Automatic construction of meta-evaluation benchmarks for evaluation metrics

Abstract

The evaluation of long-form text, particularly in domains requiring factual accuracy, is increasingly reliant on automated metrics. However, the reliability of these metrics themselves is often assumed rather than rigorously tested, especially for use cases where long-form generations are expected as output. This paper addresses this gap by proposing a framework to evaluate the quality of reference-based evaluation metrics. We introduce a methodology that iteratively perturbs candidate texts to assess the sensitivity and discrimination power of reference-based text evaluation metrics. Our experiments, conducted on the ACI-Bench medical dataset, demonstrate the importance of evaluating evaluation metrics for long-form text, highlighting the need for robust validation methodologies.

1 Introduction

The surge in generative models has led to the proliferation of long-form text generation use cases across various domains. Evaluating the quality of these generated texts is crucial, which has subsequently driven the development of new evaluation metrics, which vary in their purpose, methodology, complexity, and sensitivity to the specific task they are developed for. While metrics are used to inform critical decisions, the errors that may propagate from the metrics themselves are often an afterthought. As the quality of such generated text improves, the gaps between the models that generated them shrink, making it harder for evaluation metrics to discern the subtleties. Prior work ((Hanna and Bojar, 2021), (Moramarco et al., 2022) has demonstrated that traditional evaluation metrics have low correlation with human evaluation judgments. More recently, the use of Large Language Models (LLMs) like GPT-4 (Zheng et al., 2023) as judges/evaluators, has become commonplace. While they exhibit superior correlation with human evaluation metrics, their failure modes, reliability, and ability to discern between high quality inputs is under-explored. In this work, we propose a methodology to assess the quality of reference-based evaluation metrics with respect to their sensitivity and ability to discriminate between texts on specific criteria or notions of quality.

Automatic evaluation of machine-generated text has long been a central challenge in natural language generation (NLG), often proving as difficult as the generation itself (Celikyilmaz et al., 2020). Driven by the success of large-scale pretrained language models (PLMs) (Devlin et al., 2019), recent research has focused on developing evaluation metrics based on these models (Zhang et al., 2020; Yuan et al., 2021; Pillutla et al., 2021). For instance, BERTScore (Zhang et al., 2020) calculates similarity scores between the contextualized embeddings of the candidate and the reference text. These PLM-based metrics have seemingly demonstrated superior correlations with human annotations across various tasks (Yuan et al., 2021), leading to their increasing adoption in practical applications.

However, it is crucial to acknowledge the inherent limitations of PLMs. These models can produce degenerate, repetitive text (Holtzman et al., 2020) and exhibit insensitivity to perturbations such as word order shuffling (Pham et al., 2021) and negation (Ettinger, 2020). These shortcomings, when combined with specific design choices in metric development, can render PLM-based evaluation metrics brittle and susceptible to manipulation. Consequently, careful consideration of these factors is essential for ensuring the reliability and robustness of evaluation methodologies.

More recently, the use of commercial Large Language Models (LLMs) like GPT-4o as "LLM-as-a-judge" evaluators for pairwise comparisons in LLM alignment tasks has become increasingly prevalent (Zheng et al., 2023). While they have shown a higher agreement to human preference compared

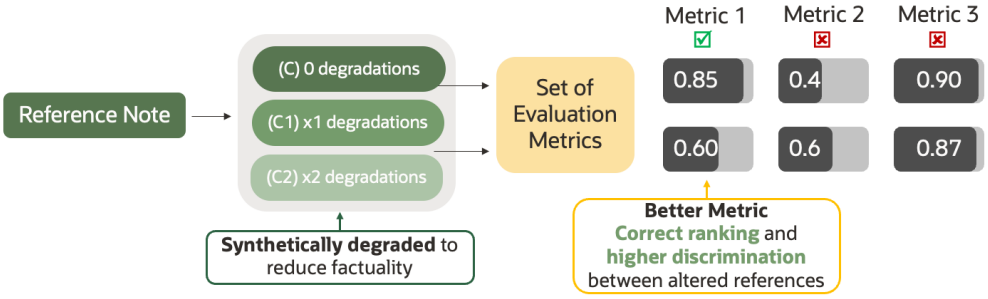

Figure 1: We construct benchmarks to test evaluation metrics by synthetically degrading the accuracy of a reference text (e.g., its factuality) and checking which evaluation metric best ranks the alterations

to PLM based methods, they suffer from various shortcomings; position bias, verbosity bias, and self-inclusion bias are some of them.

We introduce a framework that iteratively perturbs candidate texts, forming a hierarchy that allows us to evaluate the sensitivity and discrimination power of various reference-based evaluation metrics.

2 Related Work

2.1 Reference based evaluation metrics

Reference-based evaluation metrics have seen significant progress in recent years. Reference-based metrics evaluate the generated output by measuring its similarity to human-annotated reference texts. Metrics such as BERTScore (Zhang et al., 2019), BLEURT (Sellam et al., 2020), and others have demonstrated strong correlations with human judgments on short-form text. BERTScore leverages contextual embeddings from BERT to assess semantic similarity, while BLEURT is trained to predict human quality judgments. ROUGE (Lin, 2004) introduced methods to evaluate the quality of a generated summary with a reference summary.

Subsequent research discovered glaring shortcomings of each of these metrics; (Moramarco et al., 2022) show that in a note generation setting, the Levenshtein distance between a reference and generated output led to better results than using ROUGE. (Hanna and Bojar, 2021) demonstrate deficiences in BERTScore that make it less sensitive to smaller errors. This period coincides with the rise of LLM-as-a-judge paradigm (Zheng et al., 2023) as an alternative to traditional metrics. While they show stronger correlation to human judgments, various trivial failure modes have come to light. (Zheng et al., 2025) show that null models can achieve high win rates. (Ye et al., 2024) show that unrelated changes in prompts to the judge models can have significant outcomes in the judgment. In light of these findings, (Gu et al., 2025) recommend that well-calibrated judgements, uncertainty calibration and development of adversarially robust evaluation frameworks are steps to address such challenges. We position our work to better understand the ability of metrics to discriminate and calibrate an increasingly narrow regime where competing models score on the higher side with little variation in quality of their outputs.

2.2 Evaluation of metrics

(Ribeiro et al., 2020) introduced checklists to test for robustness of models under perturbations. Following a similar recipe, (Sai et al., 2021) leverage checklists but instead apply them to evaluation metrics. they design a total of 34 perturbation templates across different criteria such as Fluency, Informativeness, Negations, etc. They demonstrate that a majority of the metrics have poor correlations with human judgments across criteria. While our work builds on this foundation, the perturbation methods we use are not based on pre-fabricated templates. Additionally, we build an iterative perturbation setting which aims to not only study correlation of metrics to human judgment, but also quantitatively characterize their sensitivity to such perturbations.

(Liang et al., 2023) introduced HELM and with it included robustness tests for evaluation metrics. While they test for invariance (the stability of a model's predictions under small semantically preserved perturbations) and equivariance (the study of semantic altering perturbations on model behavior) their focus is on model behaviour and not the metrics themselves.

3 Methodology

Our methodology focuses on evaluating reference-based evaluation metrics. Given a reference text R (gold standard) and a candidate text C, we iteratively perturb C n times to generate a sequence of degraded candidates $C_1, C_2, ..., C_n$. Each perturbation introduces a controlled level of degradation, such that the quality of C_i is lower than C_{i-1}. Figure 1 visually represents this workflow.

Perturbations can target various criteria of text quality, including but not limited to grammatical correctness, brevity, simplicity, factual completeness, and factual correctness. To limit the scope of this study, we focus on factual correctness.

3.1 Perturbation Mechanism

We implement an unsupervised mechanism to perturb factual correctness. The mechanism splits the candidate text C into atomic facts, corrupts an increasing percentage of these facts using GPT-4o, and recombines them into a coherent long-form text to create the degraded candidates $C_1, C_2, ..., C_n$. Algorithms 1 illustrates this in detail.

3.2 Evaluation Metric Assessment

We evaluate the performance of evaluation metrics by examining two key properties:

1. **Monotonicity**: A desirable evaluation metric should assign scores that decrease as the candidate text is increasingly degraded. Mathematically, for an evaluation metric E_i, we expect:

$$E_i(R, C) > E_i(R, C_1) > ... > E_i(R, C_n)$$

2. **Discrimination Power**: A good evaluation metric should be able to discriminate between candidates with differing quality. If two evaluation metrics E_1 and E_2 are used to score the candidates, E_1 is deemed to have higher discrimination power if the differences between its scores are larger than those of E_2.

4 Experiments

We experiment with the ACI-Bench dataset (wai Yim et al., 2023). ACI-Bench is a medical dataset containing SOAP notes. A SOAP note is a structured document used by healthcare professionals to record patient information, comprising Subjective, Objective, Assessment, and Plan components. For each patient case, ACI-Bench provides a gold reference SOAP note.

We extract a candidate solution C by paraphrasing the gold reference answer R using GPT-4o. The degraded candidates $C_1, C_2, ..., C_n$ are obtained by the method described in Section 3

We evaluate the performance of a metric $E_i \in E$ on the pairs

$$E_i(R, C), E_i(R, C_1), E_i(R, C_2), ..., E_i(R, C_n)$$

4.1 Evaluation Metrics

4.1.1 BERTScore

BERTScore (Zhang et al., 2019) leverages contextual embeddings from BERT to assess the similarity between a candidate sentence and a reference sentence. It computes a fine-grained similarity score for each token in the candidate sentence with each token in the reference sentence, using contextual embeddings. These scores are then aggregated to produce an overall similarity score.

4.1.2 BLEURT

BLEURT (Sellam et al., 2020) is a learned evaluation metric that is trained to predict human quality judgments. It is designed to be robust and generalize well across different domains and tasks. BLEURT uses a large pre-trained language model and is fine-tuned on a dataset of human ratings, allowing it to capture subtle aspects of text quality that are often missed by other metrics.

4.1.3 AutoAIS

AutoAIS (Huang et al., 2022) is an automatic evaluation metric that focuses on assessing the factual accuracy and consistency of generated text. It uses a combination of information retrieval and natural language inference techniques to compare the generated text with a knowledge source or reference text. AutoAIS aims to provide a more reliable and interpretable evaluation of factuality than traditional metrics.

4.1.4 SBERTScore

SBERTScore is a modification of BERTScore that calculates sentence-level embeddings using Sentence-BERT (SBERT). While BERTScore computes token-level similarities, SBERTScore computes the embedding for the entire sentence and compares those embeddings. This change aims to capture the overall semantic similarity between sentences more effectively, potentially improving

Algorithm 1 Generate Degraded Candidates

Require: Reference Text R, Candidate Text C, Number of Iterations n, LLM model
Ensure: List of Degraded Candidate Texts $[C_1, C_2, ..., C_n]$

function GENERATEDEGRADEDCANDIDATES(C, n, LLM)
    $DegradedCandidates \leftarrow []$
    $CurrentCandidate \leftarrow C$
    $\alpha \leftarrow \frac{0.3}{n}$
    for $i \leftarrow 1$ to n **do**
        $Facts \leftarrow$ SPLITINTOATOMICFACTS($CurrentCandidate$, LLM)
        $CorruptedFacts \leftarrow$ CORRUPTFACTS($Facts, \alpha * count(Facts)$, LLM)
        $DegradedCandidate \leftarrow$ RECOMBINEFACTS($CorruptedFacts$, LLM)
        APPEND($DegradedCandidates, DegradedCandidate$)
        $CurrentCandidate \leftarrow DegradedCandidate$
        $\alpha \leftarrow \alpha + \frac{1}{n}$
    end for
    return $DegradedCandidates$
end function

function SPLITINTOATOMICFACTS($Text$, LLM)
    $Facts \leftarrow$ USELLMTOSPLIT($Text$, LLM)
    return Facts
end function

function CORRUPTFACTS($Facts, NumToCorrupt$, LLM)
    $CorruptedFacts \leftarrow Facts$ ▷ Copy to avoid modifying original
    $IndicesToCorrupt \leftarrow$ RANDOMLYSELECTINDICES($NumToCorrupt$, length($Facts$))
    for each $Index$ in $IndicesToCorrupt$ **do**
        $CorruptedFacts[Index] \leftarrow$ USELLMTOCORRUPT($Facts[Index]$, LLM)
    end for
    return $CorruptedFacts$
end function

function RECOMBINEFACTS($Facts$, LLM)
    $RecombinedText \leftarrow$ USELLMTORECOMBINE($Facts$, LLM) ▷ Use LLM to recombine
    return $RecombinedText$
end function

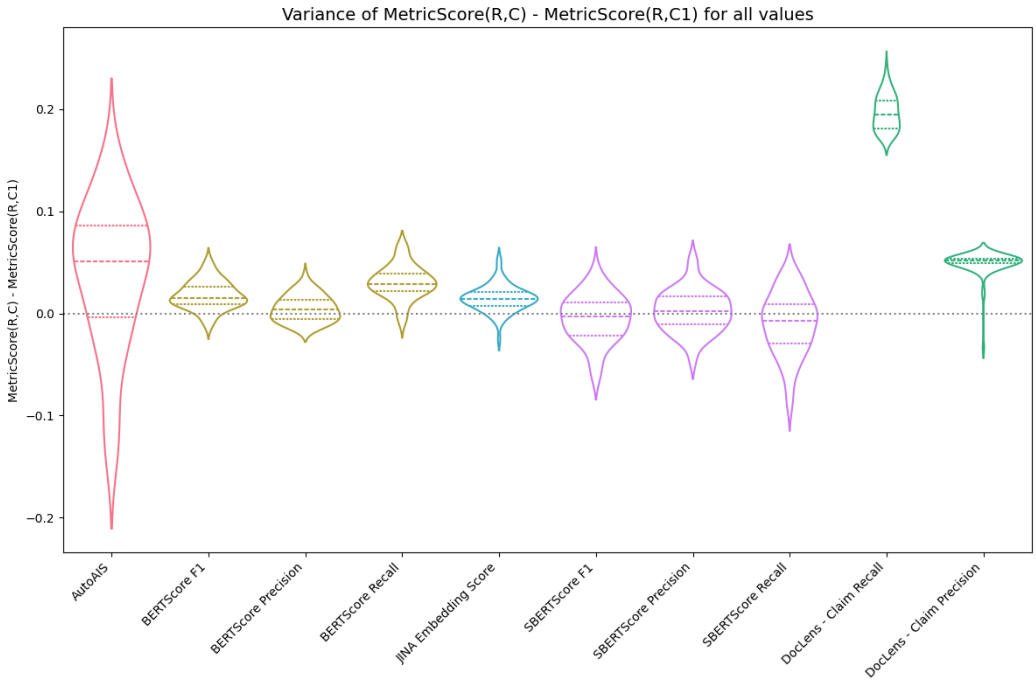

Figure 2: Each violin plots the value corresponding to $E_i(R, C) - E_i(R, C_1)$ along with the mean, and quartiles of the distribution denoted by dotted lines.

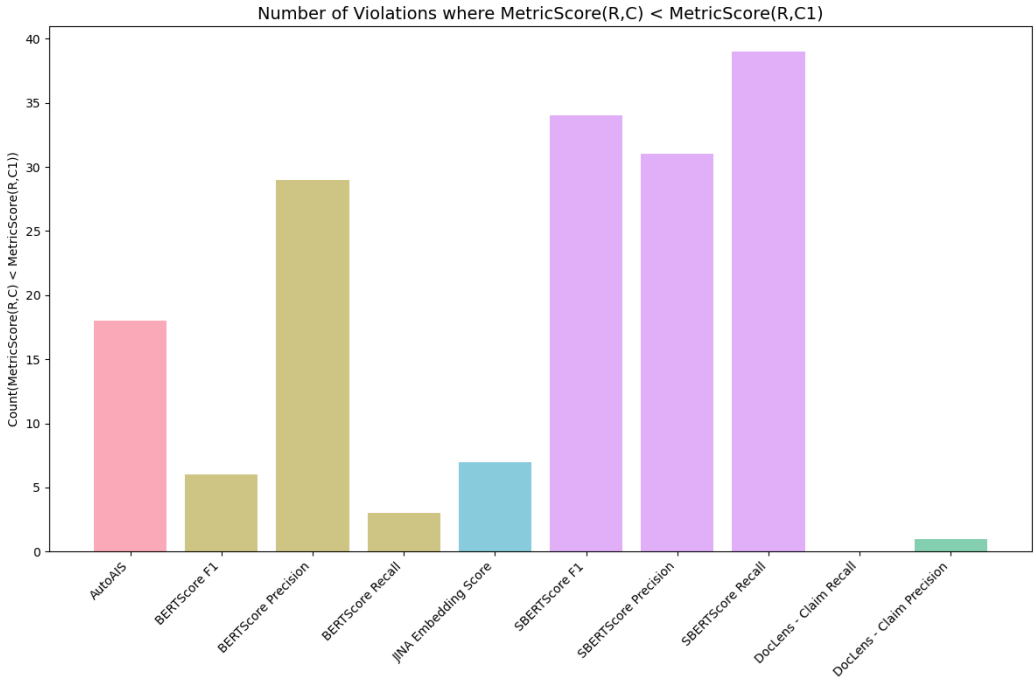

Figure 3: Each bar counts the instance where the Metric scored the degraded candidate C1 higher than candidate C, out of a total of 63 examples. DocLens has the lowest number of such violations.

the correlation with human judgments, particularly for longer texts where sentence-level coherence is important.

4.1.5 Jina Similarity Embeddings

Jina Similarity Embeddings (AI, 2024) provide a method for generating dense vector representations of text that are optimized for similarity search and comparison. These embeddings can be used for var-

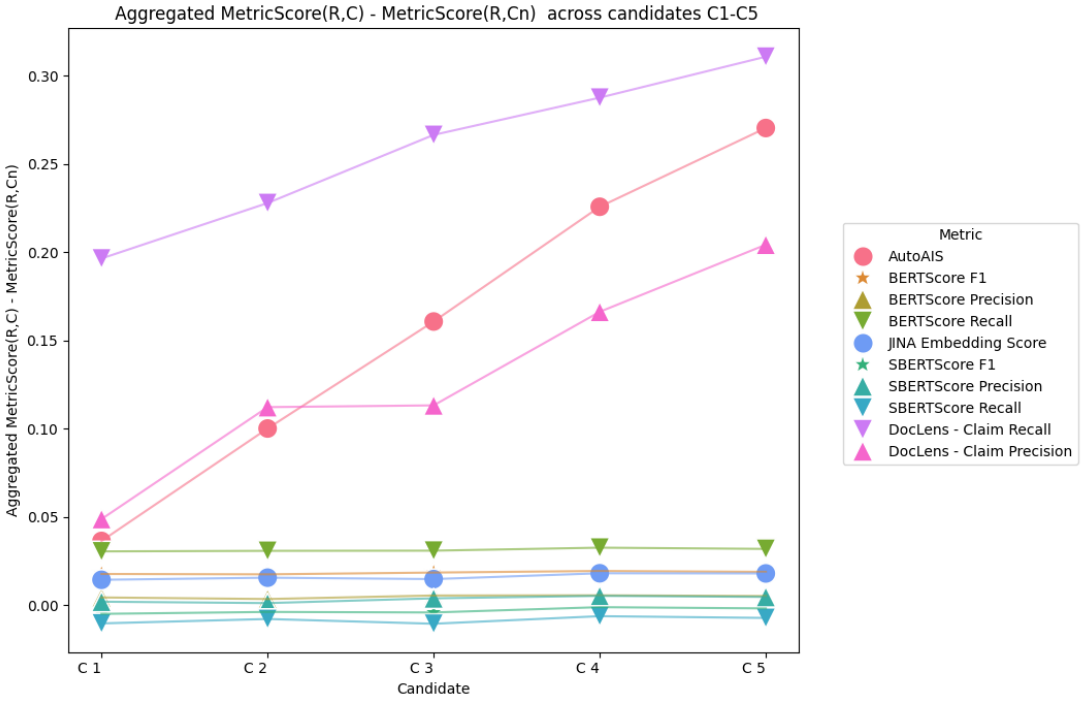

Figure 4: Each scatter in the plot corresponds to the aggregated $E_i(R, C) - E_i(R, C_n)$ across all samples for $n = 1...5$.

ious natural language processing tasks, including text evaluation. By comparing the embeddings of generated and reference texts, one can assess their semantic similarity. Jina embeddings are designed to be efficient and scalable, making them suitable for large-scale evaluation tasks.

4.1.6 DocLens

DocLens (Chen et al., 2023) is an evaluation framework designed to assess the quality of text generation tasks in the medical domain. The framework calculates the conciseness, and completeness of the generated text for clinical note generation. The metrics can be computed by various types of evaluators including instruction-following (both proprietary and open-source) and supervised entailment models. While multiple models can be executed with this framework, GPT4-o is widely used, so we replicate the same setting.

5 Results and Discussion

In figure 2 we plot the distribution of $E_i(R, C) - E_i(R, C_1)$ where C is the candidate answer and $C1$ is the first degraded candidate answer. We observe that DocLens (Chen et al., 2023) has the highest average score with no percentile below the $y = 0$ line. All instances below this line correspond to cases where the metric attributes a higher score to $E_i(R, C_1)$ than $E_i(R, C)$; we refer to these instances as a violation. The frequency of such violations are plotted in 3. This provides a measure of how often the metric can discern a higher quality text. The appendix plots these same graphs for $C_2, C_3...C_5$.

Figure 4 calculates the mean of $E(R, C) - E_i(R, C_i)$ over 63 samples for an evaluation metric E for a candidate i. As i increases, the number of degradations in C_i increase. As observed in the figure, $E(R, C) - E_i(R, C_i)$ stays flat for all metrics except AutoAIS and DocLens, indicating that they respect monotonicity.

In conclusion, by studying the distribution across all data samples of $E(R, C) - E_i(R, C_i)$ for each level C_i of degradation and comparing their means across all levels, one can quantiatively characterize both the monotonicity and the discrimination power of a metric E. On this basis, the right evaluation metric can be deployed by practitioners.

6 Human Annotation Study

We conduct a human annotation check by sampling 10 candidate pairs C and C_1. The annotator checks marks a pair as a valid degradation if it follows the following criteria.

- There is at least one incorrect fact in can-

Criteria	Count(Out of 10)
$Err_{minimum}$	10
$Err_{greater}$	10
$Err_{missing}$	9

Table 1: Human annotation on a randomly sampled set of 10 samples. There was one sample out of 10 where candidate C had a missing fact that was present in $C1$.Err$_{missing}$

didate C_1 with respect to the reference R - $Err_{minimum}$

- The number of incorrect facts in candidate C if any are lesser than in $C1$ - $Err_{greater}$

Table 1 presents the results, showing that all 10 samples qualify as valid degradations.

The annotator also flags cases where the number of missing facts in candidate C if any are lesser than the facts missing in $C1$ - $Err_{missing}$ with respect to reference R. While this count is ideally expected to always be 0 as a fact cannot appear in $C1$ that did not exist in candidate C, we observe one instance where this happens. We speculate that the data sample being present in the LLM's training data could possibly explain this behavior.

7 Conclusion and Future Work

This paper introduces a framework to evaluate the quality of reference-based evaluation metrics for long-form text. Our methodology, based on iterative perturbation of factuality, allows us to assess the sensitivity and discrimination power of various metrics. The experiments on ACI-Bench dataset highlight the importance of evaluating evaluation metrics in the context of long-form text.

While in this work we experiment with degradation of factuality, it can be based on other dimensions such as grammatical correctness, cohesiveness, simplicity or others. One can also reduce the fraction of errors introduced in the degraded candidates to better compare evaluation metrics that may have similar capabilities and quality. However, a reduced fraction of errors may have the effect of increased false positives in the benchmark (when a degraded candidate C_i is as good C), warranting a human in the loop-based verification, to ensure stringent quality.

In the future, for every new evaluation metric it can be run on the benchmark to check how it compares against previous metrics in the suite, better guiding development of evaluation metrics.

References

Jina AI. 2024. Jina embeddings: A simple, efficient, and scalable approach to similarity search. *arXiv preprint arXiv:2409.10173*.

Asli Celikyilmaz, Stephen Clark, Jianfeng Gao, Bill Grammer, Yingce Li, Lidong Lu, Wei Wang, and Wen-tau Yih. 2020. Evaluation of text generation: From bleu to bertscore. In *arXiv preprint arXiv:2006.14799*.

Zhiyuan Chen, Heyan Zhang, and Yang Liu. 2023. Doclens: Evaluating long-form document quality by decomposing document-level coherence. *arXiv preprint arXiv:2311.09581*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Allyson Ettinger. 2020. What bert is not: Lessons from natural language inference. In *Transactions of the Association for Computational Linguistics*, volume 8, pages 345–360.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo. 2025. A survey on llm-as-a-judge.

Michael Hanna and Ondřej Bojar. 2021. A fine-grained analysis of BERTScore. In *Proceedings of the Sixth Conference on Machine Translation*, pages 507–517, Online. Association for Computational Linguistics.

Ari Holtzman, Jan Groeneveld, Yejin Choi, and Luke Zettlemoyer. 2020. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Po-Nien Huang, Yu-Siang Chen, and Hung-yi Lee. 2022. Autoais: Automatic assessment of information seeking dialogues. *arXiv preprint arXiv:2212.08037*.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher Ré, Diana Acosta-Navas, Drew A. Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue Wang, Keshav Santhanam, Laurel Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. 2023. Holistic evaluation of language models.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Francesco Moramarco, Alex Papadopoulos Korfiatis, Mark Perera, Damir Juric, Jack Flann, Ehud Reiter, Anya Belz, and Aleksandar Savkov. 2022. Human evaluation and correlation with automatic metrics in consultation note generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5739–5754, Dublin, Ireland. Association for Computational Linguistics.

Hieu Pham, Trung Tran, and Long Nguyen. 2021. Towards robust evaluation of text-to-text models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6811–6822.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, Chandra Bhagavatula, and Yejin Choi. 2021. Robust evaluation of text generation without human references. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6467–6482.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

Ananya B. Sai, Tanay Dixit, Dev Yashpal Sheth, Sreyas Mohan, and Mitesh M. Khapra. 2021. Perturbation checklists for evaluating nlg evaluation metrics.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. Bleurt: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892.

Wen wai Yim, Yujuan Fu, Asma Ben Abacha, Neal Snider, Thomas Lin, and Meliha Yetisgen. 2023. Aci-bench: a novel ambient clinical intelligence dataset for benchmarking automatic visit note generation.

Jiayi Ye, Yanbo Wang, Yue Huang, Dongping Chen, Qihui Zhang, Nuno Moniz, Tian Gao, Werner Geyer, Chao Huang, Pin-Yu Chen, Nitesh V Chawla, and Xiangliang Zhang. 2024. Justice or prejudice? quantifying biases in llm-as-a-judge.

Wei Yuan, Graham Neubig, and Pengfei Liu. 2021. Revisiting automatic evaluation of machine translation with effective metrics. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5775–5787.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Mikel Artetxe. 2019. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Mikel Artetxe. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena.

Xiaosen Zheng, Tianyu Pang, Chao Du, Qian Liu, Jing Jiang, and Min Lin. 2025. Cheating automatic llm benchmarks: Null models achieve high win rates.

A Example Appendix

A.1 Prompts

Prompt (Fact splitting)
You are a helpful bot, your task is to Break the user text into the smallest independent atomic facts possible. Do not number them. Print them out line by line. Each fact needs to be simple.

Here is an example

INPUT: The Java Development Kit (JDK) is an implementation of either one of the Java Platform , Standard Edition , Java Platform , Enterprise Edition , or Java Platform , Micro Edition platforms released by Oracle Corporation in the form of a binary product aimed at Java developers on Solaris , Linux , macOS or Windows .

OUTPUT: The Java Development Kit (JDK) exists.
The JDK is an implementation.
The JDK implements the Java Platform.
The Java Platform has three editions: Standard Edition, Enterprise Edition, and Micro Edition.
The JDK is released by Oracle Corporation.
The JDK is released as a binary product.
The binary product is aimed at Java developers.
Java developers use Solaris.
Java developers use Linux.
Java developers use macOS.
Java developers use Windows.
INPUT: <TEXT>

Figure 5: GPT-4o prompts for splitting facts

Prompt (Fact corruption)
You are a helpful bot, that is good at changing or modifying the facts in the user input provided to you. There are to be no follow up questions. Just provide the answer.
<TEXT>

Figure 6: GPT-4o prompts for corrupting facts

Prompt (Fact recombination)
Use the following facts provided in the user input to generate a coherent paragraph. Do not add in your response information that does not exist in the input provided.
<TEXT>

Figure 7: GPT-4o prompts for recombining facts

Prompt (Paraphrasing)
You are a helpful assistant who is great at paraphrasing text. Take the text provided to you and paraphrase it strongly but correctly.
Do not ask follow up questions.
Do not ask to perform any other actions.
Do not copy from the original text as is.
Do not try to continue the conversation.
<TEXT>"

Figure 8: GPT-4o prompts for generating the paraphrased version of a text

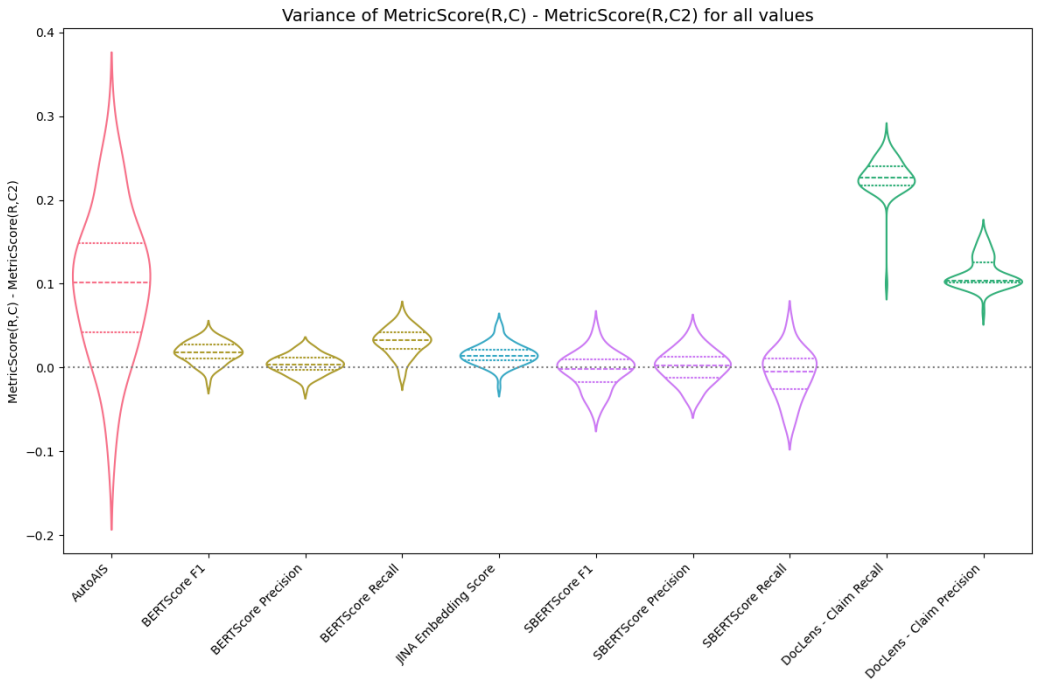

Figure 9: Each violin plots the value corresponding to $E_i(R, C) - E_i(R, C_2)$ along with the mean, and quartiles of the distribution denoted by dotted lines.

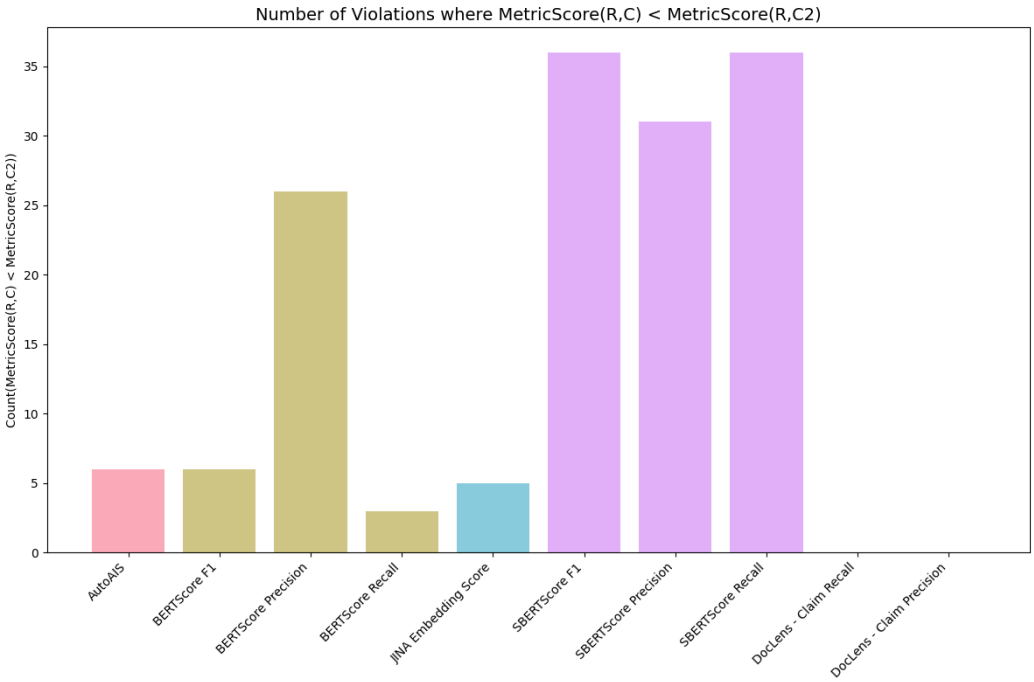

Figure 10: Each bar counts the instance where the Metric scored the degraded candidate C2 higher than candidate C, out of a total of 63 examples. DocLens has the lowest number of such violations.

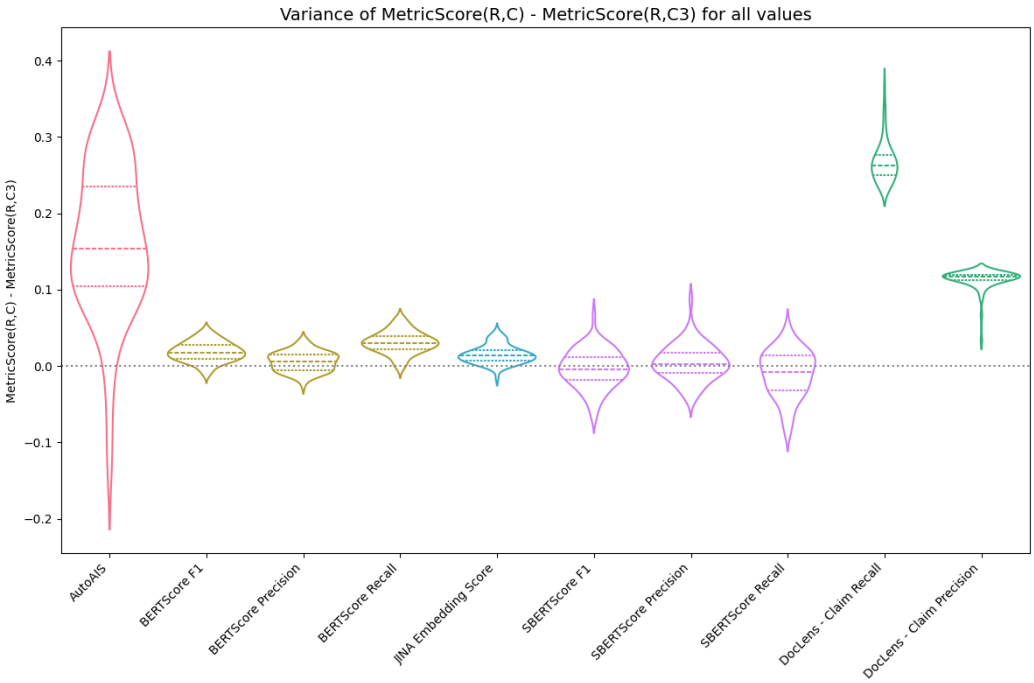

Figure 11: Each violin plots the value corresponding to $E_i(R, C) - E_i(R, C_3)$ along with the mean, and quartiles of the distribution denoted by dotted lines.

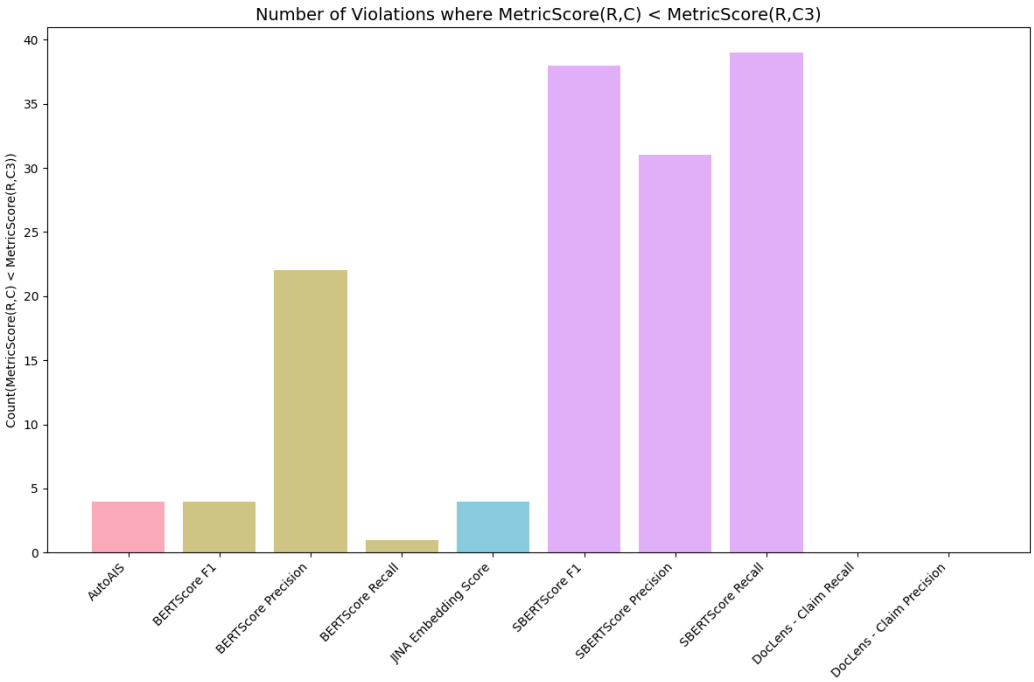

Figure 12: Each bar counts the instance where the Metric scored the degraded candidate C3 higher than candidate C, out of a total of 63 examples. DocLens has the lowest number of such violations.

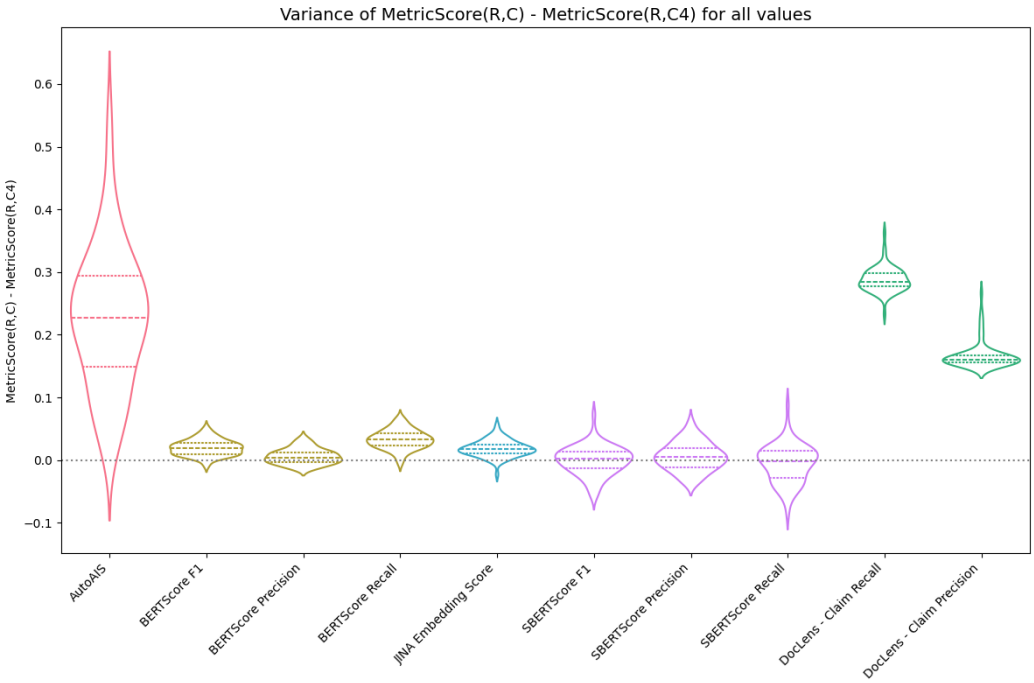

Figure 13: Each violin plots the value corresponding to $E_i(R, C) - E_i(R, C_4)$ along with the mean, and quartiles of the distribution denoted by dotted lines.

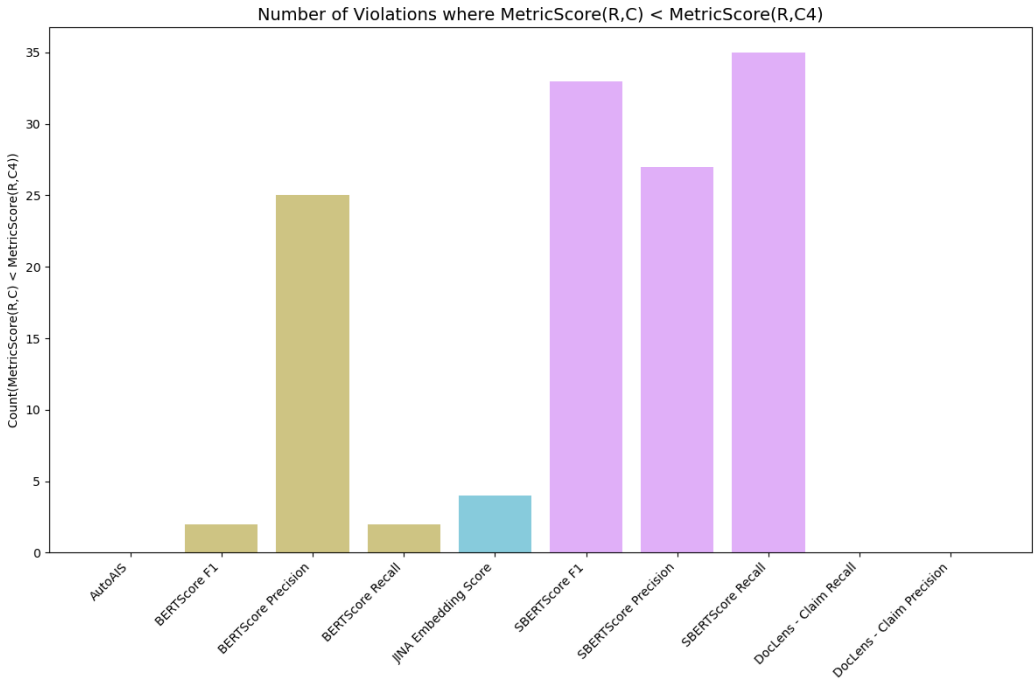

Figure 14: Each bar counts the instance where the Metric scored the degraded candidate C3 higher than candidate C, out of a total of 63 examples. DocLens has the lowest number of such violations.

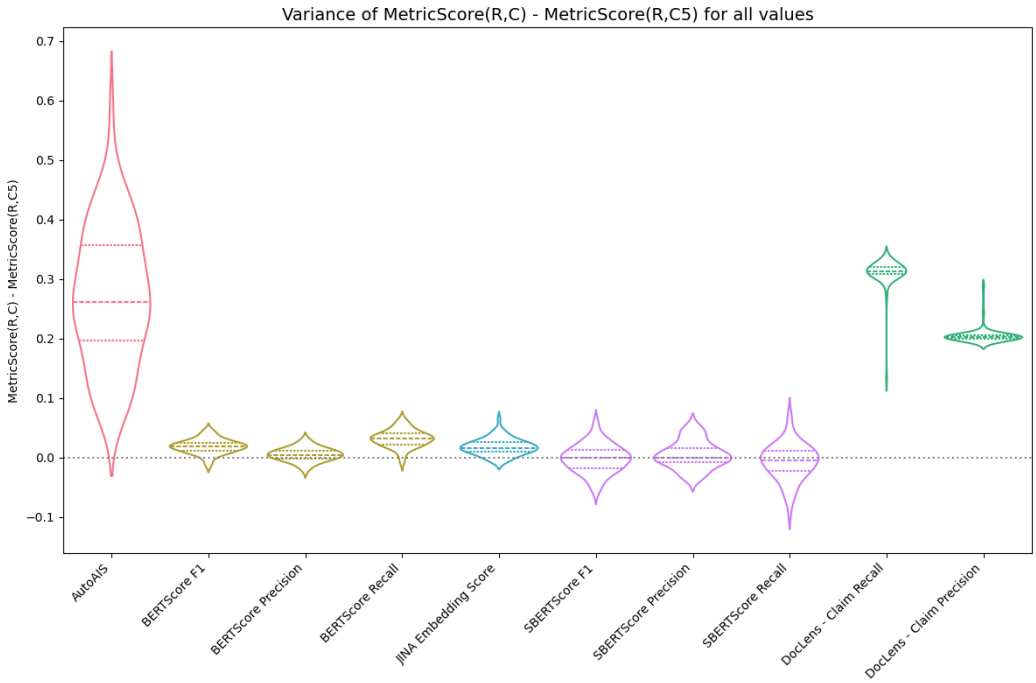

Figure 15: Each violin plots the value corresponding to $E_i(R, C) - E_i(R, C_5)$ along with the mean, and quartiles of the distribution denoted by dotted lines.

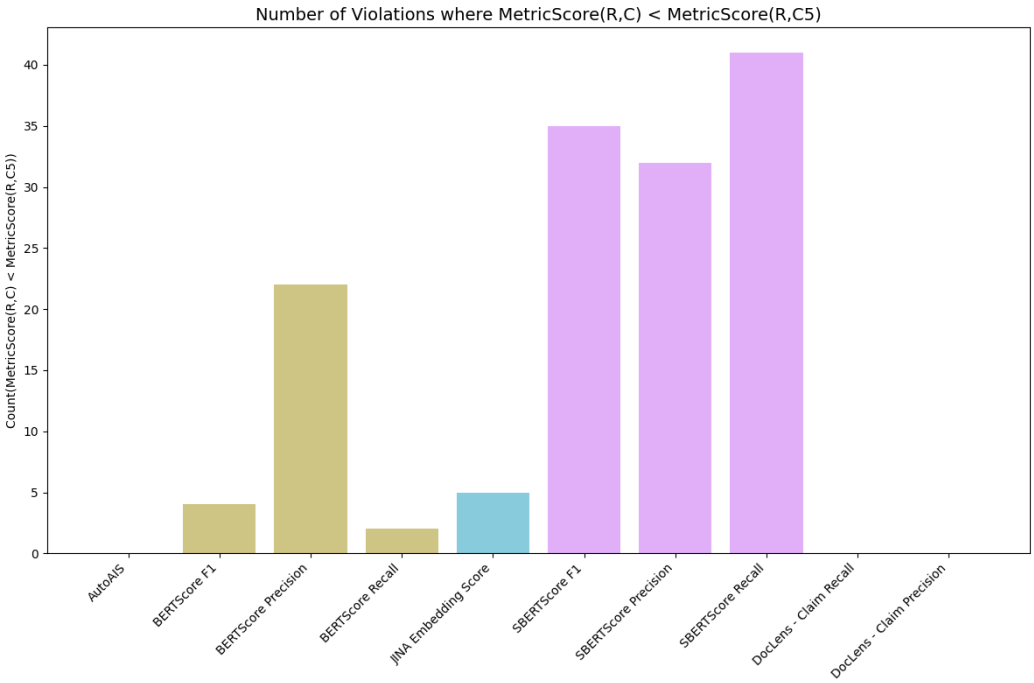

Figure 16: Each bar counts the instance where the Metric scored the degraded candidate C5 higher than candidate C, out of a total of 63 examples. DocLens has the lowest number of such violations.