0.52). As a result, the silence is inherited by the combinations, and the cascade's 99% cost saving leads to almost zero detection. The three judges from different model families (Llama-3.1-8B, Qwen2.5-7B, Gemma-2-9B) produce weak results individually (recall equals 14.3%, 21.4%, and 16.7% respectively with at best one true alarm in three). They often fail on different claims, as no pair identifies the same errors more than four times, and flagging any claim by any of the judges increases recall to 40.5% in an exploratory analysis. The best exploratory combination, the two stronger judges together with the span detector, achieves an F1 score of 38.1% compared to 26.1% for the best single judge, and the two-judge union alone reaches 43.1% after excluding meta-claims which are claims about what the sources do not say that the judge prompt cannot verify and that create most false alarms. Requiring judges to agree fails completely: every claim all three judges flag together is a false alarm. We conclude that benchmark performance did not transfer to natural errors arising in this domain. For local judges, diversity across families, rather than agreement, provides a productive ensemble signal, resulting in increased recall at the expense of precision depending on which families have been grouped together.**

***Index Terms*—retrieval-augmented generation, hallucination detection, LLM-as-judge, claim-level verification, local LLMs**

I. Introduction

Large language models store information in their weights — a parametric memory that is fixed during training and can produce confident yet false statements. Retrieval-augmented generation (RAG) addresses this by introducing a non-parametric memory: an external archive of documents from which relevant passages are retrieved and given to the model to base its output on (Lewis et al., 2020). Precisely because of this promise of factuality, RAG is increasingly used to answer questions over authoritative document collections, including government guidance. Yet grounding does not eradicate the problem completely. In our setting, a hallucination is a generated claim that lacks support from the retrieved sources (Ji et al., 2023) — and this is not a cosmetic defect. Guidance such as HM Treasury's Green Book governs the appraisal of public spending, meaning that a fabricated discount rate or risk category may lead to erroneous decisions. Therefore, automatic hallucination detection is proposed as a safety measure for RAG. However, the detectors in the literature are almost all trained and tested on benchmarks. Examples such as RAGTruth (Niu et al., 2024) contain responses produced by earlier and less well-grounded generators carrying out standard benchmark tasks, in which hallucinations occur frequently (in 43% of the responses). It remains largely untested whether this detection skill transfers to errors that happen naturally in a carefully designed, temperature-0, grounded pipeline. By natural we mean hallucinations that emerged from the RAG system itself during normal operation, rather than being synthetically inserted.

We examine this transfer question end-to-end under a deliberately practical constraint: everything runs locally (Apple M4 Pro, 24 GB unified memory), using 7–9B-parameter models for generating and judging the output. This is the regime available to a government team that cannot send documents to external APIs. Local open-weights models also ensure that the study is reproducible, since each model, prompt, and generation is recorded and held fixed across runs, whereas

commercial APIs are subject to silent updates.

We ask:

- **RQ1**: How often, and where, does a well-constructed local RAG system produce hallucinations on this corpus?
- **RQ2**: How well do an encoder span detector (LettuceDetect) and an LLM-as-judge detect these natural hallucinations at the claim level?
- **RQ3**: Do combinations of the two (strict AND, voting with an uncertainty class, screening cascade) perform better than either one alone?
- **RQ4**: What do the trade-offs between cost and accuracy look like?

Contributions. (1) A hand-labelled, corpus-relative gold dataset consisting of 100 questions across five difficulty types over six UK government guidance documents, resulting in 346 atomic claims, 345 of which are labelled green/yellow/red against the retrieved evidence (one excluded as an extraction error), with difficult cases resolved by consensus — a claim is judged by what the retrieved passages say, not by what is true in the real world. (2) A controlled comparative analysis of five verification systems on frozen generations, replicated across three independent judge families (Llama-3.1-8B, Qwen2.5-7B, Gemma-2-9B), the third run carried out under pre-registration. (3) Evidence that a RAGTruth-trained span detector is almost unable to detect natural hallucinations in this domain, and we identify the mechanism: for 40/42 hallucinated claims the detector produces zero spans at any threshold. (4) The disjoint-catches phenomenon and its limitations: the judge families identify mostly non-overlapping sets of true errors, so an OR combination enhances both precision and recall at the same time (exploratory), while the pre-registered third family shows that the disjointness depends on the pair; unanimous agreement catches nothing, and a 2-of-3 majority catches only 5 of 42. (5) A cost analysis showing that the cascade reduces the number of LLM calls by 99% (from 345 to 2) with almost zero detection.

II. Related work

Span-level detectors. RAGTruth (Niu et al., 2024) established word-level hallucination annotation in the context of RAG: approximately 18,000 naturally generated model responses from question answering, summarisation, and data-to-text tasks were each annotated with hallucinated spans using a four-way taxonomy (evident and subtle conflict, evident and subtle baseless information). The results show that the fine-tuned Llama-2-13B baseline outperformed the prompted GPT-4 model, indicating that trained span-level detector models can outperform far larger prompted models. LettuceDetect (Kovács and Recski, 2025) takes this further by training a much smaller ModernBERT (Warner et al., 2024) token classifier on RAGTruth. It emits binary unsupported-span predictions with confidences, and outperforms the fine-tuned 13B baseline at a fraction of its cost. Crucially, both are trained and evaluated in-distribution: on responses from the same tasks and the same 2023-era generators. However, whether that skill extends to the less frequent, subtle error types produced by a carefully engineered pipeline is precisely the gap that we measure.

LLM-as-judge. Prompted LLM evaluation (Zheng et al., 2023; Liu et al., 2023) is flexible but fragile since the judges tend to favour answers in certain positions and to reward verbosity, the degree to which they agree with human judgments depends heavily on the scale of the judge, and the scores change according to the design of the prompt. These limitations have mainly been identified in the case of frontier-scale judges; we investigate what still applies at a scale of about 8B that is feasible with local deployment, using a JSON judge that generates its label only after carrying out the reasoning.

RAG evaluation frameworks. RAGAS (Es et al., 2023) and ARES (Saad-Falcon et al., 2024) assess pipeline-level faithfulness, typically reference-free: RAGAS decomposes the responses into statements, followed by an LLM-based evaluation of each statement against the context provided by the retrieved information. On the other hand, ARES trains lightweight judges on synthetic data. RAGChecker (Ru et al., 2024) refines this further into claim-level entailment checks that separate retrieval failures from generation failures. Our approach is different from all of these methods in that it involves verifying each claim against the evidence retrieved for its own question, with the verification being scored on human gold labels rather than on an automatic substitute.

Closest work. A concurrent thesis from Lund University (Larsson and Kutaiba, 2026) makes a comparative assessment of hallucination detectors for RAG — including LettuceDetect and LLM-based judging — using a strict cascade approach in the context of regulatory documents. We differ in five ways: we use fully local ~8B models rather than a frontier API; our voting rule operates per claim and includes an explicit uncertainty class; our labels are graded, derived from span coverage; we evaluate on naturally occurring errors in a new hand-labelled dataset; and we carry out a mechanistic error analysis of why systems miss.

Ensemble theory (Dietterich, 2000) predicts what we observe: combining detectors is advantageous when their errors are uncorrelated, and a rule which depends on agreement fails when the correct detections made by its members do not overlap.

III. Methodology

A. Corpus

The corpus consists of six UK government 'coloured book' guidance documents (721 pages of text extracted): the Green Book (2026), the Magenta Book, the Orange Book (2026 update), the Aqua Book (2025), the Rose Book, and the Teal Book. The documents consist of two native PDF documents and four web-print captures of official government web pages. Each document's edition statement, date captured, and any known differences are listed in a provenance table which is included with the supporting material. Gold labels are relative to corpus, meaning claims are judged against the frozen passages retrieved from these exact documents and

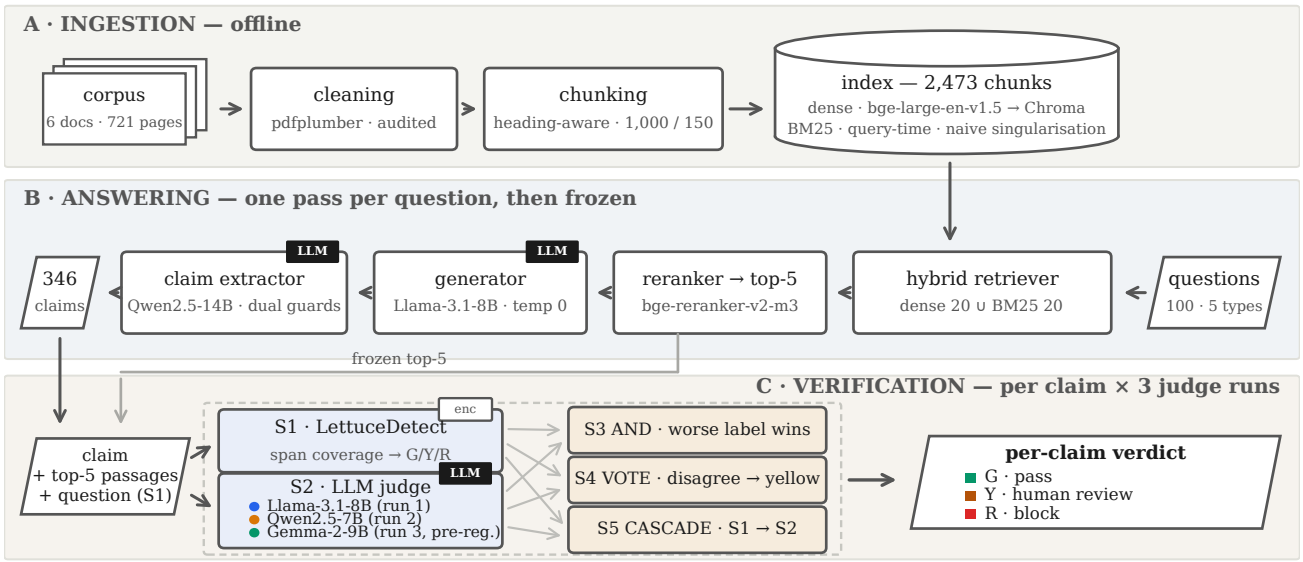


Fig. 1. The fully-local pipeline architecture: (A) offline ingestion, (B) one frozen answering pass per question, and (C) per-claim verification by the five systems across three judge runs.

never against the live web — a distinction documented by the provenance record.

B. Retrieval and generation

The PDFs are extracted using `pdfplumber` and the web-print artefacts are removed by applying a set of rules which are audited individually for each page to ensure that no content is lost. The resulting pages were split at the points of headings (Pillar/Chapter/Section/Annex/Part) before going through the recursive chunking (1,000 characters, 150 as overlap), giving 2,473 chunks embedded with `bge-large-en-v1.5` (Xiao et al., 2023) stored in ChromaDB. The retrieval method is hybrid which results in top-20 dense candidates together with top-20 BM25 candidates (Robertson and Zaragoza, 2009), i.e. applying simple singularisation, without which BM25 buried a critical chunk at rank 168, far outside its top-20 candidate window, and finally the use of a cross-encoder (`bge-reranker-v2-m3`; Chen et al., 2024) that rescored the union and produced the top-5 context. The generation model is Llama-3.1-8B (Dubey et al., 2024) at temperature 0 with a context-only prompt, numbered citations, and a fixed refusal string. The generator, extractor, and judges are run locally through Ollama using its standard quantised versions, the same in every instance; the embedder, reranker, and span detector run locally through Python libraries. Figure 1 summarises the pipeline.

C. Claim extraction

Qwen2.5-14B (Yang et al., 2024) breaks down answers into atomic claims at temperature 0. A model deliberately larger than the generator, since extraction fidelity contaminates all subsequent measurements, while the 8B generator is the system being evaluated. Two deterministic guards turned out to be more effective than prompt engineering at this scale: a refusal pre-filter (discarding rejection sentences prior to extraction and preventing the retrieval of false claims formed out of rejection echoes), and a lexical grounding guard (claims whose content words are less than 60% recoverable from the answer are dropped as extractor inventions).

D. Label scheme: green, yellow, red

Every claim receives one of three colours, evaluated solely against the five passages obtained for its question and not against the web or any external knowledge. **Green** means supported: the statement made by the claim is stated in the passages or directly follows from them. **Red** means unsupported: the claim's content is absent from the passages or contradicted by them. **Yellow** means that it is only partly supported, which is the reason why a two-way system was insufficient: a claim may have a true core and an ungrounded addition to it, overstated certainty (e.g. "must" in the case where the text uses "should") or wrong attribution (the right fact being assigned to the wrong document or section). A binary system either accepts these cases or assigns them excessive punishment; the yellow label allows for proper treatment of these claims. The labels also have deployment implications like a traffic light system: green means approval, yellow means the case should be referred to human review, and red means it is blocked. When it comes to scoring, the binary flag task simplifies the scheme by treating yellow and red together as flagged.

E. Five verification systems

Each system maps (1) a claim, (2) the top-5 passages, and (3) the question to a green/yellow/red label; internally, the span

detector uses all three, while the LLM judge conditions only on the claim and passages.

System 1 (S1): LettuceDetect produces binary unsupported spans with confidences, we obtain graded labels from span coverage which is our design not the tool's. The way this works is as follows:

- Green → No Spans
- Yellow → Spans covering <50% of the claim
- Red → Spans covering ≥50% of the claim

We ignore spans below confidence 0.3 (threshold sweep: Appendix B). The system is deterministic with no LLM calls.

System 2 (S2): Reasoning-based LLM judge in JSON format in which the model provides its reasoning first and then the label, maintaining a single-best-passage rubric (including contradictions). We used Llama-3.1-8B model for run 1; the Qwen2.5-7B model in run 2 (chosen over Qwen3-8B because its thinking mode interacts unpredictably with forced-JSON output) and the Gemma-2-9B (Gemma Team, 2024) model in run 3 (executed under pre-registration; see Section IV-G).

System 3 (S3 AND): S1 and S2 run together; the one with the worse label wins (as per Larsson and Kutaiba, 2026). It is important to note that in the case of the binary flag task, this maximum severity acts as the union of flags; the name is a result of its strict severity semantics. This severity-based AND is distinct from the agreement-based AND combinations between judges explored in Sections IV-E to IV-G.

System 4 (S4 Vote): Agreement returns the agreed label; any disagreement returns yellow — an explicit uncertainty class sending claims to human review. In this case, yellow indicates a disagreement that is different from a gold yellow (partial support).

System 5 (S5 Cascade): In S1, every claim is screened; the final 'green' label would indicate a fast path with no LLM calls. Flagged claims would go through S2, which would give a final verdict.

F. Evaluation dataset and gold labelling

We wrote 100 questions in five categories: 66 answerable, 7 partial-support (designed to produce a yellow label), 13 false-premise, 8 fabrication, and 6 abstain/unanswerable (the documents simply did not have enough information to answer). Every question was first checked against the source documents. The system produced 346 claims (12 questions resulted in no claims after refusal filtering; Section IV-A). Claims were labelled by the author against the frozen top-5 passages, with the author blind to all system outputs. Claims the author had doubts about were further read by two additional reviewers, who saw only this small subset of difficult cases; disagreements were discussed together, and the final label was always the author's call against the frozen passages. For the very difficult claims, the author also asked newer, much larger language models (Claude Fable 5 and GPT-5.6) for a second opinion, in the same way as the reviewers; neither model is among the evaluated systems, and the author decided every final label. As the reviewers did not independently label the full claim set, inter-annotator agreement could not be calculated, which is a limitation. One claim was excluded as an extraction error, and a total of 345 claims were analysed: 303 green (87.8%), 27 yellow, and 15 red.

G. Metrics and protocol

As 87.8% of the claims were green, raw accuracy would not give a useful picture of how well the system performed. For example, a system could simply label everything green and still get 87.8% accuracy, despite failing to identify any incorrect claims. Therefore, a binary flagging task was used for all scoring, where a claim is positive if its gold label is yellow or red (42 claims) and flagged if the system outputs yellow or red. Precision and recall are reported with Wilson 95% intervals (Wilson, 1927). F1 intervals are derived from a bootstrap that resamples whole questions (10,000 draws), because claims from the same question share an answer and passages and are therefore not independent. To compare two systems we conduct exact McNemar tests (McNemar, 1947) on the claims where they disagreed, run separately for the gold non-green claims (who detects more?) and the gold green claims (who false-alarms more?); we do this because there were many more green claims, and combining everything could make a system that rarely flags anything appear better than it really is. With ten system pairs per judge run, p-values are Holm-corrected (Holm, 1979). Wilson intervals and McNemar tests treat claims as independent; only the F1 intervals are resampled at question level. Recomputing the precision and recall intervals at question level shifts each bound by only a few points, and the 11 detections behind the strongest McNemar result come from 9 different questions rather than one, so the conclusions remain the same. S4's yellow label means that its detectors disagreed, rather than meaning that the claim had partial support. Therefore, systems were compared only on whether they flagged a claim or did not flag it. Generation was run *once* and frozen: every system judged the same claims against the same passages. The thresholds were decided before scoring, each run's configuration is stored with its outputs, and one analysis script recomputes the principal derived statistics.

IV. Results

A. Natural hallucinations are rare and hide in answerable questions (RQ1)

Of 345 evaluated claims, 42 (12.2%) are gold non-green. This is a claim-level statistic regarding the extracted claims and not the hallucination rate at the level of the answers; at the level of the questions, no less than 30 out of 88 claim-bearing questions contain at least one gold non-green claim, leaving a full answer-level correctness and abstention analysis as future work. Twelve questions produced no claims after refusal filtering: eleven from the adversarial types (4/6 abstain, 3/8 fabrication, and 4/13 false-premise) and one partial-support question, for which refusal is a conservative miss rather than correct behaviour. Out of 42 erroneous claims, 6 originate from the three adversarial question types, and a further 3 come

TABLE I

FLAG TASK (POSITIVE = GOLD YELLOW∪RED, n=42 OF 345 SCORED CLAIMS). WILSON 95% INTERVALS IN BRACKETS; F1 INTERVALS ARE QUESTION-CLUSTER BOOTSTRAPS (10,000 RESAMPLES; SCRIPT IN SUPPORTING MATERIAL). "REDS CAUGHT" COUNTS GOLD-RED CLAIMS FLAGGED AT ANY SEVERITY. LLM CALLS ARE PER CLAIM; PER-CLAIM LATENCY APPEARS IN SECTION IV-D.

System	Judge	Flagged	TP	FP	P % [95% CI]	R % [95% CI]	F1 % [95% CI]	Reds	LLM calls/claim
S1 lettuce	—	2	2	0	100.0 [34.2–100]	4.8 [1.3–15.8]	9.1 [0–27.3]	0/15	0
S2 llm	llama	25	6	19	24.0 [11.5–43.4]	14.3 [6.7–27.8]	17.9 [6.2–30.3]	2/15	1.0
S3 and	llama	26	7	19	26.9 [13.7–46.1]	16.7 [8.3–30.6]	20.6 [6.8–34.9]	2/15	1.0
S4 vote	llama	26	7	19	26.9 [13.7–46.1]	16.7 [8.3–30.6]	20.6 [6.8–34.9]	2/15	1.0
S5 cascade	llama	1	1	0	100.0 [20.7–100]	2.4 [0.4–12.3]	4.7 [0–14.6]	0/15	0.006
S2 llm	qwen	27	9	18	33.3 [18.6–52.2]	21.4 [11.7–35.9]	26.1 [11.8–40.0]	3/15	1.0
S3 and	qwen	29	11	18	37.9 [22.7–56.0]	26.2 [15.3–41.1]	31.0 [15.4–44.7]	3/15	1.0
S4 vote	qwen	29	11	18	37.9 [22.7–56.0]	26.2 [15.3–41.1]	31.0 [15.4–44.7]	3/15	1.0
S5 cascade	qwen	0	0	0	—	0.0 [0–8.4]	0.0	0/15	0.006
S2 llm	gemma	36	7	29	19.4 [9.8–35.0]	16.7 [8.3–30.6]	17.9 [7.2–28.2]	2/15	1.0
S3 and	gemma	37	8	29	21.6 [11.4–37.2]	19.0 [10.0–33.3]	20.3 [8.0–32.2]	2/15	1.0
S4 vote	gemma	37	8	29	21.6 [11.4–37.2]	19.0 [10.0–33.3]	20.3 [8.0–32.2]	2/15	1.0
S5 cascade	gemma	1	1	0	100.0 [20.7–100]	2.4 [0.4–12.3]	4.7 [0–14.6]	0/15	0.006

from partial-support questions. Thus, **33 out of 42 hallucinated claims came from ordinary answerable questions**, and they are subtle: a risk-response block attributed to the wrong categories (five claims in one answer), a corrupted welfare-weight formula, and a fabricated purpose statement for the Magenta Book. Finally, in terms of three-way accuracy, the base rate is essentially unbeatable (the best system achieves 88.1% vs. 87.8% for predicting all-green), which motivates the flag-task view in Table I.

B. The span detector is almost blind to natural errors (RQ2)

LettuceDetect is capable of flagging 2 claims out of 345 seen claims, both of them being true ones resulting in 100% precision and 4.8% recall. Even without any sort of confidence threshold, when rescoring all claims, **40 of the 42 gold non-green claims produce no spans whatsoever**, producing an AUROC of 0.524, which is near chance (tie-aware average precision 0.164 at prevalence 0.122). As a determinism check, the detector produced identical labelling on all 346 labels across the two runs that included it. A detector which has been trained on the frequent hallucinations of earlier generators when performing standard benchmark tasks (Niu et al., 2024) is unable to identify mistakes that are just fluent rewordings or missing-support problems of a grounded temperature-0 pipeline in this domain.

C. 8B judges are weak — and differently weak (RQ2)

The Llama judge catches 6/42 with the precision of 24%. Qwen judge catches 9/42 having 33% precision, improving every metric. The pre-registered third family, Gemma-2-9B catches 7/42 at the worst precision 19%. They all fail to detect the 15 gold-red errors when the severity is red; when it is flagged with yellow, the counts are: 2 for Llama, 3 for Qwen, and 2 for Gemma, as shown in the "Reds" column of Table I. For the scale involved, the poor judging isn't just due to one bad model choice since all three ∼8B families we could run locally are a long way from being usable on subtle natural errors.

D. Ensembles cannot rescue a silent member (RQ3, RQ4)

Since S1 contributes almost no signal, S3/S4 will effectively become the same as S2 because S1 has flagged only two claims, and the cascade will inherit the screener's blindness. Under Qwen, the cascade drops to 0/42 because the referee has overturned the screener's only two legitimate flags (under Gemma it confirms one: 1/42). The cascade's cost win is real since it involves only 2 calls from LLMs rather than 345, the time per claim dropping from roughly 5–7 seconds to about 0.7–1.9 seconds depending on the judge run; yet it provides almost no detection regardless of which judge is used. A cascade's recall is bounded by its screener, and a disagreeing referee can drop it to zero.

E. Exploratory: judge families catch disjoint errors

This analysis was developed after seeing the results and is reported as exploratory, not as a pre-registered system. It has been found that while the two judges agree on 89.9% of claims overall, they never identify the same error (Llama 6, Qwen 9, shared 0), while their false alarms significantly overlap (19 and 18, shared 11; Figure 2, which also shows the third judge of Section IV-G; Section IV-F shows that the overlap is mostly one claim type). The result (Table II, Figure 3) shows that when either of the two judges flags a claim, recall rises to 35.7% and precision to 36.6%, both greater than the results of either judge individually. The reason being that the judges catch different real errors while many of their false alarms are the same. Therefore, their union includes all the different errors they catch whilst the 11 overlapping false alarms are only counted once. This increases the F1 to 36.1%: 1.4× the better single judge, Qwen (26.1%), and 2.0× the Llama judge. Adding LettuceDetect finds one more error and increases precision, recall, and F1 all to 38.1%. The opposite rule, where a claim is flagged only when both judges

TABLE II
EXPLORATORY JUDGE COMBINATIONS ON THE FLAG TASK. F1 INTERVALS: QUESTION-CLUSTER BOOTSTRAP.

Combination	Flag	TP	FP	P %	R %	F1 % [95% CI]
llama alone	25	6	19	24.0	14.3	17.9 [6.2–30.3]
qwen alone	27	9	18	33.3	21.4	26.1 [11.8–40.0]
gemma alone	36	7	29	19.4	16.7	17.9 [7.2–28.2]
OR(llama, qwen)	41	15	26	36.6	35.7	36.1 [22.7–48.3]
+ lettuce	42	16	26	38.1	38.1	38.1 [24.1–50.5]
OR(all three)	56	17	39	30.4	40.5	34.7 [21.1–46.9]
Majority (2-of-3)	22	5	17	22.7	11.9	15.6 [3.8–27.3]
AND(llama, qwen)	11	0	11	0.0	0.0	0.0
AND(all three)	10	0	10	0.0	0.0	0.0

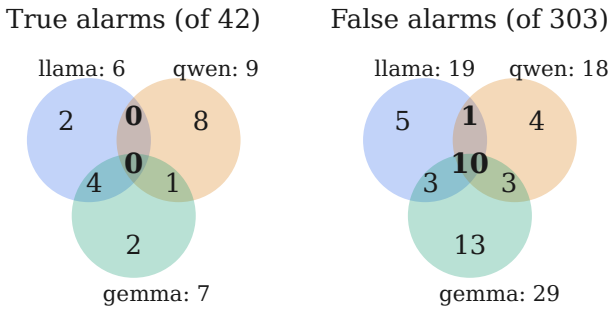


Fig. 2. True-alarm and false-alarm overlap across the three judge families.

flag it, results in 11 flags, all of them being false alarms as the judges agree only on their shared mistakes. This shows that the ensemble works solely due to the diversity amongst the judge families rather than the consensus between them.

F. The shared false alarms are mostly meta-claims

Our labelling flagged 23 scored claims as meta-claims, statements about what the sources do or do not mention rather than statements about the real world — for example, "the passages do not mention X". Twenty-two are gold green and one yellow. The judge's task is to look for a passage that supports the claim. However, a meta-claim tells us that something does not exist in the sources, so no passage directly states it. A judge that is trying to find supporting text will flag it even when the absence is real. This causes many false alarms: 12 of Llama's 19 false alarms are meta-claims, 14 of Qwen's 18, and **10 of the 11 overlapping false alarms**. If we exclude these 23 meta-claims (a post-hoc analysis; 322 claims, 41 positives; Table III), Llama's and Qwen's precision improves to almost double (Qwen 69.2%) while Gemma's improves less (27.3%), the rest of the judges' false alarms barely overlap (one claim), and the OR-combination still has the highest judge-only F1 of 43.1% (45.5% when the span detector is added), although its precision of 58.3% is lower than Qwen's 69.2%. Therefore a conclusion can be reached that diversity in judge families helps in improving F1 and recall at the expense of precision, and the apparent "shared family bias" in false alarms is not a deep weakness of the model families themselves; it mostly comes from meta-claims not fitting the judges' task of finding supporting evidence. This is an actionable insight: since meta-claims can plausibly be detected automatically by simple text patterns, they could be sent to a separate, dedicated check instead.

G. A pre-registered third judge: disjointness is pair-dependent

Before testing Gemma-2-9B, we made two predictions (Appendix B): (P1) Gemma would catch mostly different errors from Llama and Qwen, (P2) its false alarms would mostly be meta-claims. The result is mixed. For P1, the hypothesis is partially refuted as Gemma catches 7 out of 42 real errors, but 4 of these overlap with Llama's catches (pairwise overlaps: Llama–Qwen 0, Qwen–Gemma 1, Llama–Gemma 4; three-way 0). This shows that the disjointness between Llama and

TABLE III
FLAG TASK EXCLUDING THE 23 META-CLAIMS (322 CLAIMS, 41 POSITIVES). F1 INTERVALS: QUESTION-CLUSTER BOOTSTRAP.

System	Flag	TP	FP	P %	R %	F1 % [95% CI]
llm/llama	12	5	7	41.7	12.2	18.9 [4.9–33.3]
llm/qwen	13	9	4	69.2	22.0	33.3 [16.2–48.6]
llm/gemma	22	6	16	27.3	14.6	19.0 [7.1–30.8]
OR(llama, qwen)	24	14	10	58.3	34.1	43.1 [28.0–56.3]
+ lettuce	25	15	10	60.0	36.6	45.5 [30.5–59.0]
OR(all three)	39	16	23	41.0	39.0	40.0 [24.7–53.2]
AND(llama, qwen)	1	0	1	0.0	0.0	0.0

Qwen was only specific to their case and doesn't necessarily mean that it will happen among every group of judges. Gemma only adds 2 catches and 13 new false alarms beyond the Llama–Qwen union (its 29 false alarms in total are the most of any judge). As a result, while adding Gemma as a third judge increases recall to 40.5%, it also lowers F1 (34.7% vs. 36.1% with meta-claims; 40.0% vs. 43.1% without), showing that adding another judge can increase the number of errors detected but can also introduce too many false alarms. P2 — registered as "its false alarms would concentrate on meta-claims" — is partially supported: 45% of Gemma's own false alarms are meta-claims — a six-fold enrichment over their 6.7% base rate, though not a majority — and the false alarms it shares with Llama or Qwen are mostly meta-claims: 11 out of 13 with Llama and 12 out of 13 with Qwen. All 10 claims flagged by all three judges are false alarms, and even a 2-of-3 majority manages only 5 true positives, with an F1 of 15.6%. Overall the results show that the judges tend to agree on the same false alarms because of the meta-claim problem shared by all, but different judges do not necessarily catch different errors like Llama and Qwen did, and the benefits of combining judges depend on which judges are combined.

H. Statistical considerations

With only 42 positives, differences among the judge-based systems (llm, AND, vote) are not statistically resolvable — consistent with S3/S4 collapsing to approximately S2 (Section IV-D). For the Llama judge, no detection-subset pair at all is significant after Holm correction, as all adjusted p-values were at least 0.31. For the Qwen judge, the pairs involving the near-silent screeners are significant: AND/vote detect more

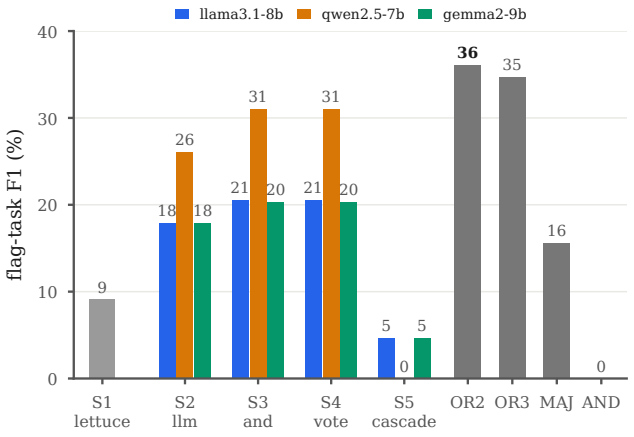


Fig. 3. Flag-task F1 by system and judge, with exploratory judge combinations.

than the cascade (11 vs. 0 discordances, $p_{\text{adj}} = 0.0098$), the LLM judge more than the cascade (9 vs. 0, $p_{\text{adj}} = 0.031$), and AND/vote more than Lettuce (9 vs. 0, $p_{\text{adj}} = 0.031$). As for the Gemma judge, none of the detection pairs are statistically significant, with the smallest adjusted p-value being 0.156. The differences in false alarms are much clearer in all three runs: lettuce/cascade versus the LLM-based systems, 19 (Llama), 18 (Qwen), and 29 (Gemma) discordant claims all in one direction, Holm-corrected $p \leq 7.6 \times 10^{-5}$ (Llama/Qwen) and 3.7×10^{-8} (Gemma). With 15 gold reds, an observed red-recall of 0/15 remains consistent (95%) with a true recall up to ~18%.

V. Discussion

A. Error analysis: what the 42 hallucinations are

I manually examined all 42 gold non-green claims and divided them into six categories (Table IV; single-coder qualitative coding by the author, shipped claim-by-claim with the supporting material). The most common type, with 16 out of 42 errors, is *wrong attachment*, where all facts in a claim are present in the passages but are connected incorrectly. For example, one answer attached the definitions of the Teal Book's risk-response categories to the wrong categories, producing 5 red claims where every word is individually well supported by the passages. Another common type is overstatement, with 11 claims, all yellow. These claims exaggerate what the sources say, for example turning "should" into "must" or broadening scope beyond the passages' content. Cross-document blending happens when the AI blends information from different documents or incorrectly connects information from one to another; for example, a "purpose" statement for the Magenta Book was created by the AI using the Green Book's value-for-money language. Overall, **24 of 42 errors (57%) could not be identified by any system under any of the three judges**; wrong-attachment errors account for the largest number of missed errors (9 of the 24).

TABLE IV
Taxonomy of the 42 gold non-green claims (manual coding). "Caught" = flagged by at least one of the five systems under any of the three judges.

Category	Red	Yellow	Caught
Wrong attachment / relation swap	11	5	7/16
Overstated modality or scope	0	11	5/11
Cross-document blending, invented links	1	5	2/6
Premise-echo survivals (trap questions)	1	4	2/5
Numeric / enumeration corruption	2	1	1/3
Meta-claims about the sources	0	1	1/1
Total	15	27	18/42

Why is the span detector blind? The hallucinations here are not like RAGTruth's. The responses in RAGTruth come from earlier, less reliable generators which were tasked with answering benchmark questions such as those found on the web, along with reviews and news items, where hallucinations are common, whereas our errors consist largely of paraphrased recombinations — connecting the correct facts in the wrong way — and are expressed in formal, bureaucratic language that differs from the distribution of data used to train the detector. Our single-corpus design cannot isolate which shift is responsible — the domain, the generator, the task format, the claim-level scoring, or the error-type mix all differ. Wrong-attachment errors are the worst case for a token-level detector, as no individual token is wrong or unsupported; the problem is the relationship between the words. This results in a gap between benchmark performance and real-world performance rather than simply being a problem fixable by altering the detector's threshold (Section IV-B rules out thresholds).

Why judges disagree productively. The disagreement between judges is beneficial because they have different training and instruction-tuning, so they are sensitive to different kinds of errors and each catches a different slice of them. However, the mistakes they share are almost all meta-claims (Section IV-F). This means the shared false alarms come from the judging task itself rather than from a deep shared bias in the model families: once meta-claims are excluded, the two judges' false alarms overlap on only a single claim — if the families shared a deep bias, that overlap would be expected to remain. This is ensemble theory made concrete. It is the diversity of errors that brings in value, and in turn an agreement rule collects exactly the mistakes both members share. The third family reinforces this concept: whether or not a new judge's catches add value depends on the pair (Gemma largely re-caught Llama's slice), while the meta-claim problem remains the one error all the judges share.

Practitioner implications. (1) Do not deploy a detector based on benchmark numbers; validate on your own traffic — ours went from benchmark-competitive to near-chance on our corpus. (2) If ensembling LLM judges, make sure to use

diverse pairs of them and apply union or uncertainty semantics; consensus rules can give worse results than any individual member. (3) A cascade is only as good as its screener, and its referee can reject even the screener's correct catches. (4) Meta-claims should be identified and handled separately before the normal judging, since they are recognisable by simple text patterns and caused most of the judges' false alarms in this study.

Limitations. The number of genuine errors was small, with only 42 positive claims, which makes some of the statistical comparisons less reliable. The 12.2% rate is the claim-level rate within our constructed evaluation set, whose question mix was deliberately varied; it is not an estimate of how often hallucinations would occur in real-world usage. The OR-combination and the meta-claim analysis provide post-hoc results and require confirmation on data they were not designed on. Gold labels were assigned by a single primary annotator, with difficult cases resolved through consensus, but there was no independent agreement statistic. The error-taxonomy categories are also a single coder's judgments. Wilson intervals and McNemar tests assume the claims are independent, even when several come from the same question. Additionally, the study used only one corpus and domain and one generator, and every system depends on the same claim extractor, so the results may not apply to other datasets or AI systems. Finally, the labels are corpus-relative by design, meaning that a claim supported by the corpus may still be false in the real world.

VI. Conclusion and future work

On naturally occurring hallucinations in a fully-local RAG pipeline over UK government guidance: hallucinations are relatively uncommon (12.2%) and mostly occur in normal answerable questions (RQ1). Adding to this, a RAGTruth-trained span detector has almost no success in detecting any hallucinations (2 out of 42 cases with an AUROC of 0.52) and an 8B LLM judge performs quite poorly (from 6 to 9 out of 42 cases), with different judge families detecting different types of errors (RQ2). Combination rules cannot save a silent member, as AND/vote collapse into the judge, and the cascade trades 99% of LLM calls for nearly no detection (RQ3 and RQ4). In contrast, the exploratory OR of two judge families increases F1 markedly from 26.1% to 36.1% (43.1% excluding meta-claims) by exploiting disjoint catches, at the expense of precision once meta-claims are excluded. Most of the judges' false alarms (39 of the 66 across the three judge runs) are on meta-claims — statements about what the sources do not say, which cannot be verified by searching for supporting text. A pre-registered third family showed the disjointness is dependent on the pair rather than being universal, as Gemma often detected errors that Llama had already found and added only two new catches beyond the two-judge union while producing 29 false alarms of its own, while the meta-claim problem remained consistent and the consensus between the three judges still produced only false alarms.

There are many possible avenues for future research. The first of these consists in carrying out the study once again using larger local judges, for example the 27B or 32B models on a more powerful machine, in order to find out whether the weak judging is due to the limitation of the 8B model or whether it is a general problem with local judging. Meta-claims should be identified and handled separately before being passed on to the LLM judges, and a judge prompt which is specifically instructed how to verify absence claims should be tested on new questions under pre-registration. A pre-registered induced-hallucination stratum should also be used to see if the OR-advantage still shows up and whether the span detector recovers on errors closer to its training distribution. As the advantage obtained from combining judges depends on which judges are combined together, it would be helpful for practitioners if there were a low-cost method for estimating the degree of overlap between two judges in advance. Moreover, answer-level correctness and abstention — that is, when the system should refuse to answer — should also be examined. Finally, the span detector could be fine-tuned using data that is specific to the domain, and multiple independent annotators should be employed in order to calculate an inter-annotator agreement statistic.

Acknowledgements

The author thanks the two reviewers who gave second opinions on the difficult gold-label cases, and Syed Rafee for supervision of this project.

References

Chen, J., Xiao, S., Zhang, P., Luo, K., Lian, D. and Liu, Z. (2024), BGE M3-Embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation, *in* 'Findings of the Association for Computational Linguistics: ACL 2024'.

Dietterich, T. G. (2000), Ensemble methods in machine learning, *in* 'Multiple Classifier Systems'.

Dubey, A. et al. (2024), 'The Llama 3 herd of models', *arXiv preprint arXiv:2407.21783*.

Es, S., James, J., Espinosa-Anke, L. and Schockaert, S. (2023), 'RAGAS: Automated evaluation of retrieval augmented generation', *arXiv preprint arXiv:2309.15217*.

Gemma Team (2024), 'Gemma 2: Improving open language models at a practical size', *arXiv preprint arXiv:2408.00118*.

Holm, S. (1979), 'A simple sequentially rejective multiple test procedure', *Scandinavian Journal of Statistics* **6**(2).

Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A. and Fung, P. (2023), 'Survey of hallucination in natural language generation', *ACM Computing Surveys* **55**(12).

Kovács, Á. and Recski, G. (2025), 'LettuceDetect: A hallucination detection framework for RAG applications', *arXiv preprint arXiv:2502.17125*.

Larsson, D. and Kutaiba, Y. (2026), A comparative evaluation of hallucination detection methods for RAG systems, Master's thesis, Lund University. LUP Student Papers 9240231.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel,

T., Riedel, S. and Kiela, D. (2020), Retrieval-augmented generation for knowledge-intensive NLP tasks, *in* 'Advances in Neural Information Processing Systems 33'.

Liu, Y., Iter, D., Xu, Y., Wang, S., Xu, R. and Zhu, C. (2023), G-Eval: NLG evaluation using GPT-4 with better human alignment, *in* 'Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing'.

McNemar, Q. (1947), 'Note on the sampling error of the difference between correlated proportions or percentages', *Psychometrika* **12**(2).

Niu, C., Wu, Y., Zhu, J., Xu, S., Shum, K., Zhong, R., Song, J. and Zhang, T. (2024), RAGTruth: A hallucination corpus for developing trustworthy retrieval-augmented language models, *in* 'Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics'.

Robertson, S. and Zaragoza, H. (2009), 'The probabilistic relevance framework: BM25 and beyond', *Foundations and Trends in Information Retrieval* **3**(4).

Ru, D., Qiu, L., Hu, X. et al. (2024), RAGChecker: A fine-grained framework for diagnosing retrieval-augmented generation, *in* 'Advances in Neural Information Processing Systems 37'.

Saad-Falcon, J., Khattab, O., Potts, C. and Zaharia, M. (2024), ARES: An automated evaluation framework for retrieval-augmented generation systems, *in* 'Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics'.

Warner, B., Chaffin, A., Clavié, B., Weller, O., Hallström, O., Taghadouini, S., Gallagher, A., Biswas, R., Ladhak, F., Aarsen, T., Cooper, N., Adams, G., Howard, J. and Poli, I. (2024), 'Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference', *arXiv preprint arXiv:2412.13663*.

Wilson, E. B. (1927), 'Probable inference, the law of succession, and statistical inference', *Journal of the American Statistical Association* **22**(158).

Xiao, S., Liu, Z., Zhang, P. and Muennighoff, N. (2023), 'C-Pack: Packaged resources to advance general Chinese embedding', *arXiv preprint arXiv:2309.07597*.

Yang, A. et al. (2024), 'Qwen2.5 technical report', *arXiv preprint arXiv:2412.15115*.

Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E. P., Zhang, H., Gonzalez, J. E. and Stoica, I. (2023), Judging LLM-as-a-judge with MT-Bench and Chatbot Arena, *in* 'Advances in Neural Information Processing Systems 36'.

Appendix A
Generative AI Tools – Student Accountability Statement

Student Name: Moiz Shahid
Project Title: Detecting Natural Hallucinations in Local RAG: Span Detectors, LLM Judges, and the Limits of Ensembling
Supervisor Name: Syed Rafee

Generative AI Student Declaration

- Yes, I have used generative AI tools in this project.
- I have declared all uses of AI tools (e.g., coding assistants, text generators, data analysis tools) in this project.
- I have reviewed and verified all AI-assisted content to ensure accuracy and suitability.
- I take full responsibility for the originality and integrity of my project submission.
- I have referenced all sources, including those suggested or produced with AI support.
- I have not used AI in ways that undermine academic honesty (e.g., plagiarism, unverified code submission).
- My use of AI was supplementary and did not replace my own independent learning and problem-solving.

Signed: Date: 26 August 2026

TABLE V
List of AI tools used in this project.

AI Tool	Version	Publisher	Access URL
ChatGPT	5.6 Sol Ultra	OpenAI	`https://chatgpt.com/`
Claude	Opus 4.8	Anthropic	`https://claude.ai/`
Claude	Opus 5	Anthropic	`https://claude.ai/`
Claude	Fable 5	Anthropic	`https://claude.ai/`

TABLE VI
Declared use of generative AI tools in this project.

Project Section	Description	Justification
Software development — code syntax	I used AI's assistance with the Python syntax for the pipeline code as well as helping out with difficult sections involving code interpretation and debugging.	Python is not my primary programming language, since I work in JavaScript regularly, so this was often confusing. The AI generated syntax, with me adapting it. The way the pipeline logic works is of my making, as I ran and debugged each and every line of code using my database.
Software development — results and evaluation tooling	I used Claude to help me come up with ideas about how to compare my results, and with the syntax of the evaluation scripts. I also used it in the generation of statistical tests and metric calculations.	Many result files (gold labels, verdict files from numerous runs) are produced in this project that must be compared consistently. While the comparison design, the selection of metrics and the comprehension of all results are my own, I took help with script syntax where I needed to. Every reported number is regenerated from my own experiments' outputs.
Gold labelling — difficult claims	While labelling the gold dataset I occasionally asked very large language models (such as Claude Fable 5 and GPT-5.6) for a second opinion on difficult claims, where I was sometimes unsure about which of the labels to give out after reading the passages (e.g., yellow or red).	Several claims were very difficult to judge against the passages. I have used AI as an extra check in addition to the two human reviewers. However, the final label for every claim was decided by me against the frozen passages, and the systems under evaluation never had access to the gold labels.
Report — drafting and language	I wrote the drafts of all sections myself and used AI in order to help me in paraphrasing, proofreading for grammar errors, checking sentence clarity and suggesting relevant vocabulary for my drafts. Moreover, I also used AI to help me understand complicated concepts and formulas clearly as well as for brainstorming and LaTeX assistance.	Because English is not my native language, I am not always confident about the clarity of my writing. While I used my own ideas and findings to write content, make arguments and reach conclusions, AI-suggested wording was used here and was revised by me before being kept.
Report — tables, figures and consistency checks	I took help in creating the report tables and verifying them, shifting results into tables, making sure no figure or result was missing, and keeping the numbers and cross-references consistent throughout the report.	The report contains various numbers included in the tables and figures. I asked AI to cross-check them against my result files so that any mistake in transcription would be caught. However, the results themselves are obtained from my own experiments and I reviewed every check myself.

Appendix B
Supplementary material

Corpus provenance

Gold labels are based only on the frozen passages retrieved from the corpus (never the live web), so provenance is part of the methodology (Table VII). Web-print artefacts (per-page date/URL headers and footers, `(cid:N)` glyphs) were removed during ingestion and each procedure was checked page by page to ensure the actual content had not been lost. The Teal Book capture contains statements that differ from the public record (for example, a "third edition" self-description). Because labels are checked only against the corpus, the affected questions can be used to test whether the AI follows the information provided in the document or relies on its own knowledge that it has been trained on.

TABLE VII
Corpus documents.

Document	Edition / capture	Pages
Green Book	2026 edition (native PDF)	85
Magenta Book	May 2026 (dated cover page; native PDF)	157
Orange Book	updated 3 June 2026; GOV.UK print, 25/06/2026	50
Aqua Book	2025 update; GOV.UK print, 25/06/2026	55
Rose Book	updated 27 Mar 2024; GOV.UK print, 25/06/2026	63
Teal Book	projectdelivery.gov.uk capture	311

Confusion matrices

Table VIII provides complete three-class confusion cells obtained from every system under all three judges (345 scored claims). S1 is independent of the judge, with identical labelling under the two runs that included it (346 out of 346 claims). The only difference between the output files was the recorded latencies. It was not tested again in the Gemma run.

LettuceDetect threshold sweep

Sweeping the span-confidence threshold t (flag if any span with confidence $\geq t$): $t \in \{0.001, 0.05, 0.1, 0.2, 0.3, 0.5\}$ all flag the same 2 claims (both true positives); t=0.7 flags 1; t=0.9 flags 0. Out of the 42 gold non-green claims, 40 produce zero spans at every threshold while the remaining two nonzero claims (max confidences 0.76 and 0.65, coverages 1.0 and 0.4) come from a single question. AUROC of max span confidence: 0.524. Because 343 out of 345 claims had a score of zero, naive average precision is order-dependent; using the tie-aware calculation, the value is 0.164, compared with a 12.2% prevalence of non-green claims.

Pre-stated predictions for a third judge family

Before any Gemma-2-9B verdicts, I predicted for a third judge family (gemma2:9b) on the same frozen claims, based on Sections IV-E and IV-F: (P1) its true catches would mostly be disjoint from both existing judges' and (P2) its false alarms would concentrate on meta-claims. Results partially disproved P1 while P2 was partially supported, mainly for shared errors between the judges, as reported in Section IV-G.

Supporting material archive

The submitted archive is comprised of all pipeline and experiment code, the frozen generations from 100 questions with 346 claims and their top-5 passages, the gold labels, the verdict files for all three judge runs with per-run configuration records, results from the evaluations, the per-claim taxonomy assignments (`analysis/taxonomy.csv`), one analysis script (`analysis/paper_stats.py`) that recomputes the principal derived statistics in this paper, and a README with instructions about how to reproduce results (models, versions, and commands).

TABLE VIII
Confusion cells, gold→predicted. Column g→y = gold green predicted yellow, etc.

System	Judge	g→g	g→y	g→r	y→g	y→y	y→r	r→g	r→y	r→r
lettuce	—	303	0	0	25	1	1	15	0	0
llm	llama	284	16	3	23	4	0	13	2	0
and	llama	284	16	3	22	4	1	13	2	0
vote	llama	284	19	0	22	5	0	13	2	0
cascade	llama	303	0	0	26	1	0	15	0	0
llm	qwen	285	9	9	21	6	0	12	3	0
and	qwen	285	9	9	19	7	1	12	3	0
vote	qwen	285	18	0	19	8	0	12	3	0
cascade	qwen	303	0	0	27	0	0	15	0	0
llm	gemma	274	23	6	22	5	0	13	2	0
and	gemma	274	23	6	21	5	1	13	2	0
vote	gemma	274	29	0	21	6	0	13	2	0
cascade	gemma	303	0	0	26	1	0	15	0	0