DEMONSTRATION

Nugget-based Annotation Protocol and Tool For Evaluating Long-form Retrieval-Augmented Generation

EUGENE YANG, Johns Hopkins University, Baltimore, MD, United States

DAWN J LAWRIE, Johns Hopkins University, Baltimore, MD, United States

HOA TRANG DANG, National Institute of Standards and Technology, Gaithersburg, MD, United States

IAN M SOBOROFF, National Institute of Standards and Technology, Gaithersburg, MD, United States

JAMES CLIFTON MAYFIELD, Johns Hopkins University, Baltimore, MD, United States

Open Access Support provided by:

Johns Hopkins University

National Institute of Standards and Technology

Published: 13 July 2025

Citation in BibTeX format

SIGIR '25: The 48th International ACM SIGIR Conference on Research and Development in Information Retrieval
July 13 - 18, 2025
Padua, Italy

Conference Sponsors:
SIGIR

Nugget-based Annotation Protocol and Tool For Evaluating Long-form Retrieval-Augmented Generation

Eugene Yang
eugene.yang@jhu.edu
Johns Hopkins University HLTCOE
Baltimore, MD, USA

Dawn Lawrie
lawrie@jhu.edu
Johns Hopkins University HLTCOE
Baltimore, MD, USA

Hoa Dang
hoa.dang@nist.gov
NIST
Gaithersburg, MD USA

Ian Soboroff
ian.soboroff@nist.gov
NIST
Gaithersburg, MD USA

James Mayfield
mayfield@jhu.edu
Johns Hopkins University HLTCOE
Baltimore, MD, USA

Abstract

Retrieval-augmented generation (RAG) summarizes retrieved documents into a text passage that fulfills the information need expressed by the user. Such generated responses should faithfully distill the relevant information and provide sufficient attribution back to the source documents. Nugget-based evaluation was proposed for text summarization and has been adapted to evaluate RAG output in recent shared tasks such as 2024 TREC RAG, BioGen, and NeuCLIR tracks. However, annotating such detailed and nuanced information is complex and errorful. Multiple pieces of information need to be labeled, extracted, linked, and cross-referenced. In this work, we present an annotation protocol and tool tailored to collecting information for evaluating RAG systems. Our tool has four steps: nugget creation, nugget revision, document support assessment, and finally, nugget alignment. Each step aims to minimize the annotator's cognitive load, improving the efficiency and reliability.

CCS Concepts

• **Information systems** → **Summarization**; **Evaluation of retrieval results**; **Information retrieval**.

Keywords

Retrieval-Augmented Generation, Annotation, Annotation Protocol, User Interface

ACM Reference Format:
Eugene Yang, Dawn Lawrie, Hoa Dang, Ian Soboroff, and James Mayfield. 2025. Nugget-based Annotation Protocol and Tool For Evaluating Long-form Retrieval-Augmented Generation. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '25), July 13–18, 2025, Padua, Italy*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3726302.3730156

1 Introduction

Retrieval-augmented Generation (RAG) systems consist of a retrieval system followed by a generation model. Despite different system designs [1, 2, 13], the purpose of the retrieval system in the pipeline is to assist the generation model by providing credible information from a corpus based on the user request. The retrieved information can be used to verify the responses from the generation model [3] or to synthesize the responses [2]. In either case, retrieval is used to improve the credibility and factuality of the RAG system response (which we refer to as a *report*).

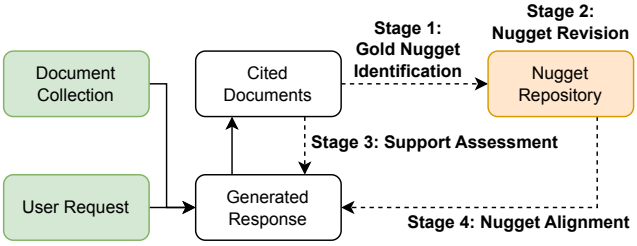

Figure 1: Overview of the four stages of our proposed annotation process. Green boxes indicate system inputs, white boxes are system outputs, and the orange box in the upper right is the artifact retained after annotation. Solid and dashed arrows indicate steps taken by the system and the annotators, respectively.

One crucial aspect of credibility is provision of references for each piece of information in the response for attribution and improved trustworthiness [3, 4]. Assessing the quality of the citation and the faithfulness of the generated responses and their associated citations are two key aspects when evaluating RAG systems. Prior works, such as FactScore [8] and AttrScore [20], aim to automatically assess the quality of the attribution based on a reference document. However, the quality of the evaluation is then conditioned on the effectiveness of the matching models used in the evaluation, which are usually implemented as LLM prompt models [8]. Since RAG systems also use LLMs for their work, this connection risks bias in favor of RAG systems that use the same or similar LLM as the evaluation [10, 19]. It also opens the door for distilling the evaluator into the system [14]. Therefore, human assessment is still critical for the evaluation of RAG systems.

ARGUE [7] integrates aspects of both summarization and retrieval evaluation identified in prior works. It constructs a conceptual framework for managing the dependencies and implications of the combination of these aspects. It provides a clear analysis of identifying misinformation, hallucination, and irrelevant information in the system responses through a finite state machine (or flowchart). However, the amount of annotation required to evaluate a paragraph of information-rich response is enormous. Annotators need to assess the relevance of the cited document, the faithfulness of the generated content, and the usefulness of including such information in the response. While feasible, such assessments are tedious and errorful without proper process design and tooling.

SIGIR '25, Padua, Italy

ACM ISBN 979-8-4007-1592-1/2025/07
https://doi.org/10.1145/3726302.3730156

In this work, we introduce an annotation protocol and tool for implementing the annotation needed for ARGUE. Our annotation protocol separates the dependencies in ARGUE as individual, self-contained annotation units into four stages (illustrated in Figure 1); this minimizes the cognitive load for the annotators and reduces potential annotation errors. Additionally, we provide a web-based annotation tool with a UI design that limits the possibility of annotation errors and provides better management. The tool is deployed for annotation of the report generation pilot task at the 2024 TREC NeuCLIR Track [5] and is publicly available on GitHub.[1]

2 Background

The ARGUE framework [7] aims to evaluate RAG responses that contain multiple sentences, multiple pieces of information, and citations to multiple documents in the corpus. It constructs a process to determine whether a generated sentence should be rewarded, punished, or ignored in the evaluation. While examining documents in the corpus, the human annotator curates a set of "gold nuggets" [9], which are the pieces of information that the human annotator expects the generated report to contain. We assume that the information in the document collection is all factual, which is a typical assumption in IR evaluation [6]. The curation of the nugget set also sets the scope for the topic since it defines the set of information that should be mentioned or that is worth mentioning in the generated responses. Treating such judgments as opinions instead of facts is another common assumption in retrieval evaluation [16].

Nugget-based evaluation was first developed for utility-based summarization evaluation [9]. While reference-based metrics such as ROUGE and BLEU are more commonly used, they require a gold summary and assume that quality correlates with the amount of overlap with the gold summary; this is less true in the era of the LLM [11]. ARGUE implements nuggets as question-answer pairs with multiple possible answers for a question.

To evaluate a specific RAG response, ARGUE scores each sentence and aggregates the resulting scores. For a sentence that has a citation, it verifies whether the cited document supports the sentence and whether the sentence answers a gold nugget question correctly. For a sentence without citations, ARGUE assesses whether the sentence should have a citation based on its content. If the sentence restates information from the user input or reflects common sense, it can stand by itself without a reference. Please refer to Figure 2 in Mayfield et al. [7] for the complete scoring flowchart.

However, merely following the steps of the flowchart requires the annotator to read different documents while stepping through the sentences. Cited documents may appear multiple times in a generated response or even across different responses being evaluated under the same user input; such frequent context switching from one document to another creates a heavy cognitive load, resulting in inefficiency [15]. Furthermore, for sentences with citations, the annotator also needs to reference the gold nugget set, which occupies a large amount of short-term memory if the nugget set is large.

In the next section, we discuss the annotation protocol that decomposes the above annotation units into independent questions to reduce cognitive load. While the steps are potentially automatable, we focus on pure manual annotation in this work and leave the automation for future work.

[1] https://github.com/hltcoe/rag-annotation-tool

Table 1: The two independent aspects of sentence annotation.

	Sentence Support		
Information Contained	Supported By Citation	Not Supported By Citation	Missing Citation
Some Nuggets	Valid	Hallucinated	Hallucinated
Irrelevant	Ignored	Hallucinated	Hallucinated
No Information	Ignored	–	Ignored

3 Nugget-based Annotation Protocol

ARGUE [7] is a framework for scoring RAG output. ARGUE assumes that a human assessor creates the artifacts that are necessary to perform an evaluation. The annotator is responsible for creating 1) a description of the desired information, expressed as questions; 2) a set of acceptable answers to those question; and 3) a list of documents from the collection that attest to each such answer. The first two form the gold nugget set.

Annotation of this sort is more complicated and time-consuming than, say, relevance judgment for an information retrieval task. Because of this, it is vital to prioritize the assessor's time when designing annotation tools. Two principles aimed at reducing assessor cognitive load have guided our development of assessment tools for ARGUE:

- The assessor should need to visit any document a finite number of times, preferably only once, and here it is twice.
- The assessor should need to view the generated text a finite number of times; here it is twice.

To streamline report sentence annotation, for each sentence, the annotator answers two independent questions:

(1) Is the sentence supported by the cited document?
(2) Does the sentence convey information about at least one nugget, does it contain only irrelevant information, or does it contain no information?

These questions can be answered separately, as the second one focuses on the nuggets while the first only requires examination of the report sentence and the document.

In the following four stages, we describe the process of creating the artifacts and acquiring annotation for the two questions for each report sentence.

3.1 Stage 1: Nugget Creation and Attribution

The purpose of Stage 1 is to create assessment units (nugget questions and answers to those questions that are linked to a document in the collection). The first step in this process is to create a pool of documents that are likely to contain information that the assessor would like included in the report. These documents can be selected in a variety of ways: documents cited in a report; documents that are ranked highly by some search process; or documents that have been judged relevant to the topic of the report according to some relevance evaluation process. Both the 2024 TREC NeuCLIR Report Generation Pilot and the TREC RAG Track included

straight retrieval tasks on the same topics as the RAG task. These tasks provided document sets that were likely to contain relevant nuggets. Conversely, if the collection is designed to support traditional ranked retrieval tasks, nugget creation and attribution can support document relevance assessment [12]. If the document contains a nugget, it is a good clue that the document may be relevant. The level of relevance can be recorded by the assessor. Or, if nuggets have been assigned importance values, the level of relevance might be inferred by the importance of the nuggets extracted from it.

Once the document pool has been created, the assessor scans each document to determine whether it contains information that should be included in a good summary. For each such piece of information, the assessor writes a stand-alone question whose answer is a word or short phrase. The answer needs to be short and specific to make it unlikely that the information it contains might be split over multiple sentences in a report. The document is then linked to that answer.

If the document provides an answer to a previously written question, it is linked to that question. If it is a new answer to the question, the assessor creates a new answer entry under the question. Documents conveying the same nugget may express the information slightly differently; all such variants are acceptable. For example, a document may refer to the signing of the Declaration of Independence of the US as occurring in 1776, July 1776, or July 4 of 1776. All three answers could be acceptable if the annotator deems them as responsive to the user's need; thus, as with relevance judgments in ranked retrieval, satisfaction of an answer to a nugget question is an assessor opinion, not a fact. We advise the annotator to only record separate answers that differ semantically, not mere variations in syntax such as "July 4," "fourth of July," or "4th of July."

Such annotations can be partially automated by prompting a large language model to produce an initial set of nuggets. However, the set of information that should be included in the report should be the assessor's opinion. Therefore, assessors still need to verify and edit the automatically generated nuggets to ensure quality.

If new nuggets are created during this stage, documents that have previously been reviewed may also contain this newly-created nugget. Re-annotating past documents creates a circular loop that is extremely inefficient. We solve this problem by giving the annotator a second chance to attribute documents to the nugget; this is described under Stage 3 below. Thus, an assessor never revisits a document during this stage.

3.2 Stage 2: Nugget Revision

After reviewing all documents in the pool, an annotator reviews all the nuggets created and revises them if necessary. This process is likely to be done by an *annotation manager* to ensure no catastrophic changes are made to the nugget set. Some nuggets created earlier than others might have different wording that should be aligned or even merged. Some might seem central or worth mentioning in the generated report when they are created but later be deemed unnecessary. Such nuggets can be deleted; the annotation manager must ensure that deletion does not remove critical information.

After revision, the nugget questions are frozen. The scope of the generation response is then fixed. Additional attribution or edits in nugget *answers* can be added in the next stage, but such changes would not (or should not) introduce changes to the nugget set.

3.3 Stage 3: Document Support Assessment

The purpose of Stage 3 is to judge whether the citations in a report properly cite documents. During this stage, additional answers can be added to existing nugget questions. When a new answer is found it is linked to the documents that attest that answer. New nugget questions are never added at this stage; doing so would complicate the assessor's task and call into question their opinions on what should be included in the report.

Stage 3 is document-driven, not report-driven. This differs from the ARGUE methodology, which iterates over report sentences. For each document to be reviewed in this stage, the set of report sentences that cite it is collected. The annotator assesses whether each report sentence is supported by the document, which answers the first question (*Is the sentence supported by the cited document?*) for judging a report sentence. In most cases, each report sentence can be assessed in isolation; however, because a report sentence can refer to an item earlier in the report, our tool provides easy access to the entirety of the report containing the sentence if the assessor wishes to see it.

This annotation can be expanded beyond binary judgments to allow more nuanced evaluation. For example, in the 2024 TREC BioGen Track, the relationship between a response and the document can be one of *Supports*, *Contradicts*, *Neutral*, or *Not Relevant*.[2]

Including relevant documents that are not cited by any report increases collection reusability, because the assessment criteria can include links between answers and documents that were not identified by any system that participated in the initial evaluation of the task. This is particularly important because in the current instantiations of the task, systems are only required to cite a single document to support a given fact, rather than all documents that support it.

At the conclusion of Stage 3, a collection of nuggets has been formed that accurately captures the assessor's original opinion about what information must appear in the report. Answers to nugget questions found in the document pool have been included even if no report used in forming the pool expressed that answer. As with information retrieval, if the pools are sufficiently varied to allow us to conclude that they contain most answers to the nugget questions, using the existing collection to distinguish systems that were not part of the original evaluation should be accurate.

3.4 Stage 4: Nugget Alignment

The purpose of the final stage is to answer the second sentence assessment question listed above (*Does the sentence convey information about at least one nugget, does it contain only irrelevant information, or does it contain no information?*). This is done by iterating through all sentences from every report being assessed to identify the nugget or nuggets for which the sentence contains an answer. Unlike assessment of the relationship between the report sentence and the cited document in the previous stage, here, we judge based only on sentence content.

[2] https://dmice.ohsu.edu/trec-biogen/evaluation.html

Figure 2: Screenshot of Stage 3 annotation in the web annotation tool.

If a report sentence supports a nugget, we can record the identity of that nugget. This information can be used to determine how repetitive the generated response is and to avoid penalizing sentences with the same nugget for having no citation after the first mention. Since the annotator is already referencing the gold nugget set when determining whether the sentence contains a nugget, recording this information does not introduce additional burden on the annotator, only some bookkeeping in the annotation tool.

Table 1 shows how each combination of an answer to the second sentence assessment question with the citation assessment is handled. 'Valid' means that the sentence is rewarded. 'Ignored' means the sentence does not affect the score. 'Hallucinated' means that the sentence should be penalized.

At the conclusion of these four stages, nuggets in all pooled documents have been extracted, and all report sentences have been judged for citation support and have been aligned with the appropriate nugget or nuggets. All reports can be scored. In addition, the resulting collection can be used as part of a reusable benchmark if we assume that automatic citation evaluation and nugget alignment are reliable. These techniques are active research topics in both IR and NLP. There is some evidence that they may be feasible [8, 17, 18].

4 Tool Implementation Details

To provide a better user experience and prevent annotation errors, we provide a web interface built using Streamlit. Figure 2 is a screenshot of the tool. The tool implements the four annotation stages with a lightweight user and project management tool that manage multiple simultaneous annotation projects.

The task configuration is defined as a JSON file stored at the server side. Topic files, pooled documents, and generated responses for evaluation are all separate files linked from the configuration file. In particular, three items are not implemented as part of the tool (though supporting scripts are provided in the repository): document pooling, sentence definition, and the final scoring mechanism. Each task and annotation project has different pooling decisions, different annotation units for the generated responses (we refer to them as sentences, but they can be as small as phrases), and different scoring and evaluation schemes. These decisions are orthogonal to the annotation process and should be explicit decisions made by the project manager. Therefore, they are explicitly excluded from the scope of the tool.

The annotated data is primarily stored in an SQLite database that is independent for each annotation project. Nuggets are also stored in a separate JSON file in the output directory for easy access. The project manager can decide to pre-populate nuggets created during another annotation process, such as topic development, before the start of Stage 1; this can accelerate annotation. If the pre-populated nugget questions are believed to be complete, Stages 1 and 2 can also be skipped, advancing directly to the document support stage.

To allow the annotator to acknowledge annotation errors during nugget creation, Stage 4 allows the annotator to identify a question that could have been a nugget but was not included in the previous stages. This allows the project manager to investigate annotation results post hoc and assess whether the gold nugget set is complete.

5 Conclusion

This work presents an annotation protocol and associated tool for evaluating retrieval-augmented generation with nuggets. Our process consists of four stages to ensure that the gold nugget set is complete and generated responses are judged for support by cited documents and topicality to the user requirements. Our tool is currently deployed for annotation of the 2024 TREC NeuCLIR Track Report Generation Pilot.

Disclaimer

Certain software or materials are identified in this paper in order to specify the experimental procedure adequately and is not intended to imply recommendation or endorsement of any product or service by NIST. These opinions, recommendations, findings, and conclusions do not necessarily reflect the views or policies of NIST or the United States Government.

References

[1] Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-rag: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511* (2023).

[2] Bolei He, Nuo Chen, Xinran He, Lingyong Yan, Zhenkai Wei, Jinchang Luo, and Zhen-Hua Ling. 2024. Retrieving, Rethinking and Revising: The Chain-of-Verification Can Improve Retrieval Augmented Generation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*. 10371–10393.

[3] Yue Huang, Lichao Sun, Haoran Wang, Siyuan Wu, Qihui Zhang, Yuan Li, Chujie Gao, Yixin Huang, Wenhan Lyu, Yixuan Zhang, et al. 2024. Trustllm: Trustworthiness in large language models. *arXiv preprint arXiv:2401.05561* (2024).

[4] Kalpesh Krishna, Erin Bransom, Bailey Kuehl, Mohit Iyyer, Pradeep Dasigi, Arman Cohan, and Kyle Lo. 2023. LongEval: Guidelines for Human Evaluation of Faithfulness in Long-form Summarization. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*. 1650–1669.

[5] Dawn Lawrie, Sean MacAvaney, James Mayfield, Paul McNamee, Douglas W. Oard, Luca Soldanini, and Eugene Yang. 2025. Overview of the TREC 2024 NeuCLIR Track. In *The Thirty-Third Text REtrieval Conference (TREC 2024) Proceedings*.

[6] Christina Lioma, Birger Larsen, Wei Lu, and Yong Huang. 2016. A study of factuality, objectivity and relevance: three desiderata in large-scale information retrieval?. In *Proceedings of the 3rd IEEE/ACM International Conference on Big Data Computing, Applications and Technologies* (Shanghai, China) *(BDCAT '16)*. Association for Computing Machinery, New York, NY, USA, 107–117. https://doi.org/10.1145/3006299.3006315

[7] 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.

[8] Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. FActScore: Fine-grained Atomic Evaluation of Factual Precision in Long Form Text Generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 12076–12100.

[9] 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*. Association for Computational Linguistics, Boston, Massachusetts, USA, 145–152. https://aclanthology.org/N04-1019

[10] Arjun Panickssery, Samuel R Bowman, and Shi Feng. 2024. Llm evaluators recognize and favor their own generations. *arXiv preprint arXiv:2404.13076* (2024).

[11] Xiao Pu, Mingqi Gao, and Xiaojun Wan. 2023. Summarization is (almost) dead. *arXiv preprint arXiv:2309.09558* (2023).

[12] Shahzad Rajput, Matthew Ekstrand-Abueg, Virgil Pavlu, and Javed A. Aslam. 2012. Constructing test collections by inferring document relevance via extracted relevant information. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management* (Maui, Hawaii, USA) *(CIKM '12)*. Association for Computing Machinery, New York, NY, USA, 145–154. https://doi.org/10.1145/2396761.2396783

[13] Yijia Shao, Yucheng Jiang, Theodore Kanell, Peter Xu, Omar Khattab, and Monica Lam. 2024. Assisting in Writing Wikipedia-like Articles From Scratch with Large Language Models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, Kevin Duh, Helena Gomez, and Steven Bethard (Eds.). Association for Computational Linguistics, Mexico City, Mexico, 6252–6278. https://doi.org/10.18653/v1/2024.naacl-long.347

[14] Ian Soboroff. 2024. Don't Use LLMs to Make Relevance Judgments. *arXiv preprint arXiv:2409.15133* (2024).

[15] John Sweller. 1988. Cognitive Load During Problem Solving: Effects on Learning. *Cognitive Science* 12, 2 (1988), 257–285. https://doi.org/10.1207/s15516709cog1202_4

[16] Ellen M. Voorhees and Dawn M. Tice. 2000. The TREC-8 Question Answering Track. In *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC'00)*, M. Gavrilidou, G. Carayannis, S. Markantonatou, S. Piperidis, and G. Stainhauer (Eds.). European Language Resources Association (ELRA), Athens, Greece. https://aclanthology.org/L00-1018/

[17] Miriam Wanner, Seth Ebner, Zhengping Jiang, Mark Dredze, and Benjamin Van Durme. 2024. A Closer Look at Claim Decomposition. *arXiv preprint arXiv:2403.11903* (2024).

[18] Miriam Wanner, Benjamin Van Durme, and Mark Dredze. 2024. DnDScore: Decontextualization and Decomposition for Factuality Verification in Long-Form Text Generation. *arXiv preprint arXiv:2412.13175* (2024).

[19] Koki Wataoka, Tsubasa Takahashi, and Ryokan Ri. 2024. Self-preference bias in llm-as-a-judge. *arXiv preprint arXiv:2410.21819* (2024).

[20] Xiang Yue, Boshi Wang, Ziru Chen, Kai Zhang, Yu Su, and Huan Sun. 2023. Automatic Evaluation of Attribution by Large Language Models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*. 4615–4635.