UCPH-RMT at CheckThat! 2026: Evaluating Text Summaries, Atomic Facts, and Semantic Triplets as RAG Context for Fact-Checking Article Generation

CheckThat! Lab at CLEF 2026

Robin Svahn[1,*], Maria Maistro[1]

[1]Department of Computer Science, University of Copenhagen, Universitetsparken 1, 2100 Copenhagen, Denmark

Abstract

This paper describes the UCPH-RMT submission to Task 3 of the CLEF 2026 CheckThat! Lab, which asks systems to turn a claim, its veracity label, and a set of evidence documents into a complete fact-checking article with inline citations. The submission is a deliberately simple, modular Retrieval-Augmented Generation pipeline built entirely on small open-source Llama models. Within this fixed pipeline the work studies a single factor, namely how the retrieved evidence is represented before it reaches the writer, comparing raw chunks, claim-informed Text Summaries, Atomic Facts, and Semantic Triplets. On the official test set the system reached the highest entailment score among all participating teams while placing in the middle of the leaderboard, behind the leaders on citation precision and recall.

Keywords

Fact-checking, Retrieval-Augmented Generation, Evidence representation, Article generation, CLEF CheckThat!

1. Introduction

Professional fact-checking organizations such as PolitiFact and Snopes do not stop at assigning a verdict to a claim. They publish full articles that give context and explain how the gathered evidence supports the conclusion. Automated fact-checking systems, by contrast, have largely stopped at a veracity label with a short justification [1]. Task 3 of the CLEF 2026 CheckThat! Lab targets this gap. Given a claim, its veracity label, and a set of evidence documents, a participating system must produce a complete fact-checking article with inline citations [2].

Large Language Models can generate fluent long-form prose, but they are prone to hallucination and to inventing or misattributing citations [3, 4, 5]. Retrieval-Augmented Generation grounds generation in supplied documents [6], yet passing the retrieved evidence as raw, noisy passages can still degrade factual accuracy and citation correctness [7, 8].

This work takes a deliberately simple position. Instead of engineering a complex writing process, it builds a straightforward modular pipeline on small open-source models and varies only one factor, the representation of the retrieved evidence before it reaches the writer. The retrieval step, the article structure, and the writing model are held fixed. The objective of the experiments is to compare four evidence representations under the official Task 3 metrics while every other part of the pipeline stays constant.

1. **Baseline (raw chunks).** The retrieved chunks are passed to the writer without any transformation.
2. **Text Summaries.** Each chunk is compressed into a short, claim-informed summary.
3. **Atomic Facts.** Each chunk is decomposed into isolated, single-information statements.
4. **Semantic Triplets.** Each chunk is converted into explicit subject-predicate-object relations.

CLEF 2026 Working Notes, 21 – 24 September 2026, Jena, Germany

*Corresponding author.

gcs195@alumni.ku.dk (R. Svahn); mm@di.ku.dk (M. Maistro)

0009-0005-9143-8255 (R. Svahn); 0000-0002-7001-4817 (M. Maistro)

The central research question is the following. To what extent does the structural representation of retrieved evidence influence the quality of generated fact-checking articles? The contributions are modest and practical. This work provides a simple pipeline that guarantees citation coverage and correct source attribution by construction, a controlled comparison of four evidence representations under the official metrics, and a transparent account of how a source-accounting design choice inflates one metric while slightly depressing others.

2. Background

The task of generating full fact-checking articles was formalized by Sahnan et al. [1], who also proposed QRAFT, a multi-agent framework that targets article quality through an iterative writing and editing process. However, the format of the retrieved text can easily misdirect a language model. Cuconasu et al. [7] showed that while models can handle some irrelevant data, unstructured text and poorly aligned context significantly reduce generation accuracy. At the same time, asking a model to synthesize several documents at once raises hallucination rates and makes the output sensitive to input ordering [9, 10]. Inspired by these challenges, the proposed system evaluates whether transforming retrieved text into a structured evidence format reduces noise. It also avoids multi-document synthesis issues by keeping the writing pipeline simple and processing one source at a time.

3. Task and Data

In Task 3 a system receives a claim, a normalized veracity label, and a set of evidence documents that a professional fact-checker consulted, each stored as the scraped text of a source page. The system does not decide whether the claim is true. It writes an article that explains the verdict and attributes each factual statement to a source through an inline citation [2].

Submissions are scored on four metrics, each in the range from 0 to 1, using the evaluation provided by the task organizers [2]. The *bidirectional entailment* score is a reference-based natural language inference metric that measures semantic agreement between the generated article and a human-authored reference article [1, 2]. *Citation recall* measures whether the cited sources contain enough information to support each sentence, and *citation precision* measures whether every cited source is necessary. This necessity and sufficiency formulation of citation precision and recall originates with the ALCE benchmark [11]. *Evidence coverage* measures the fraction of available premise articles that are actually cited in the output.

Development used WatClaimCheck [12], which provides both the evidence documents and the published reference article for each claim. The work used the validation split restricted to claims reviewed by PolitiFact and Snopes, since these two organizations align with the test set and carry relatively low noise. This yields 2,454 of the 3,372 validation claims. The official test set on which the submission was scored contains 1,158 claims, made up of 940 claims from ExClaim and 218 from AmbiguousSnopes, and does not include reference articles. The task overview reports slightly different planning figures for the test set [2]. A profiling analysis found that roughly 20% of premise articles across the corpus are unusable in their provided form, for example skeleton pages, error pages, or near-empty files, which the pipeline must cope with at runtime.

4. System Description

The system is a modular pipeline[1] in which each claim passes through six stages, namely loading, cleaning, chunking, evidence extraction, planning, and writing. A visualization of the stages is shown in Figure 1.

[1]The source code, configurations, and the exact prompts for every stage are available at https://github.com/robinsvahn/ucph-rmt.

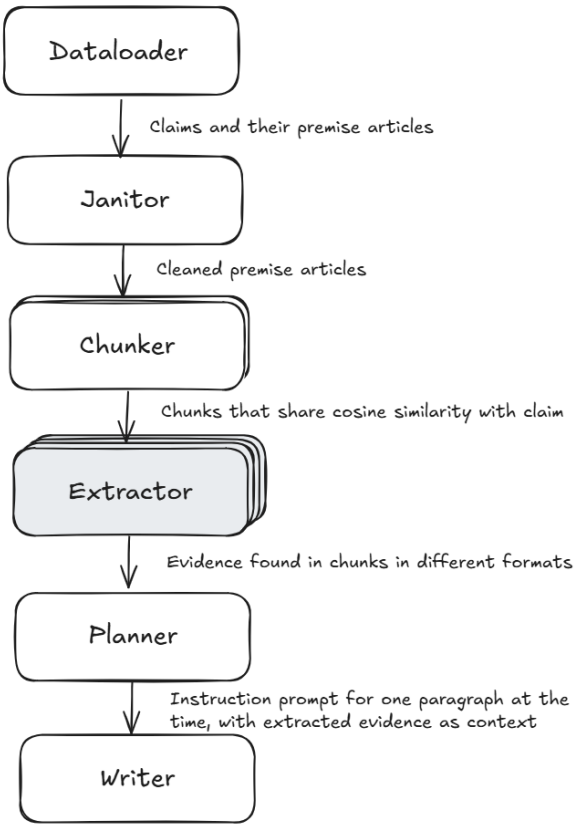

Figure 1: Overview of the modular pipeline architecture. Shared components (white) remain constant across all configurations. The extraction stage (shaded) is the primary controlled variable.

Loading, cleaning, planning, and writing are identical across all configurations. The evidence extraction stage is the primary studied variable, and the chunking stage is a secondary one. The pipeline processes a single claim as follows.

1. **Load** the claim, its veracity label, and the set of premise article URLs with their scraped text.
2. **Clean** each premise article with the janitor, rejecting unusable documents and stripping boilerplate.
3. **Chunk** the cleaned text and keep only the segments whose embedding similarity to the claim passes a threshold.
4. **Extract** evidence from the surviving chunks using one of the four representations (the studied variable).
5. **Rank** and keep the ten most claim-relevant extracted elements per source.
6. **Plan and write** the article, generating one paragraph per source plus an introduction and a verdict, then append the deterministic citations and the source-accounting end-note.

Loader and janitor. The loader reads a dataset split and selects claims from the chosen reviewers. The janitor then cleans the raw premise text at two levels. At the file level it rejects whole documents that consist of navigation links, error pages, login pages, or are empty. At the line level it strips navigation bars, social media boilerplate, image metadata, and copyright notices with a rule-based matcher. Removing this noise matters because roughly a fifth of premise articles are unusable in their provided form, and noisy input degrades both extraction and citation.

Chunker and ranking. The chunker splits the cleaned text and keeps only chunks whose cosine similarity to the claim passes a threshold, using the all-MiniLM-L6-v2 [13] sentence encoder. Two

chunker variants are compared as a secondary ablation. A multi-sentence variant groups text into roughly 200-word blocks and keeps chunks at or above a similarity of 0.35. A single-sentence variant applies a two-pass filter, first dropping low-relevance blocks below 0.20 and then keeping individual sentences at or above a stricter 0.50. These thresholds were set manually after initial tests on the validation set. The two thresholds trade recall of relevant text against precision. After extraction, a ranking step keeps the ten most claim-relevant elements per source, which bounds the writer's prompt size and avoids context saturation.

Per-source isolation and deterministic citation. The planner groups evidence by source and calls the writer once per source, so that each source produces exactly one paragraph. The writer generates only what the evidence says in context of the claim. It never sees or produces a URL. The planner appends the source URL as the citation after the writer finishes. This design has two useful properties. It avoids multi-document synthesis, where small models are known to hallucinate and to be sensitive to input ordering [9, 10]. It also removes citation hallucination as a failure mode, because the model never generates citations at all. If the writer returns nothing usable for a source, the planner falls back to a verbatim rendering of the extracted evidence, so every source with evidence appears in the article.

Every generated article therefore follows the same fixed three-part structure. An introduction paraphrases the claim and states what is being checked, a body of premise-analysis paragraphs presents exactly one paragraph per source, and a closing "Our Verdict" section weighs the evidence and explains the rating. This layout mirrors the structure of typical PolitiFact and Snopes articles and keeps the generation task small and predictable for the writer.

Evidence representations. The four representations differ only in how each chunk is rewritten before it reaches the writer. The baseline forwards the raw chunks unchanged and serves as the control. The Text Summaries extractor produces a short claim-informed summary per chunk, where the claim is included in the prompt so that the model keeps only the passages that bear on the specific verification question, acting as a targeted relevance filter. The Atomic Facts extractor decomposes each sentence into self-contained statements, guided by a few-shot prompt, inspired by FActScore [14]. The Semantic Triplets extractor converts text into open-vocabulary subject-predicate-object relations using an LlamaIndex [15] path extractor. Unlike the Text Summaries extractor, the Atomic Fact and Semantic Triplet extractors do not see the claim, because the chunker has already filtered for relevance and their task is faithful decomposition rather than further filtering. Extraction runs at temperature 0.0 for determinism, and the writer runs at temperature 0.3 for a more natural result.

Table 1 makes the four representations concrete on a single retrieved chunk from one claim in the development set. The same chunk yields fluent prose under the Text Summaries extractor, a list of discrete statements under the Atomic Facts extractor, and typed relations under the Semantic Triplets extractor.

For the same claim, the writer turns the evidence from one source into a single body paragraph, after which the planner appends the source URL as a citation. The following excerpt shows one such paragraph together with the deterministic end-note that accounts for the remaining premises.

> According to a source, Milwaukee County's savings from Act 10 were not sufficient to cover the full extent of the state aid cuts, with the county's estimated $18 million in savings from employee benefits offsetting only about half of the nearly $29 million in lost state aid. (source: jsonline.com)
>
> *Additional sources were investigated, but they either had corrupt information or were not deemed relevant. (source: example_one.com; example_two.com)*

Source-accounting end-note. As a final step, the pipeline lists any premise URL that is not cited in the article body in a deterministic end-note, stating that the source was investigated but found unusable or not relevant. The intent is transparency, so that the article accounts for every provided premise, including those that contributed no usable evidence. As shown in Section 5, this choice has a strong effect on the evaluation metrics.

Table 1
One retrieved chunk from a single source (claim 26978, "While Act 10 allowed (Milwaukee County) to save some money, it was millions short of what we needed to fill the hole left by the $28 million cut in state aid", rated half-true) shown under each evidence representation. Only the representation stage differs, and all other pipeline components are identical.

Representation	Output for the chunk
Baseline (raw)	The cuts in the county's state aid reached nearly $29 million, but saving through employee benefits offset only about $18 million of that, according to a county estimate.
Text Summaries	The county's state aid cuts were nearly $29 million, but the estimated savings from employee benefits only covered about $18 million of that amount.
Atomic Facts	The cuts in state aid were worth hundreds of millions of dollars. Milwaukee Public Schools received $82 million in state aid cuts. The county's state aid was cut by nearly $29 million.
Semantic Triplets	(state aid, CUTS_IN, hundreds of millions of dollars); (Milwaukee Public Schools, HAS_BEEN_AFFECTED_BY, state aid cuts); (state aid cuts, AMOUNT_OF, 82 million)

Models and configurations. All generation roles use Llama-3.2-3B-Instruct in the 3B tier and Llama-3.1-8B-Instruct in the 8B tier [16], run through the HuggingFace Transformers library. The Llama family was chosen partly because the official citation judge is also a Llama model, but also due to being available in both 3B and 8B parameter instruct settings. The official evaluation uses roberta-large-mnli [17] for entailment and Llama-3.2-1B-Instruct as the citation judge. The generation models run in fp16 on the cluster GPUs and are cached by identifier so that all roles in a run share one loaded instance. Prompts are capped at a budget of 3072 tokens and left-truncated when they exceed it.

Compute. All experiments were run on a shared, limited-access Slurm cluster. The 3B configurations were scheduled on whichever commodity GPUs were free, including NVIDIA Titan RTX and Titan X, Xp, and V cards. The 8B configurations require around 20 GB of VRAM and were scheduled on NVIDIA A40, L40S, or A100 nodes.

Experimental design. Each of the four evidence representations is evaluated under four configurations, formed by crossing the two chunking strategies (multi-sentence and single-sentence) with the two model scales (3B and 8B). This gives 16 configurations in total. Every other component, along with all similarity thresholds, the top-ten ranking, and the generation temperatures, is held fixed across the 16 runs. This means the differences in the scores can be attributed to the representation, the chunker, or the model scale. Each configuration is evaluated on two claim sets that represent different input quality levels. The unfiltered set keeps all premise articles, including noisy ones, and is the condition closest to the official test set. The strict set keeps only claims for which every premise article passes the janitor and yields at least one chunk above the relevance threshold, which separates the effect of the representation from the effect of input quality.

5. Results

The configuration submitted to Task 3 was Text Summaries with the multi-sentence chunker at the 8B scale. It was selected because it achieved the highest entailment score in development, where entailment was treated as the primary quality signal, and because it was substantially faster than the triplet pipeline under heavy cluster load before the deadline. Runtimes are not comparable across the full set, because the GPUs varied from run to run. But when comparing every 8B configuration that ran on the same L40S GPU, averaged over both the single-sentence and multi-sentence chunkers there is a clear picture. The average time per claim was about 0.3 minutes for the baseline, 0.5 minutes for Text Summaries, 1.7 minutes for Atomic Facts, and 4.6 minutes for Semantic Triplets. The LlamaIndex Semantic Triplets

extractor was therefore the slowest by a wide margin, more than an order of magnitude over the baseline.

Text Summaries trailed the structured representations on citation, so the submission traded citation correctness for the stronger entailment signal and a safer runtime. The closest alternative on entailment, Atomic Facts, carried a bug at submission time that lowered its scores and was fixed only afterwards, so it was not a viable choice then.

Table 2 places the submission on the official leaderboard, where teams are ranked by the arithmetic mean of the four metrics. The submission (UCPH-RMT) placed fifth. It achieved the highest entailment score of all participating teams, including the teams ranked above it, which supports the focus on evidence representation as a lever for semantic faithfulness. The gap to the leaders comes almost entirely from citation precision and recall, where several teams scored much higher. Evidence coverage was near the ceiling for several teams and does not separate systems in this task.

Table 2
Official CLEF 2026 CheckThat! Lab Task 3 leaderboard. UCPH RMT is our submission. Mean is the arithmetic mean of all four metrics. Best value per column in bold.

Rank	Team	Ent.	Cit. Prec.	Cit. Rec.	Ev. Cov.	Mean
1	Outsider	0.278	**0.671**	**0.671**	0.563	**0.546**
2	UTS	0.341	0.299	0.299	**0.996**	0.484
3	DS GT CheckThat	0.295	0.240	0.276	0.902	0.428
4	Facty	0.229	0.231	0.254	0.995	0.427
5	**UCPH-RMT (ours)**	**0.364**	0.144	0.174	0.993	0.419
6	Ro	0.321	0.463	0.463	0.343	0.398
7	Sourceminds	0.245	0.337	0.339	0.394	0.329
8	CheckMate	0.221	0.350	0.350	0.391	0.328
9	KNU Fact	0.227	0.287	0.295	0.440	0.312
–	*Baseline*	*0.298*	*0.223*	*0.240*	*0.329*	*0.272*

5.1. Effect of Evidence Representation

Table 3 reports the four representations at the 8B model scale under both chunking strategies, on two claim sets. Run 1 uses 150 unfiltered claims that include noisy premise articles and is the condition closest to the official test set. Run 3 uses 150 claims pre-selected with the strictest filter, where every premise article passes the janitor cleaning module and also yields at least one chunk that survives the relevance filter. This size balances evaluation coverage against the computational time needed to run all 16 configurations on limited available hardware. The focus here is on the 8B tier, which gives the strongest configurations. The 3B variants were comparable on entailment but generally lower on citation precision and recall, so the larger model mainly helps the citation metrics rather than the semantic faithfulness.

The submitted configuration, Text Summaries with the multi-sentence chunker, reached the best entailment on the noisy Run 1 set (0.392), which is why it was selected for the official submission. On the strict Run 3 set the strongest configuration overall is Semantic Triplets with the single-sentence chunker, which reaches the highest entailment (0.419) together with the highest citation precision and recall (0.206). Atomic Facts follows closely on both dimensions. Text Summaries keeps an entailment advantage over the baseline but trails the structured representations on citation precision and recall, and on Run 1 it even falls below the baseline there. A plausible reading is that summaries produce fluent prose that matches the reference style but lose the per-source specificity the citation judge rewards, whereas structured representations keep each statement tied to a single source. The citation metrics rise markedly from Run 1 to Run 3 for every configuration, for example from 0.170 to 0.206 recall for Semantic Triplets with the single-sentence chunker, which shows how much input quality affects the pipeline's ability to produce well-supported citations. Single-sentence chunking tends to help on the

Table 3
Per-configuration scores at the 8B model scale (Llama-3.1-8B-Instruct) for each evidence representation under both chunking strategies. Run 1 is the unfiltered claim set, Run 3 the strict subset where every premise article passes the janitor and yields at least one relevant chunk. Ent. is entailment, Prec. and Rec. are citation precision and recall. Evidence coverage is omitted because it is uniform at about 0.993 by design across all rows, as discussed in Section 5. Best value per column in bold.

		Run 1 (unfiltered)			Run 3 (strict)		
Representation	**Chunker**	**Ent.**	**Prec.**	**Rec.**	**Ent.**	**Prec.**	**Rec.**
Baseline	Multi	0.353	0.118	0.118	0.364	0.171	0.171
Baseline	Single	0.363	0.118	0.133	0.397	0.154	0.154
Text Summaries	Multi	**0.392**	0.103	0.108	0.395	0.131	0.132
Text Summaries	Single	0.381	0.119	0.135	0.410	0.133	0.134
Atomic Facts	Multi	0.369	0.137	0.137	0.403	0.187	0.187
Atomic Facts	Single	0.386	0.155	0.166	0.407	0.190	0.190
Semantic Triplets	Multi	0.378	**0.167**	0.169	0.401	0.204	0.204
Semantic Triplets	Single	0.380	0.157	**0.170**	**0.419**	**0.206**	**0.206**

clean Run 3 set, while its effect on the noisy Run 1 set is mixed. The absolute citation scores remain low across the board, which reflects the difficulty of the task and the limits of the small citation judge.

5.2. The Evidence-Coverage Caveat

The evidence-coverage scores in Table 2 sit near a perfect 1.00 score. This is not a model capability. It is a direct result of the source-accounting end-note, which causes nearly every provided URL to appear somewhere in the output. Evidence coverage counts a premise URL as covered whenever it appears in a citation anywhere in the output. The end-note lists the unused sources in that same citation format, so they count toward coverage even though the body draws no evidence from them. To make this explicit, the two best configurations from the strictly filtered development data were re-evaluated with the end-note removed. Table 4 reports the result on the unfiltered claim set, which is closest to the test-set conditions.

Table 4
Effect of removing the source-accounting end-note on the unfiltered claim set. Without the end-note, evidence coverage reflects the fraction of premise articles that actually contributed usable content.

Configuration	**Entailment**	**Cit. Prec.**	**Cit. Recall**	**Ev. Cov.**
Atomic Facts, single, 8B	0.406	0.153	0.153	0.499
Semantic Triplets, single, 8B	0.410	0.158	0.158	0.498

Once the end-note is removed, evidence coverage falls to about 0.50 on unfiltered data. In other words, the pipeline finds usable content in roughly half of the premise articles, and the end-note was the only mechanism citing the rest. Entailment improves slightly when the end-note is removed, because the end-note text does not match any content in the reference article. Citation precision and recall are largely unchanged, because the end-note adds at most one cited sentence and so carries little weight in the per-sentence average that the citation judge computes. The end-note was designed for transparency toward a human reader who wants to know which sources were considered. Under the automated metrics it inflates coverage while contributing nothing to the more important citation scores. The reported coverage near 0.99 therefore does not reflect how much evidence the article body actually uses, and the figure closer to 0.50 is the more honest measure.

6. Conclusion and Future Work

This paper described a simple, modular pipeline for generating fact-checking articles, built on small open-source models, and used it to study how the representation of retrieved evidence affects article quality. The system reached the highest entailment score among all participating teams while placing in the middle of the leaderboard, which shows that careful but simple design can compete on semantic faithfulness even with small models. The more honest reading of the result, however, is that the system is outperformed on the most important metrics for the task, citation precision and recall, by everyone.

The reasons are not certain, but a few stand out. The judge scores one sentence at a time, while the writer builds a full paragraph with the claim as context. The last sentence, which carries the citation, may reason about the claim rather than restate what the source says. That can leave the judge unable to link it back to the cited page. Both the writer and the judge are small models, which adds further noise.

Atomic Facts and Semantic Triplets both improve citation precision and recall over Text Summaries and over the raw baseline, while Text Summaries keep only an entailment edge, so the submitted summaries are not the strongest choice for attribution. The effect is small and clearest on clean input. Future work will explore a citation strategy better suited to the leave-one-out precision test, larger models, and human evaluation to check whether the metric differences reflect differences a reader would notice.

Declaration on Generative AI

During the preparation of this work, the authors used Claude Opus 4.7 in order to: Formatting assistance, Improve writing style, Grammar and spelling checks. After using this tool, the authors reviewed and edited the content as needed and takes full responsibility for the publication's content.

References

[1] D. Sahnan, D. Corney, I. Larraz, G. Zagni, R. Miguez, Z. Xie, I. Gurevych, E. Churchill, T. Chakraborty, P. Nakov, Can llms automate fact-checking article writing?, Transactions of the Association for Computational Linguistics 14 (2026) 489–509. URL: https://direct.mit.edu/tacl/article/doi/10.1162/TACL.a.644/136336/Can-LLMs-Automate-Fact-Checking-Article-Writing. doi:10.1162/TACL.a.644.

[2] J. M. Struß, S. Schellhammer, S. Dietze, V. V, V. Setty, T. Chakraborty, P. Nakov, A. Anand, P. Chungkham, S. Hafid, D. Sahnan, K. Todorov, The CLEF-2026 CheckThat! Lab: Advancing Multilingual Fact-Checking, 2026. URL: https://arxiv.org/abs/2602.09516. doi:10.48550/ARXIV.2602.09516, version Number: 1.

[3] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. Bang, D. Chen, W. Dai, H. S. Chan, A. Madotto, P. Fung, Survey of Hallucination in Natural Language Generation, ACM Computing Surveys 55 (2023) 1–38. URL: http://arxiv.org/abs/2202.03629. doi:10.1145/3571730, arXiv:2202.03629 [cs.CL].

[4] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, T. Liu, A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions, ACM Transactions on Information Systems 43 (2025) 1–55. URL: http://arxiv.org/abs/2311.05232. doi:10.1145/3703155, arXiv:2311.05232 [cs.CL].

[5] G. Zuccon, B. Koopman, R. Shaik, ChatGPT Hallucinates when Attributing Answers, 2023. URL: https://arxiv.org/abs/2309.09401v1.

[6] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, D. Kiela, Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks, 2021. URL: http://arxiv.org/abs/2005.11401. doi:10.48550/arXiv.2005.11401, arXiv:2005.11401 [cs.CL].

[7] F. Cuconasu, G. Trappolini, F. Siciliano, S. Filice, C. Campagnano, Y. Maarek, N. Tonellotto, F. Silvestri, The Power of Noise: Redefining Retrieval for RAG Systems, 2024. URL: https://arxiv.org/abs/2401.14887v4. doi:10.1145/3626772.3657834.

[8] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, M. Wang, H. Wang, Retrieval-Augmented Generation for Large Language Models: A Survey, 2024. URL: http://arxiv.org/abs/2312.10997. doi:10.48550/arXiv.2312.10997, arXiv:2312.10997 [cs.CL].

[9] C. G. Belem, P. Pezeshkpour, H. Iso, S. Maekawa, N. Bhutani, E. Hruschka, From Single to Multi: How LLMs Hallucinate in Multi-Document Summarization, 2025. URL: http://arxiv.org/abs/2410.13961. doi:10.48550/arXiv.2410.13961, arXiv:2410.13961 [cs.CL].

[10] J. Ma, Input Order Shapes LLM Semantic Alignment in Multi-Document Summarization, 2025. URL: http://arxiv.org/abs/2512.02665. doi:10.48550/arXiv.2512.02665, arXiv:2512.02665 [cs.CL].

[11] T. Gao, H. Yen, J. Yu, D. Chen, Enabling Large Language Models to Generate Text with Citations, in: H. Bouamor, J. Pino, K. Bali (Eds.), Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Singapore, 2023, pp. 6465–6488. URL: https://aclanthology.org/2023.emnlp-main.398/. doi:10.18653/v1/2023.emnlp-main.398.

[12] K. Khan, R. Wang, P. Poupart, WatClaimCheck: A new Dataset for Claim Entailment and Inference, in: S. Muresan, P. Nakov, A. Villavicencio (Eds.), Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Dublin, Ireland, 2022, pp. 1293–1304. URL: https://aclanthology.org/2022.acl-long.92/. doi:10.18653/v1/2022.acl-long.92.

[13] N. Reimers, I. Gurevych, Sentence-bert: Sentence embeddings using siamese bert-networks, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2019. URL: https://arxiv.org/abs/1908.10084.

[14] S. Min, K. Krishna, X. Lyu, M. Lewis, W.-t. Yih, P. W. Koh, M. Iyyer, L. Zettlemoyer, H. Hajishirzi, FActScore: Fine-grained Atomic Evaluation of Factual Precision in Long Form Text Generation, 2023. URL: https://arxiv.org/abs/2305.14251v2.

[15] J. Liu, Llamaindex, 2022. URL: https://github.com/jerryjliu/llama_index, released on 2022-11-01.

[16] A. Grattafiori, A. Dubey, A. Jauhri, et al., The Llama 3 Herd of Models, 2024. URL: http://arxiv.org/abs/2407.21783. doi:10.48550/arXiv.2407.21783, arXiv:2407.21783 [cs.AI].

[17] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, Roberta: A robustly optimized bert pretraining approach (2019). URL: http://arxiv.org/abs/1907.11692. doi:10.48550/arXiv.1907.11692, arXiv:1907.11692.