HLTCOE Evaluation Team at TREC 2025: RAG, RAGTIME, DRAGUN, and BioGen

Laura Dietz,[‡] Bryan Li,[*] James Mayfield,[†] Dawn Lawrie,[†] Eugene Yang,[†] William Walden[†]
[†]Human Language Technology Center of Excellence, Johns Hopkins University, USA
[‡]University of New Hampshire, USA
[*]University of Pennsylvania, USA
dietz@cs.unh.edu,bryanli@seas.upenn.edu,mayfield@jhu.edu,lawrie@jhu.edu,eugeneyang@jhu.edu,wwalden1@jhu.edu

ABSTRACT

The HLTCOE Evaluation team participated in several tracks focused on Retrieval-Augmented Generation (RAG), including RAG, RAGTIME, DRAGUN, and BioGen. Drawing inspiration from recent work on *nugget-based* evaluations, we introduce the Crucible system, which scrambles the traditional retrieval → generation → evaluation workflow of a RAG task by automatically curating a set of high-quality question-answer pairs (*nuggets*) from retrieved documents and then conditioning generation on this set. This not only enables us to study how effectively we can recover the set of gold nuggets for each request but additionally how nugget set quality impacts final performance.

1 INTRODUCTION

The HLTCOE Evaluation team participated in several tracks focused on Retrieval-Augmented Generation (RAG), including RAG, RAGTIME, DRAGUN, and BioGen. Our core contribution is the Crucible system for RAG tasks, which inverts the standard retrieval → generation → evaluation workflow.

Typical RAG systems begin with retrieval, then summarize or distill relevant portions of the retrieved content to produce the output. These outputs can then be evaluated in a variety of ways. Recently, *nugget-based* evaluations—which assess generated content for coverage of key facts or questions, have received significant attention [2, 5, 6, 8]—although use of nuggets for evaluation dates back to the TREC 2003 Question Answering track [7].

With Crucible, we turn this process on its head: We first obtain a set of topic-relevant nuggets, retrieve documents relevant to those nuggets, and finally generate task outputs conditioned on the nuggets and their associated documents.

Crucially, this enables us to investigate not only the impact of retrieval and generation modules on task performance, but also the impact of predicted nugget quality: we can compare our automatically predicted nugget sets to those that were manually curated for evaluation—both in terms of their semantic similarity and in terms of the performance they yield on the target RAG task when incorporated into our system.[1]

We ensure that our nugget-driven generation process is thoroughly grounded in documents from the collection: an initial set of candidate nuggets are initially derived from individual retrieved documents and aggregated (*nugget ideation*); passages addressing these nuggets are then directly extracted from retrieved documents and summarized. We further experiment with a filtering step that uses an LLM judge to verify whether these summaries are attested by their associated citations.

Although we find that every step makes useful contributions, we find that starting with the best possible nugget set makes the largest difference to performance by far.

[1]This comparison applies *only* to meta-evaluations, as test nuggets are held-out during system development.

2 RAG SYSTEM OVERVIEW

2.1 Nugget Ideation

We generate (*ideate*) nuggets with a document-grounded approach that follows a four-stage pipeline: we generate a large number of candidate nuggets, merge related ones, filter the resulting set based on various criteria, and finally select the most relevant subset. The data requirements are a request/query r and a set of relevant documents $D = d_1, d_2, \ldots, d_i$.

For each request r we obtain a set of documents D from one of several IR systems: BM25, PLAIDX, Qwen3, LSR (see Section 2.2). For monolingual collections (e.g. RAG) we retrieve 100 documents from the corpus. For multilingual collections (e.g. RAGTIME), we take different approaches depending on the system: For BM25, we take the top 25 documents for each language; for other systems, we retrieve the top 100 documents across all languages.

Stage 1: Generation. Given some documents $d \in D$, we first prompt an LLM to generate a summary s of each. We then use another prompt that takes the the summary s and the request r as input and queries the LLM to write a set of 1–6 nuggets N in the form of question and answer (Q&A) pairs, $N = (q_1, a_1), \ldots, (q_j, a_j)$, where $1 \leq j \leq 6$. This summarize-then-generate approach is inspired by recent work (XRAG [1]), which highlights that summarizing a document first yields higher-level and more thematic Q&A pairs—which is desirable for reports—than when generating them from a document directly.

Stage 2: Merge. As the generation process is applied to each document d, for a given request r we obtain multiple nugget sets $N^{d_1}, N^{d_2}, \ldots$. Naturally, very similar questions may appear across different nugget sets, as the sets are generated independently. We term sets of very similar nugget questions *question paraphrases* and the goal of this stage is to cluster such questions, where each cluster is represented by a single canonical nugget. We do this in two stages. First, we finetune a custom paraphrase identification model,[2] which generates an embedding for each nugget question. We shortlist candidate pairs whose cosine similarity exceeds $k = 0.9$.

[2]The paraphrase model is initialized with bge-large-en-v1.5

Second, we prompt an LLM to verify each shortlisted question pair, and apply single-link clustering to merge paraphrase clusters. The result is a set N^r of canonical nugget questions with associated answers.

Stage 3: Answer Filter. To improve the quality and relevance of the nugget sets, we further filter all nuggets where:

- The answer is "None," "No information," "Not discussed," or similar.
- The answer is labeled by an LLM judge as an irrelevant or unreasonable response to the question.

Any nugget for which all answers are removed is excluded from the final nugget set.

Stage 4: Selection. For some requests, Stages 1–3 result in *many* nuggets, which we cull to only the 10–20 most important in Stage 4. This stage is motivated by the desire to mimic human-assessed data (which typically has only this many nuggets per topic) and, accordingly, to be able to cover all nuggets within a length-limited report (e.g. 2000 characters for RAGTIME). We rank all questions using the following heuristics, retaining only the top 20 nuggets:

Most Common: Rank by nugget frequency across documents in D. We track the provenance of all documents associated with a given nugget cluster through Stages 1–3. We sort clusters in descending order of the number of unique documents from which nuggets in that cluster were extracted. We then take the canonical nugget associated with each of the top 20 clusters.

SVC: Rank using a support vector classifier (SVC) trained on 19 manually crafted nugget quality features. The SVC is trained on the development data from the NEUCLIR 2024 Report Generation Pilot task to distinguish nuggets that are paraphrases of a human-generated nugget from those that are not. Our preliminary analysis indicated that SVC heuristic can identify more useful questions than the Most Common heuristic.

DRAGUN. The DRAGUN task differs from RAG and RAGTIME in that each topic comes with an article to be fact-checked. Therefore, for each article, we apply only Stage 1 (*Generation*). We author a customized prompt, which borrows both from our original generation prompt and DRAGUN track guidelines.

BioGen. Our sole BioGen run was an early prototype of the full Crucible system. This system performs nugget ideation and retrieval as outlined above, but takes an alternative approach to generation: It uses an LLM (Claude 4 Sonnet) to rewrite each selected nugget as a single answer sentence, arbitrarily selecting one of the nugget references as a citation for that sentence. Sentences are generated in descending order of nugget rank.

2.2 Retrieval

To enhance the diversity of the retrieved document set $d \in D$, we employ multiple retrieval strategies: BM25, sparse, and dense retrieval and reranking.

Sparse approaches. In addition to BM25, we adopt learned sparse retrieval (LSR [4]), which emphasizes weighted lexical matching while incorporating query and document expansion to capture broader semantic meaning. For (multilingual) RAGTIME, we employ multi-LSR [3] to address non-English corpora by aligning multilingual language model heads.

Dense approaches. To mitigate the lexical mismatch problems in sparse methods, we add two dense retrieval approaches: PLAIDX [9] and Qwen3 [12]. Both leverage multilingual language models, enabling us to encode both English and multilingual corpora. On top of these dense models, we apply LLM-based reranking using `Llama-3.3-70B-Instruct` with a listwise reranking strategy.

2.3 Response Generation

The system responds with a report (or long answer) that is generated by aligning ideated nuggets $(q, a) \in N$ to a set of retrieved documents $d \in D$—obtained with one of the retrieval models discussed in Section 2.2. For each nugget question, we extract sentences that contain answers, then rephrase each extraction into a concise sentence that clarifies how the document addresses the question.

Stage 1: Obtain source documents. We consider two document sources:

Nugget references: We compile the set of documents D from which *some* nugget was extracted during nugget ideation. This approach thus favors documents used during ideation.

Ranking: We compile the top k documents D returned by a provided ranking (cf. Section 2.2). This approach reflects the IR model priorities.

Stage 2: Candidate Sentence Extraction. For every nugget $(q, a) \in N$ and every document $d \in D$, we issue extraction prompts to the LLM to obtain a sentence that attests that nugget (if one exists). To avoid the 4000-token limit of our LLM, we segment each document into 1000-character chunks, split at sentence boundaries, and issue prompts on each chunk to extract one sentence per nugget.

We use several prompt templates for this purpose:

Question/Answer: (SupportedAnswerExtractorRequest) This prompt provides the document chunk, the question, and a list of answers. It asks for a passage from the chunk that substantiates an answer, then requests a concise sentence that makes the question clear.

Answerability: (SupportedAnswerabilityExtractorRequest) This prompt omits the answer list and asks the model to locate a passage that answers the question, then to produce a concise sentence as above.

All prompts also include information about the request, such as title query, background, and problem statement.[3]

Stage 3a: Verify Supported Citations. To ensure that summarized sentences are genuinely supported by their citations, we use the sentence support check (the `check_sentence_attested` function) from the `argue_eval` library [8]. If enabled, any citation for which we receive the answer "NO" is removed. Only sentences without remaining citations are retained.

[3]Due to a misconception with DSPy, prompts in this experiments ran without instruction, the LLM's response was only based on field names and descriptions.

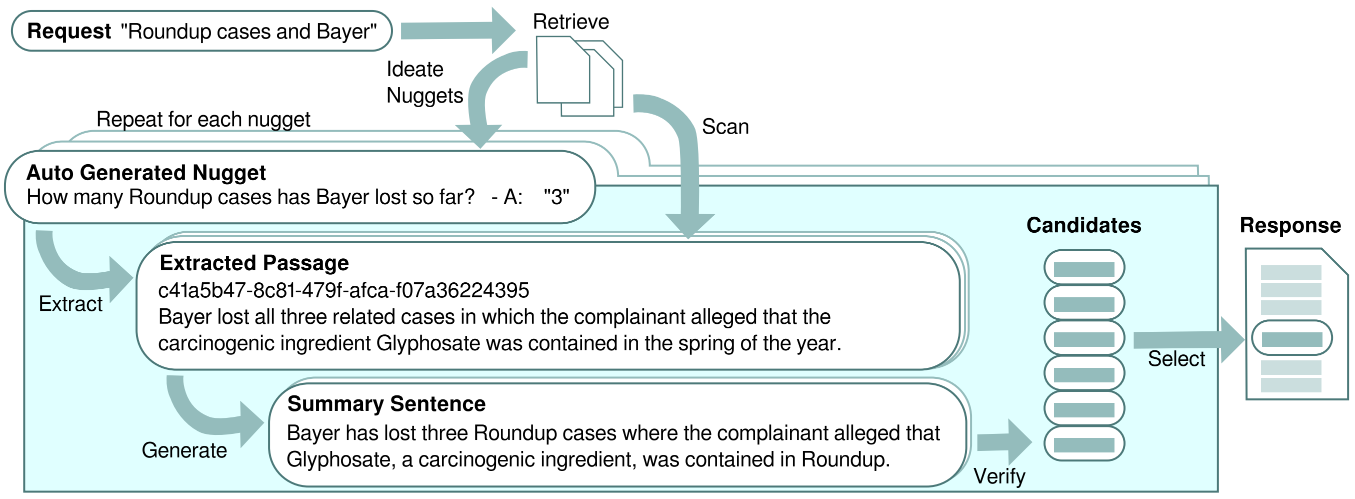
Request "Roundup cases and Bayer"
Retrieve
Ideate Nuggets
Repeat for each nugget
Scan
Auto Generated Nugget
How many Roundup cases has Bayer lost so far? - A: "3"
Extract
Extracted Passage
c41a5b47-8c81-479f-afca-f07a36224395
Bayer lost all three related cases in which the complainant alleged that the carcinogenic ingredient Glyphosate was contained in the spring of the year.
Generate
Summary Sentence
Bayer has lost three Roundup cases where the complainant alleged that Glyphosate, a carcinogenic ingredient, was contained in Roundup.
Verify
Candidates
Select
Response

Figure 1: Overview of the Crucible pipeline. For each ideated nugget, sentence candidates are extracted from retrieved documents. Of this candidate set, only the best sentence is included in the system's response.

Stage 3b: Verify Nugget Coverage. To ensure that summarized sentences faithfully address the target nugget,[4] we use the nugget attestation check (the `check_nugget_matches` function) from the `argue_eval` library. Only sentences where both the extractive and abstractive summary attest the target nugget are retained.

Stage 4: Choose Sentence. Despite verification, the candidate extraction process may yield multiple sentences for one nugget. We select the sentence with the highest extraction confidence[5] and add it to the report. Nuggets with no extractions are omitted.

We also employ a simple quality check that omits sentences with extraction confidence < 0.5 as well as those judged too similar to previous sentences in the report. Similarity is computed from a fingerprint of stemmed and stopword-removed sentence text. We avoid adding sentences that include the keyphrase "source document," as this was often indicative of low-quality summaries.

Stage 5: Chop. If the report exceeds a 2000-character limit, we iteratively remove sentences in ascending order of extraction confidence until the report respects the limit.

Stage 6: Polish. Metadata for each sentence is tracked throughout the pipeline. This sentence-level metadata, along with stage-level metadata, are collated into a unified report structure. For the final report, we discard the per-sentence metadata, as it violates the required format. We also create a description and a run ID that record the hyperparameters chosen in the earlier stages.

3 RESULTS

3.1 RAGTIME Track (Dry Run)

The dry run evaluation required assessors to select useful snippets and then to derive nuggets from these snippets. Citation support was also assessed.

The results are presented in Table 2. To our surprise, retrieval with Milco (LSR) obtained the best nugget coverage—even better than extraction from documents that the nuggets were derived from. Unfortunately, sentence support was also lowest in these cases. However, we believe that sentence support can be easily addressed by filtering out unsupported citations (as in Stage 3 of Report Generation).

Strict settings of our sentence selector did not seem to obtain improvements. The extractive approach obtained the best F1 combination with 0.55 (median 0.51, max 0.60).

3.2 RAGTIME Track

The RAGTIME coordinator uses human assessors to create question-answering nuggets. Next they used both an LLM-prompt of AutoArgue [8] and manual assessors to identify which generated answer contained which nugget. To date, only information about AutoArgue is available, and among those only short reports (2k characters) were judged.

Moreover, AutoArgue is uses a prompt to identify whether referenced citations faithfully support their sentence sentence.

The results are presented in Table 1. Our system's sentence support scores are very high. For all submitted runs except `ansR-bare-conf`, this is by design, since our verification step uses exactly the same prompt to filter sentence candidates that do not meet this requirements. In other words, our system is designed to obtain a very high score on sentence/citation support. In contrast, `ansR-bare-conf` demonstrates the system behavior without any verification step (i.e., only selecting the most confident extractions). And indeed, this run only obtains a sentence support score of 0.778.

Our nugget coverage score is barely meeting the median performance. This is also true for runs with nugget-coverage verification step. This is because the verification step will remove sentences that do not align with crucible's system generated nugget set, but (obviously) has no knowledge about the gold nuggets which are

[4]Here, we use only the nuggets predicted by our Crucible system—not the gold nuggets that used for task evaluation.
[5]We compute extraction confidence using DSPy's chain of thought module.

Table 1: TREC 25 results across the four tracks we participated in. All measures are higher=better, except Contradict, where lower=better. (*) denotes reference for two-sided paired-t tests with $\alpha = 5\%$.

Track		Nugget Ideation				Retrieval			Extraction		Verification			Evaluation Score	
	Run	Most Common	SVC	Claude	Claude2	Nugget Refs	LST	PlaidX	Question/Answer	Answerability	None	Citation	Citation & Nugget		
RAG														Sub-Nuggets	Weighted Precision
1	ansR		✓			✓			✓				✓	0541.±0.035 *	0.686±0.042 ▼
2	ansR-conf		✓			✓			✓			✓		0.509±0.030 ▼	0.585±0.055 ▼ ▼
3	ansR-bare-conf		✓			✓			✓		✓			0.535±0.035 ▼	
4	ablR		✓			✓				✓			✓	0.481±0.031 ▼ ▼	**0.798**±0.033 ▲ *
5	ablR-conf		✓			✓				✓		✓		0.490±0.039 ▼ ▼	
	median (*)													**0.622**±0.027 *	0.618±0.033 * ▼
DRAGUN														Support	Contradict (lower)
1	ablR		✓			✓				✓			✓	**0.173**±0.024 ▲ *	**0.009**±0.004
2	ablR-conf		✓			✓				✓		✓		0.166±0.019 ▲	0.025±0.011
3	confirm-ansR		✓			✓			✓				✓	**0.170**±0.024 ▲	0.011±0.006
4	clod-ablR-conf			✓		✓				✓		✓		0.158±0.020 ▲	0.013±0.006
5	cloch-ablR				✓	✓				✓		✓		0.157±0.017 ▲	0.018±0.006 ▲
	median (*)													0.108±0.010 * ▼	0.003±0.002 *
RAGTIME														Sentence Support	Nugget Coverage
1	ansR		✓			✓			✓				✓	0.912±0.062 ▲	0.331±0.056
2	ansR-conf		✓			✓			✓		?	✓		0.922±0.062 ▲	**0.348**±0.058
3	ablR		✓			✓				✓			✓	0.921±0.062 ▲	0.330±0.053
4	ansR-LSR		✓				✓		✓				✓	0.887±0.062 ▲	0.308±0.055
5	ansR-plaidx		✓					✓	✓				✓	0.881±0.064 ▲	0.336±0.059
6	ansR-most-common	✓				✓			✓				✓	**0.973**±0.007 ▲ *	0.267±0.041 ▼ ▼
7	ansR-bare-conf		✓			✓			✓		✓			0.778±0.061 ▼	0.342±0.058
8	ablR-LSR		✓				✓			✓			✓	0.909±0.062 ▲	0.298±0.045
9	ablR-plaidx		✓					✓		✓			✓	0.905±0.062 ▲	0.319±0.054
10	ablR-conf		✓			✓				✓	?	✓		0.925±0.062 ▲	0.303±0.052 ▼
	median (*)													0.772±0.060 * ▼	0.332±0.051 * ▲
BioGen														Answer Accuracy	Citation Coverage
1	svc-smoothed-sonnet		✓			✓			✓		✓			1.000±0.000	0.640±0.000
	median (*)													1.000†	0.740†

manually created by assessors. Any mismatch between our system nuggets and the gold nuggets will only be exacerbated by the nugget coverage verification step. We see this in the results, as the two best nugget coverage runs, `ansR-conf` and `ansR-bare-conf` do not use the coverage-verifiation step.

The worst performing run selected a nugget set with the `most-common` heuristic, showing that reranking the nuggets with SVC leads to a nugget bank that is better aligned with human assessor's interests.

3.3 RAG Track

Only preliminary relevance assessments for RAG are available at the time of this writing. Regarding the nugget coverage, the track coordinators evaluated both coverage of all topical sub-nuggets and a selection of vital nuggets. In both cases, our best approach is comparable to the median. Extracting with questions and answers is faring better than plain open-ended answerability. Citation and nugget filtering do not significantly change the performance. Since our reports contain exactly one citation per sentence, weighted precision and weighted recall are the same. Interestingly, filtering with our system nuggets (in addition to citations) improves our citation score slightly. Sadly, none of the runs without citation filtering were manually judged for citation relevance.

Our best citation score is obtained when extracting sentences based on questions only (holding out the expected answers). This result is statistically significant. We hypothesize that providing the gold answer might lead to hallucinated sentences that don't hold up to manual assessment. Since reports generated with this 'answerability' extractions are significantly worse, we conclude that the reports contain less relevant content, but the provided content is more faithfully supported.

Finally, we note that the manual citation verification results are at odds with the results on RAGTIME, where citation support

Normal	Strict	Extractive	Most Common	Nugget Refs	Ranking	LST	PlaidX	Answer	Answerability	Nugget Coverage	Sentence Support
	✓		✓		✓	✓		✓		0.42	0.74
✓			✓		✓	✓		✓		0.41	0.72
		✓	✓	✓				✓		0.41	0.92
	✓		✓	✓					✓	0.40	0.93
✓			✓	✓				✓		0.40	0.86
✓			✓		✓	✓			✓	0.39	0.84
	✓		✓		✓		✓	✓		0.39	0.79
	✓		✓		✓	✓			✓	0.39	0.89
	✓		✓	✓				✓		0.38	0.93
	✓		✓		✓		✓		✓	0.38	0.86
Max										0.47	0.93
Median										0.39	0.82

Table 2: RAGTIME Dryrun results

is verified with am LLM-judge approach. In RAGTIME, among all runs that used citation support verification, no difference in citation support was detected between `ansR` and `ablR` (with or without nugget verification).[6]. Since the Crucible verification steps (purposefully) create a circularity with the LLM Judge, it is possible that this difference reveals a vulnerability in the Argue evaluation system.

3.4 DRAGUN Track

The main goals of the DRAGUN track [11] is to provide a basis for fact-checking an article. Task 1 ("Question") asks systems identify key questions a reader should investigate when fact-checking the article. Task 2 ("Report") asks systems to generate a longer response that addresses these questions using a provided corpus.

For evaluation, three assessors are designing a rubric of broad "need-to-know" questions along with claim-style answers (referred to as the "nuggets") that should be covered in the report.

Below is an example rubric:

- Question 1 (Need to Know): What should I know about the source of this article, Distractify.com?
  - **Q1 / Answer Nugget 1:** Distractify is a pop culture online publication.
  - **Q1 / Answer Nugget 2:** Distractify has been sued for violations of copyrighted photographs.
  - **Q1 / Answer Nugget 3:** Distractify has a left-center bias, a Mixed rating for Factual Reporting, and a Medium Credibility rating.

Report Task. Each question has an importance score (4,2,1) and each answer can be matched wholly (1), partially (0.5) or not (0).

Report runs are evaluated by a recall-based evaluation measures for "supportive" (higher=better) and "contradictory" (lower=better) to the assessor's rubric. For each question, a report can earn up to '$ question_importance · $num_answers' many points. The evaluation measure is $\frac{\text{obtained points}}{\text{max points}}$.

We deviate from our approach in terms of retrieval (described in Section 2.2), by obtaining passages from the DRAGUN Starter Kit [10]. (We do not use any other components of the kit.)

The results are presented in Table 1. We find that verification of both citation support and nugget coverage increases the performance (albeit only slightly). Overall, we obtain a relative improvement over the median of +50% to +70%.

[6] 0.912, 0.922, 0.921, 0.925

Table 3: DRAGUN Question Task Results. *: Reference for paired t-test.

	Evaluation Score
MC	0.364±0.062$^{\blacktriangledown\blacktriangledown}$
Claude	0.493±0.057 $^{\blacktriangledown}$
Claude2	**0.621**±0.072 *
Median	0.574±0.048 *

Question Task. We submitted our generated nugget questions to the report task. We realize only now, that our usage of the word "nugget" refers to a question, where DRAGUN coordinators refer to claims/answers as "nuggets".

Example questions from Claude2:

> What is the documented date when Sam Panopoulos invented Hawaiian pizza?
> What was the name of the restaurant where Sam Panopoulos invented Hawaiian pizza?
> When did Sam Panopoulos immigrate from Greece to Canada?

The evaluation aligns each of the submitted question to the closest question on of the assessor's rubric (using two different LLM-based approaches). Assessors are manually annotating whether the predicted question indeed aligns with the sentiment of the manual rubric entry. The assessment labels are Very similar (1 point), Similar (0.5 point) and 0 points different or very different.

Our results are presented in Table 3. While our main nugget generation approach (Most Common) is significantly below the median, our alternative "Claude2" prompt is on par (or slightly above) the median.

Are good questions helping with good reports? We analyze the per-topic correlation between performance on the report task and question task. While on a some topic with high question task score, we also obtain a high report generation score, overall there is no strong correlation between performance under both tasks.

Internal LLM-as-a-Judge analysis.

Run 1: The DRAGUN task differs in that each topic comes with an article which will be fact-checked. Therefore, for this single article, we apply only the first stage, "Generation". We design a prompt which draws both from the original generation prompt and the guidelines of DRAGUN. This is shown in Figure **??**. We also use the stronger LLM Claude Sonnet 4 here.

Informally, we use GPT5 to assess our submission. While all report generation methods obtain similar and reasonable reports, the report generation method "Answerability" seemed to be slightly more comprehensive. According to GPT5, three topics, Instant noodles, Coca-Cola, Satan Shoes, among generated reports with citation verification. Example:

> "The preservative in instant noodles linked to potential cancer risks is tertiary-butyl hydroquinone (TBHQ)." "A study published in The Journal of Nutrition in 2014 found that consuming instant noodles two to three times a week increases the risk of developing cardiometabolic syndrome, which can lead to heart disease, diabetes, and stroke." "The chemical TBHQ, found in instant noodles, has been linked to neurotoxic effects, including convulsions and paralysis in lab rats, and vision disturbances in humans." "The FDA allows a maximum of 0.02 percent TBHQ in the oil and fat content of food." "Dr. Braden Kuo of Massachusetts General Hospital conducted a study on instant noodle digestion using a pill-sized camera." "Women who eat more than two portions of instant noodles per week are at a higher risk of developing metabolic syndrome, which can lead to heart disease, stroke, and type 2 diabetes." "The comprehensive study on instant noodle consumption was conducted in the Republic of Korea." "The diet pattern rich in meat, soda, and fast food is likely referred to as the 'Western diet' or 'fast food diet pattern', characterized by high intake of processed foods." "Instant Ramen noodles remain intact in the stomach for a longer period than homemade ramen noodles, taking more than 2 hours to digest." "The consumption of instant noodles has been linked to an increased risk of heart disease, stroke, and type 2 diabetes."

3.5 BioGen Track

Our team participated only in the BioGen Track's Task B ("Reference Attribution"), in which systems must generate complete answers to biomedical questions and provide attribution to PubMed documents for each sentence in the answer. Task B reports a variety of scores focused on answer quality, citation quality, and cited document relevance. In Table 1, we report two of the most important metrics: Answer Accuracy (the proportion of answers judged acceptable) and Citation Coverage (the number of answer sentences with ≥ 1 supporting citation). Here, we match the median of perfect Answer Accuracy, though we fall below the median on Citation Coverage. On other Task B metrics (not shown), we also tend to perform at or somewhat below the median. Topic-level results were not available at the time of writing to support statistical significance testing.

4 CONCLUSION

This work demonstrates that starting the pipeline with nugget ideation, which are used to drive the remaining stages of the generation is a significant factor in the effectiveness of retrieval-augmented generation systems evaluated under nugget-style metrics. While retrieval and generation components contribute meaningfully, their impact is secondary to the alignment between the nugget set and the user's information need.

Looking ahead, several directions warrant further exploration. First, improving automatic nugget extraction remains a critical challenge. Techniques that incorporate broader context, leverage more diverse document sets, or apply user-guided refinement may offer better alignment with human-annotated nuggets. Second, integrating nugget prediction more tightly with retrieval presents an opportunity to close the gap between fully automatic and oracle-like systems. Systems that jointly optimize for nugget coverage and citation support may better capture both relevance and fidelity.

Finally, we aim to scale this framework to new domains, using structured nugget representations as an interface between retrievers, generators, and evaluators. By grounding generation more explicitly in nugget-centric planning, we hope to improve both transparency and control in long-form answer generation.

REFERENCES

[1] Xin Cheng, Xun Wang, Xingxing Zhang, Tao Ge, Si-Qing Chen, Furu Wei, Huishuai Zhang, and Dongyan Zhao. 2024. xRAG: Extreme Context Compression for Retrieval-Augmented Generation with One Token. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[2] James Mayfield, Eugene Yang, Dawn Lawrie, Sean MacAvaney, Paul McNamee, Douglas W Oard, Luca Soldaini, Ian Soboroff, Orion Weller, Efsun Kayi, et al. 2024. On the evaluation of machine-generated reports. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1904–1915.

[3] Thong Nguyen, Yibin Lei, Jia-Huei Ju, Eugene Yang, and Andrew Yates. 2025. Milco: Learned sparse retrieval across languages via a multilingual connector. *arXiv [cs.IR]* (2025).

[4] Thong Nguyen, Sean MacAvaney, and Andrew Yates. 2023. A unified framework for learned sparse retrieval. In *European Conference on Information Retrieval*. Springer, 101–116.

[5] Ronak Pradeep, Nandan Thakur, Shivani Upadhyay, Daniel Campos, Nick Craswell, and Jimmy Lin. 2024. Initial nugget evaluation results for the trec 2024 rag track with the autonuggetizer framework. *arXiv preprint arXiv:2411.09607* (2024).

[6] Ronak Pradeep, Nandan Thakur, Shivani Upadhyay, Daniel Campos, Nick Craswell, Ian Soboroff, Hoa Trang Dang, and Jimmy Lin. 2025. The great nugget recall: Automating fact extraction and rag evaluation with large language models. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 180–190.

[7] Ellen M Voorhees and L Buckland. 2003. Overview of the TREC 2003 Question Answering Track.. In *TREC*, Vol. 2003. 54–68.

[8] William Walden, Marc Mason, Orion Weller, Laura Dietz, Hannah Recknor, Bryan Li, Gabrielle Kaili-May Liu, Yu Hou, James Mayfield, and Eugene Yang. 2025. Auto-ARGUE: LLM-Based Report Generation Evaluation. *arXiv preprint arXiv:2509.26184* (2025).

[9] Eugene Yang, Dawn Lawrie, and James Mayfield. 2024. Distillation for Multilingual Information Retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2368–2373.

[10] Dake Zhang. 2025. An Iterative Multi-agent RAG System for the TREC 2025 DRAGUN Track. In *The Thirty-Fourth Text REtrieval Conference Proceedings (TREC 2025) (NIST Special Publication)*. National Institute of Standards and Technology (NIST).

[11] Dake Zhang, Mark D. Smucker, and Charles L. A. Clarke. 2025. Overview of the TREC 2025 DRAGUN Track: Detection, Retrieval, and Augmented Generation for Understanding News. In *The Thirty-Fourth Text REtrieval Conference Proceedings (TREC 2025) (NIST Special Publication)*. National Institute of Standards and Technology (NIST).

[12] Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, et al. 2025. Qwen3 Embedding: Advancing Text Embedding and Reranking Through Foundation Models. *arXiv preprint arXiv:2506.05176* (2025).