Evaluating the Reliability of LLMs in Faithfully Updating Text: An Empirical Study

Ayan Datta[1]* **Paheli Bhattacharya**[2] **Rishabh Gupta**[2]
[1]International Institute of Information Technology, Hyderabad, India
[2]Bosch Research and Technology Centre, India
ayan.datta@research.iiit.ac.in
{paheli.bhattacharya, gupta.rishabh}@in.bosch.com

Abstract

We provide a comprehensive review of the FRUIT (Faithfully Reflecting Updated Information in Text) task, which formalizes the challenge of accurately updating textual information with large language models (LLMs). Our work begins with an in-depth analysis of the FRUIT dataset, revealing key structural insights. We also investigate the unsupervised capabilities of LLMs—such as zero-shot learning, chain-of-thought reasoning, self-reflection, and evidence ordering. Experimental results demonstrate that unsupervised approaches perform competitively with supervised methods in faithful text updating. Qualitative analysis shows that updates utilizing table-structured evidence outperform those based on unstructured text. We also discuss important limitations, including the need for new datasets and the risks of information leakage in this domain. These findings have significant implications for applications requiring precise document updates, such as software engineering, technical documentation, and legal document maintenance.

1 Introduction

The task of Faithfully Reflecting Updated Information in Text (**FRUIT**) involves updating existing textual content to integrate new information accurately and consistently with the available evidence. FRUIT was formally introduced to automate the process of updating documents with recent information (Iv et al., 2022). Given a source document A^t at time t and a set of evidence $\mathcal{E}^{t \to t'} = \{E_1, E_2, ...E_{|\mathcal{E}|}\}$ representing updates from time t to t', the goal is to generate an updated document $A^{t'}$. This task involves not only incorporating the new facts but also preserving the coherence and integrity of the original content. Many domains require frequent and iterative document updates to accommodate evolving data, regulations, or operational requirements. For instance, in software engineering, continuous updates to requirement documents are crucial throughout the development lifecycle (Bhatia et al., 2020; Napoleão et al., 2022). Similarly, in contract drafting, regular revisions are essential to produce clear documentation that mitigates disputes and ensures project success (Rameezdeen and Rodrigo, 2014). Technical and procedural documentation for tool maintenance and business processes (M et al., 2016; Klimant and Kollatsch, 2022; Singh and Gupta, 2024) also demand frequent updates, as do organizational policies that must be revised to comply with new legal and industry standards (Bandler, 2024). These tasks are challenging because they require understanding both the original content and the new information, making manual editing time-consuming and complex. This underscores the necessity for automated systems to enhance efficiency while maintaining accuracy.

Recent advances in large language models (LLMs) and in-context learning paradigms have significantly impacted text generation and editing tasks, including simplification, paraphrasing, and grammatical error correction—capabilities that are highly relevant to the FRUIT task (Dwivedi-Yu et al., 2024; Shu et al., 2024). Despite these developments, several gaps remain in the literature that limit our understanding of LLM performance in complex text updating scenarios:

• **Limited exploration of unsupervised capabilities**: While fine-tuning approaches have been studied, there is insufficient research on how LLMs perform under unsupervised settings, such as zero-shot prompting, chain-of-thought (CoT) reasoning, and other prompt-based strategies.

• **Understudied impact of data structure formats**: The role of structured versus unstructured evidence in influencing LLM performance for FRUIT remains largely unexplored, leaving a critical knowledge gap.

*Work done during the internship at Bosch Research and Technology Centre, India

Proceedings of the Fifth Workshop on Generation, Evaluation and Metrics (GEM), pages 271–284

• **Shallow dataset-level analysis**: Existing work provides limited insight into the FRUIT dataset's composition, including its mix of text and tabular evidence and the effect of varying evidence counts on update quality.

To address these gaps, this work makes the following contributions:

• **Comprehensive dataset analysis**: We examine the FRUIT dataset in depth, uncovering structural and content-related characteristics. For example, updated articles are typically about 100 tokens longer than their originals, suggesting a potential bias toward appending information rather than performing efficient in-place edits.

• **Systematic evaluation of unsupervised LLM behavior**: We conduct extensive experiments on the FRUIT task using diverse methodologies—zero-shot prompting, CoT reasoning, self-reflection, and evidence ordering—across two state-of-the-art models: Llama-3-8B (open-source) and GPT-4 (closed-source). To the best of our knowledge, this is the first work to systematically benchmark these unsupervised strategies for text updating.

• **Analysis of data format effects**: We present a detailed quantitative and qualitative study of how different evidence formats influence model performance. Notably, models perform better when evidence is presented in tabular form compared to unstructured text, despite the reduced contextual richness of tables. These findings offer forward-looking insights into designing effective evidence representations for future systems.

By synthesizing these observations, this work aims to provide a taxonomy of approaches, highlight emerging trends in prompt-based and fine-tuning strategies, and identify open challenges for real-world text updating scenarios where labeled data is scarce. We make the implementations publicly available at https://anonymous.4open.science/r/faith-update-llm-1381/

2 Dataset Analysis

In this section, we present the detailed statistics of the FRUIT dataset, which is sourced from the official website[1] and introduced in (Iv et al., 2022). The dataset comprises source documents A^t written at time t and their corresponding updated target documents $A^{t'}$ created at a later time t'. These target documents incorporate new information from the period t to t', based on a set of evidences $\mathcal{E}^{t \to t'} = \{E_1, E_2, ...E_{|\mathcal{E}|}\}$. The dataset consists of the train, dev and gold set comprising of articles from Wikipedia from 2019 to 2021. The gold set is a manually annotated subset of the development (dev) set. We exclude the samples used for the gold set from the dev set. We use the gold set to evaluate our methods for fair comparison, and refer readers to (Iv et al., 2022) for details on dataset construction.

Table 1: Statistics of the FRUIT dataset

Feature	Train	Dev	Gold
# samples	92,388	54,729	914
Original Article length	668	690	742
Updated Article length	808	826	902
# Evidences/article	6.2	5.6	8
Ev: Text only	31%	32%	15%
Ev: Table only	10%	12%	7%
Ev: Table+Text	59%	56%	78%
Tabular Ev. # Rows	2	2	2
Tabular Ev. # Columns	5.7	5.5	6
# Entities in Original	9.3	9.8	10.5
# Entities in Updated	11.6	12	13.7

Dataset Statistics: Table 1 summarizes the FRUIT dataset statistics. Updated articles are, on average, 100 tokens longer than originals, potentially biasing models toward appending rather than editing in-place. Evidences appear in two formats: plain text and tabular. In the training set, 10% of article pairs use only tabular evidence, 31% only plain text, and the rest both. For the test set, 12% are tabular-only, 32% text-only; for the gold set, 7% are tabular-only, 15% text-only. Each table has at most 2 rows (including header) and about 5 columns. Articles contain an average of 6 evidences in training/test sets and 8 in the gold set. Of these, 2 (training/test) and 3 (gold) are tabular. Updated articles mention 2 more entities on average. Each edited sentence is typically influenced by 1 evidence, supporting streaming-based update settings.

3 Approaches

In this section, we describe the LLM-based approaches investigated for the text updating task on the FRUIT dataset. For this survey, we focus on widely used prompt-based methods and LoRA-style fine-tuning, and exclude instruction-based editing techniques. These methods are categorized into classes such as Base Prompting, Reflect and

[1] https://github.com/google-research/language/tree/master/language/fruit

Refine, Evidence Ordering, Few-Shot Prompting, and LoRa and QLoRa Fine-tuning. This hierarchical organization reflects the progression from minimal intervention (Base Prompting) to structured reasoning (Reflect and Refine), evidence-aware strategies (Evidence Ordering), and data-driven adaptation (Few-Shot and Fine-Tuning). By analyzing these categories, we aim to highlight their relative strengths, limitations, and applicability to real-world text updating scenarios.

3.1 Base Prompting

• **Zero-shot**: In this approach, we supply the model with carefully crafted basic prompts that include the source information along with updates.

• **Chain-of-Thought**: Inspired by the reasoning abilities of LLMs demonstrated through multi-step reasoning processes (Wei et al., 2023), we developed a multi-step prompting approach that leverages these capabilities. Initially, the model is prompted to identify discrepancies between the evidence and the original article. Subsequently, it is asked to update the article to address and correct the identified discrepancies.

3.2 Reflect and Refine

This approach follows a two-step process inspired by (Madaan et al., 2023) and the LLM-as-a-Judge paradigm (Gu et al., 2024). First, the model generates an updated version of the article. Subsequently, this updated article is *reflected* upon through evaluation across various aspects. Based on this assessment, the model *refines* its original output to produce a final, improved version of the article. In this approach, we experiment with different settings, as follows:

• **Self Reflection**: In this we prompt the same LLM to assess its previous answer, identify any mistakes, and then regenerate the output accordingly.

• **External Evaluators**: We use open-source evaluator LLMs, TIGERScore (Jiang et al., 2024) and Prometheus (Kim et al., 2024), to address the self-preference bias of base LLMs, which tend to favor their own outputs (Laskar et al., 2024; Panickssery et al., 2024). These evaluators are tasked to identify errors and evaluate the editing task and then generate a JSON report with errors and a score, which is then fed back to the base LLM to produce the final updated article.

3.3 Evidence Ordering

In this class of approaches, we focus on filtering and ranking the evidences under the assumption that when the LLMs are provided with the most relevant evidence before less relevant ones can enhance their performance. Such approaches are particularly important when there are limitations on the input context length of LLM models. For relevancy, we hypothesize that evidences that are more inconsistent with the source play a greater role in updating, as they can introduce new information not already contained within the source.

• **Default Ordering**: We evaluate the default order of evidences as extracted from the Wikipedia dataset. Although this ordering is not explicitly optimized for our task, it still carries useful signal because evidences are collected from sequentially ordered Wikipedia sections. As a result, neighboring evidences may preserve topical and temporal continuity, which can help the model prioritize context that is naturally related to the current update.

• **Filter + Rank: Similarity**: Inspired by (Goldstein and Carbonell, 1998), where redundancy reduction is performed through similarity-based reranking (MMR), we adopt a similar intuition to filter evidences based on their closeness to the source. We compute the cosine similarity between the embeddings of the source and evidence texts to assess their level of similarity. We first filter out highly similar evidences that offer little new information about the source, as well as highly dissimilar evidences that are unrelated to the source article. The remaining evidences are then ranked and provided to the LLM alongside the source article.

• **Filter + Rank: Hallucination**: This approach is inspired by (Sui et al., 2024), which shows that LLM confabulations often exhibit narrativity and semantic richness and can therefore serve as a useful source of novel information. The higher the hallucination score of an evidence relative to the source, the more significant it is. We use the fact that a high hallucination score indicates the evidence offers novel information not already contained in the source. We filter out the factually consistent evidences based on a threshold and then ranked them based on the hallucination score and provided it to the LLM along with the source article. We utilize a hallucinations detection model to evaluate the evidences.

• **Rank: Random**: We randomly shuffle and rank the evidence to establish a more effective baseline.

• **Stream Evidences**: We evaluate an alternate setup for the problem where we process the evidences in batches of three, updating the article after each batch. The updated article then serves as the basis for the next batch. We follow the default order of the evidences as provided in the dataset, which is essential for realistically modeling real-world scenarios, where updates are revealed gradually over time rather than all at once.

We use batches of three as a middle ground between providing enough context to support coherent updates and keeping each step small enough to reduce cognitive load and error accumulation across iterations. This choice also aligns with the dataset's typical evidence-to-edit granularity, while keeping the number of update steps manageable. We keep the default evidence ordering to preserve temporal and contextual sequencing from Wikipedia edits and to avoid injecting future information earlier than it appears in the data. Alternatives include k=1 (finer-grained updates but more steps and compounding errors), larger k (fewer steps but less benefit from incremental revision), or learned/relevance-based orderings (potentially higher relevance but risk of disrupting temporal coherence and increasing computation).

3.4 Few-Shot Prompting

Inspired by the in-context learning capabilities of LLMs (Brown et al., 2020), this class of approaches investigates few-shot prompting methods that involve selecting representative examples from the dataset and incorporating them into the prompts. Here we have two approaches based on the method used to select the examples:

• **Fixed**: We provide a fixed example from the training set for each source document along with its corresponding evidence which imitates the task.

• **Similar**: Since LLMs are sensitive to the examples provided during in-context learning (Liu et al., 2022; An et al., 2023), we select examples that closely match the input source document. This matching is based on length similarity between the given document and the examples, using the differences in the number of evidences as a tie breaker

3.5 LoRa and QLoRa Finetuning

We also explore low-rank adaptation (LoRA) techniques, specifically LoRA and QLoRA (Hu et al., 2021; Dettmers et al., 2023), which provide parameter-efficient strategies for fine-tuning LLMs. These methods reduce memory and computational requirements by constraining updates to low-rank subspaces, allowing us to train high-capacity models even under limited resource budgets. For the FRUIT text-updating task, we leverage these techniques to adapt the model so that it can generate revised articles conditioned jointly on the source article and the new evidence set. Using a sampled subset of the training corpus, the model is fine-tuned to integrate the new information faithfully while preserving the original structure and linguistic style where appropriate.

4 Experimental Setup

In this section, we explore various models and novel approaches applied to the FRUIT task dataset to evaluate their effectiveness in accurately updating text. Our assessment encompasses both quantitative and qualitative metrics to measure the accuracy, coherence, and faithfulness of the generated updated text.

4.1 Baseline Method

As baseline, we consider EdiT5 introduced in (Iv et al., 2022) which is a variant of T5 that enhances text updating tasks by using a compressed output format that reduces the need to generate the entire text from scratch. It features Diff-Formatted Output, which replaces copied sentences with a copy token, and Reference and Control Tokens that guide the model on whether to add, edit, or delete content, promoting effective content planning.

4.2 Evaluation Metrics

To systematically analyze models' performance for the FRUIT task, we use the following automatic evaluation metrics, which provide quantitative measures to assess various aspects of the models' effectiveness and accuracies:

• **Generation Metrics**: We utilize UpdateROUGE, a variant of ROUGE introduced in (Iv et al., 2022), to assess the quality of the updated text against a reference by applying ROUGE only to the updated sentences which ensures the score is not inflated with non-edited content. Notably, UpdateROUGE does not account for synonym variations; therefore, we complement our evaluation with METEOR (Banerjee and Lavie, 2005) and BERTScore (Zhang et al., 2020) metrics.

• **Faithfulness Metrics**: We evaluate faithfulness by comparing named entities extracted from the

source, evidence, and updated text (Iv et al., 2022). This comparison employs *Precision*, *Recall*, and *F1 Score* to measure the overlap of entities between the source and the updated article.

We also employed the metric *Unsupported Entities (Unsupp)*, which calculates the average number of entity tokens in generated updates that are absent from the source article or evidence. Higher scores on this metric suggest reduced faithfulness. Since prior work does not provide an implementation of this metric, we utilize a named entity recognizer (Devlin et al., 2018; Tjong Kim Sang and De Meulder, 2003), which detects locations, organizations, persons, and miscellaneous entities.

4.3 Large Language Models

For the FRUIT text update task, we conducted experiments using two state-of-the-art LLMs: the open-source model Llama-3-8b (Llama-3, 2024) and the proprietary GPT-4o (OpenAI et al., 2024). All experiments were performed with greedy decoding and temperature set to 0 to ensure reproducibility.

For the external evaluator models discussed in Section 3.2, we utilize TigerScore (Jiang et al., 2024) and Prometheus (Kim et al., 2024). We utilize the vectara hallucination model (Li et al., 2024) for detecting hallucinations and ranking evidence as described in Section 3.3. LoRa and QLoRa fine-tuning was performed on the Llama-3-8B model using Axolotl (Axolotl maintainers and contributors, 2023) on an NVIDIA A100 GPU. Training was halted when the validation loss on a sample of the dev set ceased to decrease.

5 Results and Discussion

In this section, we present the experimental results for the approaches described above, utilizing two LLMs: Llama-3-8b and GPT-4o, as summarized in Table 2. Additionally, we include the top-performing approach (EdiT5-3B) from the baseline paper (Iv et al., 2022) to facilitate a comparative analysis of our findings.

5.1 Quantitative Analysis

• **Generation vs. Faithfulness**: We observe that the baseline supervised model (EdiT5-3B) achieves comparable performance in Generation (measured by UpdateROUGE metric) to the unsupervised LLM-based approaches utilizing Llama-3-8b and GPT-4o models (Table 2). However, these methods demonstrate a significant improvement in Faithfulness (measured by the entities). This indicates that the LLM-based approaches have a greater ability to faithfully update the text, while their text generation capabilities remain comparable to the baseline method.

• **Zero-shot vs. Few-shot vs. Fine-tuning**: Our experiments show that few-shot prompting slightly improves generation performance (measured by UpdateROUGE) for both Llama-3-8b and GPT-4o compared to zero-shot prompting. However, both models still trail behind the EdiT5-3B baseline (Table 2). Notably, in terms of faithfulness (measured by Entities), the zero-shot approach outperforms the baseline for both models. Additionally, transitioning to few-shot prompting yields significant faithfulness improvements for both Llama-3-8b and GPT-4o. This emphasizes the importance of including examples while faithfully updating the text documents.

Fine-tuning the model Llama-3-8b using QLoRA techniques, significantly enhances its generation capabilities as measured by BERTScore while making it top-performing model when compared to GPT-4o, which has a much larger parameter count. Recent studies on faithful updating (Zhang et al., 2024; Shu et al., 2024) demonstrate that fine-tuning combined with advanced RL capabilities plays a significant role in improving the faithfulness of text updates.

• **Reflect & Refine and Evidence Ordering**: The effectiveness of the Reflect and Refine techniques varies: TigerScore-based refinement outperforms for Llama-3 model, while self-reflection yields better results for GPT-4. This indicates that GPT-4's self-reflection capabilities make it a more effective critique than Llama-3 for the faithful updation task.

Evidence Ordering does not surpass zero-shot prompting, as random shuffling produces comparable results. The default ordering performs best, likely because of the inherent temporal or semantic structure in the Wikipedia-based dataset.

Streaming Evidences decreases performance, possibly due to incomplete context during incremental updates. Although Reflect & Refine and Evidence Ordering show no improvements on the FRUIT dataset, their potential benefits should be explored further in domain-specific contexts.

Table 2: The evaluation results of the different approaches over two LLMs – Llama-3-8b and GPT-4o. Best results are in bold. The first row reports the best performing approach from the baseline work (Iv et al., 2022). F+R is abbreviated for Filter+Rank, R for Rank.

Model	Approach Type	Approach	UpdateROUGE			BERTScore			Meteor	Entities			
			R1	R2	RL	Pr	R	F1		Pr	R	F1	Unsupp
Baseline method: EdiT5-3B (Iv et al., 2022)			**47.4**	34	**41.1**	–	–	–	–	69.9	52.2	59.77	1.58
Llama-3-8b	Base Prompting	Zero Shot	47.02	30.29	36.84	80.21	83.89	81.66	63.55	64.4	76.8	70.05	2.52
		CoT	44.96	28.36	34.78	78.72	83.79	80.81	63.04	62.2	76.2	68.49	2.48
	Reflect & Refine	Self Reflection	45.03	28.42	34.62	77.52	83.93	80.24	62.7	60.1	77.4	67.66	3.07
		TigerScore	46.8	29.9	36.49	80.13	83.62	81.48	63.03	64.2	76	69.6	2.47
		PrometheusScore	44.6	27.82	33.91	75.82	83.4	79.12	61.53	58.5	77.2	66.56	3.13
	Evidence Ordering	F+R: Similarity	45.47	28.77	35.27	79.04	83.97	81.07	63.56	62.5	77.2	69.07	2.62
		F+R: Hallucination	45.58	28.75	35.36	79.5	83.81	81.19	63.14	62.8	76.9	69.13	2.62
		R: Random	45.82	29.21	35.59	79.71	83.72	81.27	63.06	63.2	76.8	69.33	2.59
		Stream Evidences	41.85	25.5	31.68	73.77	82.3	77.4	58.92	74.6	53.4	62.24	3.67
	Few Shot	Fixed	45.17	31.78	38.59	87.4	79.91	83	59.42	**79.6**	64.9	71.5	0.75
		Similar	43.61	30.22	36.88	87.71	78.4	82.23	56.71	78.4	67.4	72.48	0.9
	Finetuning	LoRa	36.57	26.2	31.79	89.01	83.73	85.96	66.62	80	71.1	75.28	1.01
		QLoRa	44.8	32.9	39.5	**90.8**	**84.87**	**87.76**	57.32	79.5	70.1	74.5	1.12
GPT-4o	Base Prompting	Zero Shot	46.88	32.45	39.2	86.84	85.21	85.73	**69.24**	76.7	77.2	76.95	1.32
		CoT	43.22	25.86	31.92	78.12	85.32	81.29	65.49	59.8	79.7	68.33	3.04
	Reflect & Refine	Self Reflection	45.62	28.13	34.5	78.78	85.37	81.71	66.15	61.4	**80.05**	69.5	3.02
		TigerScore	44.07	26.7	30.8	76.71	83.32	79.87	64.58	60.4	79.1	68.5	3.06
		PrometheusScore	42.4	24.25	29.74	74.67	82.98	78.6	63.21	58.82	77.55	66.9	3.1
	Evidence Ordering	F+R: Similarity	45.12	31.04	37.63	84.65	83.37	83.72	67.52	76.4	75	75.69	1.37
		F+R: Hallucination	44.67	32.15	37.24	83.87	84.03	83.94	65.78	75.92	74.78	75.35	1.38
		R:Random	45.2	30.96	37.58	84.82	83.46	83.86	67.46	76.2	75.18	75.69	1.38
		Stream Evidences	46.16	29.18	36.7	84.04	82.85	83.44	66.54	64	77.3	70.02	2.57
	Few Shot	Fixed	46.04	33.81	39.4	80.24	84.61	82.09	65.78	77.45	78.36	**77.9**	**0.34**
		Similar	45.97	**34.12**	38.75	78.75	82.85	80.74	63.81	76.57	77.92	77.24	0.52

5.2 Qualitative Analysis

In this section, we present our qualitative findings for the zero-shot approach utilizing both Llama-3-8b and GPT-4o models. Our goal is to analyze how different counts and data structures influence the performance of the faithful updating task. Additionally, we offer specific observations related to copy cases and the practice of simply appending updates to the source during the faithful text updating process.

• **Number of edits and number of evidences**: We evaluate cases that require significant edits by calculating the translation edit rate (Snover et al., 2006) between the original source and the gold reference. This captures the extent of rewriting needed and serves as a proxy for update difficulty. As illustrated in Figure 1, both LLMs perform poorly on these cases, with their performance decreasing as the number of edits increases. Figure 2 shows that the performance generally declines as the number of evidences increases. For Llama-3-8b, the optimal performance is achieved with approximately 1 or 2 evidences. In contrast, GPT-4o demonstrates a better ability to handle a larger number of evidences. Here, we define *required edits* as the minimum sequence of insertions, deletions, and substitutions needed to transform the source article into the gold update; we measure this using Translation Edit Rate (TER) because it provides a length-normalized, operation-level proxy for update effort that is standard in text-to-text editing and robust to variable article lengths. TER complements evidence count: evidence count captures input breadth, while TER captures the magnitude of output change; they are correlated but not identical (e.g., many evidences can be redundant, and a few evidences can trigger large edits). We include both figures to show how performance degrades along distinct difficulty axes, which clarifies which aspects of FRUIT (input volume vs. edit intensity) most stress current models and motivates evidence ordering and streaming choices.

• **Copy Cases and Source Appends**: For the zero-shot method, the model copies the original article unchanged in approx. 2.6% of cases, disregarding the update instructions. Only one of these instances was justified by the reference updated article. This rate drops to about 0.5% when using chain-of-thought prompting. The update task involves not just adding evidence to the source but making targeted edits to integrate new information. We assess cases where the model appends information at the end rather than editing the source directly – contrary to how the reference update is performed. In the zero-shot approach, we observe this behavior in 15.6% of Llama-3-8b and 18.2% of GPT-4o cases.

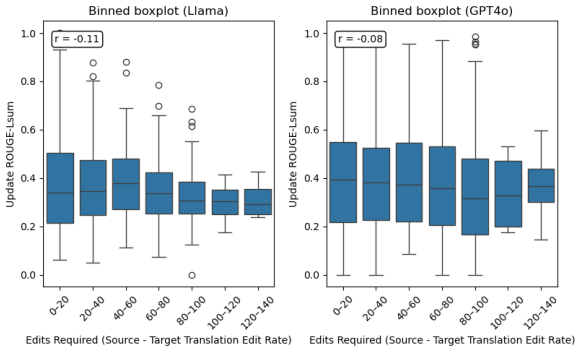

Figure 1: Performance (as measured by Rouge-L) vs Required Edits (measured by the Translation Edit Rate between the source article and the gold reference article). Performance diminishes as the number of required edits increases.

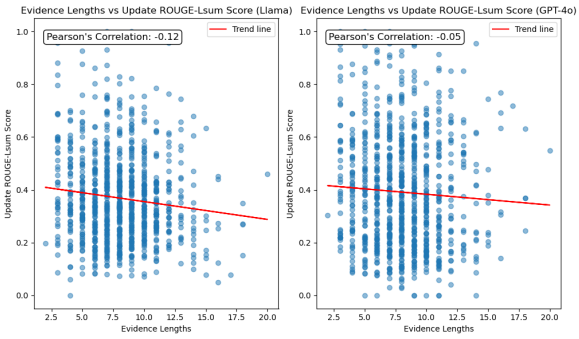

Figure 2: Performance (as measured by Rouge-L) vs Evidence Counts for Llama-3-8b and GPT-4. Performance drops with increasing number of evidences.

• **Choice of external evaluators**: We inspect the Llama-3, GPT-4o, TigerScore and Prometheus evaluations and find that these models are very poor at evaluating this task. We observe low correlation of these metrics and the UpdateROUGE metrics, showing that these models are not good evaluators of the faithful updation task. These correlations matter because they indicate whether automatic evaluators track the core notion of faithful updating; weak alignment implies that evaluator-driven refinement can optimize the wrong objective, which is critical for interpreting results and designing future benchmarks.

• **Effect of Tabular Evidences**: As presented in Table 1, 7% of gold samples consist solely of tabular evidence, whereas 78% contain both text and tables. To assess the influence of structured (tabular) evidence on performance, refer to Figure 3. For Llama-3-8b, incorporating tabular inputs leads to significant performance gains across all approaches. Specifically, in the zero-shot setting, samples with only tabular evidence achieve an average UpdateROUGE-LSum score of 45, compared to 33 for those with plain text. This pattern is not observed with GPT-4o, indicating that smaller models derive greater benefit from structured formats.

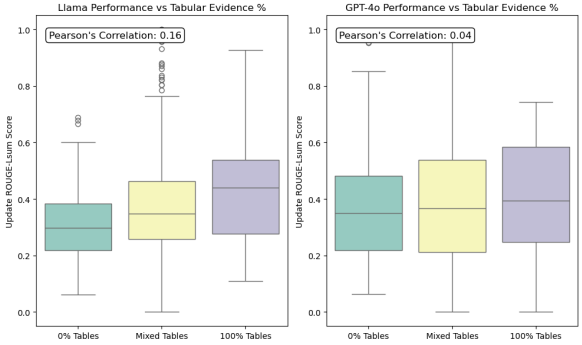

Figure 3: Tabular Evidence vs Performance (as measured by Rouge-L) vs % of Tabular Evidences. Performance increases slightly with more tabular evidences.

5.3 Discussion

This section discusses how the current work can inform future research and provides forward-looking insights into the broader trajectory of the field.

• **Generalization of the Prompting Methods**: We analyze the generalizability of the zero-shot method across articles from various domains within the FRUIT dataset. Articles are classified into different domains using (Rep et al., 2024), and domain-specific performance is evaluated, as illustrated in Figure 4. Our findings show that both Llama-3-8b and GPT-4o perform comparably across these domains, indicating robust domain generalization.

• **Reflect & Refine and Evidence Ordering**: These prompting methods (ref. Section 3.5) are particularly well-suited for domain-specific tasks that benefit from iterative prompting. One such domain is software engineering. In software requirements documents, iterative prompting proves valuable due to the need for step-by-step refinement – such as extracting requirements, creating system models, and generating code or summaries – by leveraging previous responses to enhance output quality (Alhanahnah et al., 2024; Austin et al., 2021; Frydenlund et al., 2024; Krishna et al., 2024). These techniques have been employed in requirements engineering settings, where model outputs are refined using feedback mechanisms or verification tools (Li et al., 2023; Luitel et al., 2024; Mandal et al., 2023), and further improved through multi-step refinement strategies (Wu et al., 2023).

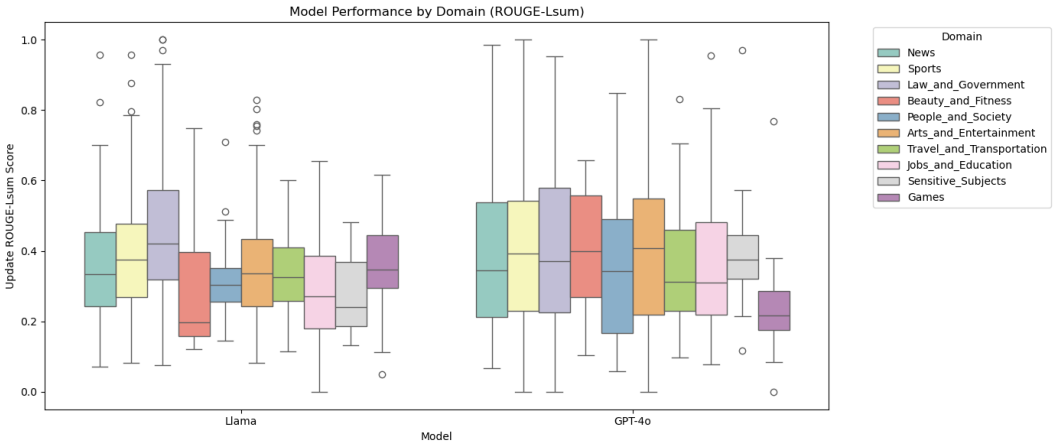

Figure 4: Domain Specific Performance (in Update Rouge-L). Performance is comparable across domains.

Table 3: The overlap between the target article in the FRUIT dataset and the parametric knowledge of the LLMs about the main entity, on a random sample 20 examples.

Model	UpdateROUGE			BertScore			Meteor	Entities			
	R1	R2	RL	P	R	F		Pr	R	F1	Unsupp
Llama-3-8b	15.92	6.2	12.34	64.34	62.56	63.43	24.8	47.43	43.82	45.55	7.31
GPT-4-o	18.8	6.9	13.58	67.78	61.99	64.43	26.6	49.13	45.6	47.3	6.54

• **Requirement for a new dataset**: A key limitation of the current evaluation setup lies in the temporal and structural mismatch between recent LLMs and the FRUIT dataset. While models such as GPT-4o and Llama-3-8b incorporate knowledge up to 2023, FRUIT is constructed from Wikipedia revisions dated 2020–2021. This gap creates a risk of information leakage, as models may have already internalized post-update facts that the dataset intends to test, thereby inflating their measured performance. Our manual experiment with 20 samples – where entities such as Jaylen Guy Twyman were extracted and compared against model responses – confirmed that overlaps are often lower because LLMs provide more up-to-date information than what FRUIT reflects (Table 3). Beyond this temporal concern, the reliance on Wikipedia as the sole source also constrains the dataset's representativeness. Wikipedia revisions capture certain structural and stylistic patterns but fail to reflect the complexities of domain-specific documents such as technical manuals, medical records, or policy documentation, which typically contain stricter semantics, formatting, and constraints. As a result, models optimized for FRUIT may not generalize effectively for the enterprise or high-stakes domains, limiting the dataset's practical utility. To address these issues, future benchmarks should either be curated from post-2024 data, ensuring alignment with LLM training windows, or evaluations should be restricted to earlier-generation models. Additionally, expanding beyond Wikipedia to incorporate diverse corpora will be critical for assessing model reliability in realistic, domain-rich scenarios.

6 Conclusions

In this paper, we present an in-depth analysis of LLMs' ability to perform faithful document updates on the FRUIT task. We find that prompting strategies—especially zero-shot and few-shot setups—substantially affect performance, and that structured evidence generally improves update quality. Our benchmarking shows clear model-dependent trends: fine-tuning performs best for open-source models, while in-context learning is most effective for closed-source models. We also observe that few-shot prompting with well-matched examples improves both generation quality and faithfulness, highlighting the importance of careful prompt design. At the same time, LLMs struggle when updates require many edits or large evidence sets, suggesting limitations in memory and attention. Methods such as Reflect-and-Refine and evidence ordering can help, but gains vary across models. Performance is also sensitive to evidence domain and format, with structured sources yielding more reliable updates than noisy, unstructured web text.

7 Ethics statement

This work evaluates large language models for faithful text updating using the FRUIT benchmark and does not involve human-subject experimentation or collection of personally identifiable information. We use publicly available data and report aggregate results only. Our experiments are intended to improve reliability in document revision settings, but we acknowledge that benchmark-based evaluation may not fully capture real-world harms.

A key ethical risk is misuse: systems for automatic text updating can introduce subtle factual errors, omit critical context, or produce plausible but unsupported edits. Such failures could be harmful in high-stakes domains (e.g., legal, medical, or policy documents). To mitigate this risk, we emphasize faithfulness-focused evaluation, compare multiple prompting and adaptation strategies, and discuss model limitations (especially under large evidence loads and unstructured inputs).

8 Limitations

Although this study offers a thorough evaluation of prompting strategies for the FRUIT task, certain limitations restrict the generalizability and practical applicability of the results.

• **Limited Exploration of Iterative Prompting Methods**: Although Reflect & Refine and Streaming Evidence prompting methods were introduced and tested, their performance showed mixed results. These techniques are theoretically well-suited for scenarios involving step-wise document refinement (e.g., software requirements engineering, contract updates). However, we did not evaluate their effectiveness on real-world domain-specific datasets where such iterative updating is common. Further empirical studies are required to understand how these methods perform in specialized workflows with varying evidence complexity and update frequency.

• **Dependence on Wikipedia-Based Data**: The FRUIT dataset, built from Wikipedia revisions, may not adequately represent the structure, semantics, and constraints of domain-specific documents such as technical manuals, medical records, or policy documentation. As a result, models optimized for FRUIT may struggle with transferability to real-world enterprise or industry use cases. Evaluating performance on more diverse corpora is crucial for understanding model behavior across domains.

• **Risk of Knowledge Leakage**: Since the FRUIT dataset was constructed from 2020–2021 Wikipedia articles and the LLMs used have knowledge cutoffs in late 2023, there is a potential for information leakage. That is, model performance might be inflated by prior exposure to post-update facts during training. While we attempted a small-scale manual verification of overlap, a broader audit would be needed to assess the extent of this issue. Ideally, either models with earlier knowledge cutoffs or newly curated datasets beyond their training window should be used.

References

Mohannad Alhanahnah, Md Rashedul Hasan, and Hamid Bagheri. 2024. An empirical evaluation of pre-trained large language models for repairing declarative formal specifications. *arXiv preprint arXiv:2404.11050*.

Shengnan An, Bo Zhou, Zeqi Lin, Qiang Fu, Bei Chen, Nanning Zheng, Weizhu Chen, and Jian-Guang Lou. 2023. Skill-based few-shot selection for in-context learning. In *EMNLP*, pages 13472–13492.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and 1 others. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Axolotl maintainers and contributors. 2023. Axolotl: Open source llm post-training.

John Bandler. 2024. Building and updating organization policies and procedures.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

MPS Bhatia, Akshi Kumar, Rohit Beniwal, and Tushar Malik. 2020. Ontology driven software development for automatic detection and updation of software requirement specifications. *Journal of Discrete Mathematical Sciences and Cryptography*, 23(1):197–208.

Tom B. Brown and 1 others. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *Preprint*, arXiv:2305.14314.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Jane Dwivedi-Yu, Timo Schick, Zhengbao Jiang, Maria Lomeli, Patrick Lewis, Gautier Izacard, Edouard Grave, Sebastian Riedel, and Fabio Petroni. 2024. EditEval: An instruction-based benchmark for text improvements. In *CoNLL*, pages 69–83.

Erika Frydenlund, Joseph Martínez, Jose J Padilla, Katherine Palacio, and David Shuttleworth. 2024. Modeler in a box: how can large language models aid in the simulation modeling process? *Simulation*, 100(7):727–749.

Jade Goldstein and Jaime Carbonell. 1998. Summarization: Using MMR for diversity- based reranking and evaluating summaries. In *Tipster Text P PHASE III: Proceedings of a Workshop held at Baltimore, Maryland*, pages 181–195.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, and 1 others. 2024. A survey on llm-as-a-judge. *arXiv preprint arXiv:2411.15594*.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *Preprint*, arXiv:2106.09685.

Robert Iv, Alexandre Passos, Sameer Singh, and Ming-Wei Chang. 2022. Fruit: Faithfully reflecting updated information in text. In *NAACL*, pages 3670–3686.

Dongfu Jiang, Yishan Li, Ge Zhang, Wenhao Huang, Bill Yuchen Lin, and Wenhu Chen. 2024. Tigerscore: Towards building explainable metric for all text generation tasks. *Preprint*, arXiv:2310.00752.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2024. Prometheus 2: An open source language model specialized in evaluating other language models. *Preprint*, arXiv:2405.01535.

Philipp Klimant and Christian Kollatsch. 2022. Concepts for creating augmented reality based technical documentations for the maintenance of machine tools. In *International Journal on Interactive Design and Manufacturing (IJIDeM)*, pages 467–474. IEEE.

Madhava Krishna, Bhagesh Gaur, Arsh Verma, and Pankaj Jalote. 2024. Using llms in software requirements specifications: an empirical evaluation. In *2024 IEEE 32nd International Requirements Engineering Conference (RE)*, pages 475–483. IEEE.

Md Tahmid Rahman Laskar and 1 others. 2024. A systematic survey and critical review on evaluating large language models: Challenges, limitations, and recommendations. In *EMNLP*, pages 13785–13816.

Jia Li, Ge Li, Yongmin Li, and Zhi Jin. 2023. Enabling programming thinking in large language models toward code generation. *arXiv preprint arXiv:2305.06599*.

Miaoran Li, Rogger Luo, and Ofer Mendelevitch. 2024. HHEM-2.1-Open.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, William B Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for gpt-3? In *Proceedings of DeeLIO: The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114.

Llama-3. 2024. Llama 3 model card.

Dipeeka Luitel, Shabnam Hassani, and Mehrdad Sabetzadeh. 2024. Improving requirements completeness: Automated assistance through large language models. *Requirements Engineering*, 29(1):73–95.

Kumar M and 1 others. 2016. Optimization of process time in the creation and updation of technical publication documents using business process reengineering: Case study. *Optimization of Process Time in the Creation and Updation of Technical Publication Documents Using Business Process Reengineering: Case Study (April 20, 2016)*.

Aman Madaan and 1 others. 2023. Self-refine: Iterative refinement with self-feedback. *Preprint*, arXiv:2303.17651.

Shantanu Mandal, Adhrik Chethan, Vahid Janfaza, SM Mahmud, Todd A Anderson, Javier Turek, Jesmin Jahan Tithi, and Abdullah Muzahid. 2023. Large language models based automatic synthesis of software specifications. *arXiv preprint arXiv:2304.09181*.

Bianca Minetto Napoleão, Fabio Petrillo, Sylvain Hallé, and Marcos Kalinowski. 2022. Towards continuous systematic literature review in software engineering. In *2022 48th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 765–773.

OpenAI, :, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, and 401 others. 2024. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

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

Raufdeen Rameezdeen and Anushi Rodrigo. 2014. Modifications to standard forms of contract: The impact on readability. volume 14, pages 31–40.

Ivan Rep, David Dukić, and Jan Šnajder. 2024. Are electra's sentence embeddings beyond repair? the case of semantic textual similarity. In *EMNLP Findings 2024*, pages 9159–9169.

Lei Shu, Liangchen Luo, Jayakumar Hoskere, Yun Zhu, Yinxiao Liu, Simon Tong, Jindong Chen, and Lei Meng. 2024. Rewritelm: An instruction-tuned large language model for text rewriting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18970–18980.

Shruti Singh and Rishabh Gupta. 2024. Sparse graph representations for procedural instructional documents. *Preprint*, arXiv:2402.03957.

Matthew Snover, Bonnie Dorr, Rich Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*, pages 223–231.

Peiqi Sui, Eamon Duede, Sophie Wu, and Richard Jean So. 2024. Confabulation: The surprising value of large language model hallucinations. In *ACL*, pages 14274–14284.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models. *Preprint*, arXiv:2201.11903.

Yonghao Wu, Zheng Li, Jie M Zhang, Mike Papadakis, Mark Harman, and Yong Liu. 2023. Large language models in fault localisation. *arXiv preprint arXiv:2308.15276*.

Haopeng Zhang, Hayate Iso, Sairam Gurajada, and Nikita Bhutani. 2024. Xatu: A fine-grained instruction-based benchmark for explainable text updates. In *LREC-COLING*, pages 17739–17752.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. *Preprint*, arXiv:1904.09675.

A Appendix

A.1 Dataset Preparation Details

Text from Wikiepedia dump is stripped of markup, normalised, and has its tables serialized. Then stylistic updates containing no new information is removed by detecting updates with no new entities.

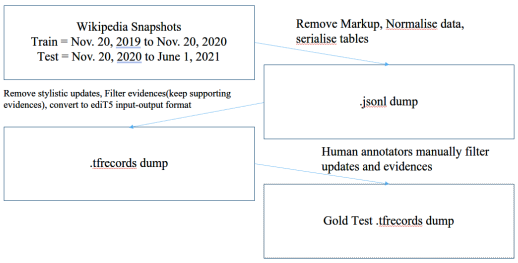

Figure 5: Data Processing Pipeline

The test set was further sampled and manually annotated to filter updates and evidences by human annotators to produce a gold set. The pipeline is shown in Figure 5. We use the gold set in our experiments.

The data before removal of stylistic updates and evidence filtering comes in a .jsonl format, and the data after the stylistic updates removed and the evidences filtered comes is present in a .tfrecord format. This is in the ediT5 input-output format. EdiT5 (Iv et al., 2022) is a t5-model which was trained to copy sentences that are unedited and to refer to the evidences before using them to generate a new edited sentence.

A.2 FRUIT dataset Format

The ediT5 input format (for example, see Figure 6) consists of indexed sentences (using square brackets) and indexed evidences (using parentheses) both separated with a [CONTEXT] token. The tables consist of the heading, a caption, and a header delimited using [COL], [ROW], and [HEADER] tokens.

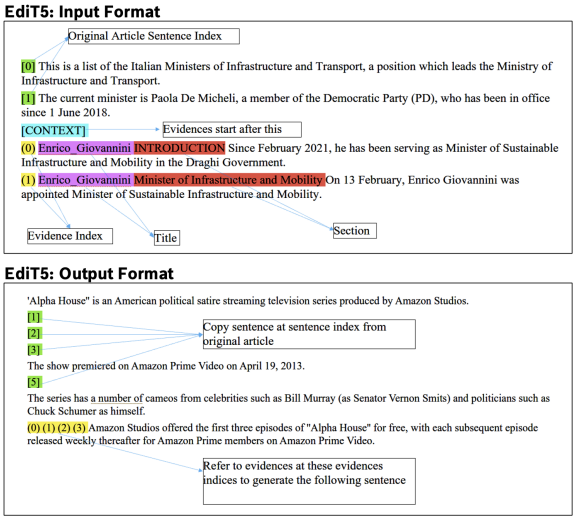

Figure 6: ediT5 format of the FRUIT dataset.

The ediT5 output format (for example, see Fig-

Table 4: Baseline results from (Iv et al., 2022)

	UpdateROUGE			Entities			
	R1	**R2**	**RL**	**Pr**	**R**	**F1**	**Unsupp**
Copy Source	0	0	0	0	0	0	0
+ All Evidence	18.8	6.9	12	37.9	64.9	47.85	0
T5-Large	31.1	18.4	24.4	52.7	44.9	48.49	2.67
+ Evidence Input	44.3	29.4	36.8	62.2	50.7	55.86	2.34
EdiT5-Small	41.2	27.3	35.3	62.4	44.9	52.22	1.71
EdiT5-Base	47	32.1	39.7	62.2	**54.9**	58.32	2.28
EdiT5-Large	46.3	32.4	39.6	67.2	53.1	59.32	**1.54**
EdiT5-3B	**47.4**	**34**	**41.1**	**69.9**	52.2	**59.77**	1.58

ure 6) consists of references to the original article's sentences using the sentence index whenever the model wishes to use the sentence from the original article as is without any changes. For any updates that the model does make, it first grounds its generation by generating evidence indices from the input and then generates the updated sentence. The training data for this was generated by matching updated sentences with evidences by matching entities.

This output format was matched with the original .jsonl format to obtain the filtered original version of the text without any sentence indices.

A.3 Baseline methods and Results

Here, we briefly discuss the methods introduced in the work Iv et al. (2022).

• **Copy Source**: This method generates a copy of the source article without any modifications.

• **Copy Source + Evidence:** This approach concatenates the source article with the new evidence and outputs the combined text.

• **T5 Fine-Tuning**: The T5 model is fine-tuned using only the source article as input (T5) and also with both the source article and the new evidence as inputs (T5+Evidence Inputs)

• **EdiT5**: EdiT5, a variant of T5 proposed by (Iv et al., 2022), enhances text updating tasks by using a compressed output format that reduces the need to generate the entire text from scratch. It features Diff-Formatted Output, which replaces copied sentences with a copy token, and Reference and Control Tokens that guide the model on whether to add, edit, or delete content, promoting effective content planning.

A.4 Prompts over the different approaches

System Prompt

We use the following system prompt for all our experiments.

> You are a knowledgeable, efficient, and direct AI assistant. Provide concise answers, focusing on the key information needed. Engage in productive collaboration with the user.

Zero Shot Prompt

We use this prompt after the system prompt, and provide it with the user role.

> Given a source article and evidence documents, edit the source article to incorporate new information from the evidence documents. Prefer substitutions and editing sentences over adding new ones. Just generate the updated article and not the evidences.
> Source Article: $source
> Evidence 1:
> Title: $title_1
> Section: $section_1
> $content_1
> Evidence 2: ...

COT Prompt

We use a two step prompt where the model first lists discrepancies then uses that chain of thought to answer.

> Given a source article and evidence documents, find cases where information given in the evidences does not agree with the source article or where the source article does not contain information it should from the evidences. Generate the evidence number and the disagreement/missing information in the source article.
> Source Article: $source $evidences

> Now, given the above source article, evidence documents and the disagreements / missing information, edit the source article to incorporate new information by updating or adding from the evidence documents to correct the disagreements or add the missing information. Prefer substitutions and editing sentences over adding new ones. Just generate the updated article and not the evidences

Self Reflection Prompt

We use a three step prompt to generate an initial article, then critique it using the model itself, TigerScore, and Prometheus 2, and then using the critique refine the answer. The initial generation

prompt is same as in zero shot. We use the following prompt to evaluate the model's own output.

> Now, evaluate the generated updated article. Find cases where the updated article does not agree or contains missing information when compared with the supplied evidences. List out all such discrepancies with the evidence number and the reason.

We use the following rubric for the prometheus model to grade the model output with

> Criteria: Is the model proficient in updating articles based on new evidence, making correct and precise edits in place wherever possible?
> Score 1: The model neglects to identify or incorporate new evidence into the article, resulting in outdated and inaccurate information.
> Score 2: The model intermittently acknowledges new evidence but often fails to make correct and precise edits in place, leading to incomplete or inaccurate updates.
> Score 3: The model typically identifies new evidence and attempts to make correct and precise edits in place, yet the updates might sometimes miss important details or lack precision.
> Score 4: The model consistently identifies and incorporates new evidence into the article, making correct and precise edits in place. Nonetheless, there may still be sporadic oversights or deficiencies in the accuracy and precision of the updates.
> Score 5: The model excels in identifying and incorporating new evidence into the article, persistently making correct and precise edits in place that demonstrate a thorough understanding of the subject matter. The updates are accurate, precise, and comprehensive, leaving no room for inaccuracies or incomplete information.

Finally, we use the following prompt to refine our answer.

> Now, given the evaluation and the previously updated article, fix the discrepancies and generate a new updated article. Prefer substitutions and editing sentences over adding new ones.

One Shot Prompt

We use the chat history to provide in context examples, autocompleting the assistant role with the reference output.

A.5 Need for a new dataset

The Table 5 shows a example on Jaylen Guy Twyman from the FRUIT dataset. The first two rows are the source article and the target article, directly from the dataset. The last two rows show the predictions from the Llama-3 and the GPT-4 models respectively. The highlighted phrase shows the updated texts that are absent in the source article. As we can see the LLMs (Llama-3-8b and GPT-4) already contains the updated knowledge (2021 NFL Draft) and also more recent information pertaining to Minessota Vikings. This shows that the LLMs have more updated knowledge since it had access to more recent public sources while training. The FRUIT dataset on the other hand has outdated knowledge (and therefore lesser information and entities). Hence there is a need to develop new datasets in this domain.

Table 5: Example from the dataset on Jaylen Guy Twyman

Source Article	Jaylen Guy Twyman (born July 19, 1999) is an American football defensive tackle for the Pittsburgh Panthers.
Target Article	Jaylen Guy Twyman (born July 19, 1999) is an American football defensive tackle for the Minnesota Vikings of the National Football League (NFL). He played college football at Pittsburgh and was drafted by the Vikings in the sixth round of the 2021 NFL Draft.
Prediction : Llama-3-8b	Jaylen Guy Twyman is an American football defensive tackle who played college football at the University of Pittsburgh. He gained attention for his impressive performance during his time at Pitt, particularly in the 2019 season when he recorded 10.5 sacks. Twyman declared for the NFL Draft and was selected by the Minnesota Vikings in the sixth round of the 2021 NFL Draft. His career faced a significant challenge when he was shot four times while visiting family in Washington, D.C., in June 2021, but he survived and has been working towards making a return to football.
Prediction : GPT-4	Jaylen Guy Twyman is an American professional football player who plays on the defensive line. Born on July 19, 1999, in Washington, D.C.,he starred at H. D. Woodson High School before playing college football at the University of Pittsburgh. At Pitt, he earned significant recognition— racking up impressive sack totals and earning first-team All-ACC honors—mirroring the legacy of notable defensive players like Aaron Donald. Drafted by the Minnesota Vikings in the sixth round of the 2021 NFL Draft, Twyman has since spent time with several NFL teams (including the Miami Dolphins) and New York Jets) and alternative football leagues. Off the field, he has overcome serious challenges, notably recovering from being shot four times in 2021. Standing 6’2" and weighing around 285–290 pounds, Twyman’s physicality and performance continue to define his career in professional football.