UNH @ Rag4Reports: A Broad Exploration of LLM-Judges for RAG

Minna Tran, Ryan McCarthy, Aiden Parsons, Jaren Unzen, Laura Dietz
University of New Hampshire, USA
minna.tran@unh.edu, mryan9393@gmail.com, aidenmp2323@gmail.com, jarenunh@gmail.com, dietz@cs.unh.edu

Abstract

We submitted a breadth of LLM-as-a-Judge approaches to Rag4Reports Task A; our top method ranked first among all submitted systems. We find that citation faithfulness is the most essential signal, and that content is best verified by checking whether cited documents cover nuggets generated from the LLM's internal knowledge.

1 Introduction

An AutoJudge, also called LLM-as-a-Judge, is an automated pipeline for predicting the quality of Retrieval-Augmented Generation (RAG) systems. Given a set of topics, each with a problem statement and background information, and the responses produced by multiple RAG systems, an AutoJudge assigns each system a quality score and produces a leaderboard that ranks systems from best to worst. The Rag4Reports challenge is based on the TREC 2025 RAGTIME dataset, in which each topic consists of a short title, a problem statement, and a user background description.

An AutoJudge evaluates each system output as a triple: topic, retrieved documents, and generated response. The quality score decomposes along the three pairwise relationships among these elements: whether the response addresses the topic, whether the retrieved documents are relevant to the topic, and whether the response is faithful to and properly cited from the retrieved documents. These are aggregated along two dimensions, *nuggets* (atomic pieces of information a good response must contain) and *citation* (whether claims are backed by valid citations to retrieved documents), which we combine into an F1 score.

To evaluate the AutoJudge itself, we compare its system-level ranking against a reference ranking produced by human annotators using Kendall's τ, Spearman's ρ, and Pearson's r. We focus on Kendall's τ as leaderboard quality which it measures pairwise ordering agreement among systems.

The AutoJudge systems in this paper were developed by students in a Spring 2026 undergraduate/graduate course, *Research on Text, Knowledge, and LLMs*. Each student independently designed an AutoJudge and entered it in the Rag4Reports challenge. For method development we used the RAGTIME subset of the TREC AutoJudge Pilot Dataset v0.2. The remainder of this paper presents these systems and compares their leaderboards against the human reference.

2 Related Work

Nugget-based Evaluation. A nugget is an atomic piece of information that a human assessor decides is essential to a response (Voorhees, 2004; Nenkova and Passonneau, 2004). Counting how many essential nuggets a response covers yields a content-relevance metric that is more tractable to annotate consistently than full-response relevance. Voorhees distinguished *vital* nuggets, which any good answer must contain, from other relevant nuggets, which are nice-to-have, and computed recall only over vital nuggets so that importance is embedded into the metric.

Citation Evaluation. Where nuggets ask whether the right content is present, citation evaluation asks whether the content is attributable to a cited source. Liu et al. (2023) found that only 51.5% of sentences generated by commercial search engines were fully supported by their citations, and only 74.5% of citations actually supported the claims they were attached to. Citation quality cannot be assumed. Nugget-based content and citation-based attribution form the dominant axes along which RAG outputs are evaluated in AUTO-ARGUE (Walden et al., 2026), and they are the basis for the scoring criteria used in AutoJudge.

Proceedings of the First Workshop on Multilingual Report Generation via Retrieval Augmented Generation (RAG4Reports), pages 71–76
July 4, 2026 ©2026 Association for Computational Linguistics

LLM-as-a-Judge Biases. LLM-based judges are easier to scale than human assessment but exhibit systematic biases: passage length and surface lexical cues inflate scores (Yu et al., 2026), the mere presence of reasoning regardless of its factual content sways judgement (Tu et al., 2026), and judges favor outputs that resemble their own style (Roytburg et al., 2026). These behaviors are artifacts of training on signals (length, fluency, lexical overlap) that correlate with quality without causing it.

Decomposing the Judgement. Several methods make LLM judgement more auditable by decomposing it into checkable parts. Rubric-based evaluation breaks a scalar judgement into explicit, checkable criteria (Dhole and Agichtein, 2026; Dietz, 2024). Chain-of-thought prompting asks the model to produce a reasoning trace before its answer (Wei et al., 2022). The Selection-Inference framework of Creswell et al. (2022) goes further by splitting each step into two modules: Selection picks relevant facts, and Inference produces a new fact from that selection without access to the original question, denying the model the inputs that would let it produce a memorized answer.

3 Approaches

3.1 Citation Accuracy

LLM: `cross-encoder/nli-deberta-v3-base` (DeBERTa-v3 NLI, 3-class).

For each response, fragments with non-empty citations are retained. For each cited fragment, the first cited document that resolves in the response's document set is paired with the fragment text; remaining citations on the fragment are ignored. Each pair is scored by 3-class NLI yielding logits `[contradiction, entailment, neutral]`, and the pair is marked supported iff entailment is strictly the maximum (`s[1] > s[0]` and `s[1] > s[2]`). The final score is supported fragments divided by total cited fragments, or 0 if no fragment cited anything.

3.2 Retrieval Quality

LLM: `gpt-4.1-nano` (used for both nugget generation and grading).

For each topic, the LLM is prompted via the Nugget Creation prompt to decompose the question into atomic sub-questions ("nuggets"). For each response, a single retrieval context is built by taking all retrieved documents, keeping the first 20, truncating each to 1000 characters, and joining with a separator. For each (retrieval context, nugget) pair, the LLM is prompted via the Retrieval Grading prompt to return a single number (2 = fully answers, 1 = partially answers, 0 = does not answer). The final score is the sum of nugget grades divided by $2 \times n_{\text{nuggets}}$.

Nugget Creation prompt.

```
You are an evaluation expert. Given a question, break it into
specific sub-questions that a complete answer must address.

Each sub-question should be atomic (cover only 1 aspect),
independently assessable (don't need further sub-questions to
answer it), and cover a distinct aspect of the question.

Return ONLY a JSON array of strings.

Example: ["What year was the US founded?",
"How many children did Britney Spears have?"]
```

Retrieval Grading prompt.

```
Do these retrieved documents contain information that answers
this question? Reply with a single number:

2 = documents fully answer the question,
1 = documents partially answer the question,
0 = documents do not answer the question.
```

Example Nuggets (topic: "Geoffrey Hinton's AI Concerns"):

- Who is Geoffrey Hinton and what is his significance in artificial intelligence?
- What are Geoffrey Hinton's main contributions to artificial intelligence?
- Why did Geoffrey Hinton leave Google in 2023?
- What specific warnings did Hinton issue about the potential dangers of AI?
- Why did Hinton's warnings about AI become newsworthy or make headlines?

3.3 Attribution Score

LLM: `vectara/hallucination_evaluation_model` (HHEMv2) for scoring; `gpt-4.1-nano` for claim extraction.

For each response, the LLM is prompted via the Claim Extraction prompt to split the report text into self-contained verifiable claims (facts, opinions, predictions, and hedged statements with hedging preserved); claims shorter than 10 characters are dropped. Each claim is then paired with every document retrieved for the response (not truncated) and each pair is scored by HHEMv2, which returns a continuous entailment-style score in $[0, 1]$. For each claim, the maximum score across its document pairings is kept as the "best supporting evidence" score. The final score is the mean of these per-claim max scores, or 0 if no claims were extracted.

Claim Extraction prompt.

```
You are a claim extractor. Break this response into specific
claims that could be verified against a source document.

Each claim should be a single, self-contained statement.
Include all types: facts, opinions, predictions, and hedged
statements. For hedged claims, preserve the hedging language.
Return ONLY a JSON array of strings.

Example: ["The war started in 1914.",
"Everybody in the family agreed that it was a bad meal."]
```

Example Extracted Claims (topic `1001`, system `aloe`):

- Hinton has been worried about the potential of AI to do harm and often talks about the existential threat AI might pose to the human species.
- He has used OpenAI's ChatGPT and this has made him uneasy.
- Dr Hinton told the BBC: "Right now, they're not more intelligent than us, as far as I can tell. But I think they soon may be."
- The Biden White House is concerned about AI being developed by the tech sector and then unleashed on the public with little to no guardrails.
- AI could match human intelligence in five years, according to Google DeepMind CEO Demis Hassabis.

3.4 Final

The FINAL score combines Citation Accuracy (CA), the Attribution Score (AS), and Retrieval Quality (RQ) into a single system quality score. For each (system, topic) pair, the stronger of CA and AS is selected via $\max$ and the weaker is discarded; this value is then averaged with RQ:

$$\text{Final} = \tfrac{1}{2}\big[\max(\text{CA},\, \text{AS}) + \text{RQ}\big].$$

3.5 Checklist

LLM: `gpt-4o-mini`.

For each topic, the LLM is prompted via the Checklist Builder prompt to produce a fixed checklist of five items capturing the most important content a good response should contain. For each response, the LLM is then prompted via the Checklist Evaluation prompt to award one point per checklist item the response satisfies. The final score is the total number of points awarded.

Checklist Builder prompt.

```
You are generating a checklist to evaluate how well a response
satisfies a report request.

Request: {request_text}
Instructions:
- Create EXACTLY {num_items} checklist items.
- Each item must represent ONE distinct requirement.
- Items must be specific, objective, and clearly checkable.
- Avoid overlap between items.
- Cover the most important aspects of the request.
- If needed, merge or prioritize less important details to stay
  within {num_items} items.
- Ensure a balanced checklist (not all items about the same
  aspect).

Output format:
- Return a numbered list (1., 2., 3., ...).
- Each item must be a single sentence.
- Do NOT include explanations or extra text.
Return ONLY the checklist.
```

Checklist Evaluation prompt.

```
You are evaluating how well a response satisfies a checklist.

Checklist: {checklist_request.checklist}
Response:  {response_text}

Instructions:
- Each checklist item is worth 1 point.
- Award 1 point if the response clearly satisfies the item.
- Award 0 points if it does not.
- Be strict: partial or vague matches do NOT count.
- Sum the total points.

Return ONLY the total score as a number.
Do not explain your answer.
```

Example Checklist:

- The report includes a brief biography of Geoffrey Hinton, highlighting his key contributions to artificial intelligence.
- The report explains the context and significance of Hinton's resignation from Google in 2023.
- The report details Hinton's specific warnings about the potential dangers of AI and why they are considered important.
- The report summarizes responses from other experts in the field regarding Hinton's concerns.
- The report provides insights on what the public should understand about the implications of Hinton's warnings for the future of AI.

3.6 Negation Judge

LLM: `gpt-4o-mini`.

For each topic, the problem statement is taken as the original query and a negated query is constructed by prepending the prefix "Not about, related to, or relevant to the following:". The original query, the negated query, and the response are passed to the LLM, which scores the relevance of the response against each query on a 0–3 scale using the prompt below. The final score for the topic is a function of the two relevance scores: the penalty grows as the two scores converge, so a system whose response appears equally relevant to a query and its negation receives the lowest score, while a system that correctly assigns low relevance to the negated query receives no penalty. The intuition is that score separation across inverse prompts indicates a Rag system that generates content of semantic relevance rather than lexical overlap.

Scoring prompt.

```
You are a relevance evaluator that evaluates the relevance
between a query and a response. Score relevance between each
query and response (0 to 3). Consider each query and response
pair separately. Return ONLY one integer that represents the
relevance between each query and response pair. The first
integer is the relevance between the first query and the
response. The second integer is the relevance between the second
query and the response. Example output: 3 1.

Response: {full_response}
Query 1:  {query}
Query 2:  Not about, related to, or relevant to the following:
          {query}
```

3.7 Graded Relevance

LLM: DeepSeek v3.2.

For each topic, a query is constructed by concatenating the title, problem statement, and background. The LLM is prompted to rate the relevance of the response to this query on a 1–5 scale; the scale endpoints are repeated in both the system and user message to reduce malformed outputs. The final score is the LLM's 1–5 rating.

Scoring prompt.

```
query = f"{topic_title} {topic_problem_statement}
          {topic_background}"

System: You are a relevance evaluator. Rate the relevance of
the following output on a scale of 1-5, where 1 is 'Completely
Irrelevant' and 5 is 'Perfectly Relevant'. Provide a score only
in JSON, example: {"score" : "1"}

User: Is this passage: {full_report_text} relevant to this
user's needs? Query: {query}

Rate the relevance on a scale of 1-5, where 1 is 'Completely
Irrelevant' and 5 is 'Perfectly Relevant'. Provide a score only
in JSON, example: {"score" : "1"}
```

3.8 Naive: Length

For each topic, the response is scored by the number of characters in the response text; systems producing longer responses are ranked higher. No LLM used.[1]

3.9 Naive: Random

Each response receives a uniformly random score, providing a sanity-check baseline for the AutoJudge evaluation. No LLM used.

4 Empirical Results

To evaluate our approaches, we use two datasets based on the TREC 2025 RAGTIME data for report generation. For method development we use the RAGTIME subset of TREC AutoJudge Pilot dataset v0.2 (Dietz et al., 2025). Our test dataset is the Rag4Reports challenge.

4.1 Results

Table 1 compares our methods on AutoJudge Pilot and official Rag4Reports data set.

The best results are obtained by CITATIONACCURACY, which placed on first rank in the challenge, obtaining results only slightly below the organizer's AUTO-ARGUE system. This is likely because CITATIONACCURACY directly measures citation faithfulness, which the other judges do not. The official leaderboard places a high emphasis on citation faithfulness, marking CITATIONACCURACY the winner, despite its low nugget-coverage performance. In our internal evaluation with the AutoJudge Pilot dataset, FINAL performed stronger than CITATIONACCURACY.

Table 1: Kendall's τ for each method on the AutoJudge development set ("nugget coverage" and "citation support") and the Rag4Reports test set. Best results in bold. ($\star$) did not finish in time for submission.

Method	AutoJudge Nugget	AutoJudge Citation	Rag4Rep. Kendall
CITATIONACCURACY	0.245	**0.527**	**0.558**
RETRIEVALQUALITY	0.613	0.452	0.468
FINAL	0.669	0.482	0.430
ATTRIBUTION	0.634	0.481	0.555
CHECKLIST	0.669	−0.048	0.299
NEGATIONJUDGE	0.646	0.269	0.284
GRADEDRELEVANCE	**0.686**	0.265	($\star$)
Naive: Length	0.208	0.195	0.222
Naive: Random	-0.072	0.018	-0.143

Across both datasets, NEGATIONJUDGE performed less well, reaching a Kendall score of 0.284. It did not attempt to maximize citation support or incorporate nuggets, which is likely its primary weakness, and its scoring rule is naive. CHECKLIST performed better, especially on nugget measures, suggesting that explicit relevance criteria applied directly by the LLM outperform a rigid hard-coded rule. Tthe naive baselines perform the worst.

GRADEDRELEVANCE obtained the best correlation with AutoJudge nugget coverage, but the method did not finish in time for submission to this challenge.

5 Conclusion

Across our submissions to Rag4Reports two broad takeaways emerge. First, citation faithfulness and topical content coverage (measured via nuggets) appear to be complementary strengths of different judges. Methods that target one axis tend not to lead on the other, and our explicit attempt to combine the two, FINAL, did not dominate either: integrating the two axes into a single judge that is competitive on both remains an open problem.

Second, on a more encouraging note, methods that scored higher on the AutoJudge Pilot development set also tended to score higher on Rag4Reports: the naive baselines came in last on both, and the LLM-based judges led on both. Despite its AUTO-ARGUE-augmented truth on preliminary manual TREC judgments from late 2025, AutoJudge Pilot data serves as a useful proxy for selecting methods to enter into the downstream evaluation.

[1] Test implementation from autojudge-starterkit.

Limitations

Method differences confounded with model and prompt choices. Each method was designed independently by a different author, so differences in Kendall's τ between methods reflect not only the underlying scoring criterion (citation faithfulness vs. nugget coverage, etc.) but also unrelated choices: LLM provider, embedding model, prompt wording, retrieval-set construction, and output post-processing. We did not run ablations to isolate which of these factors drives the observed gaps, so our comparisons are between *systems* rather than between scoring principles.

Single test set. Our only test ranking is the Rag4Reports leaderboard, derived from a single set of human assessments. We cannot say whether the methods that ranked highly here would generalize to other report-generation benchmarks or to non-report RAG tasks. The development set (TREC AutoJudge Pilot v0.2) uses a AUTO-ARGUE-derived reference with very few preliminary human judgments rather than a thoroughly verified human curated truth, so agreement on the development set is itself a noisy signal. GRADEDRELEVANCE did not finish in time for the Rag4Reports submission, so we cannot tell whether its strong AutoJudge Pilot result transfers.

Reproducibility of closed-weight LLMs. All of our LLM-based methods except CITATIONACCURACY rely on commercial APIs (OpenAI `gpt-4o-mini`, `gpt-4.1-nano`, DeepSeek v3.2). Their outputs are non-deterministic and may shift with provider-side updates, so exact replication of our Kendall's τ values is not guaranteed.

Course-project scope. Each method was developed within a single semester course. None has been subjected to the iterative tuning, prompt engineering, or cost analysis that a deployed AutoJudge would require, and our results should be read as a wide first-pass survey rather than a calibrated benchmark of mature methods.

Ethical considerations

Data provenance and anonymization. The datasets used to develop and test our methods originated from the public TREC 2025 evaluation. Participating teams submitted RAG systems. As team identity is immaterial to method comparison, run names are anonymized in both Rag4Reports and the TREC AutoJudge Pilot v0.2. The manual annotations that form the Rag4Reports reference ranking were produced by NIST under their established rules for research involving human subjects.

Risks of LLM-as-a-Judge. The purpose of an AutoJudge is to scale, and in practice often replace, human assessment. An AutoJudge that correlates well with human judges on one benchmark can still embed and amplify biases of its underlying LLM, including length, fluency, and self-preference biases (Section 2). Methods that target a narrow signal, such as the citation-faithfulness focus of our best-performing approach, are auditable but should not be mistaken for general report-quality measures. In consequential settings, automated rankings should not be treated as ground truth without human review.

Compute and access costs. Several of our methods make many LLM calls per topic (e.g., one call per nugget per system, or one call per claim per document). At evaluation scale this incurs monetary cost and energy consumption that we do not measure. Teams without LLM access cannot reproduce these methods on equal footing; future work should consider compute cost alongside Kendall's τ and investigate smaller open-weight alternatives where feasible.

Table 2: Evaluation metrics by team and run, sorted by τ_{gap}. Our team's entries and baselines shown in bold.

Team / Run	τ_{gap}	Kendall	Topic
coordinators / autoargue-f1	*0.470*	*0.636*	*0.607*
unh / citation accuracy	**0.414**	**0.559**	**0.168**
unh / retrieval quality	**0.226**	**0.469**	**0.415**
unh / final	**0.221**	**0.430**	**0.386**
crucible / maxgrade	0.177	0.390	0.356
unh / attribution	**0.173**	**0.291**	**0.306**
crucible / cover4	0.128	0.360	0.190
tiet / f1 (weighted)	0.127	0.334	0.573
unh / negation-judge	**0.109**	**0.284**	**0.260**
crucible / avggrade	0.107	0.355	0.201
ju-nlp-ug / rank-argue-v4	0.099	0.375	0.546
unh / checklist	**0.096**	**0.300**	**0.343**
rgipt / Qwen2-NRB	0.077	0.133	0.221
tiet / nugget coverage	0.075	0.289	0.599
ju-nlp-pg / cite-first	0.026	0.123	0.161
tiet / sentence support	−0.056	0.183	0.247
unh / naive-length	**−0.084**	**0.222**	**0.229**
rgipt / qwen0-5	−0.125	0.154	0.251
rgipt / semiauto-run1	−0.125	0.154	0.251
rgipt / Qwen2-0.5B-NRB	−0.269	−0.297	0.185
unh / naive-random	**−0.277**	**−0.143**	**0.010**

References

Antonia Creswell, Murray Shanahan, and Irina Higgins. 2022. Selection-inference: Exploiting large language models for interpretable logical reasoning.

Kaustubh D. Dhole and Eugene Agichtein. 2026. Rubricrag: Towards interpretable and reliable llm evaluation via domain knowledge retrieval for rubric generation.

Laura Dietz. 2024. A workbench for autograding retrieve/generate systems. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1963–1972.

Laura Dietz, Ian Soboroff, and Coordinators of various TREC tracks. 2025. Trec autojudge pilot data. Licensed under CC BY-SA 3.0. Based on content from TREC DRAGUN, TREC RAG, TREC RAGTIME, their license applies.

Nelson F. Liu, Tianyi Zhang, and Percy Liang. 2023. Evaluating verifiability in generative search engines. In *Findings of the Association for Computational Linguistics: EMNLP 2023*.

Ani Nenkova and Rebecca Passonneau. 2004. Evaluating content selection in summarization: The pyramid method. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 145–152, Boston, Massachusetts, USA. Association for Computational Linguistics.

Dani Roytburg, Matthew Bozoukov, Matthew Nguyen, Jou Barzdukas, Mackenzie Puig-Hall, and Narmeen Oozeer. 2026. Are llm evaluators really narcissists? sanity checking self-preference evaluations.

Minzhu Tu, Shiyu Ni, and Keping Bi. 2026. How long reasoning chains influence llms' judgment of answer factuality.

Ellen M. Voorhees. 2004. Overview of the TREC 2003 question answering track. In *Proceedings of the Twelfth Text REtrieval Conference (TREC 2003)*, pages 54–68, Gaithersburg, MD. NIST.

William Walden, Marc Mason, Orion Weller, Laura Dietz, John Conroy, Neil Molino, Hannah Recknor, Bryan Li, Gabrielle Kaili-May Liu, Yu Hou, Dawn Lawrie, James Mayfield, and Eugene Yang. 2026. Auto-argue: Llm-based report generation evaluation.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*.

Chuting Yu, Hang Li, Guido Zuccon, Joel Mackenzie, and Teerapong Leelanupab. 2026. When llm judges inflate scores: Exploring overrating in relevance assessment.