A Framework of Reflective Agents with Adaptive Collaboration for Attributed Summary Generation

Yu Chen[1], **Peng Chen**[1], **Ziwei Zheng**[2], **Bang Wang**[2,*]
[1]School of Software Engineering,
Huazhong University of Science and Technology, Wuhan, China
[2]School of Electronic Information and Communications,
Huazhong University of Science and Technology, Wuhan, China
{hustchenyu, hustchenpeng, zhengziwei, wangbang}@hust.edu.cn

Abstract

Despite progress in LLM summarization, factual hallucinations persist, motivating Attributed Summary Generation (ASG), which requires sentence-level citations. However, existing prompt-based approaches face severe challenges such as positional preference, poor citation quality and sensitivity to uninformative documents. In view of these limitations, we propose **RAAC**, a framework of **R**eflective **A**gents with **A**daptive **C**ollaboration for attributed summarization. RAAC performs iterative summarization via reflective agents' collaboration, where a post reflection module evaluates the consistency between the summary and the input documents, based on which it critiques the summary and uses the resulting feedback to recalibrate the inputs to the next adaptive iteration. The agents' collaboration involves two components: `TextAgent` and `CitationAgent`. Experimental results on the ALCE benchmark demonstrate that our framework outperforms existing baselines in both factual correctness and citation quality.

1 Introduction

Large Language Models (LLMs) have made remarkable progress in summarization; however, factual errors and hallucinations severely limit their reliability in real-world applications (Ji et al., 2023; Shuster et al., 2021). If generated summaries are attributed, their reliability can be significantly improved. To address these issues, a task called **A**ttributed **S**ummary **G**eneration (ASG) (Deng et al., 2024; Sun et al., 2023; Gao et al., 2023b; Slobodkin et al., 2024) has been proposed, as shown in Figure 1. In this task, the LLM is required to generate a summary given a question and a set of documents ranked by semantic similarity to the question. And the LLM must also provide citations for each summary sentence. Attributed summaries

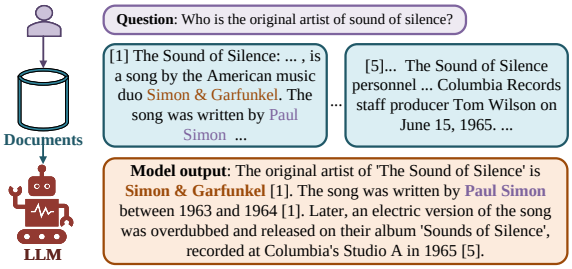


Figure 1: The ASG task involves generating a summary in response to a given question from an ordered set of documents, while providing citations for each sentence.

offer dual benefits: enhancing verifiability and improving faithfulness to the given documents.

Current research on ASG can be categorized into two broad lines of research: fine-tuning LLMs and guiding LLMs through prompting strategies. The first approach, fine-tuning LLMs (Aly et al., 2024; Huang et al., 2024; Ye et al., 2023), aims to enhance LLMs' attribution capability through supervised training and reinforcement learning. However, this approach currently faces two major challenges. First, most existing datasets suffer from serious deficiencies in high-quality citation annotations, making them inadequate for supporting effective fine-tuning (Deng et al., 2024). Moreover, fine-tuning approaches that focus too heavily on specific scenarios, which may limit their effectiveness in commercial systems that must adapt to diverse application contexts.

The second approach, prompting strategies (Kamalloo et al., 2023; Gao et al., 2023b), attempt to produce attributed summaries through leveraging LLMs' in-context learning ability, where attribution instructions are embedded in prompt. However, these approaches have several limitations.

First, LLMs often exhibit positional preference when processing long input of multiple documents (Zhu et al., 2025). Specifically, they tend to

*Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2026, pages 27406–27425
July 2-7, 2026

overemphasize the beginning and ending portions of the input while neglecting the middle, resulting in incomplete summary content (Wright et al., 2025). Second, most existing LLMs lack a built-in mechanism for attribution (Gao et al., 2023a), which leads to incorrect citations and missing citations for some sentences. Third, the quality of attributed summaries is sensitive to uninformative documents in the input (Gao et al., 2023b), which can distract the LLM and cause incorrect or missing citations. Document informativeness should be judged against the summary rather than the question, since some documents may become uninformative to the summary during generation. Therefore, it is better not to include such documents in the input.

Although LLMs possess strong generative capabilities and the potential for self-feedback and refinement (Madaan et al., 2023), the above analysis indicates that existing prompt-based approaches for attributed summarization, on the one hand, struggle to mitigate positional preference and to perform precise citation verification. On the other hand, reasoning-based citation validation can improve attribution accuracy, and a dynamic refinement mechanism can leverage LLMs' prior knowledge and semantic understanding to reassess document informativeness against the generated summary, removing summary-uninformative documents to form an adaptive document set, on which low-quality summaries can be improved.

These observations underscore the need for an adaptive agent collaboration framework with iterative reasoning, which dynamically refines the input document set and iteratively regenerates the summary for more accurate and reliable attribution.

Motivated by the above analysis, we propose a framework of **R**eflective **A**gents with **A**daptive **C**ollaboration (**RAAC**) for attributed summarization. RAAC performs iterative summarization via reflective agents' collaboration. After each collaboration round, a post-reflection module evaluates the consistency between the generated summary and the input documents. It then critiques the summary and produces feedback. This feedback is used to adjust the inputs for the next collaboration round, enabling feedback-driven adaptive collaboration among agents.

Specifically, RAAC consists of three core modules: `TextAgent`, `CitationAgent` and `Summary Calibration`. (1) `TextAgent` alternates between using the LLM as an evaluator and as an optimizer, performing multi-aspect self-assessment and feedback-directed optimization to reintegrate overlooked information and improve summary quality; (2) `CitationAgent` leverages the logical-consistency judgment capability of an NLI model to perform post-hoc reasoning over the generated summary, thereby improving citation recall and precision; (3) `Summary Calibration` (SC) serves as a post-reflection module. After agent collaboration, it reflects on the generated summary. Specifically, it assesses the consistency between the input documents and the summary. `SC` then uses the resulting feedback to adaptively adjust the inputs, ensuring that they remain informative for subsequent summarization.

The method pipeline is shown in Figure 2. A discussion of related work is provided in Appendix A.

2 Method

2.1 Task Definition

Following (Gao et al., 2023b; Huang et al., 2024; Sun et al., 2023), the ASG task is formalized as follows: given a question Q and a set of top-5 documents $\mathcal{D}$ ranked by semantic similarity to Q, an LLM $\mathbb{L}$ is required to generate a summary $S = \{s_i\}_i$ consisting of sentences each associated with a set of citations $\{c_{ik}\}_k$. Specifically, the citation format for each s_i is like $[c_{i1}][c_{i2}]$.

2.2 TextAgent: Summary Generation

An LLM often exhibits positional preference when its input is too long (Liu et al., 2023; Wright et al., 2025). Recent study (Madaan et al., 2023) has reported that even if an LLM performs poorly on its first attempt for summarization, it can provide useful feedback to be exploited for further refining the output. Motivated by these considerations, we design the `TextAgent` focusing on high quality summarization first. The operation of the `TextAgent` consists of the following steps:

Initial generation The LLM $\mathbb{L}$ first produces a summary $S^{(0)}$ for the given Q and $\mathcal{D}$ by using the *summary generation prompt* P_{sg}. Details of P_{sg} is provided in Appendix Table 12. $S^{(0)}$, as shown in Figure 3, is used as the initial input for the next *feedback stage*.

Feedback stage We also employ the LLM $\mathbb{L}$ serving as an evaluator to assess the summaries $S^{(\rho)}$,

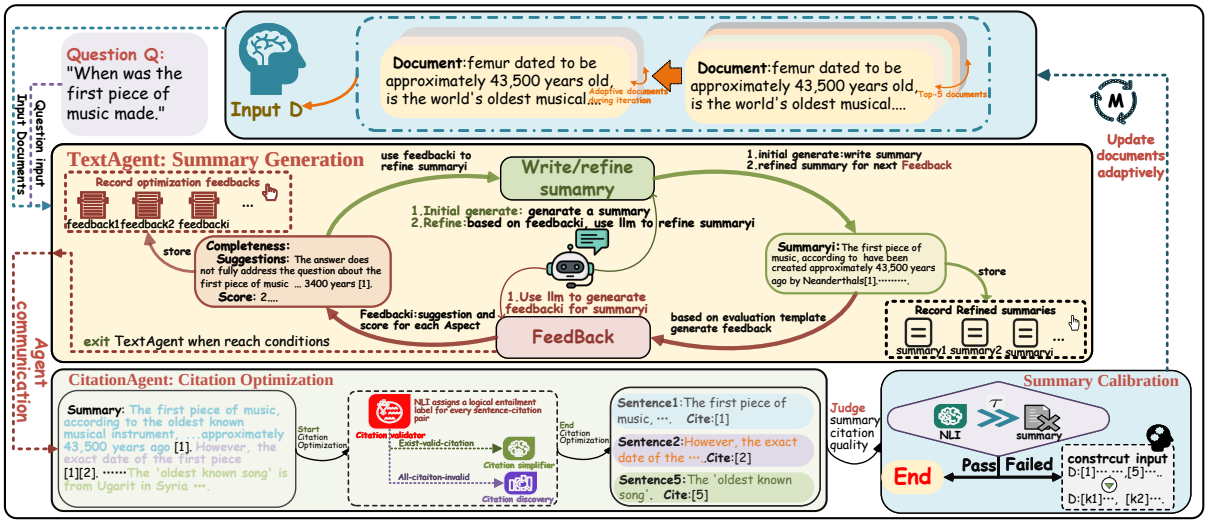


Figure 2: The illustration of RAAC, which mainly consists of three modules: `TextAgent`, `CitationAgent`, and `Summary Calibration`, where iterative agent-collaboration is triggered only when the generated summary fails the `Summary Calibration` evaluation.

Instructions: Your task is to generate a comprehensive summary based on the provided documents that directly answers the user's question, including proper citations.
Question: "When was the first piece of music made."
Document: {Documents}

Output Summary ($S^{(0)}$)**:**
The first piece of music, according to archaeological findings, was created approximately 43,500 years ago by Neanderthals [1]. Evidence suggests ... Paleolithic age [4].

Figure 3: *Initial generation* : $\mathbb{L}$ generates $S^{(0)}$ based on Q and $\mathcal{D}$.

Instructions: Evaluate the summary according to the following criteria. For each criterion, give a score from 1 to 3 points, and provide suggestions for improvement.
Evaluation Criteria:
Completeness: The generated summaries can fully address the input questions.
Question: "When was the first piece of music made."
Document: {Document Context}
Summary ($S^{(0)}$)**:** The first piece of music, ..., was created approximately 43,500 years ago by Neanderthals [1]...

Output Feedback:
Completeness:
– **suggestions**: The answer does not fully address the question about the first piece of music. It would be better to mention the oldest known song from Ugarit in Syria, which dates back 3,400 years.
– **Score**: 2/3

Figure 4: *feedback stage* example: generating *scores and suggestions* for $S^{(0)}$.

with ρ denoting the ρ-th round iteration of generated summary. Specifically, we design several evaluation aspects for the LLM-based summary evaluation, including *completeness, objectivity, specificity* etc. We design a *summary evaluation prompt* P_{se} to instruct the LLM $\mathbb{L}$ to output a score and suggestion for each aspect. P_{se} is provided in Table 13 in the Appendix.

For instance, this stage evaluates the summary $S^{(0)}$ on predefined aspects. As shown in the orange part of Figure 4, $\mathbb{L}$ generates a specific suggestion: *"Ugarit in Syria, which dates back 3400 years."*, alongside with a score of 2. Ultimately, the output feedback comprises a set of *scores and suggestions* across all predefined aspects.

Refine stage we also employ the LLM $\mathbb{L}$ serving as a reviser to refine the summary $S^{(\rho-1)}$, where ρ denotes the current iteration. Specifically, we design a *summary refinement prompt* P_{sr} that provides $\mathbb{L}$ with the summary $S^{(\rho-1)}$ and the aspect-level suggestions from the ρ-th *Feedback stage*, and instructs $\mathbb{L}$ to revise $S^{(\rho-1)}$ accordingly, producing a refined summary $S^{(\rho)}$. Detailed P_{sr} is provided in Table 14 in the Appendix.

For instance, given $S^{(0)}$, $\mathbb{L}$ is guided by the *scores and suggestions* in Figure 5 to refine $S^{(0)}$ by adding the missing temporal detail 3400 years. As a result, the refined summary explicitly states the key temporal information *3400 years ago*.

The *Feedback and Refine* iteration will stop when one of the following conditions is met. It includes two cases: the iteration count reaches k or the summary achieves the maximum score (3 points on all predefined evaluation aspects during the *Feedback stage*. The output of `TextAgent` is $S_{\text{ta}} = \{s_i\}_i$ and each s_i has an associated citation list $\mathcal{C}_i = \{c_{ik}\}_k$. The detailed algorithmic work-

Instructions: Our goal is to revise responses that improve over time based on points and Modification Suggestions.

scores and suggestions:
Completeness: score: 2 suggestion: The answer does not fully address the question about the first piece of music... It would be better to mention the oldest known song from Ugarit in Syria, which dates back 3400 years.

Question: "When was the first piece of music made."
Document: {Document}

Refine Summary:
The first piece of music, in the form of the oldest known song from Ugarit in Syria, dates back to approximately 3400 years ago[1] ..... dating back to between 7000 and 6600 BC [3].

Figure 5: Refine stage example for summary $S^{(0)}$ based on scores and suggestions from *Feedback stage*.

flow is provided in Algorithm 1.

2.3 CitationAgent: Citation Optimization

Relying solely on the holistic reasoning of an LLM $\mathbb{L}$ often fails to simultaneously ensure high-quality summary content and reliable citations (Gao et al., 2022, 2023b). To mitigate this, we introduce `CitationAgent`, a modular post-processing process that refines the citation for the `TextAgent` output through a sentence-level citation optimization pipeline. Specifically, for summary S_{ta}, the pipeline executes three stage:

Citation validator For each $s_i \in S_{\text{ta}}$, we evaluate all non-empty citation subsets of its existing citation list $\mathcal{C}_i = \{c_{ik}\}_k$. Specifically, we enumerate all subsets $\mathcal{C} \subseteq \mathcal{C}_i$ with $\mathcal{C} \neq \emptyset$.

For each pair $(s_i, \mathcal{C})$, we concatenate the documents referenced by $\mathcal{C}$ to form the premise and treat s_i as the hypothesis. We adopt TrueTeacher(NLI model) (Sun et al., 2023) to determine whether the premise entails the hypothesis, yielding a binary label $label \in \{0, 1\}$. Here, $label = 1$ indicates that the premise entails s_i. We thus obtain a tuple $(s_i, \mathcal{C}, label)$ for each pair $(s_i, \mathcal{C})$.

Citation simplifier For each sentence s_i, we collect all candidate citation subsets $\mathcal{F}_i$ validated by the *Citation validator*,

$$\mathcal{F}_i = \{\mathcal{C} \subseteq \mathcal{C}_i \mid \mathcal{C} \neq \emptyset,\ \text{label}(s_i, \mathcal{C}) = 1\}.$$

To reduce redundant citations, we select a minimum-cardinality subset from $\mathcal{C} \in \mathcal{F}_i$:

$$\mathcal{C}_i^* = \arg\min_{\mathcal{C} \in \mathcal{F}_i} |\mathcal{C}|.$$

If multiple subsets attain the same minimum cardinality, we break the tie by selecting the subset that minimizes the sum of citation indices in $\mathcal{C}_i$. *Citation simplifier* outputs the optimized citation set $\mathcal{C}_i^*$ for each s_i, resulting in either (i) *preserved citations* when $\mathcal{C}_i^* = \mathcal{C}_i$, or (ii) *reduced citations* when $|\mathcal{C}_i^*| < |\mathcal{C}_i|$.

Citation discovery this stage handles each sentence s_i whose $\mathcal{C}_i = \emptyset$ after *Citation simplifier*. We use Dense Passage Retriever(DPR) (Karpukhin et al., 2020) to encode the sentence s_i into a vector representation z_i and encode each document $d \in \mathcal{D}$ accordingly. Each document d is then ranked according to their embedding similarity to z_i. Subsequently, we design an *AddCitation prompt* P_{a} that provides an LLM $\mathbb{L}$ with s_i and the reranked documents, and ask $\mathbb{L}$ to select an index set of document that entail s_i. The selected citations form $\mathcal{C}_i^*$ and are attached to s_i in format of $[c'_{i1}][c'_{i2}]$. Details P_{a} can be found in Table 15 in the Appendix.

The output of *CitationAgent* is $S_{\text{ca}} = \{s_i\}_i$, where each si has an associated citation list $\mathcal{C}_i^* = \{c'_{ik}\}_k$.

2.4 Summary Calibration (SC)

During attributed summarization with `TextAgent` and `CitationAgent`, sentence-level citation alignment can be compromised by uninformative documents in the input. To mitigate this, we introduce `Summary Calibration` (SC) as a scheduler that separates low-quality input samples and re-dispatches the agents for iterative refinement. Specifically, SC leverages its submodule `Summary Assessment` to compute a *citation F1* for the generated summary S_{ca} of each input sample and compares it against a predefined threshold τ. Summaries with *citation F1* exceeding τ are considered satisfactory and require no further optimization. In contrast, for input samples with *citation F1* below the τ, `SC` dynamically calibrates the input document set $\mathcal{D}$ and feeds the refined document set into a new round of iterative summarization.

Summary Assessment `SC` assesses the citation quality of S_{ca}. For each sentence s_i with optimized citations $\mathcal{C}_i^*$, we compute sentence-level *citation recall* as

$$r_i = \mathbf{1}\big[\text{label}(s_i, \mathcal{C}_i^*) = 1\big]$$

Only when $r_i = 1$ (i.e., the full citation set entails s_i) do we further assess *citation precision* by identifying essential citations. Specifically, a citation $c'_{ik} \in \mathcal{C}_i^*$ is essential if it either entails s_i

Method	ASQA					ELI5				
	Corr.EM	ROUGE-Lsum	Citation Rec.	Citation Pre.	Citation F1	Corr. Claim.	ROUGE-Lsum	Citation Rec.	Citation Pre.	Citation F1
FRONT (Huang et al., 2024)	40.84	36.95	77.70	69.89	73.59	9.18	19.09	**58.60**	55.33	56.92
Self-RAG 7B (Asai et al., 2024)	30.00	35.70	66.90	67.80	67.30	9.70	16.90	23.30	33.90	27.60
RAAC (ours)	**41.23**	**37.22**	**79.69**	**79.08**	**79.38**	**13.43**	**20.50**	56.77	**63.19**	**59.81**

Table 1: We compare RAAC with baselines that fine-tune LLMs and report main results on ASQA and ELI5. Bold and underlined numbers denote the best and second-best results, respectively.

Method	Dataset: ASQA												
	Evaluate Summary Content										Evaluate Citation		
	Corr. EM	Rouge-1	Rouge-2	Rouge-4	Rouge-L	Rouge-Lsum	BERTscore	Meteor	BLEU	SAC	Citation Rec.	Citation Pre.	Citation F1
VANILLA	36.14	33.05	17.06	6.52	24.36	34.57	85.88	21.58	7.30	2.22	65.84	61.97	63.85
SUMM	34.78	29.46	14.85	4.98	22.10	31.24	85.74	18.11	4.58	2.06	64.72	64.55	64.63
SNIPPET	29.07	28.02	14.28	5.42	21.26	29.90	85.34	17.46	4.89	1.74	49.37	57.63	53.18
RERANK	38.06	24.58	17.23	6.41	25.00	35.31	86.22	22.71	8.25	2.34	61.23	62.89	62.05
RAAC (ours)	**41.23**	**36.59**	**18.12**	**7.16**	**26.15**	**37.22**	**86.41**	**24.98**	**10.38**	**2.42**	**79.69**	**79.08**	**79.38**

Method	Dataset: ELI5												
	Evaluate Summary Content										Evaluate Citation		
	Corr. EM	Rouge-1	Rouge-2	Rouge-4	Rouge-L	Rouge-Lsum	BERTscore	Meteor	BLEU	SAC	Citation Rec.	Citation Pre.	Citation F1
VANILLA	13.40	**22.77**	3.12	0.18	**12.70**	20.17	82.98	17.16	**1.41**	1.82	39.42	39.67	39.54
SUMM	9.70	18.82	2.40	0.14	11.35	16.44	82.89	12.32	0.61	1.47	43.00	41.20	42.08
SNIPPET	10.53	20.78	2.91	0.18	12.07	17.86	**83.24**	13.81	0.87	1.44	24.51	34.88	28.79
RERANK	**13.90**	22.69	3.00	0.18	12.62	19.99	83.02	15.90	1.32	**1.85**	40.39	41.49	40.93
RAAC (ours)	13.43	22.75	3.08	0.16	12.63	**20.50**	82.63	**17.72**	1.37	1.64	**56.77**	**63.19**	**59.81**

Table 2: Main results on ASQA and ELI5. Bold and underlined numbers denote the best and second-best results, respectively. Comparisons between RAAC and prompt-based baselines.

individually or is necessary for $\mathcal{C}_i^*$ to entail s_i:

$$\begin{aligned}\mathrm{Ess}(s_i, c'_{ik}) = \mathbf{1}\left[\mathrm{label}(s_i, \mathcal{C}_i^*) = 1\right] \\ \cdot \mathbf{1}\big[\mathrm{label}(s_i, \{c'_{ik}\}) = 1 \\ \vee\, \mathrm{label}(s_i, \mathcal{C}_i^* \setminus \{c'_{ik}\}) = 0\big]\end{aligned}$$

We then define sentence-level *citation precision* (computed only for sentence s_i with $r_i = 1$) as

$$p_i = \frac{\sum_{c'_{ik} \in \mathcal{C}_i^*} \mathrm{Ess}(s_i, c'_{ik})}{|\mathcal{C}_i^*|}$$

We report overall *citation recall* and *precision* by averaging r_i and p_i over sentences, and derive the *citation F1* accordingly. If the *citation F1* falls below a predefined threshold τ, *Summary Assessment* identifies the current S_{ca} as low-quality sample and triggers a calibration routing to reconfigure the input $\mathcal{D}$ for subsequent refinement.

Implicit LLM Preference Reconstruction When calibration is triggered, we redefines the input document set $\mathcal{D}$ by constructing an adaptive subset based on the `TextAgent`'s ouput. Specifically, given $S_{\mathrm{ta}} = \{s_i\}_i$, we use regular expression matching to extract the citation set $\mathcal{C}_i$ for each sentence s_i, and take the union of all extracted citations:

$$\mathcal{C}_{union} = \bigcup_{i=1}^{n} \mathcal{C}_i$$

The adaptive input $\mathcal{D}_{adp}$ then includes the documents referenced by $\mathcal{C}_{union}$. This design leverages the LLM's internalized prior knowledge to prune uninformative documents from the previous input $\mathcal{D}$. By treating the `TextAgent`'s selection as an implicit informativeness heuristic, we distill a concentrated input $\mathcal{D}_{adp}$ that aligns with the LLM's semantic preference. This reduction of input alleviates the burden of uninformative context in subsequent stages.

Quality Gating With the input $\mathcal{D} = \mathcal{D}_{adp}$, `SC` invokes the agents sequentially in each iteration and controls the stopping criteria. To ensure the efficacy of refinement, a candidate summary is accepted only if its *citation F1* shows a substantive improvement over the previous round. If this condition is not met, `SC` discards the new summary, reverts to the previous version, and terminates further optimization for the current question.

The process continues until the iteration count reaches a predefined maximum M, detailed analysis can be found in Appendix D, or the *citation F1* exceeds the threshold τ. Moreover, to balance performance and cost, the threshold τ is treated as a tunable hyperparameter, as detailed in Section 4.4. The workflow of SC is provided in Algorithm 2.

Method	Correct (Em Rec)	ROUGE-Lsum	Citation Rec.	Citation Pre.	Citation F1.
Qwen2.5-7b					
VANILLA	36.14	34.57	65.84	61.97	63.85
SUMM	34.78	31.24	64.72	64.55	64.63
SNIPPET	29.07	29.90	49.37	57.63	53.18
RERANK	38.06	35.31	61.23	62.89	62.05
RAAC (ours)	**41.23**	**37.22**	**79.69**	**79.08**	**79.38**
Qwen2.5-14b					
VANILLA	32.73	32.48	63.75	71.07	67.21
SUMM	35.63	32.38	58.36	61.00	59.65
SNIPPET	34.16	31.91	56.17	66.53	60.91
RERANK	34.90	32.13	71.94	71.28	71.61
RAAC (ours)	**41.43**	**37.05**	**79.33**	**77.91**	**78.62**
Qwen2.5-32b					
VANILLA	41.99	37.32	74.02	68.60	71.21
SUMM	41.67	34.56	73.39	62.73	67.64
SNIPPET	34.21	33.94	54.10	62.60	58.04
RERANK	40.55	36.17	73.17	70.65	71.89
RAAC (ours)	**45.06**	**40.03**	**85.04**	**83.99**	**84.51**

Table 3: Performance comparison across different Qwen2.5 model sizes on ASQA Dataset. Bold and underlined numbers denote the best and second-best results, respectively.

3 Experiments

Datasets and Evaluation Metrics We conduct our experiments on the long-form datasets in the ALCE (Gao et al., 2023b), namely ASQA (Stelmakh et al., 2022) and ELI5 (Fan et al., 2019). Following ALCE, our evaluation focuses on two aspects: summary content quality and citation quality. Detailed descriptions of all metrics and dataset are presented in the Appendix B.2.

Baselines For an equitable comparison, we adopt the best-performing baseline methods proposed in the ALCE benchmark, which are also adopted by (Sun et al., 2023). These methods include VANILLA, SUMM, SNIPPET, and RERANK. Detailed descriptions of these baselines are available in Appendix B.1. To ensure a fair comparison against our RAAC, we implement all four baselines using Qwen2.5-7B as the backbone. Furthermore, we include results from strong training-based methods, including Self-RAG 7B (Aly et al., 2024) and FRONT (Huang et al., 2024), both fine-tuned on LLaMA2-7B.

Implementation Details We implement RAAC using the Qwen2.5-7B model as the backbone LLM. In the `CitationAgent` module, we employ `t5_11b_trueteacher_and_anli7` as the NLI model introduced in (Sun et al., 2023). Each experiment is conducted three times, and we report the average results. More details about the experimental settings are presented in Appendix B.3 and B.4.

4 Analysis

4.1 Main Result

We contrast the performance of our method with existing approaches based on model fine-tuning, and the results are presented in Table 1. Our framework improves ASG performance while preserving the base LLM's generative capabilities, which facilitates plug-and-play deployment in diverse commercial systems.

Moreover, we compare our method with four prompting-based methods (Li et al., 2023) on the ASG, and the results are presented in Table 2. Our method substantially improves both content quality and citation quality on ASQA and ELI5, indicating stronger attribution accuracy while retaining salient information from long documents.

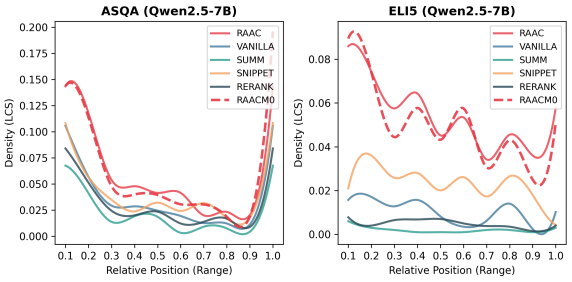
Scalability of RAAC performance with model size. To assess scalability, we evaluate RAAC across LLMs of different sizes on ASQA and results are presented in Table 3. Performance on ASG consistently improves as model scale increases, mainly because larger LLMs have stronger semantic understanding and generate higher-quality calibration signals, enabling more effective iterative refinement.

Method	ALCE Evaluator			DeepSeek-V3 Evaluator			Qwen2.5-32B Evaluator		
	Cite. Rec.	Cite. Pre.	Cite. F1	Cite. Rec.	Cite. Pre.	Cite. F1	Cite. Rec.	Cite. Pre.	Cite. F1
ASQA Dataset									
VANILLA	65.84	61.97	63.85	74.06	72.82	73.43	69.85	67.68	68.75
SUMM	64.72	64.55	64.63	66.57	61.98	64.19	62.15	57.74	59.87
SNIPPET	49.37	57.63	53.18	48.67	51.87	50.22	46.33	50.57	48.35
RERANK	61.23	62.89	62.05	67.46	67.57	67.51	64.45	64.51	64.48
RAAC	**79.69**	**79.08**	**79.38**	**87.04**	**85.69**	**86.36**	**81.34**	**80.12**	**80.73**
ELI5 Dataset									
VANILLA	39.42	39.67	39.54	60.11	61.61	60.85	55.77	56.89	56.33
SUMM	43.00	41.20	42.08	58.60	53.41	55.88	51.73	47.63	49.56
SNIPPET	24.51	34.88	28.79	30.67	39.94	34.70	28.89	37.66	32.69
RERANK	40.39	41.49	40.93	60.95	65.60	63.19	58.63	63.13	60.79
RAAC	**56.77**	**63.19**	**59.81**	**67.62**	**75.62**	**71.39**	**64.55**	**72.45**	**68.27**

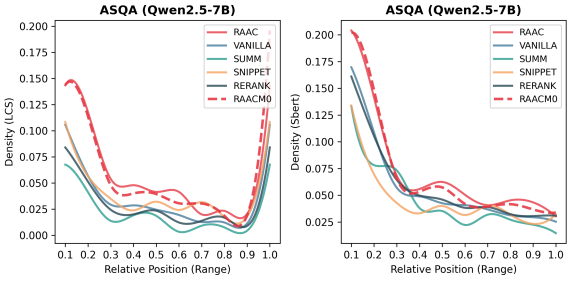
Table 4: Performance comparison using ALCE, DeepSeek-V3 and Qwen2.5-32B as citation evaluators. Bold and underlined numbers denote the best and second-best results, respectively.

Robustness of citation evaluation. To further im-

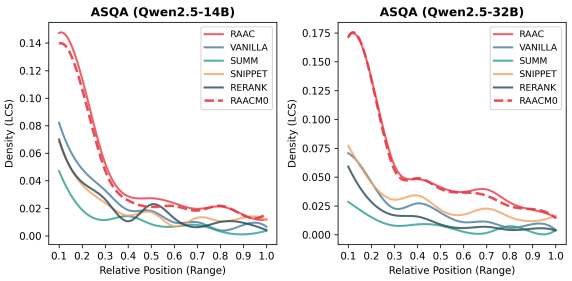
prove the reliability of our result in citation metric, we introduce an additional LLM-based evaluation protocol. The corresponding *evaluation prompt* P_e is provided in Table 16. Specifically, we use Deepseek-V3 and Qwen2.5-32B as automated evaluators. Given each sentence and its cited documents, the evaluator judges whether the documents entail the sentence. As presented in the Table 4, RAAC outperforms all baselines across all citation metrics under both evaluators, indicating that the improvements are robust and suggesting the gains are not evaluator-specific.



(a) Positional distribution of document used in summaries across methods on ASQA and ELI5 (Qwen2.5-7B)



(b) Positional distribution on ASQA under different matching metrics (LCS vs SBERT; Qwen2.5-7B).



(c) Effect of model scale on positional distribution: Qwen2.5-14B vs Qwen2.5-32B on ASQA.

Figure 6: Distribution of document positions from which summary sentences are derived across different methods.

4.2 Evaluation of positional preference

To quantitatively evaluate the effectiveness of RAAC in mitigating LLMs' positional preference,we analyze where in the input documents the summary content originates. Specifically, we compute the longest common subsequence(LCS) between each summary sentence and the input documents. We then associate each sentence with its relative position in the input documents based on the matched span (using an overlap threshold of 50%). In particular, by comparing RAAC with RAACM0 (RAAC without SC), we can control for the effect of SC and isolate the positional-distribution changes attributable to TextAgent.

The resulting distributions are shown in Figure 6a. The analysis reveals that prompting-based methods (VANILLA, SUMM, SNIPPET, and RERANK) have limited ability to extract information from the middle sections of long input documents. In contrast, RAAC shows a clear advantage in capturing information from the central region of the input and improving the overall information acquisition ratio.

This result indicates that the mechanism in TextAgent is effective. Combined with the main results, these findings suggest that, by mitigating positional preference, RAAC encourages the LLM to form a more comprehensive understanding of the input and improves the content quality of the generated summaries.

To further validate this conclusion, we conduct two additional checks. *Evaluation robustness.* To reduce dependence on a specific matching algorithm, we replace LCS with SBERT-based semantic similarity (80% overlap threshold). As shown in Figure 6b, the results shows trends consistent with the LCS-based results. *Scalability.* We repeat the LCS-based source-distribution analysis with larger models (Qwen2.5-14B and Qwen2.5-32B) in Figure 6c, and observe consistent improvements across model sizes.

4.3 Ablation Study

We conduct an ablation study on ASQA and ELI5 datasets to isolate the contribution of each component in our method. The results, presented in Table 5, compare our full method against the following configurations.

Qwen2.5 (Base): The Qwen2.5-7B model without any components of RAAC, serving as the baseline performance of the underlying LLM. *w/o* TextAgent: This configuration removes iterative self-refinement and replaces TextAgent with the vanilla strategy. *w/o* CitationAgent: This configuration removes CitationAgent to evaluate the impact of post-hoc verification. *w/o* SC: In this set-

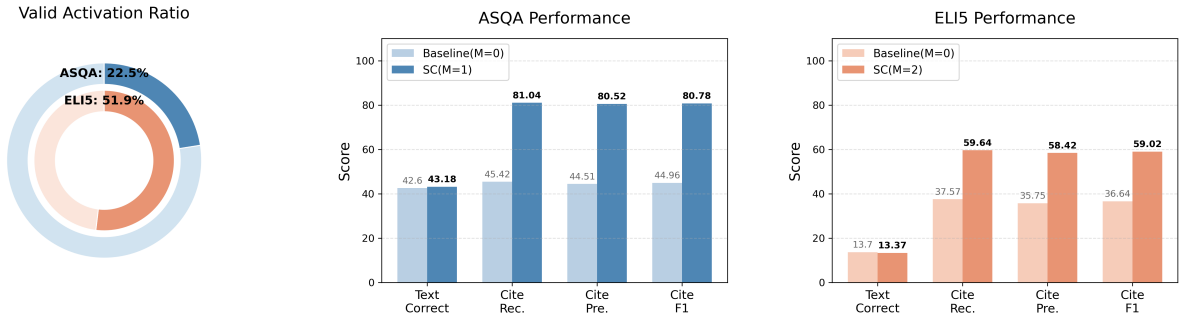


Figure 7: This figure provides a detailed comparison of content correctness and citation quality metrics for low-quality questions under Qwen2.5-7B, with and without SC.

ting, SC is removed. The summary is generated in a single pass by TextAgent and CitationAgent.

CitationAgent improves summary quality by enhancing citation faithfulness. As presented in Table 5, adding CitationAgent yields substantial improvements in both citation recall and precision. In addition, Citation Discovery supplements missing citations for sentences the LLM fails to attribute, strengthening overall citation faithfulness in TextAgent's summaries.

Summary Calibration (SC) strengthens citation quality, as shown in Table 5. We further inspect cases with poor attribution in the agents' cooperative outputs to examine how SC affects both factual correctness and citation quality and results are shown in Figure 7.

Overall, these results indicate that SC supports reflective agent adaptive collaboration by dynamically updating the document set based on agents' feedback during iterative summarization. This adaptive reconstruction filters summary-uninformative documents, improves citation quality, and ultimately yields higher-quality summaries. From these observations, we draw two conclusions: (i) a subset of questions in ASQA and ELI5 benefit from adaptive input reconstruction; and (ii) the calibrated document set $\mathcal{D}$ improves citation quality by filtering uninformative documents.

Method	ASQA				ELI5			
	Correct Em Rec	Cite Rec.	Cite Pre.	Cite F1.	Correct (Em Claim.)	Cite Rec.	Cite Pre.	Cite F1.
RAAC	41.23	79.69	79.08	79.38	13.43	56.77	63.19	59.81
Qwen2.5-7b(Base)	36.14	65.84	61.97	63.85	13.40	39.42	39.67	39.54
RAAC w/o TextAgent	35.39	79.31	79.10	79.20	12.4	56.27	54.72	55.48
RAAC w/o CitationAgent	40.89	76.86	67.90	72.10	12.9	47.61	56.73	51.77
RAAC w/o SC	40.10	70.63	69.90	70.26	13.70	47.46	56.93	51.76

Table 5: Ablation experiment results. The table shows the effect on the results on the ALCE-ASQA and ALCE-ELI5 when removes different modules in our method.

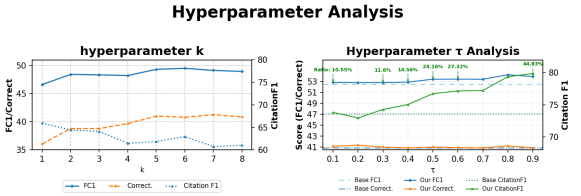


Figure 8: Hyperparameter Analysis in TextAgent and SC.

4.4 Hyperparameter Analysis

We analyze two key hyperparameters of our method: the number of refinement iterations, k, in TextAgent, and the SC threshold: τ. Results are shown in Figure 8.

To evaluate the robustness of our framework with respect to k, we sweep k from 1 to 8. As shown in Figure 8, the $FC1$ metric (see Appendix B.2 for details) stabilizes for $k \in [5, 8]$. Moreover, performance fluctuations within this range are minimal, indicating that the TextAgent module is relatively insensitive to k.

The SC threshold τ determines the citation-quality level below which citations are considered poor, thus activating SC. We conduct a comprehensive study with τ varying from 0.1 to 0.9. Figure 8 shows that $FC1$ increases steadily as τ increases, and $\tau = 0.8$ consistently achieves the best overall performance. In addition, when $\tau \geq 0.3$, our overall quality measured by $FC1$, as well as citation F1 and correctness, surpasses *Front*, while a smaller proportion of samples enters the SC. This indicates that τ can be treated as a tunable hyperparameter to balance performance and cost.

5 Fixed Budget Evaluation

To further ensure a fair comparison between RAAC and the baselines (VANILLA, SNIPPET, SUMM,

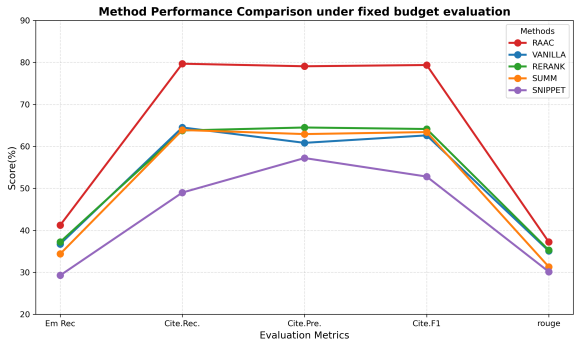


Figure 9: Method Performance Comparison under fixed budget evaluation.

and RERANK), we control the token consumption of all methods to the same budget in this module and further evaluate their performance. The results in Figure 9 show that RAAC still significantly outperforms the competing methods under the same token-budget constraint, indicating that the improvement mainly stems from the effectiveness of our method itself rather than increased token usage.

6 Conclusion

We propose RAAC for attributed summary generation (ASG), a reflective agent framework that improves both summary content and citation faithfulness through feedback-driven, adaptive collaboration. RAAC consists of three components: (i) `TextAgent`, which alternates the LLM between evaluator and optimizer to mitigate positional preference and improve content coverage and summary quality; (ii) `CitationAgent`, which post-processes `TextAgent`'s summaries by verifying and supplementing supporting citations, thereby strengthening attribution faithfulness; and (iii) `Summary Calibration (SC)`, which uses collaboration reflection to adaptively filter and reselect input documents during iterative refinement, further improving summary quality.

Overall, RAAC makes two main contributions: we introduce LLMs into TextAgent as multiple role-specific agents to perform self-refine-based summarization and iterative refinement; unlike prior paradigms that rely on retrieval-based approaches to update documents, in the SC we optimize the inputs for samples that need improvement by leveraging implicit signals produced by the model, thereby proposing a new reflection–iteration optimization framework.

As future work, we will develop more cost-efficient techniques to streamline the multi-step pipeline, reducing LLM invocations and token usage while maintaining or further improving performance. In parallel, we will explore alternative verification signals to reduce reliance on external NLI models, aiming for more robust and portable attribution across settings.

Limitations

Although our method demonstrates strong performance, it still has certain limitations. First, the quality of the NLI model affects the performance of the `CitationAgent` during the verification. If the NLI model performs poorly, mismatches may occur between documents and generated sentences, which in turn reduces the effectiveness of filtering out low-quality samples. Second, our framework requires multiple LLM invocations, leading to higher computational cost and slower real-time response. This overhead mainly comes from two sources: (i) `TextAgent` alternates between the roles of evaluator and optimizer across multiple steps; and (ii) `SC` may trigger an additional calibration-and-regeneration cycle when the current document set is deemed suboptimal. As shown in Appendix Figure 10, our method incurs higher token consumption and runtime than prompt-based baselines. This overhead reflects a deliberate trade-off: we sacrifice some efficiency to achieve better output quality and more reliable attribution. And a detailed analysis of the fixed budget evaluation is provided in Section 5.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China under Grant 62172167.

Ethics Statement

This study does not involve any specific ethical considerations.

References

Rami Aly, Zhiqiang Tang, Samson Tan, and George Karypis. 2024. Learning to generate answers with citations via factual consistency models. *arXiv preprint arXiv:2406.13124*.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-rag: Learning to retrieve, generate, and critique through self-reflection.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, and 1 others. 2022a. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, and 1 others. 2022b. Constitutional ai: harmlessness from ai feedback. 2022. *arXiv preprint arXiv:2212.08073*, 8(3).

Haolin Deng, Chang Wang, Li Xin, Dezhang Yuan, Junlang Zhan, Tian Zhou, Jin Ma, Jun Gao, and Ruifeng Xu. 2024. Webcites: Attributed query-focused summarization on chinese web search results with citations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15095–15114.

Ahmed Elgohary, Christopher Meek, Matthew Richardson, Adam Fourney, Gonzalo Ramos, and Ahmed Hassan Awadallah. 2021. Nl-edit: Correcting semantic parse errors through natural language interaction. *arXiv preprint arXiv:2103.14540*.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. Eli5: Long form question answering. *arXiv preprint arXiv:1907.09190*.

Zhangyin Feng, Weitao Ma, Weijiang Yu, Lei Huang, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and 1 others. 2023. Trends in integration of knowledge and large language models: A survey and taxonomy of methods, benchmarks, and applications. *arXiv preprint arXiv:2311.05876*.

Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. 2023. Gptscore: Evaluate as you desire. *arXiv preprint arXiv:2302.04166*.

Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Zhao, N. Lao, Hongrae Lee, Da-Cheng Juan, and Kelvin Guu. 2022. Rarr: Researching and revising what language models say, using language models. *ArXiv*, abs/2210.08726.

Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Zhao, Ni Lao, Hongrae Lee, Da-Cheng Juan, and 1 others. 2023a. Rarr: Researching and revising what language models say, using language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16477–16508.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. 2023b. Enabling large language models to generate text with citations. *arXiv preprint arXiv:2305.14627*.

Or Honovich, Roee Aharoni, Jonathan Herzig, Hagai Taitelbaum, Doron Kukliansy, Vered Cohen, Thomas Scialom, Idan Szpektor, Avinatan Hassidim, and Yossi Matias. 2022. True: Re-evaluating factual consistency evaluation. *arXiv preprint arXiv:2204.04991*.

Jie Huang and Kevin Chen-Chuan Chang. 2023. Citation: A key to building responsible and accountable large language models. *arXiv preprint arXiv:2307.02185*.

Lei Huang, Xiaocheng Feng, Weitao Ma, Yuxuan Gu, Weihong Zhong, Xiachong Feng, Weijiang Yu, Weihua Peng, Duyu Tang, Dandan Tu, and 1 others. 2024. Learning fine-grained grounded citations for attributed large language models. *arXiv preprint arXiv:2408.04568*.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM computing surveys*, 55(12):1–38.

Ehsan Kamalloo, Aref Jafari, Xinyu Zhang, Nandan Thakur, and Jimmy Lin. 2023. Hagrid: A human-llm collaborative dataset for generative information-seeking with attribution. *arXiv preprint arXiv:2307.16883*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick SH Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *EMNLP (1)*, pages 6769–6781.

Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio Savarese, and Steven Chu Hong Hoi. 2022. Coderl: Mastering code generation through pretrained models and deep reinforcement learning. *Advances in Neural Information Processing Systems*, 35:21314–21328.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Xinze Li, Yixin Cao, Liangming Pan, Yubo Ma, and Aixin Sun. 2023. Towards verifiable generation: A benchmark for knowledge-aware language model attribution. *arXiv preprint arXiv:2310.05634*.

Jiacheng Liu, Skyler Hallinan, Ximing Lu, Pengfei He, Sean Welleck, Hannaneh Hajishirzi, and Yejin Choi. 2022. Rainier: Reinforced knowledge introspector for commonsense question answering. *arXiv preprint arXiv:2210.03078*.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023. Lost in the middle: How language models use long contexts. *Preprint*, arXiv:2307.03172.

Ximing Lu, Sean Welleck, Jack Hessel, Liwei Jiang, Lianhui Qin, Peter West, Prithviraj Ammanabrolu, and Yejin Choi. 2022. Quark: Controllable text generation with reinforced unlearning. *Advances in neural information processing systems*, 35:27591–27609.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, and 1 others. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36:46534–46594.

Jianmo Ni, Chen Qu, Jing Lu, Zhuyun Dai, Gustavo Hernández Ábrego, Ji Ma, Vincent Y Zhao, Yi Luan, Keith B Hall, Ming-Wei Chang, and 1 others. 2021. Large dual encoders are generalizable retrievers. *arXiv preprint arXiv:2112.07899*.

Baolin Peng, Michel Galley, Pengcheng He, Hao Cheng, Yujia Xie, Yu Hu, Qiuyuan Huang, Lars Liden, Zhou Yu, Weizhu Chen, and 1 others. 2023. Check your facts and try again: Improving large language models with external knowledge and automated feedback. *arXiv preprint arXiv:2302.12813*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Timo Schick, Jane Dwivedi-Yu, Zhengbao Jiang, Fabio Petroni, Patrick Lewis, Gautier Izacard, Qingfei You, Christoforos Nalmpantis, Edouard Grave, and Sebastian Riedel. 2022. Peer: A collaborative language model. *arXiv preprint arXiv:2208.11663*.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, pages 31210–31227. PMLR.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. *arXiv preprint arXiv:2104.07567*.

Aviv Slobodkin, Eran Hirsch, Arie Cattan, Tal Schuster, and Ido Dagan. 2024. Attribute first, then generate: Locally-attributable grounded text generation. *arXiv preprint arXiv:2403.17104*.

Ivan Stelmakh, Yi Luan, Bhuwan Dhingra, and Ming-Wei Chang. 2022. Asqa: Factoid questions meet long-form answers. *arXiv preprint arXiv:2204.06092*.

Hao Sun, Hengyi Cai, Bo Wang, Yingyan Hou, Xiaochi Wei, Shuaiqiang Wang, Yan Zhang, and Dawei Yin. 2023. Towards verifiable text generation with evolving memory and self-reflection. *arXiv preprint arXiv:2312.09075*.

Niket Tandon, Aman Madaan, Peter Clark, Keisuke Sakaguchi, and Yiming Yang. 2021. Interscript: A dataset for interactive learning of scripts through error feedback. *arXiv preprint arXiv:2112.07867*.

Sean Welleck, Ximing Lu, Peter West, Faeze Brahman, Tianxiao Shen, Daniel Khashabi, and Yejin Choi. 2022. Generating sequences by learning to self-correct. *arXiv preprint arXiv:2211.00053*.

Dustin Wright, Zain Muhammad Mujahid, Lu Wang, Isabelle Augenstein, and David Jurgens. 2025. Unstructured evidence attribution for long context query focused summarization. *arXiv preprint arXiv:2502.14409*.

Shicheng Xu, Liang Pang, Huawei Shen, Xueqi Cheng, and Tat-seng Chua. 2023. Search-in-the-chain: Towards the accurate, credible and traceable content generation for complex knowledge-intensive tasks. *arXiv preprint arXiv:2304.14732*.

Kevin Yang, Yuandong Tian, Nanyun Peng, and Dan Klein. 2022. Re3: Generating longer stories with recursive reprompting and revision. *arXiv preprint arXiv:2210.06774*.

Michihiro Yasunaga and Percy Liang. 2020. Graph-based, self-supervised program repair from diagnostic feedback. In *International Conference on Machine Learning*, pages 10799–10808. PMLR.

Xi Ye, Ruoxi Sun, Sercan Ö Arik, and Tomas Pfister. 2023. Effective large language model adaptation for improved grounding and citation generation. *arXiv preprint arXiv:2311.09533*.

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2023. Making retrieval-augmented language models robust to irrelevant context. *arXiv preprint arXiv:2310.01558*.

Mengna Zhu, Kaisheng Zeng, Mao Wang, Kaiming Xiao, Lei Hou, Hongbin Huang, and Juanzi Li. 2025. Eventsum: A large-scale event-centric summarization dataset for chinese multi-news documents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 26138–26147.

A Related Work

A.1 Attributed Summarization

Retrieval-Augmented Generation (RAG) has emerged as a promising paradigm for knowledge-intensive tasks in recent years (Lewis et al., 2020; Karpukhin et al., 2020; Feng et al., 2023). However, recent studies (Shi et al., 2023; Yoran et al., 2023; Xu et al., 2023) have reported that existing RAG-based LLMs struggle to handle uninformative or contradictory documents and often fail to fully comprehend contextual information. These deficiencies can lead to factual hallucinations. One strategy to

mitigate this issue is to have the LLM generate citations alongside the summaries, which not only reduces hallucinations but also enhances the verifiability of the LLM's output (Huang and Chang, 2023; Li et al., 2023). Existing approaches to this problem vary. Some research, such as (Gao et al., 2023b), has explored prompting-based methods and post-retrieval processing techniques. While these methods are simple to implement, they often yield poor performance. In contrast, another line of research focuses on fine-tuning LLMs on custom-built training datasets (Huang et al., 2024; Aly et al., 2024; Ye et al., 2023; Asai et al., 2024). However, these methods typically require task-specific or domain-specific labeled data, leading to high human annotation cost, and the resulting models may not transfer well to new settings without additional fine-tuning. The method proposed in this paper is iterative optimization framework that enhances LLM performance without fine-tuning.

A.2 Self-Feedback-Refine

Humans have consistently served as a valuable source of feedback (Tandon et al., 2021; Elgohary et al., 2021; Bai et al., 2022b). However, due to the high cost associated with human feedback, some methods have employed scalar reward functions as a proxy or an alternative, such as (Bai et al., 2022a; Liu et al., 2022; Lu et al., 2022; Le et al., 2022; Welleck et al., 2022). Recently, LLMs have been utilized to generate feedback in general domains (Fu et al., 2023; Peng et al., 2023; Yang et al., 2022). Feedback-refinement pairs of have been used to train supervised refinement models (Yasunaga and Liang, 2020; Schick et al., 2022). A limitation of this approach is its domain-specific nature and the high cost of collecting supervised data. In contrast to previous work, our method leverages the same LLM to evaluate its own output and provide suggestions for revision. This is achieved by guiding the LLM through a multi-aspect chain-of-thought process. Based on these self-generated suggestions, the LLM then refines its output without training any additional models.

B More Experimental Details

B.1 Details of Baselines

The following describes the prompt-based baselines used in our experiments. All baselines use the same underlying LLM and input, and differ only in prompting and post-processing.

VANILLA (Gao et al., 2023b): A direct approach where the LLM generates a summary with citations based on a set of input documents.

SUMM (Gao et al., 2023b): A two-step method where the LLM first summarizes the input documents to distill relevant information. Subsequently, this summary is used in a new prompt to guide the LLM in generating the summary with citations.

SNIPPET (Gao et al., 2023b): This method first prompts the LLM to extract key passages or snippets from the input documents that directly address the user's question. These extracted snippets are then provided as context for the LLM to generate summary with citations.

RERANK (Gao et al., 2023b): A two-step generate-then-rank approach. The LLM first generates multiple candidate summaries based on the input documents. Each summary is then evaluated, and the one with the highest citation recall is selected as the final summary.

B.2 More Details of Datasets & Evaluation Metrics

Following (Gao et al., 2023b), we use the Wikipedia dump from December 20, 2018 as the retrieval corpus. For ASQA, we employ the Generalizable T5-based dense Retriever (GTR) (Ni et al., 2021). For ELI5, we adopt the sparse retriever BM25. The datasets used in our evaluation, **ASQA** and **ELI5**, are described below. And Table 6 presents the dataset statistics for ASQA and ELI5.

ASQA is a long-form factual QA dataset featuring ambiguous questions. Summarizing these questions requires synthesizing information from multiple sources to cover different facets and interpretations of the question.

ELI5 contains complex or abstract questions that require explanations to be presented in a simple and accessible manner, as if explaining to a five-year-old child.

Citation Quality We measure the traceability of the generated summaries from two aspects: (1) *Citation Recall*, which assesses whether the cited documents entails the sentence, and (2) *Citation Precision*, which judges whether each citation is essential to entail its associated sentence. We perform this evaluation using the TRUE (Honovich et al., 2022) model, a T5-11B (Raffel et al., 2020) model fine-tuned on a collection of NLI datasets to examine entailment between the cited documents and the model output. Additionally, we report *Citation*

F1, the harmonic mean of precision and recall:

$$F_1 = 2 \cdot \frac{\text{Citation Recall} \cdot \text{Citation Precision}}{\text{Citation Recall} + \text{Citation Precision}}.$$

Correctness EM Rec/Claim (Corr. EM Rec/-Claim) Correctness is evaluated differently for each dataset. For the ASQA, we calculate Exact Match Recall (EM Rec.), which measures whether the ground-truth short summaries appear as exact substrings in the generated output. For ELI5, we assess correctness using Claim Recall, (EM Claims), which evaluates whether the model's output logically entails the individual sub-claims in the ground truth.

Sub-Aspect Coverage(SAC) To quantitatively evaluate the sub-aspect coverage of the generated summary against the reference summary, we use Qwen2.5-32B as an automated evaluator. The model scores the content completeness of each generated summary on a scale of 0 to 5. The prompt used for this evaluation is provided in Table 17.

More details of $FC1$ To comprehensively evaluate the balance between content quality and citation quality achieved by our framework, we introduce a unified metric, $FC1$, defined as:

$$FC1 = 2 \cdot \frac{Correctness \cdot CitationF1}{Correctness + CitationF1}$$

This metric provides an integrated perspective for assessing the effects of hyperparameters.

ROUGE-N measures the recall of N-grams.

ROUGE-1 measures the recall of unigrams (single words), i.e., how many words from the reference summaries also appear in the candidate summary.

ROUGE-2: measures the recall of bigrams (pairs of words).

ROUGE-4 measures the recall of 4-grams (sequences of four words).

ROUGE-L computes the Longest Common Subsequence(LCS) between the candidate and reference summaries, and is often used to evaluate sequence-level similarity beyond exact n-gram overlap.

ROUGE-Lsum is a variant of ROUGE-L designed for longer, multi-sentence summaries. It computes LCS at the sentence level and sums then the scores to produce a summary-level score. This helps evaluate whether the summary captures main points that might be distributed across different sentences in the source.

Dataset	Split	Samples
ASQA	Test	948
ELI5	Test	1,000

Table 6: Dataset statistics.

parameters	ASQA	ELI5
Initially documents number D	5	5
k in `TextAgent`	5	5
τ in `SC`	0.8	0.8
M in `SC`	1	2

Table 7: Parameter settings used for RAAC on different datasets in the main experiments with Qwen2.5-7B.

BLEU measures n-gram overlap between a candidate text and reference texts, and is commonly used in machine translation.

METEOR is based on an F-measure that combines precision and recall. It extends exact matching with stemming and synonym matching (e.g., via WordNet), and applies a fragmentation penalty that penalizes disordered word alignments.

BERTScore is a semantic-based metric that computes cosine similarity between contextual token embeddings in the candidate and reference texts, and reports precision, recall, and F1.

B.3 More Implementation Details

In our experiments, we access Qwen2.5 models of three parameter scales (7B, 14B, and 32B) and the Deepseek-V3 LLMs are accessed via APIs provided by the Alibaba Cloud Bailian platform. For the NLI components, we locally deployed the `t5_11b_trueteacher_and_anli7` and `t5_xxl_true_nli_mixture` on a server equipped with two NVIDIA A10 GPUs. The `NLTK` version is 3.9.1.

B.4 Parameter Settings of RAAC

The parameter settings for the models used in our experiments: Qwen2.5-7B, Qwen2.5-14B, Qwen2.5-32B, and Deepseek-V3, are provided in Table 7 and Table 8.

C Prompts

The prompts used in our experiments are listed as follows. Since VANILLA and RERANK share the same prompt, we only report the VANILLA prompt in Table 9. The prompts for SUMM, SNIPPET, `TextAgent`, `CitationAgent` are provided in

parameters	Qwen2.5-14B	Qwen2.5-32B	Deepseek-V3
Initially documents number D	5	5	5
k in TextAgent	5	5	5
τ in SC	0.8	0.8	0.8
M in SC	1	1	1

Table 8: Parameter settings for Qwen2.5-14B, Qwen2.5-32B, and Deepseek-V3 on ASQA.

Table 10, Table 11, Table 12, Table 13, Table 14, Table 15.

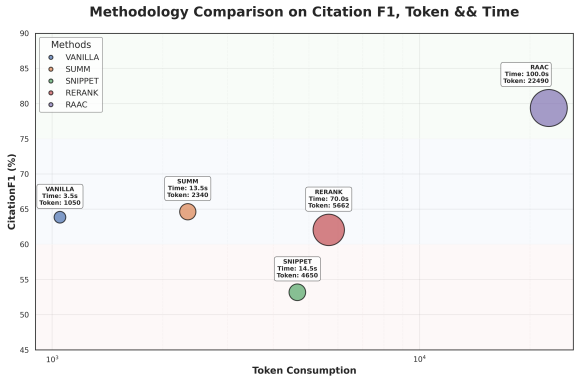


Figure 10: Performance vs Cost: Citation F1 under Token and Runtime Budgets.

D Analysis of Iteration Count M when Using SC on ASQA

To evaluate whether multiple refinement iterations are necessary for ASQA, we employ two metrics: *Jaccard Similarity* and *Bit Flip Rate*. These metrics quantify the divergence between the first iteration ($M = 1$) and the second iteration ($M = 2$), thereby assessing whether $M = 1$ is sufficient.

Metric Definition. *Jaccard Similarity* measures the degree of overlap between two sets of cited documents and is defined as:

$$J(S_1, S_2) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}$$

This metric evaluates the model at the set level. For example, if $M = 1$ cites documents $[1, 2]$ and $M = 2$ selects $[2, 3]$, the *Jaccard Similarity* is $1/3$, despite citing the same number of documents. It indicates whether the model is shifting its document set or only making minor adjustments. *Bit Flip Rate* assesses stability at the level of individual binary decisions. It computes the proportion of citation decisions that change status (cited $\leftrightarrow$ uncited) between iterations:

$$\text{Bit Flip Rate} = \frac{\#\text{flipped decisions}}{N \times K}$$

where N is the number of samples and K is the number of candidate documents (i.e., citation decision slots) per sample, and $\#$flipped decisions counts the total number of document-level citation indicators that differ between two iterations (i.e., the number of positions where $c_{i,j}^{(1)} \neq c_{i,j}^{(2)}$).

Consensus on citations between $M = 1$ and $M = 2$. The *Jaccard Similarity* reaches **0.8817**, indicating substantial overlap between the cited document sets. This suggests that the model identifies the core document set required to entail the summary in the first iteration ($M = 1$).

Minimal decision volatility between $M = 1$ and $M = 2$. The *Bit Flip Rate* is low, with a global flip rate of only 8.33% (i.e., fewer than 0.5 citation changes per sample on average). This indicates that $M = 1$ already yields a stable document-selection policy.

Overall, when the citation set remains largely unchanged, additional iterations provide limited practical benefit. Therefore, for ASQA, we set $M = 1$.

E Case Study

To better visualize the RAAC pipeline, we present a detailed walkthrough using the question: 'When did the last season of Jersey Shore air?'. As shown in Figure 11, in the first round of collaboration between the TextAgent and the CitationAgent ($M = 1$), the LLM within the TextAgent performs k rounds of role-based interaction, after which the CitationAgent refines the citations. SC then evaluates the output of this round and reports a summary citationF1 score of 0 for the current question, which does not exceed the threshold τ. Therefore, we select the documents indicated by citations [3] and [5] in the TextAgent-generated summary, and optimize the documents for the next round by constructing it from the corresponding contents of [3] and [5]. After the second round of TextAgent-CitationAgent collaboration ($M = 2$), the citation f1 of the generated result doubles compared to the previous round.

F Algorithm for our method

F.1 The algorithm of TextAgent

The detailed process of TextAgent is presented in Algorithm 1.

VANILLA Prompt

Instruction: Write an accurate, engaging, and concise summary for the given question using only the provided search results and cite them properly using [1][2][3].
Question: {Question}
Document: {Document}
summary:

Table 9: Prompt for VANILLA.

SUMM Prompt

Step1:
Summarize the following document within 50 words with the question of interest {Question}
Return `"irrelevant"` if the document is "irrelevant" to the question. Try to keep all the important dates, numbers, and names.
Title: {Title}
Text: {Text}
Notes: Your output should be a concise summary of the "Text" only.
Do not include any explanations, introductions, or formatting.
Avoid redundancy. Just return the summary content.
Summary:
Step2:
Instruction: Write a high-quality summary for the given question using only the provided search results and cite them properly using [1][2][3].
Question: {Question}
Document: {Document}
summary:

Table 10: Prompt for SUMM.

F.2 The algorithm of SC

The detailed process of SC is presented in Algorithm 2.

SNIPPET Prompt

Step1:
Instruction: Given the following passage and the question {Question}, extract a useful span from the passage that can summary the question.
Resolve all the coreference issues to make the extracted span understandable and standalone. If the passage is not helpful for summarying the question, return `"irrelevant"`. If there are multiple spans, merge them and only output one paragraph.
Title: {Title}
Text: {Text}
Extracted span:
Step2:
Instruction: Write a high-quality summary for the given question using only the provided search results and cite them properly using [1][2][3].
Question: {Question}
Document: {Document}
Notes: All sentences in the summary must be separated by a single space (do not use line breaks). Do not include statements such as "in summary" or "analysis shows". Directly summary the question using only the provided information.
summary:

Table 11: Prompt for SNIPPET.

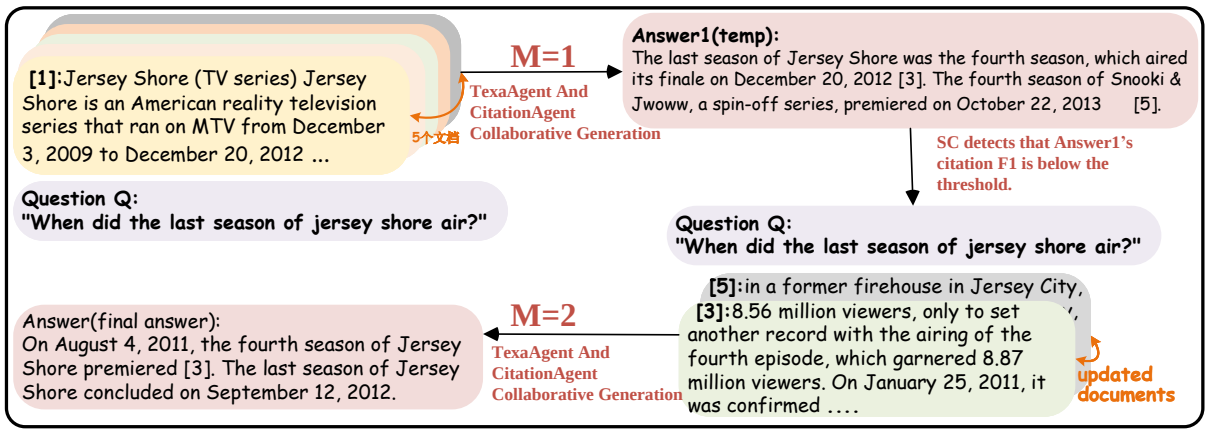


Figure 11: A case study of RAAC on the question: 'When did the last season of Jersey Shoreair?'

Algorithm 1: `TextAgent`, Click here return to Section 2.2

Input: Question Q, Documents $\mathcal{D}$, LLM $\mathbb{L}$, Prompts $\{\mathsf{P}_{sg}, \mathsf{P}_{se}, \mathsf{P}_{sr}\}$, stop condition
Output: Final Summary

$Summary_1 \leftarrow \text{LLM}(Q \parallel \mathcal{D} \parallel \mathsf{P}_{sg})$ `// Initial generation`
$fb_{\text{output}_1} \leftarrow \text{LLM}(\mathsf{P}_{se} \parallel Q \parallel \mathcal{D} \parallel Summary_1)$ `// Feedback stage`
Store data in the DB pool: $Summary_1 + fb_{\text{output}_1} + \mathcal{D} + Q$
if *stop*(fb_{output_1}[*Total Score*] $== 21$) **then**
  return $Summary_1$
end
else
  for *iteration* $i \in 1, 2, \ldots$ **do**
    $Summary_{i+1} \leftarrow \text{LLM}(Summary_i \parallel Q \parallel \mathcal{D} \parallel fb_{\text{output}_i} \parallel \mathsf{P}_{sr})$ `// Refine stage`
    $fb_{\text{output}_{i+1}} \leftarrow \text{LLM}(\mathsf{P}_{se} \parallel Q \parallel \mathcal{D} \parallel Summary_{i+1})$ `// Feedback Stage`
    Store data in the DB pool: $Summary_{i+1} + fb_{\text{output}_{i+1}} + \mathcal{D} + Q$
    if *stop*($fb_{output_{i+1}}$[*Total Score*] $== 21$ *or* $i + 1 == k$) **then**
      break
    end
  end
end
return $Summary$ with Max$\{fb_{\text{output}_i}$[Total Score]$\}$ from DB, $i = 1 \ldots k$

TextAgent *Initial Generate* Prompt

Instructions for summary Generation:
1. Your summary must be accurate and engaging.
2. The summary must correctly cite the reference materials: Indicate the source of the literature in the form of `[id]`.
If a sentence cites multiple reference materials, the format should be `[id1][id2]`.
If multiple documents support the same content, do not cite more than three. 3.Each sentence in your summary must cite at least one reference, but no more than three.
4.You must not include any content that is not found in the documents.
5. Maintain an objective and neutral tone, similar to a news report or encyclopedia.
6.To enhance citation accuracy, prefer using original wording or near-verbatim phrasing from the referenced sources.
Question: {Question}
Reference Format:
Each reference is given in the format: `[id]: Title<|>Text`
`"id"` is the identifier of the reference.
`"Title"` is the title of the reference.
`"Text"` is the body content of the reference.
Documents: {Documents}
Notes:
1.When generating content, use the original information from the documents strictly without any personal elaboration or summarization.
2.Each sentence in your summary must cite at least one reference, but no more than three.
3.Each citation must be in the format `[id]` or `[id1][id2]`, placed at the end of the sentence before the period. Example: "He scored 805 goals in official matches`[5]`."

Table 12: Prompt for TextAgent *Initial Generate* stage, Click here to return to Section 2.2.

TextAgent *Feedback stage* Prompt

You are an evaluation expert. Evaluate the summary according to the following criteria. For each criterion, give a score from 1 to 3 points, and provide suggestions for improving the summary.
Evaluation Criteria:
Completeness: The generated summarys can fully address the input questions.
Objectivity: The summary should be composed of original sentences from the documents or sentences with the same semantic meaning, ensuring strict adherence to the source content without paraphrasing or adding new interpretations.
Specificity: Avoid general expressions such as "This is important", "Many studies have shown", and "According to xxx", etc.
Length The summary should have no fewer than 50 words.
Citation Recall: Each sentence should have 1–3 citations, and there should be factual consistency between the sentence and its citation.
Citation Precision: When there are multiple cited sentences, check whether each citation is valid.
Output Format: Only return a JSON object that contains the score and explanation for each criterion, as well as a total score (the sum of all individual scores). Return the result **strictly** in the format of the example below. Each evaluation indicator must consist of the following structure:
{Example:}
{"Completeness":
{ "Score": xxxxx, "Suggestions": xxxx }, ...
, "Total Score": XX }
Question: {Question}
Reference Format: Each reference is given in the format: `[id]: Title<|>Text`
`"id"` is the identifier of the reference.
`"Title"` is the title of the reference.
`"Text"` is the body content of the reference.
Documents: {Documents}
summary: {summary}

Table 13: Prompt for TextAgent *Feedback stage*, Click here to return to Section 2.2.

TextAgent *Refine stage* Prompt

You are an AI assistant. Your goal is to revise responses that improve over time based on points and Modification Suggestions. Each interaction may include:
A question; Reference materials for grounding; A summary; Feedback scores and suggestions for improvement
Given this information, you must revise the summary accordingly. *Focus on the areas where the feedback indicates weakness*. Maintain factual consistency with the documents provided. Your output should be a high-quality, updated version of the summary that better aligns with the feedback, while preserving strengths from the original.
Question: {Question}
Reference Format:
Each reference is given in the format: `[id]: Title<|>Text`
`"id"` is the identifier of the reference.
`"Title"` is the title of the reference.
`"Text"` is the body content of the reference.
Documents: {document}
Summary: {summary}
textbfscores and suggestions:
completeness: score: {score}, suggestion: {suggestion}
.....
Output Format: Return a JSON format, outputting the summary that has been revised according to the feedback results.
Notes: 1. When adding documents, follow this rule: Each reference should be in the form of `[id]`. If citing multiple documents, use `[id1][id2][id3]`.
2. Please return the result strictly in the format of the output format example.
3.When generating content, use the original information from the documents strictly without any personal elaboration or summarization.
4.Each sentence in your summary must cite at least one reference, but no more than three.
5.Each citation must be in the format `[id]` or `[id1][id2]`, placed at the end of the sentence before the period.
Example: "He scored 805 goals in official matches [5]."
Output Format Example: {"revised": xxx}

Table 14: Prompt for `TextAgent` *Refine stage*, Click here to return to Section 2.2.

LLM AddCitation Prompt

Instructions: You will be provided with a sentence and several related documents.
Your task is to directly append citation annotations to the sentence using these documents without changing the sentence.
When citing documents, use `[1][2][3]`.
Document: {Document}
Sentence: {Sentence}
Notes:
1.Do not explain your reasoning. Only output the sentence with citation, or the original sentence if no support is found.
2.Do not cite all supporting documents. Only include the smallest sufficient subset (1–3) needed to entail the sentence.
Output: Sentence with citation:

Table 15: Prompt for LLM AddCitation, Click here return to Section 2.3.

Prompt of LLM as the evaluator

Role: Data Annotator
Instructions: You are provided with the following materials:
- Passage: passage
- Sentence: sentence
Task: Assess whether the passage fully supports the sentence.
Choices: 1. Fully Supports: Select this option if the passage completely and clearly supports every aspect of the sentence.
2. Does Not Fully Support: Select this option if any discrepancies, omissions, or inaccuracies in the passage prevent it from fully supporting the sentence.
Output:
- If the passage fully supports the sentence, output "Yes."
- If it does not, output "No."
Note: Please refrain from adding any content not requested in the instructions.

Table 16: Prompt for LLM as the evaluator to evaluate citation metrics.

Prompt of LLM evaluate Sumary Content

You will be given one Generated Summary and one Gold Summary.
Your task is to **rate the Generated Summary based on its sub-point coverage of the Gold Summary**.
Please read and understand the following instructions carefully:
–Input:
Generated Summary: summary
Gold Summary: answer
–Evaluation Metric:
Sub-point Coverage (0-5)
–Evaluation Criteria:
This metric measures **how many of the key sub-points from the Gold Summary** are also present in the Generated Summary.
5 (Full Coverage): The Generated Summary contains all key sub-points from the Gold Summary.
4.0-4.5 (High Coverage): The Generated Summary contains the vast majority of the key sub-points.
3.0-3.5 (Partial Coverage): The Generated Summary contains about half of the key sub-points.
2.0-2.5 (Low Coverage): The Generated Summary contains only a few of the key sub-points.
1.0-1.5 (Very Low Coverage): The Generated Summary contains almost no sub-points from the Gold Summary.
0 (No Coverage): The Generated Summary contains none of the sub-points from the Gold Summary.
– Evaluation Steps:
1. Read the Gold Summary carefully and internally break it down into a list of its core facts and sub-points.
2. Read the Generated Summary carefully.
3. Compare the Generated Summary against the list of sub-points from the Gold Summary.
4. Assess the percentage of sub-points from the Gold Summary that are covered by the Generated Summary.
5. Assign a score from 0 to 5 based on the coverage percentage, following the Evaluation Criteria.
–Output Format: You must output only a single numerical score (e.g., 4 or 4.5). Do not output your analysis, reasoning, or any other text.

Table 17: Prompt for LLM as the evaluator to evaluate summary content.

Method	Correct.EM	Rouge-Lsum	Citation Rec.	Citation Pre.	Citation F1	$FC1$	M=1
FRONT	40.84	36.95	77.70	69.89	73.59	52.53	-
$\tau = 0.1$	41.14	37.17	74.23	73.43	73.83	52.88	100
$\tau = 0.2$	41.38	37.39	73.39	72.59	72.97	52.81	101
$\tau = 0.3$	41.00	37.30	74.64	73.92	74.28	52.84	110
$\tau = 0.4$	40.87	37.21	75.34	74.78	75.06	52.92	138
$\tau = 0.5$	41.00	37.11	77.02	76.47	76.74	53.45	229
$\tau = 0.6$	40.91	37.25	77.48	76.86	77.17	53.47	259
$\tau = 0.7$	40.86	37.00	77.49	77.00	77.24	53.45	297
$\tau = 0.8$	41.23	37.22	79.69	79.08	79.38	54.27	383
$\tau = 0.9$	40.89	36.99	80.22	79.55	79.88	53.94	425

Table 18: Comparison of our method against FRONT across various metrics (Correctness, Rouge-Lsum, Citation Recall, Citation Precision, Citation F1, $FC1$) under different values of τ. "M=1" denotes the number of samples entering the SC stage in the ASQA dataset.

Algorithm 2: `Summary Calibration` for one sample, Click here return to Section 2.4

Require: Input from `TextAgent`, summary

$m \leftarrow 1$
while $m \leq M$ **do**
  $CitationF1 \leftarrow$ NLI$(summary, \mathcal{D})$ `// Compute F1`
  if $CitationF1 \geq \tau$ **then**
    return $summary$
  end
  else
    $\mathcal{D}$=$\mathcal{D}_{adp} \leftarrow$ regex$(summary)$ `// use regex match citation`
    $summary_{new} \leftarrow$ [`TextAgent` + `CitationAgent`]$(Q, \mathcal{D})$
    $CitationF1_{new} \leftarrow$ NLI$(summary_{new}, \mathcal{D})$
    if $CitationF1_{new} > CitationF1$ **then**
      `// control optimization direction`
      $summary_{better} \leftarrow summary_{new}$
    end
    else
      $summary_{better} \leftarrow summary$
    end
    $summary \leftarrow summary_{better}$
  end
  $m \leftarrow m + 1$
end