Modeling Iterative Text Revisions Through Chain of Revisions

Zhexiong Liu, Diane Litman
Department of Computer Science, Learning Research & Development Center
University of Pittsburgh, Pittsburgh, Pennsylvania, USA 15260
zhexiong@cs.pitt.edu, dlitman@pitt.edu

Abstract

Text revision has become an essential part of writing, as it often improves the text through iterative refinement. However, large language models (LLMs) have primarily regarded text revision as a holistic task that simultaneously revises grammar, logic, and content. This might be because LLMs are pre-trained only to generate final text rather than to learn the iterative revision process. Therefore, such LLMs largely overlook the dynamic nature of human revision, where writers progressively improve their writing through a sequence of intended edits. To bridge the gap, we investigate text revision generation from an iterativeness perspective and propose a chain of revisions that focuses on revision orders to iteratively improve text quality. Experimental results suggest that our proposed revision chains effectively capture the dynamics of text revision in multi-intent scenarios and outperform holistic revision baselines, which provide a valuable pilot study for iterative revision research.

1 Introduction

Text revision is an important part of writing, which allows writers to iteratively improve text quality through multiple rounds of editing (Sommers, 1980; Gollins and Gentner, 2016). However, a writer's actual intentions, which motivate the revision, are often complex. For example, Figure 1 shows that texts can be revised based on multiple intentions, with each edit driven by a specific purpose (i.e., intention). These examples illustrate the complexity of text revisions, and modeling such revisions with computational models is challenging. While recent advances in large language models (LLMs) have enabled researchers to generate successful revisions by addressing grammatical, logical, and content-level issues (Shu et al., 2024; Ziegenbein et al., 2024), these methods typically formulate text revision as a single-pass transformation from an initial draft to a final text. Such

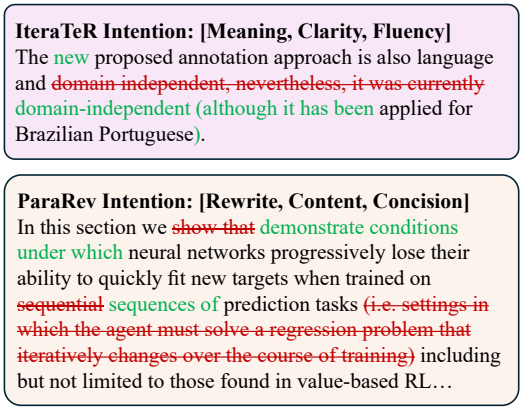


Figure 1: Revision examples: an original text can be revised based on multiple intentions, e.g., *meaning, clarity,* and *fluency* from the IteraTeR corpus (Du et al., 2022b) and *rewrite, content*, and *concision* from the ParaRev corpus (Jourdan et al., 2025).

designs largely disregard the iterativeness of human revision, in which writers progressively refine their work through a sequence of edits that are driven by distinct intentions (Kim et al., 2022), e.g., enhancing clarity, improving coherence, and altering content. As a result, these methods could fail to handle multi-intent revisions, in which intentions are entangled and revising one can influence another, e.g., fixing a grammatical text could possibly harm the text's clarity (Skitalinskaya and Wachsmuth, 2023). Therefore, it is crucial to model iterative revisions in a fine-grained manner that decomposes the revision process into explainable and controllable revision chains.

While prompting LLMs with pre-defined instructions has achieved impressive success on several NLP tasks (Zhang et al., 2025; Sun et al., 2025; Lin et al., 2025), their exploration in iterative revisions remains limited. For example, Davis et al. (2024) used zero-shot and few-shot learning for single-pass text revision but did not investigate optimal strategies for selecting helpful revision exemplars. Although Du et al. (2022a) developed an iterative revision system to optimize the final

Proceedings of the 15th International Joint Conference on Natural Language Processing and the 5th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, pages 295–307
November 6-10, 2026 ©2026 Association for Computational Linguistics

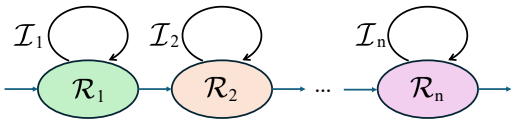


Figure 2: The *chain-of-revision* framework, where revisions are made based on n revision orders (steps), and each step involves edits based on fine-grained intentions.

text through human feedback, Ziegenbein et al. (2024) used reinforcement learning with machine feedback, and Cao and Ng (2025) designed LLM-based planners for iterative text refinement, they did not decompose complex revision tasks into fine-grained revision steps humans often implement (Faigley and Witte, 1981). Hence, such methods have limited interpretability and controllability over the revision process. While Li et al. (2026) explored a chain-of-thought prompt for iterative text editing, no optimal iterative revision order was investigated. In contrast, we propose a *chain-of-revision*[1] method with an intention-adaptive exemplar-retrieval strategy to capture the complex dynamics of iterative revision processes. Particularly, we study two research questions: **RQ1**: How to use LLMs to generate successful revisions and assess their quality effectively? **RQ2**: How to leverage revision exemplars to help LLMs generate improved revisions? We make three contributions:

- We propose a novel chain-of-revision prompt for modeling iterative revision generation.
- We investigate an intention-adaptive retrieval method for selecting useful revision exemplars.
- We develop LLM judges to complement the weaknesses of conventional revision metrics.

2 Corpora

We use a revision dataset named IteraTeR (Du et al., 2022b), which annotates 559 Wikipedia, arXiv, and Wikinews documents with 6 intentions: *clarity*, *fluency*, *coherence*, *style*, *meaning*, and *others*, as described in Appendix A.1. We create a subset, IteraTeR-multi, to exclude *others* that has unidentified intentions, comprising 397 documents with at least one sentence-level revision involving multiple intentions, as we hypothesize that revision order influences final text when revisions are based on multiple intentions. In addition, we use another dataset named ParaRev (Jourdan et al., 2025), which annotates 641 scientific paragraphs with 5 intentions: *rewrite, concision, development*, *content*, and *unusable* that has processing errors, as described in Appendix A.2. We also create a subset, ParaRev-multi, to exclude *unusable* and single-intent revisions, yielding 248 paragraphs. Furthermore, we use the IteraTeR-multi split based on Du et al. (2022b) and merge the validation and test sets for evaluation, given their small sizes. We split the ParaRev-multi into 70% for training and 30% for evaluation. The dataset statistics and number of intentions are shown in Appendix Tables 3 and 4.

3 Methods

Chain of Revisions. Prior work typically regards multi-intent revision as a single-pass generation (Liu and Litman, 2026), but overlooks the revision orders that can affect outcomes (Li et al., 2026). To explore this, we propose a *chain-of-revision* (CoR) method that explicitly models iterative revision as a sequence of revision orders (steps). Specifically, given an original text $\mathcal{X}$ and a set of revision intentions, our method enumerates plausible permutations of different revision chains and prompts the LLM to automatically select one chain to generate a revised text $\mathcal{Y}$. Specifically, each revision step $\mathcal{R}$ handles a single intention $\mathcal{I}$ applied to one or several sentences, e.g., improving *clarity* in one or multiple sentence(s), and the length of the revision chain is therefore determined by the number of given intentions, as illustrated in Figure 2. The CoR prompt decomposes multi-intent revisions into an ordered chain to enable the LLM to focus on one revision objective at a time and iteratively produce desirable revisions, as detailed in Appendix Tables 8 and 9.

LLM as a Revision Judge. We develop LLMs as judges to evaluate the quality of generated revisions to supplement the limitations of conventional revision metrics, e.g., SARI (Xu et al., 2016), that primarily focus on n-gram alignment and often correlate poorly with human preferences in high-level revisions (Maddela et al., 2023). In addition, existing metrics do not leverage intention information that motivates the revisions, which is important for assessing revision quality (Liu and Litman, 2025). Inspired by this, we design LLM judges to include intention references and focus on revisions over three span-level edit operations, *keep*, *delete*, and *add*, and assess whether the predicted edits perform the same operation as the gold revision. It rewards aligned edits from the perspective of the *surface* and *content* dimensions based on Faigley and Witte

[1] https://github.com/ZhexiongLiu/Chain-of-Revision

LLMs	Method	IteraTeR-multi						ParaRev-multi					
		SARI	GLEU	Update-R	Surface	Content	Average	SARI	GLEU	Update-R	Surface	Content	Average
-	Copy-Baseline	26.91	57.18	0.00	15.35	21.05	24.10	26.75	46.60	0.00	26.00	17.00	23.27
GPT-5.2	Holistic-Baseline	39.50	34.03	53.30	61.84	55.26	48.79	**39.11**	18.13	36.70	56.67	51.33	40.39
	CoT-Baseline	38.86	31.77	51.98	62.28	55.26	48.03	37.68	18.45	37.22	57.00	51.67	40.40
	CoR (ours)	**40.51**	**38.03**	**55.02**	**63.16**	**56.58**	**50.66**	**39.11**	**23.04**	**37.99**	**59.33**	**53.33**	**42.56**
Llama3.1-8B	Holistic-Baseline	35.07	17.95	53.01	60.96	51.75	43.75	38.00	19.12	41.33	53.67	45.33	39.49
	CoT-Baseline	**35.42**	18.57	**54.17**	58.77	49.12	43.21	**38.39**	**21.47**	**41.97**	57.33	46.00	**41.03**
	CoR (ours)	35.32	**22.39**	53.45	**61.84**	**53.51**	**45.30**	34.36	18.54	39.45	**58.00**	**47.00**	39.47

Table 1: The LLM performance in terms of baselines and our chain of revision (CoR) in zero-shot learning, evaluated with conventional and our LLM-as-a-revision-judge metrics. The bold numbers are the best results in each block.

(1981), which primarily categorizes the revisions into surface and text-based levels. The two perspectives are scored independently on a five-point Likert scale, then normalized to 0-1, with higher scores indicating that the predictions more closely align with those of the gold references. To improve the evaluation interpretability, the judge additionally requires a rationale that grounds the decision in concrete facts. Such metrics leverage the linguistic capabilities of LLMs to assess revisions beyond n-gram overlap and to capture alignment based on desirable intentions, as detailed in Appendix Table 10.

Intention-Adaptive Retrieval. We use revision exemplars to guide LLMs in generating intended revisions. Existing retrieval-based few-shot learning typically uses a *semantic-retriever* that employs Sentence-BERT (Reimers and Gurevych, 2019) to encode text and to retrieve similar exemplars. Although this design captures similar revision patterns that can support desirable revision generation, it relies solely on revision semantics, disregarding the important intention that motivates the revisions. As a result, retrieved exemplars often exhibit revision behaviors inconsistent with the target intentions, e.g., returning *meaning* edits when *clarity* revisions are desired, misleading LLMs to generate degraded revisions. To address this, we propose an *intention-retriever* that conditions retrieval on the revision intention relevant to each test instance, filtering out candidates whose revision intentions do not match the target, e.g., discarding *meaning* exemplars when the goal is *clarity*. This simple mechanism ensures the selected exemplars are both semantically similar and intention-adaptive.

4 Experiments

We use both open-source Llama3.1-8B (Grattafiori et al., 2024) and closed-source GPT-5.2[2] to study revision generation, and GPT-4.1[3] for LLM judges since GPT-4.1 achieves human-like annotation performance (Luo et al., 2025). We use vLLM (Kwon et al., 2023) framework on an Nvidia 3090 GPU for open-source LLMs and the Azure Foundry API[4] for closed-source LLMs. We set the temperature to 0 and the maximum tokens to 2048. We use the training set for exemplar retrieval, and report results on the evaluation set. We use **SARI** (Xu et al., 2016), **GLEU** (Napoles et al., 2015), and Update-ROUGE (**Update-R**) (Logan IV et al., 2022) scores as metrics based on Shu et al. (2024), as well as our proposed LLM-judge metrics **Surface** and **Content** as described in Sec. 3. We experiment with the following methods:

- **Copy**: Copy the original text as revised text for a non-edit baseline (Liu and Litman, 2026).
- **Holistic**: Use a prompt to address all the intentions, based on Raheja et al. (2023) and Ziegenbein et al. (2024), as in Appendix A.3.
- **Chain-of-Thought (CoT)**: Use a prompt to instruct LLMs to generate revisions step by step based on Wei et al. (2022), as in Appendix A.3.
- **Chain-of-Revision (CoR)**: Use a CoR prompt to instruct LLMs to generate revisions using a dynamically determined order. We also include a fixed-order CoR based on Li et al. (2026) for revision order analysis. The detailed prompts are described in Appendix A.3.

5 Results

We present the results regarding baselines and the proposed CoR in zero-shot learning in Table 1. It shows that GPT-5.2 with CoR mostly outperforms the holistic and CoT baselines, except for the SARI score on ParaRev-multi; Llama3.1-8B with CoR mostly achieves the best results on IteraTeR-multi, except for the SARI and Update-R scores, but is

[2] https://openai.com/index/introducing-gpt-5-2/

[3] https://openai.com/index/gpt-4-1/

[4] https://ai.azure.com/

Dataset	Revision Chain (3 intentions)	Win	Tie	Lose
IteraTeR	CoR vs. Clarity-Coherence-Meaning	**64.29%**	14.29%	21.43%
	CoR vs. Clarity-Meaning-Coherence	**57.14%**	21.43%	21.43%
	CoR vs. Coherence-Clarity-Meaning	**64.29%**	14.29%	21.43%
	CoR vs. Coherence-Meaning-Clarity	42.86%	**50.00%**	7.14%
	CoR vs. Meaning-Clarity-Coherence	**42.86%**	28.57%	28.57%
	CoR vs. Meaning-Coherence-Clarity	**42.86%**	**42.86%**	14.29%
ParaRev	CoR vs. Concision-Content-Rewrite	**50.00%**	33.33%	16.67%
	CoR vs. Concision-Rewrite-Content	**66.67%**	16.67%	16.67%
	CoR vs. Content-Concision-Rewrite	**41.67%**	16.67%	**41.67%**
	CoR vs. Content-Rewrite-Concision	**41.67%**	25.00%	33.33%
	CoR vs. Rewrite-Concision-Content	33.33%	**41.67%**	25.00%
	CoR vs. Rewrite-Content-Concision	**50.00%**	25.00%	25.00%

Table 2: LLM-judge performance on dynamic CoR and fixed revision chains in zero-shot learning across two datasets. The winning, tied, and losing rates are the rates for CoR on 3-intent revisions generated by GPT-5.2. The results are the average of two runs with CoR as the first and fixed chains as the first inputs in the prompt.

less effective on ParaRev-multi, e.g., lower SARI, GLEU, and Update-R scores. Note that the Copy baseline (no revision) has the highest GLEU score but zero Update-R, indicating that SARI is a critical metric. However, the SARI is less convincing when the revisions use paraphrased text (see the examples in Appendix Figure 5 and the discussions in Appendix A.4). In contrast, our proposed LLM judge yields consistent results across LLMs and datasets, which reveals its benefits in capturing both semantics and intention alignment. We manually evaluate the judge with 30% IteraTeR-multi data, and the QWK scores between one annotator and the judge are 0.45 for *surface* and 0.41 for *content*, both indicating moderate agreement (Landis and Koch, 1977), which is comparable to the alignment between human and LLM judges for writing assessment tasks in Wang et al. (2025).

To analyze the revision chains, we use a subset of 3-intent revisions as a case study, since the number of intentions varies across examples in Table 1 (e.g., from 2 to 5 intentions), and using more intentions would cause revision chains to grow exponentially (e.g., 5 intentions yield 120 different chains, as shown in Appendix Table 4). Here, we use another GPT-4.1 judge to evaluate pairwise performance between the CoR and each fixed-order chain based on the prompt in Appendix Table 11, which has a fair-agreed 0.32 QWK between human and the LLM judge on one-third of IteraTeR-multi pairwise data. Results in Table 2 show that CoR mostly has more wins than losses across the two datasets, and is likely to tie when the chain starts with *coherence* and *meaning* for IteraTeR-multi and *rewrite* for ParaRev-multi, which suggests that CoR might prefer to start with these intentions to generate improved revisions. To validate it, we visualize

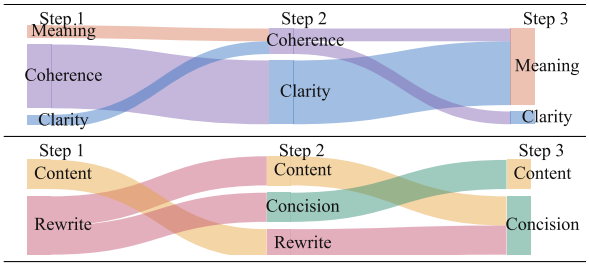


Figure 3: The revision steps (chains) generated by GPT-5.2 with zero-shot learning and CoR on the 3-intent examples in IteraTeR-multi (upper) and ParaRev-multi (lower) datasets. The purple, orange, blue, red, yellow, and green represent *coherence*, *meaning*, *clarity*, *rewrite*, *content,* and *concision*, respectively.

the dynamic CoR steps in Figure 3. The upper Sankey diagram shows that CoR mostly starts with *coherence* (purple), followed by *clarity* (blue) and *meaning* (orange), which is in line with Kim et al. (2022) that proficient writers are likely to revise *coherence* before *clarity*. Regarding ParaRev-multi, CoR prefers starting with *rewrite* and *content* and then does *concision*. These observations suggest that the CoR prompt dynamically selects revision chains and that LLM judges could supplement conventional revision metrics for revision assessment, which answers **RQ1**.

We further investigate retrieval methods for selecting helpful exemplars. Figure 4 shows the Average performance across LLMs and datasets. Regarding semantic retrievers (blue lines), more shots generally degrade performance with GPT-5.2, but one or two shots could improve performance for Llama3.1-8B across two datasets. While our intention retrievers (yellow lines) achieve the best results with one or two shots, except for GPT-5.2 with zero-shot on ParaRev-multi, three shots are mostly not helpful. This suggests that the quantity of the exemplar does not compensate for the quality of the exemplars, and more shots would have a higher chance to include useless examples with low similarity scores, as shown in Appendix Figure 6. This finding is consistent with prior work (Davis et al., 2024), which suggests that more exemplars may not benefit grammar-level revisions, and few-shot learning could help such tasks. Hence, intention retrievers equipped with appropriate exemplars could achieve improved results, which answers **RQ2**.

6 Conclusion

We propose the chain of revisions, a novel method for effectively capturing the dynamic and iterative revision process. We explore open-source and

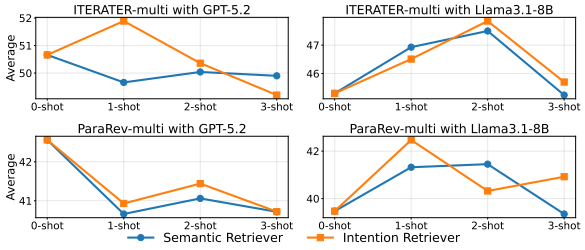


Figure 4: The CoR Average results in terms of different numbers of exemplars retrieved by the semantic-retriever and our intention-retriever on both IteraTeR-multi and ParaRev-multi datasets.

closed-source LLMs for modeling multi-intent revisions and demonstrate that the chain of revisions, equipped with intention-adaptive retrievers, can improve revision quality, and that LLM judges are useful for capturing semantic and intention alignment, which provides a valuable study for explainable and controllable iterative revision generation.

Ethics Statement

We use publicly available human-annotated revision corpora, which were collected and annotated for academic purposes and pose no ethical implications, such as those related to identity, gender, or race. The two corpora include text revisions from Wikipedia, arXiv, news, and scientific articles, which increases the diversity of the revision study. Furthermore, we acknowledge the existing issues with LLM-generated revisions, including inaccuracies and hallucinations, which need to be addressed in future work.

Limitations

Experiment. Our experiments use relatively small annotated corpora due to the limited availability of multi-intent revision corpora; however, mostly consistent results across two datasets and LLMs demonstrate the generalizability of our proposed CoR. Also, the studied intentions may not cover all types of revisions, e.g., lexical and sentence-reordering revisions, which could be extended in future work. Although we visualized dynamic revision orders (steps) selected by the LLMs, a more detailed analysis of why a specific revision order was preferred could further explain the dynamics of revision iterativeness and could also be our future work. In addition, we compare CoR with an agent-based planner-executor framework adapted from Cao and Ng (2025). We only experiment with closed-source GPT-5.2 through API calls due to computational constraints, which obtains Average scores of 46.19 and 40.05, compared to CoR scores of 50.66 and 42.56, for IteraTeR-multi and ParaRev-multi, respectively. This might be because agent-based frameworks are less interpretable and controllable, as the revision planning process might not be well justified; thus, an improved agent framework could be a focus of our future investigation.

Complexity. Regarding the complexity, our CoR method would require more tokens than holistic prompts, since additional reasoning and judgment are needed, as indicated in the prompt examples in Appendix A.3. For instance, GPT-5.2 with the Azure Foundry API requires an average of 741 tokens for the holistic, 754 tokens for the CoT, and 1,579 tokens for the CoR without intermediate revision outputs, while running on the examples from the IteraTeR-multi dataset. However, their response time remains close, e.g., 3.5, 3.3, and 3.4 seconds per example for the holistic, CoT, and our CoR, respectively, even though the CoR has doubled token assumptions. Additionally, given that the orders of revision chains grow exponentially, e.g., 5 intentions would have 120 different orders, scaling to longer revision chains would require substantial tokens. Therefore, future work could focus on pruning revision chains to drop useless orders.

Revision Chain. In terms of the case study on the revision chains, we focus on 3 intentions, e.g., *clarity-coherence-meaning* for IteraTeR-multi and *concision-content-rewrite* for ParaRev-multi. These subsets are small, with seven IteraTeR examples and six ParaRev examples, because each example in the full datasets contains a different number of intentions, e.g., 2-intent, 3-intent, 4-intent, etc., and a different combination of intentions, e.g., *clarity-coherence-meaning* and *fluency-coherence-meaning* being both 3-intent chains. Even when the number and set of intentions are fixed, different orderings create additional chain permutations, e.g., *clarity-coherence-meaning* and *clarity-meaning-coherence*. Such degrees of complexity make it a particularly challenging low-resource task. In future work, we could prune the permutations of the revision chains. For example, we could add human-preferred heuristics, e.g., *coherence* before *fluency* and *meaning* before *clarity*, to remove certain chains.

Acknowledgments

This research was supported by the U.S. National Science Foundation award #2202347 and by a sup-

plement made in response to NSF DCL 24-093. It also used Microsoft Azure credits at Cloudbank through allocation #CIS260792 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, which is supported by U.S. National Science Foundation grants #2138259, #2138286, #2138307, #2137603, and #2138296. The opinions expressed are those of the authors and do not represent the views of the institutes. The authors would like to thank the anonymous reviewers and the Pitt PETAL groups for their valuable feedback on this work.

References

Hannan Cao and Hwee Tou Ng. 2025. A constrained text revision agent via iterative planning and searching. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 26859–26882, Vienna, Austria. Association for Computational Linguistics.

Christopher Davis, Andrew Caines, Øistein E. Andersen, Shiva Taslimipoor, Helen Yannakoudakis, Zheng Yuan, Christopher Bryant, Marek Rei, and Paula Buttery. 2024. Prompting open-source and commercial language models for grammatical error correction of English learner text. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 11952–11967, Bangkok, Thailand. Association for Computational Linguistics.

Wanyu Du, Zae Myung Kim, Vipul Raheja, Dhruv Kumar, and Dongyeop Kang. 2022a. Read, revise, repeat: A system demonstration for human-in-the-loop iterative text revision. In *Proceedings of the First Workshop on Intelligent and Interactive Writing Assistants (In2Writing 2022)*, pages 96–108, Dublin, Ireland. Association for Computational Linguistics.

Wanyu Du, Vipul Raheja, Dhruv Kumar, Zae Myung Kim, Melissa Lopez, and Dongyeop Kang. 2022b. Understanding iterative revision from human-written text. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3573–3590, Dublin, Ireland. Association for Computational Linguistics.

Lester Faigley and Stephen Witte. 1981. Analyzing revision. *College Composition & Communication*, 32(4):400–414.

Allan Gollins and Dedre Gentner. 2016. A framework for a cognitive theory of writing. In *Cognitive processes in writing*, pages 51–72. Routledge.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Léane Jourdan, Florian Boudin, Richard Dufour, Nicolas Hernandez, and Akiko Aizawa. 2025. ParaRev : Building a dataset for scientific paragraph revision annotated with revision instruction. In *Proceedings of the First Workshop on Writing Aids at the Crossroads of AI, Cognitive Science and NLP (WRAICOGS 2025)*, pages 35–44, Abu Dhabi, UAE. International Committee on Computational Linguistics.

Léane Jourdan, Florian Boudin, Nicolas Hernandez, and Richard Dufour. 2024. CASIMIR: A corpus of scientific articles enhanced with multiple author-integrated revisions. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 2883–2892, Torino, Italia. ELRA and ICCL.

Zae Myung Kim, Wanyu Du, Vipul Raheja, Dhruv Kumar, and Dongyeop Kang. 2022. Improving iterative text revision by learning where to edit from other revision tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9986–9999, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626.

J Richard Landis and Gary G Koch. 1977. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174.

Xu Li, Chengkun Wei, and Wenzhi Chen. 2026. Editcot: A stepwise chain-of-thought reasoning framework for multi-intent text revision. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 25(3):1–25.

Xin Lin, Zhenya Huang, Zhiqiang Zhang, Jun Zhou, and Enhong Chen. 2025. Explore what llm does not know in complex question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24585–24594.

Zhexiong Liu. 2025. *Understanding and Generating Text Revisions With Intention-Adaptive Large Language Models*. Ph.D. thesis, University of Pittsburgh.

Zhexiong Liu and Diane Litman. 2025. Efficient layer-wise LLM fine-tuning for revision intention prediction. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 15319–15334, Suzhou, China. Association for Computational Linguistics.

Zhexiong Liu and Diane Litman. 2026. Intention-adaptive LLM fine-tuning for text revision generation. In *Findings of the Association for Computational Linguistics: EACL 2026*, pages 1263–1281, Rabat, Morocco. Association for Computational Linguistics.

Robert L. Logan IV, Alexandre Passos, Sameer Singh, and Ming-Wei Chang. 2022. FRUIT: Faithfully reflecting updated information in text. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3670–3686, Seattle, United States. Association for Computational Linguistics.

An Luo, Xun Xian, Jin Du, Fangqiao Tian, Ganghua Wang, Ming Zhong, Shengchun Zhao, Xuan Bi, Zirui Liu, Jiawei Zhou, Jayanth Srinivasa, Ashish Kundu, Charles Fleming, Mingyi Hong, and Jie Ding. 2025. AssistedDS: Benchmarking how external domain knowledge assists LLMs in automated data science. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 18017–18060, Suzhou, China. Association for Computational Linguistics.

Mounica Maddela, Yao Dou, David Heineman, and Wei Xu. 2023. LENS: A learnable evaluation metric for text simplification. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16383–16408, Toronto, Canada. Association for Computational Linguistics.

Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. 2015. Ground truth for grammatical error correction metrics. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 588–593, Beijing, China. Association for Computational Linguistics.

Vipul Raheja, Dhruv Kumar, Ryan Koo, and Dongyeop Kang. 2023. CoEdIT: Text editing by task-specific instruction tuning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5274–5291, Singapore. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

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.

Gabriella Skitalinskaya and Henning Wachsmuth. 2023. To revise or not to revise: Learning to detect improvable claims for argumentative writing support. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15799–15816, Toronto, Canada. Association for Computational Linguistics.

Nancy Sommers. 1980. Revision strategies of student writers and experienced adult writers. *College Composition & Communication*, 31(4):378–388.

Jiankai Sun, Chuanyang Zheng, Enze Xie, Zhengying Liu, Ruihang Chu, Jianing Qiu, Jiaqi Xu, Mingyu Ding, Hongyang Li, Mengzhe Geng, and 1 others. 2025. A survey of reasoning with foundation models: Concepts, methodologies, and outlook. *ACM Computing Surveys*, 57(11):1–43.

Zhengxiang Wang, Veronika Makarova, Zhi Li, Jordan Kodner, and Owen Rambow. 2025. LLMs can perform multi-dimensional analytic writing assessments: A case study of L2 graduate-level academic English writing. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8637–8663, Vienna, Austria. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Haopeng Zhang, Philip S Yu, and Jiawei Zhang. 2025. A systematic survey of text summarization: From statistical methods to large language models. *ACM Computing Surveys*, 57(11):1–41.

Timon Ziegenbein, Gabriella Skitalinskaya, Alireza Bayat Makou, and Henning Wachsmuth. 2024. LLM-based rewriting of inappropriate argumentation using reinforcement learning from machine feedback. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4455–4476, Bangkok, Thailand. Association for Computational Linguistics.

A Appendix

A.1 IteraTeR Dataset

The dataset IteraTeR (Du et al., 2022b) contains revisions annotated from Wikipedia, arXiv, and news articles. The annotation taxonomy includes:

- **Fluency**: the revision to fix grammatical errors in the text.
- **Coherence**: the revision to make the text more cohesive, logically linked, and consistent.
- **Clarity**: the revision to make the text more formal, concise, readable, and understandable.
- **Style**: the revision to convey the writer's writing preferences, including emotions, tone, voice, etc.

Split	IteraTeR-multi			ParaRev-multi			Total
	Add	Delete	Modify	Add	Delete	Modify	
Training	475	404	1,739	22	29	637	3,306
Evaluation	110	113	450	6	10	252	941
Total	585	517	2,189	28	39	889	4,247

Table 3: Data splits and the number of revisions in IteraTeR-multi and ParaRev-multi datasets.

# of intentions	# of chains	IteraTeR-multi			ParaRev-multi		
		Train	Eval	Total	Train	Eval	Total
2 intentions	2!=2	93	9	102	149	64	213
3 intentions	3!=6	143	21	164	24	11	35
4 intentions	4!=24	86	19	105	-	-	-
5 intentions	5!=120	18	8	26	-	-	-
Total	152	340	57	397	173	75	248

Table 4: The number of intentions and the number of permutations of their revision chains in IteraTeR-multi and ParaRev-multi datasets.

- **Meaning**: the revision to update or add new information to the text.
- **Other**: the revisions that are not recognizable and do not belong to the above intentions.

A subset IteraTeR-multi that contains examples with at least one sentence-level revision involving multiple intentions is created. The data statistics and their intentions are included in Appendix Tables 3 and 4.

A.2 ParaRev Dataset

The dataset ParaRev (Jourdan et al., 2025) contains revisions annotated from scientific paragraphs extracted from CASIMIR (Jourdan et al., 2024). The annotation taxonomy includes:

- **Rewrite**: revisions involving minor word-choice changes, complete sentence rephrasing, or significant rephrasing affecting at least half of the paragraph.
- **Concision**: the same idea stated more briefly by removing unnecessary details.
- **Development**: the same idea expanded with additional details or definitions.
- **Content**: modification of content through the addition, substitution, or deletion of an idea or fact.
- **Unusable**: issues due to document processing errors, e.g., segmentation problems, misaligned paragraphs, or footnotes mixed with the text.

A subset ParaRev-multi that contains examples with multi-intent revisions is created. The data statistics and their intentions are included in Appendix Tables 3 and 4.

Baseline Holistic
Revise the following text to address its issues in sentence(s) {sentence ids}. Only focus on these sentence(s) and keep others unchanged. The issues include: {$\mathcal{I}$: the intention(s) to revise the text} $\mathcal{I}_1$: The revision needs to fix grammatical errors in the text (fluency). $\mathcal{I}_2$: The revision needs to make the text more formal, concise, readable, and understandable (clarity). $\mathcal{I}_3$: The revision needs to convey the writer's writing preferences, including emotions, tone, voice, etc. (style). $\mathcal{I}_4$: The revision needs to update or add new information to the text (meaning). $\mathcal{I}_5$: The revision needs to make the text more cohesive, logically linked, and consistent (coherence). Here are the examples of the revisions {revision exemplars}.

Table 5: The holistic prompt that addresses all the revision issues in a single pass. The prompt is based on the IteraTeR-multi dataset that includes five intentions ($\mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_5$). Note that the intention $\mathcal{I}_i$ included in the prompt is determined by each instance and can vary in different datasets.

A.3 Prompt Examples

Baseline Prompts. The holistic baseline prompt uses a simple instruction, as shown in Appendix Table 5, which provides multiple intentions to instruct LLMs to generate desirable revisions. Note that we ask LLMs to focus on specific revision sentences rather than every sentence in the document, as LLMs are inclined to make excessive edits across the document if no revision sentence id is specified in our pilot studies, which could compromise the validity and reliability of the analysis. In addition, we use chain-of-thought prompt (Wei et al., 2022) to instruct the LLMs to decompose the revision process into steps, as shown in Appendix Table 6, which drives LLMs to think and plan before generating revisions.

CoR Prompts. Our proposed CoR prompt includes three components: (1) the intention information to instruct the LLMs to focus on specific intentions while generating the text. (2) the revision order (revision chain) options, which specifically provide all the permutations of the given intention list. Note that the length of the intention list varies case by case, given that different intentions are included in each example. Appendix Table 4 shows the number of possible revision chains in different cases, where 3-intent revisions have 6 orders, and 5-intent revisions have 120 orders. This suggests the complexity of the revision itself and also various ways to revise the text. (3) instruction to

Baseline CoT

Revise the following text to address its issues in sentence(s) {sentence ids}. Only focus on these sentence(s) and keep others unchanged. The issues include: {$\mathcal{I}$: the intention(s) to revise the text}
$\mathcal{I}_1$: The revision needs to fix grammatical errors in the text (fluency).
$\mathcal{I}_2$: The revision needs to make the text more formal, concise, readable, and understandable (clarity).
$\mathcal{I}_3$: The revision needs to convey the writer's writing preferences, including emotions, tone, voice, etc. (style).
$\mathcal{I}_4$: The revision needs to update or add new information to the text (meaning).
$\mathcal{I}_5$: The revision needs to make the text more cohesive, logically linked, and consistent (coherence).

Here are the examples of the revisions {revision exemplars}.
You need to think step by step.

Table 6: The chain-of-thought prompt that addresses all the revision issues in a single pass. The prompt is based on the IteraTeR-multi dataset that includes five intentions ($\mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_5$). Note that the intention $\mathcal{I}_i$ included in the prompt is determined by each instance.

CoR with Fixed Order

Revise the following text to improve its `{label_set}` in sentences `{sent_id_set}`. Only focus on the sentences `{sent_id_set}`. Keep other sentences unchanged.

Revision Order. You need to revise in the following order: $\mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_5$

Revision Intentions.
$\mathcal{I}_1$: The revision needs to fix grammatical errors in the text (fluency).
$\mathcal{I}_2$: The revision needs to make the text more formal, concise, readable, and understandable (clarity).
$\mathcal{I}_3$: The revision needs to convey the writer's writing preferences, including emotions, tone, voice, etc. (style).
$\mathcal{I}_4$: The revision needs to update or add new information to the text (meaning).
$\mathcal{I}_5$: The revision needs to make the text more cohesive, logically linked, and consistent (coherence).

Instructions. Output the revised full text. Do not output your thinking.

Exemplars. Here are the examples of the revisions: `{revision exemplars}`.

Table 7: The chain-of-revision prompt that selects a fixed revision order over the intention set and produces the revised text. The prompt is based on the IteraTeR-multi dataset that includes five intentions ($\mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_5$). Note that the intention description will be different in the ParaRev-multi dataset. Also, the intention $\mathcal{I}_i$ included in the prompt is determined by each instance.

help LLMs select optimal orders, which is based on high-level revision heuristics that could minimize back-and-forth editing (Faigley and Witte, 1981) and avoid overwriting and vandalizing other revision aspects (Skitalinskaya and Wachsmuth, 2023).

CoR with Dynamic Orders

Revise the following text to improve its `{label_set}` in sentences `{sent_id_set}`. Only focus on the sentences `{sent_id_set}`. Keep other sentences unchanged.

Revision Order Options. You have the following revision order options:
Revision Order:
{`permutation of the given intentions` $\mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_5$
$\mathcal{I}_1$->$\mathcal{I}_2$->$\mathcal{I}_3$->$\mathcal{I}_4$->$\mathcal{I}_5$
$\mathcal{I}_1$->$\mathcal{I}_2$->$\mathcal{I}_3$->$\mathcal{I}_5$->$\mathcal{I}_4$
$\mathcal{I}_1$->$\mathcal{I}_2$->$\mathcal{I}_5$->$\mathcal{I}_3$->$\mathcal{I}_4$
...}

Revision Intentions.
$\mathcal{I}_1$: The revision needs to fix grammatical errors in the text (fluency).
$\mathcal{I}_2$: The revision needs to make the text more formal, concise, readable, and understandable (clarity).
$\mathcal{I}_3$: The revision needs to convey the writer's writing preferences, including emotions, tone, voice, etc. (style).
$\mathcal{I}_4$: The revision needs to update or add new information to the text (meaning).
$\mathcal{I}_5$: The revision needs to make the text more cohesive, logically linked, and consistent (coherence).

Instructions. Each revision order option will have a different revised text result. Evaluate which revision order option is the most promising to have an optimal revised text. The promising order could minimize back-and-forth editing and avoid overwriting and vandalizing other revision aspects. Analyze the revision orders in each option in detail, and think step by step. Output the revised full text. Do not output your thinking.

Exemplars. Here are the examples of the revisions: `{revision exemplars}`.

Table 8: The chain-of-revision prompt that selects an optimal revision order over the intention set and produces the revised text. The prompt is based on the IteraTeR-multi dataset that includes five intentions ($\mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_5$). Note that the intention description will be different in the ParaRev-multi dataset. Also, the intention $\mathcal{I}_i$ included in the prompt is determined by each instance.

We include a fixed order prompt based on Li et al. (2026) in Appendix Table 7 for comparison to the dynamic order prompt in Appendix Table 8 as well as visualized revision chains using the dynamic order prompt with intermediate results in Appendix Table 9.

LLM Judges. We include an LLM-as-a-revision-judge prompt to provide alternative revision metrics. Specifically, it evaluates a model revision by comparing the edits in the predicted revision with those in the gold revision, given the original text and the specified revision intention. The prompt first asks the judge to identify span-level edit alignment through three operations: *keep*, *delete*, and *add* based on SARI (Xu et al., 2016), where alignment depends on matching spans but

CoR with Dynamic Orders and Intermediate Revisions

Revise the following text to improve its {label_set} in sentences {sent_id_set}. Only focus on the sentences {sent_id_set}. Keep other sentences unchanged.

Revision Order Options. You have the following revision order options:
Revision Order:
{permutation of the given intentions $\mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_5$
$\mathcal{I}_1$->$\mathcal{I}_2$->$\mathcal{I}_3$->$\mathcal{I}_4$->$\mathcal{I}_5$
$\mathcal{I}_1$->$\mathcal{I}_2$->$\mathcal{I}_3$->$\mathcal{I}_5$->$\mathcal{I}_4$
$\mathcal{I}_1$->$\mathcal{I}_2$->$\mathcal{I}_5$->$\mathcal{I}_3$->$\mathcal{I}_4$
...}

Revision Intentions.
$\mathcal{I}_1$: The revision needs to fix grammatical errors in the text (fluency).
$\mathcal{I}_2$: The revision needs to make the text more formal, concise, readable, and understandable (clarity).
$\mathcal{I}_3$: The revision needs to convey the writer's writing preferences, including emotions, tone, voice, etc. (style).
$\mathcal{I}_4$: The revision needs to update or add new information to the text (meaning).
$\mathcal{I}_5$: The revision needs to make the text more cohesive, logically linked, and consistent (coherence).

Instructions. Each revision order option will have a different revised text result. Evaluate which revision order option is the most promising to have an optimal revised text. The promising order could minimize back-and-forth editing and avoid overwriting and vandalizing other revision aspects. Analyze the revision orders in each option in detail, and think step by step. Output the revised full text. Do not output your thinking. Use the following JSON format to output your revised text.
{ "order": <the revision order>
"intermediate_revision": <the intermediate revisions for each step in the revision order>,
"output": <revised full text>, }

Exemplars. Here are the examples of the revisions: {revision exemplars}.

Table 9: The chain-of-revision prompt that selects an optimal revision order and generates intermediate revisions and final revised text. The prompt is based on the IteraTeR-multi dataset that includes five intentions ($\mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_5$). Note that the intention description will be different in the ParaRev-multi dataset. Also, the intention $\mathcal{I}_i$ included in the prompt is determined by each instance.

also on whether the edits fulfill the same intended purpose. Note that we do not specify the modifications revisions as additions and deletions cover those edits. Afterward, it scores the aligned edits along two independent dimensions: *surface*, covering linguistic quality such as word choice, grammar, syntax, expression, and tone, and *content*, covering facts, claims, meaning, ideas, and information. Finally, the judge assigns separate 0-4 scores with brief rationales in a strict format. We normalize the scores in our reports. The *surface* and *content* judge prompt is shown in Appendix Table 10. In addition, we use another LLM judge for a pairwise comparison between two revision methods.

LLM-as-a-Revision-Judge (Surface and Content)

You are an expert revision evaluator. Four texts are given: *revision intention* (the purpose or motivation of the revision), *original* (the source text before revision), *gold revision* (the human reference revision of the original), and *predicted revision* (the model's predicted revision of the original).

Task. Judge how closely the *predicted revision* matches the *gold revision* based on the given *revision intention*. The comparison is edit-vs-edit: compare the changes made by the *gold revision* to the *original* with the changes made by the *predicted revision* to the *original*. Do not compare the revisions as standalone texts only; compare their intended edits relative to the *original* and *gold revision*.

Span-level Revision Operations. First, identify alignment through the following operations: KEEP, text from the *original* that the revision retains; DELETE, text from the *original* that the revision removes; and ADD, new text that the revision introduces. A predicted revision aligns with the gold revision when both revisions perform the same operation on the same span for the same dimension-relevant reason. Reward matches; penalize misses, where the gold edits are ignored, and over-revisions, where the prediction edits spans the gold did not edit. Alignment should consider meaning and intention, not only n-gram or string overlap.

Dimensions. Score the quality of the identified spans along two independent dimensions. SURFACE measures linguistic quality, including word choice, grammar, syntax, expression, and tone. CONTENT measures content quality, including facts, claims, meaning, ideas, and information.

Shared 0–4 Scale.
4: Near-perfect. Almost all predicted edits match the gold edits; divergences are trivial ($\sim \geq 80\%$ alignment).
3: Strong. Most predicted edits match the gold edits, with a few extra or missing edits ($\sim$ 60–79% alignment).
2: Moderate. About half of the predicted edits match the gold edits, with about half missing or extra edits ($\sim$ 40–59% alignment).
1: Weak. Most predicted edits do not match the gold edits, though a few overlaps exist ($\sim$ 20–39% alignment).
0: Misaligned. Most predicted edits do not match the gold edits, and the divergence is significant ($< 20\%$ alignment).

Inputs. *revision intention*: {revision_intention}; *original*: {original_text}; *gold revision*: {gold_revised_text}; *predicted revision*: {predicted_revised_text}.

Output (strict JSON).
{"surface":{"score":0-4,"rationale":"..."},
"content":{"score":0-4,"rationale":"..."}}

Rules. Score the SURFACE and CONTENT dimensions independently. Do not let one dimension's score influence the other. Each dimension's score is governed by alignment across KEEP, DELETE, and ADD operations, not by only one operation type. When in doubt between two scores, choose the lower one. Do not output anything outside the valid JSON object.

Table 10: The LLM-as-a-revision-judge prompt for scoring a predicted revision against a gold revision along *surface* and *content* dimensions.

LLM-as-a-Revision-Judge (Pairwise Comparison)

You are an expert revision evaluator. Five texts are given: *revision intention* (the purpose or motivation of the revision), *original* (source text before revision), *gold revision* (human reference revision), candidate A predicted revision (model A revision), and candidate B predicted revision (model B revision).

Task. Compare candidate A predicted revision and candidate B predicted revision, and decide which one better matches the *gold revision* based on the given *revision intention*. The comparison is edit-vs-edit: compare the changes made by *gold revision* to *original* with the changes made by each candidate to *original*. Do not compare the candidates as standalone texts only; compare their intended edits relative to *original* and *gold revision*.

Labels.
`"A_win"`: Candidate A predicted revision is clearly better aligned with the *gold revision*.
`"B_win"`: Candidate B predicted revision is clearly better aligned with the *gold revision*.
`"tie"`: both candidates are similarly aligned, or the difference is too small to confidently choose a winner.

Important. A candidate wins only if it is meaningfully better than the other candidate. Use `"tie"` when both candidates make similar edits, have similar errors, or differ only trivially. Do not reward a candidate for making a good revision if it does not match the *gold revision*'s intended edit.

Inputs. *revision intention*: `{revision_intention}`; *original*: `{original_text}`; *gold revision*: `{gold_revised_text}`; candidate A predicted revision: `{candidate_a_text}`; candidate B predicted revision: `{candidate_b_text}`.

Output (strict JSON).
```
{"result":{"winner":"<A_win | B_win | tie>",
  "rationale":"<1-2 sentences explaining which
candidate better matches the gold edits>"}}
```

Rules. Judge the candidates based on alignment with the *gold revision* under the given *revision intention*. Use `"tie"` if neither candidate is clearly better. Do not infer which candidate is the proposed method or baseline. Do not let candidate order influence the judgment. Output only the valid JSON object.

Table 11: The LLM-as-a-revision-judge prompt for pairwise comparison between two candidate revisions.

The judge decides which method wins or loses, or both methods are tied, based on alignment with the gold revision under the given revision intention, since intentions are important perspectives for the generated revisions (Liu, 2025; Liu and Litman, 2026). The pairwise judge prompt is described in Appendix Table 11.

A.4 Revision Analysis

Metric Comparison. We compare the different revision metrics in Figure 5. It shows that the human revision paraphrases the text "important for choosing" to "of primary importance for choosing",

Existing Revision Metrics: SARI=28.87, GLEU=0, Update-R=30.00
LLM-as-a-Revision-Judge Metrics: Surface=75%, Content=75%

Human Revision [Content and Rewrite]:
Fig. 3a shows that the first two components take about 90% of the explained variance. Fig. 3b shows that these components include only τ 1 , avg degree, and modularity. The fact that n is not used means that the size of the graph as well as the density are not ~~important~~ of primary importance for choosing the best measure. ~~The fact that~~ So is not τ 2 ~~is not used means~~ measuring that the ~~difference in sizes~~ diversity of ~~clusters is not important, too~~ cluster sizes.

LLM with CoR Revision [Content and Rewrite]:
Fig. 3a shows that the first two components take about 90% of the explained variance. Fig. 3b shows that these components include only τ 1 , avg degree, and modularity. ~~The fact~~ This suggests that ~~n is not used means~~ that neither the ~~size~~ number of ~~the~~ nodes (n) nor overall graph ~~as well as~~ density meaningfully influences the ~~density are not important for choosing~~ selection of the best measure. The fact that τ 2 is not used means that the difference in sizes of clusters is not important, too.

Figure 5: The comparison between existing revision metrics (SARI, GLEU, Update-R) and our proposed LLM-as-revision-judge scoring metrics. The revisions are generated by GPT-5.2 with CoR and zero-shot learning on the ParaRev-multi dataset.

while the LLM revises it to "meaningfully influences the selection of". Both revisions are successful with respect to the *rewrite* intention. Although the LLM-generated revisions differ from the human revisions in wording, the revised texts are semantically close. However, conventional revision metrics use n-grams for evaluating the matches, yielding lower scores, e.g., SARI (28.87), than those reported for GPT-5.2 in Table 1. In contrast, our proposed LLM-as-a-revision-judge metrics score 75% for both the *surface* and *content* revisions, higher than the LLM judge scores in Table 1, suggesting relatively high-level overlaps on semantics and intentions. This example demonstrates the advantages of our proposed *surface* and *content* metrics in assessing complex revisions.

Retriever Examples. Although a semantic retriever can retrieve examples with similar semantic meaning, these examples may involve different revision intentions and therefore provide limited guidance for the target revision. In other words, the quantity of the exemplar does not compensate for the quality of the exemplars, and more shots would have a higher chance of including unhelpful examples with low similarity scores, as shown in Appendix Figure 6. In contrast, an intention-adaptive retriever retrieves examples with more closely matched revision purposes, driving LLMs to generate useful outputs for intention-aware revision. While helpful, an intention-adaptive retriever may sometimes retrieve examples that are semantically less similar. Therefore, further study on the

Intention-Retriever (3-shot):	**Semantic-Retriever (3-shot):**
Intentions: **Fluency, Clarity** **#1 Retrieved Intention: clarity** (similarity score: 0.90) In this ~~work~~ paper , we pre-train a Gigawor-based bilingual language model (GigaBERT) to study these two distant languages as well as zero-short transfer learningon various IE tasks. **# 2 Retrieved Intention: clarity** (similarity score: 0.87) ~~Arabic is a morphological rich language, posing many challenges for information extraction (IE) tasks, including Named Entity Recognition (NER) , Part-of-Speech tagging (POS), Argument Role Labeling (ARL), and Relation Extraction (RE). A few multilingual~~ Multilingual pre-trained models have been proposed and show good performance for Arabic, however, most experiment results are reported on language understanding tasks, such as natural language inference, question answering and sentiment analysis. **#3 Retrieved Intention: clarity** (similarity score: 0.60) Our GigaBERT outperforms multilingual BERT and and monolingual AraBERT on these tasks, in both supervised and zero-shot ~~learning~~ transfer settings. ~~footnote~~ We have made our pre-trained models publicly available at URL.	**Intentions**: **Fluency, Clarity** **#1 Retrieved Intention: clarity** (similarity score: 0.90) In this ~~work~~ paper , we pre-train a Gigawor-based bilingual language model (GigaBERT) to study these two distant languages as well as zero-short transfer learningon various IE tasks. **# 2 Retrieved Intention: clarity** (similarity score: 0.87) ~~Arabic is a morphological rich language, posing many challenges for information extraction (IE) tasks, including Named Entity Recognition (NER) , Part-of-Speech tagging (POS), Argument Role Labeling (ARL), and Relation Extraction (RE). A few multilingual~~ Multilingual pre-trained models have been proposed and show good performance for Arabic, however, most experiment results are reported on language understanding tasks, such as natural language inference, question answering and sentiment analysis. **#3 Retrieved Intention: Meaning** (similarity score: 0.83) In this work , we pre-train a ~~Gigaword-based~~ customized bilingual ~~language model (GigaBERT) to~~ BERT , dubbed GigaBERT, that is designed specifically for Arabic NLP and English-to-Arabic zero-shot transfer learning. We study ~~these two distant languages as well as~~ GigaBERT's effectiveness on zero-short transfer ~~learningon various~~ across four IE tasks~~.~~ , named entity recognition, part-of-speech tagging, argument role labeling, and relation extraction. Our ~~GigaBERT~~ best model significantly outperforms ~~multilingual BERT~~ mBERT, XLM-RoBERTa, and and ~~monolingual~~ AraBERT ~~on these tasks~~ (Antoun et al., 2020), in both the supervised and zero-shot learning settings. footnote We have made our pre-trained models publicly available at URL

Figure 6: The Intention-Retriever retrieved unhelpful examples with low similarity scores, while the Semantic-Retriever includes an irrelevant intention example in the IteraTeR-multi dataset.

Step1: Clarity	**Step2: Meaning**	**Step3: Coherence**
Vision, ~~as~~ a central component of human perception, plays a fundamental role in shaping natural language~~.~~ . To better understand how text models ~~are connected~~ relate to ~~our~~ visual perception~~s~~. we propose a method for ~~examining~~ ~~the~~ measuring similarities between neural representations ~~extracted~~ derived from words in text and objects in images~~.~~. Our approach uses a lightweight probing model that learns to map language representations of concrete words into the visual domain.	Vision, a central component of human perception, plays a fundamental role in shaping natural language~~.~~ and conceptual meaning. To better understand how text models relate to visual perception, we propose a method for ~~measuring similarities~~ quantifying the alignment between neural representations derived from words in text and objects in images. Our approach uses a lightweight probing model that learns to map language representations of concrete words into the visual domain.	Vision, a central component of human perception, plays a fundamental role in shaping natural language and conceptual meaning. To better understand how text models relate to visual perception, we propose a method for quantifying the alignment between neural representations derived from words in text and objects in images. ~~Our approach uses~~ Specifically, we use a lightweight probing model that learns to map language representations of concrete words into the visual domain.
Step1: Meaning	**Step2: Coherence**	**Step3: Clarity**
Vision, as a central component of human perception, plays a fundamental role in shaping natural language~~.~~ . To better understand how text models ~~are connected~~ relate to our human visual perception~~s~~. perception, we propose a method for ~~examining~~ measuring the ~~similarities~~ alignment between neural representations extracted from words in text and objects~~.~~ representations in images~~.~~. Specifically, we examine whether contextualized embeddings for concrete nouns encode information that can be linearly recovered to match visual features.	Vision, as a central component of human perception, plays a fundamental role in shaping natural language. To better understand how text models relate to human visual perception, we propose a method for measuring the alignment between neural representations extracted from words in text and object representations in images. Specifically, we ~~examine~~ test whether contextualized embeddings for concrete nouns encode information that can be linearly recovered to match visual features.	Vision, ~~as~~ a central component of human perception, plays a fundamental role in shaping natural language. To better understand how text models relate to human visual perception, we propose a method ~~for measuring the~~ to measure alignment between neural representations extracted from words in text and object representations in images. Specifically, we test whether contextualized embeddings for concrete nouns encode information that can be linearly recovered to match visual features.

Figure 7: The comparison between revisions using *clarity-meaning-coherence* and *meaning-coherence-clarity* chains in the IteraTeR-multi dataset.

helpful retrieving examples would be future work.

Revision Chains. We analyze the impact of different revision chains on the same text in Appendix Figure 7, using CoR prompt with intermediate revision outputs[5], as described in Appendix Table 9. Several intermediate revision results indicate consistency in *clarity* and *meaning*, as the revisions are identical even though they occur in different chains, e.g., "as" was deleted in both *clarity* revisions and "similarities" was revised to "alignment" in both *meaning* revisions in the two chains. Also, it shows that conducting *meaning* revisions might overwrite revisions made in *clarity*, e.g., the edit from "examining the similarities" to "measuring similarities" in the *clarity* step in the revision

[5]While the CoR prompt with intermediate revision outputs in Appendix Table 9 is slightly different from the one without intermediate outputs in Appendix Table 8, their performance are close, e.g., GPT-5.2 with CoR achieving 50.20 Average score on IteraTeR-multi using the former prompt, compared to reported 50.66 in Table 1 with the latter one.

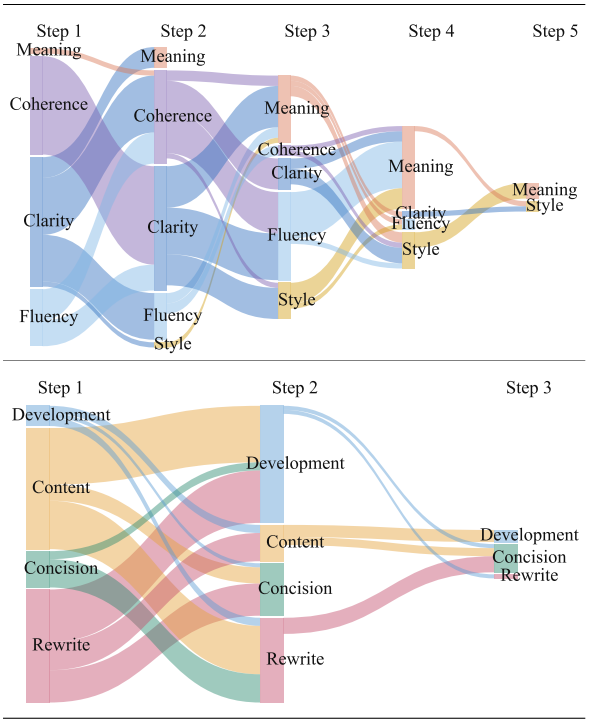


Figure 8: The revision steps (chains) generated by GPT-5.2 with zero-shot learning and CoR on the IteraTeR-multi (upper) and ParaRev-multi (lower) datasets.

chain *clarity-meaning-coherence* was later revised to "quantifying the alignment" in the *meaning* step. Again, conducting a *coherence* revision after the *meaning* would also likely alter the content, e.g., the text "our approach uses a lightweight probing" was modified to "we use a lightweight probing". These results suggest that revisions are dynamic, sometimes back-and-forth, and the revision order is important for generating successful revisions. Moreover, we visualize the revision chains with different chain lengths. Appendix Figure 8 shows that *clarity* and *coherence* are mostly the initial steps for IteraTeR-multi, and *content* and *rewrite* are preferred for ParaRev-multi[6]. Although these dynamic chains adapt to different inputs, they become increasingly complex as they grow longer; thus, developing automated methods to prune redundant steps is a promising direction for future work.

Revision Errors. While our proposed CoR models the dynamics of iterative revision, typical revision errors are observed in the revision chains. Appendix Figure 9 illustrates three types of revision errors: hallucination, mismatch, and excessiveness. For example, the hallucinated revision, which aims to revise based on the *meaning* intention, added

[6]The chain orders may differ across runs, as LLMs may select different chains to achieve similar performance.

Hallucination [Meaning]

In sequence-to-sequence learning, the decoder relies on the attention mechanism to efficiently extract information from the encoder~~s~~, and recent studies show that attending to multiple encoder layers can improve both representation utilization and translation quality. While it is common practice to draw information from only the last encoder layer, recent work has proposed to use representations from different encoder layers for diversified levels of information.

Mismatch [Coherence]

This paper presents a new method, SOLOIST , which uses transfer learning to efficiently build task-oriented dialog systems at scale. We parameterize ~~a~~ the dialog system ~~using~~ with a Transformer-based ~~auto-regressive~~ autoregressive language model~~, which subsumes~~ that unifies different dialog modules (e.g., state ~~tracker~~ tracking, dialog policy, and response ~~generator~~ generation) into a single neural model.

Excessiveness [Fluency]

~~In between~~ During the wait, Carter ~~was flying~~ flew with the UN for one day to Juba in ~~s~~South Sudan to ~~take photos of~~ photograph a barge~~, with~~ carrying food aid for the region~~.~~. But soon the situation changed again. Soon, however, the situation changed again.

Figure 9: The examples of typical revision error types with respect to intentions, including hallucinated, mismatched, and excessive revisions generated by GPT-5.2 with CoR and zero-shot learning on the IteraTeR-multi dataset.

a sentence fabricated from information in the following sentence, without supporting evidence. Although another example with mismatched revision slightly improved the text quality, it did not match the *coherence* intention and instead did a *clarity* revision. Also, the excessive revision for *fluency* made unnecessary edits, e.g., replacing "was flying" with "flew" and introducing redundant text, e.g., "the situation changed again"; such over-editing could reduce its alignment with real human revisions. Therefore, future work addressing these revision errors could improve the effectiveness of CoR and help to understand the iterativeness of revisions.