Contextual Metric Meta-Evaluation by Measuring Local Metric Accuracy

Anonymous ACL submission

Abstract

Meta-evaluation of automatic evaluation metrics—assessing evaluation metrics themselves—is crucial for accurately benchmarking natural language processing systems and has implications for scientific inquiry, production model development, and policy enforcement. While existing approaches to metric meta-evaluation focus on general statements about the absolute and relative quality of metrics across arbitrary system outputs, in practice, metrics are applied in highly contextual settings, often measuring the performance for a highly constrained set of system outputs. For example, we may only be interested in evaluating a specific model or class of models. We introduce a method for contextual metric meta-evaluation by comparing the *local metric accuracy* of evaluation metrics. Across translation, speech recognition, and ranking tasks, we demonstrate that the local metric accuracies vary both in absolute value and relative effectiveness as we shift across evaluation contexts. This observed variation highlights the importance of adopting context-specific metric evaluations over global ones.

1 Introduction

Meta-evaluation of automatic evaluation metrics—assessing evaluation metrics themselves—is crucial for accurately benchmarking natural language processing systems (Zhou et al., 2022). Because metrics are central to scientific inquiry, the development of production models, and policy enforcement (Kocmi et al., 2021), there is a constant need for new approaches to rigorously evaluate these metrics to ensure they remain reliable and contextually appropriate, supporting their effective use across diverse NLP tasks and evolving systems (Novikova et al., 2017).

Although current methods for metric meta-evaluation commonly take a *global* perspective, reporting the performance of a metric across arbitrary system outputs, coming from any system (Stanojević et al., 2015; Przybocki et al., 2009), in practice, evaluation is highly contextual, measuring the performance for a highly constrained set of system outputs. For example, we may only be interested in evaluating a specific model or class of models. From a model development perspective, a metric that is sensitive to model outputs coming from partially trained models at the beginning of the development cycle (when the outputs are far from the target distribution or close to random) may struggle to differentiate between outputs from fully trained or more effective models. For example, Fomicheva and Specia (2019) show that metric performance varies significantly across different levels of translation quality. Thus, using the same metric across various contexts will have inconsistent reliability and can result in inaccurate model selection.

	context			
metric	**X**	**Y**	**Z**	**global**
A	0.9	0.9	0.3	0.7
B	0.7	0.7	0.7	0.7
C	0.3	0.3	0.9	0.5

Table 1: Contextual metric meta-evaluation. The values in the table is the metric accuracy, which represents how often a metric correctly estimates the true ordering of a pair of systems. When comparing metrics A, B, and C, traditional meta-evaluation focuses on global accuracy across arbitrary inputs. Local metric accuracy can vary by evaluation contexts X, Y, and Z.

To illustrate the difference between global and contextual metric meta-evaluation, we constructed a toy meta-evaluation for three metrics across three contexts (Table 1). The values in the table represent the accuracy of three metrics (A, B, and C) under three different contexts (X, Y, and Z) as well as the global accuracy across the different contexts. By

looking at the average, we might think that A and B are equally accurate. However, when inspecting accuracy within individual contexts, we can see that selecting the most appropriate metric is far less straightforward than simply choosing based on the highest global accuracy, as it fails to capture the metric's variability and effectiveness in specific contexts. For example, if we want a metric that best generalizes across different contexts, we want to choose B over A for its robustness, even though their global accuracies are equal. However, if we want to specifically measure outputs in context Z, then we would want to pick C as it is especially sensitive to system outputs in that context, despite it having the lowest global accuracy.

Although prior research has focused mainly on global metric evaluations, we explicitly analyze how metric accuracy varies across different evaluation contexts by measuring their *local metric accuracies*. By evaluating metrics across three different machine learning tasks—machine translation, automatic speech recognition, ranking—we show that the metric accuracy, which measures the ability of a metric to accurately assign the true preference between a pair of system decisions, changes as evaluation context changes. Specifically, we show that the metric accuracy changes both in absolute value and relative ordering across the different contexts. In contrast with existing work on metric meta-evaluation, which relies heavily on costly and time-consuming explicit human feedback (Fabbri et al., 2021; Liu et al., 2016), our method uses output perturbations (Sai et al., 2021; He et al., 2023) to obtain the true ordering between a pair of system outputs without the need for human supervision. Overall, we show that measuring local metric accuracies is a straightforward methodology to provide a more contextual understanding of evaluation metrics which complements existing global metric meta-evaluation methods.

2 Related work

Our work is situated in the broader literature on meta-evaluation. For example, the Conference on Machine Translation (WMT) has focused on evaluating the utility of metrics in machine translation since 2008, where participants submit automatic metrics for validation against human feedback (Callison-Burch et al., 2008). Xiao et al. (2023) propose a meta-evaluation framework rooted in measurement theory for NLG metrics, highlighting issues in human evaluation that include a lack of validation, standardization, and consistency.

Our use of output perturbations is inspired by prior work in testing metric robustness. Chen and Eger (2023) proposed a preference-based adversarial attack framework using targeted perturbations to evaluate the robustness of NLI-based and BERT-based metrics, finding that NLI-based metrics are more robust in summarization but not in machine translation. Sai et al. (2021) extends perturbation-based robustness testing by creating templates that target specific criteria such as jumbled word order to test fluency. He et al. (2023) utilized perturbations to design synthetic stress tests to evaluate the robustness of various text generation metrics. Furthermore, Valcarce et al. (2018) evaluated the robustness of the ranking metrics against incompleteness by purposefully removing system outputs to introduce sparsity. Our paper adopts similar perturbation techniques to assess the preference-based evaluation capability of different metrics, eliminating the need for costly and time-intensive explicit human feedback.

Although metric meta-evaluation is often done on a global level, previous work indicates that the reliability of a metric changes from the system-level to the decision-level (Reiter and Belz, 2009; Stent et al., 2005). Though some research has investigated metric performance for different contexts based on output sources (i.e., models) or output qualities (Mathur et al., 2020; Novikova et al., 2017), our work addresses the lack of a systematic review of contextual meta-evaluation and how to conduct it.

3 Local accuracy

To formalize local metric accuracy, we introduce the following notation. Let $\mathcal{X}$ be the set of all possible system inputs (e.g., for MT, all possible strings from the source language) and $\mathcal{Y}$ the set of all possible system outputs (e.g., for MT, all possible strings from the target language). We define $X \subset \mathcal{X}$ to be the subset of system inputs observed in a specific context (e.g., for MT, a sample of source sentences from a specific university). Similarly, $\mathcal{Y}_x \subset \mathcal{Y}$ is the subset of system decisions for $x \in X$ observed for X in a specific context (e.g., for MT, a set of translations generated by a set of candidate systems). In addition, we have access to several perturbation functions that, with high probability, degrade the utility of a decision y (e.g., dropping

a random word from a translated input). Let Q_x be the set of pairs decisions conditioned on an input x and their corresponding degraded version: $Q_x = \{\langle y, y' \rangle\}_{y \in \mathcal{Y}_x}$.

An evaluation metric $\mu : \mathcal{X} \times \mathcal{Y} \rightarrow \Re$ generates a scalar number reflecting the performance according to some system property that we want to measure (e.g. correctness of translation). Each metric is an approximation of μ^*, the unobserved ideal evaluation metric (i.e., the true utility of an output). Given two pairs of system outputs, μ^* will always be able to determine the true ordering of the two outputs, including instances where it has no preference. In cases where we intentionally perturb y to obtain y', we have that $\mu^*(x, y) > \mu^*(x, y')$. We can assume that the system outputs that we are evaluating are at least better than random such that we can assume that $\mu^*(x, y) > \mu^*(x, y')$ with high probability. Under the assumption that μ approximates μ^*, we want to compute how often $\mu(x, y) > \mu(x, y')$. As suggested by Kocmi et al. (2021), we focus on the ability of μ to reproduce the ordering of decisions, rather than the magnitude of the difference between $\mu(x, y)$ and $\mu(x, y')$. From this, we define the pointwise local metric accuracy, conditioned on an input x, to be:

$$\text{ACC}_\mu(Q_x) = \frac{1}{|Q_x|} \sum_{\langle y, y' \rangle \in Q_x} \mathbb{1}\left[\mu(x, y) > \mu(x, y')\right] \tag{1}$$

This measures the ability of a metric to reproduce the true ordering of perturbations for a specific input x. We define the local metric accuracy across all inputs X by averaging over $|X|$, as follows:

$$\text{ACC}_\mu(Q) = \frac{1}{|X|} \sum_{x \in X} \text{ACC}_\mu(Q_x) \tag{2}$$

where $Q = \cup_{x \in X} Q_x$. This measures the local metric accuracy across a sample of system inputs, as we may have in a standard evaluation set. The main difference with global metric accuracy is that the accuracy is only computed over all pairs of $\langle y, y' \rangle$ belonging to a specific context (and not over all the observed outputs).

We are interested in testing two hypotheses with respect to local metric accuracy.

H1: The *absolute* local metric accuracy, $\text{ACC}_\mu(Q)$, of a metric μ changes as the context changes.

Evidence supporting this hypothesis suggests that existing evaluation methods focusing on global metric accuracy obscure how metric accuracy varies across different contexts.

H2: The *relative* local metric accuracy of a metric μ changes as the context changes.

In other words, the total ordering of all metrics by local metric accuracy within a context changes as the context Q changes. Evidence supporting this hypothesis suggests that choosing an appropriate metric to benchmark compare system outputs largely depends on the context.

4 Methods and Materials

4.1 Tasks, dataset, and metrics

We performed our evaluation on three different tasks: Machine Translation (MT), Automatic Speech Recognition (ASR), and Ranking. We detail the dataset, metrics, and their corresponding implementations in Appendix A.1.

We can divide the outputs for each task into different subsets, such as the SYSTEM that produced it. We report results on the subset with the highest number of contexts for each task. The abundance of contexts allowed us to identify trends in metric behavior across a broader range of items and helped us identify supporting evidence for or against our hypotheses. We have included the results for each of the available subsets for each task in Appendix A.3.

4.2 Perturbation techniques

To test our hypotheses, we apply perturbation functions that degrade the utility of a system output y to obtain its corresponding degraded version y', simulating the output of a degraded system. This methodology is widely used to automatically generate pairs of outputs with their corresponding true ordering for meta-evaluation, such as in work done by He et al. (2023) and Sai et al. (2021). By applying perturbations, we know that the quality of y' under a specific task is worse than y with high probability. We highlight a subset of the perturbations applied to the system outputs from their corresponding tasks in Table 4. The full list of perturbations can be found in Appendix A.2.1. We selected this set of perturbations because it represents a large enough breadth yet still mimics realistic system errors in their corresponding tasks. It is important to note that not all perturbations apply to all outputs. For example, some outputs do not contain

Task	Perturbation	Description
MT	Removal	Drop a random word from the string, e.g. "John said he did not like the cake baked."
	Swapping	Switch the position of a random pair of words from the string, e.g."John he said did not like the cake she baked."
	LLM	Prompt an LLM to perturb the string, e.g. "John mentioned he didn't enjoy the cake she made ."
ASR	Phonetic sub.	Replace a random letter (or a letter group) with phonetically similar ones, e.g. "John sed he did not like the cake she bakked ."
	LLM	Prompt an LLM to perturb the string, e.g. "John sed he did not like cake she baked."
Ranking	Swapping	Switch the position of an item with another item within the ranked list, e.g. Original ranking: [Cake, Pie, Cookie] Perturbed: [Cookie, Pie, Cake]

Table 2: A subset of perturbations for MT, ASR, and Ranking tasks. The original unperturbed sentence for the MT and ASR examples is "John said he did not like the cake she baked." We show the full table of perturbations in Appendix A.2.1 and outline how each perturbation method is implemented in Appendix A.2.2.

named entities, so we cannot apply the perturbation method involving a named entity substitution.

Certain perturbations may confound, i.e. inherently favor (or conversely, disfavor), specific metrics due to their design. For example, if we apply the swapping perturbation function, this might disproportionately penalize ROUGE-1 as it is designed only to measure lexical overlap and is agnostic to position; therefore, it will consistently achieve an accuracy of 0 across all contexts. However, we likely do not observe this behavior from ROUGE-1 under different perturbations, such as removal or insertion. Thus, if a perturbation confounds a metric, the local metric accuracies will exhibit high variance when we change the perturbation method.

Finding a single perturbation method that does not confound all the metrics we evaluate is challenging. To mitigate the impact of potential confounders, we generate four different y's for each y in machine translation and automatic speech recognition, corresponding to the result of applying four randomly selected perturbations to y. Let $\{f_1, f_2, f_3, f_4\}$ be the randomly selected perturbations corresponding to any of the perturbation methods for a specific task. Now, instead of having a single pair $\langle y, y' \rangle$ in Q_x, we will have four pairs for each y, namely $\langle y, f_1(y) \rangle$, $\langle y, f_2(y) \rangle$, $\langle y, f_3(y) \rangle$, and $\langle y, f_4(y) \rangle$. Since the different perturbation methods are independent, increasing the number of samples for y' by applying different perturbations reduces the variance in local metric accuracies across various combinations of perturbations. This, in turn, minimizes the effect of confounders. We illustrate this further in our experiments in Appendix A.2.3.

Given the inherent structure of ranked lists, our approach involves applying a single perturbation method for the system outputs in the ranking task, as there are limited meaningful ways to modify a ranked list to simulate realistic system errors. To address this limitation, we swap multiple items within the list when applying the swapping perturbation to ranked outputs. In contrast, for machine translation and automatic speech recognition, we apply four different perturbation methods to a single output y to get four distinct pairs of $\langle y, y' \rangle$, but we apply each method only once (e.g., when applying the swapping perturbation, we only swap a pair of words instead of swapping multiple pairs).

4.3 Hypothesis Testing

We plotted the metric accuracies $\text{ACC}_\mu(Q)$ for each task across different contexts within the selected context subset Q as a line graph, such that we can visualize how the metric's capability of differentiating between y and y' changes as the context changes by observing the slopes and overlaps between the lines. To test H1 and to further investigate the association between the context Q and the metric accuracy $\text{ACC}_\mu(Q)$, we used the χ^2 test of independence of variables (Pearson, 1900) in a contingency table (Pearson, 1904). We compared the resulting p-values to the significance level of $\alpha = 0.05$ to understand whether the changes in metric accuracy $\text{ACC}_\mu(Q)$ across the different contexts Q are statistically significant.

To test H2, we computed Kendall's-τ_{AP} (Kendall, 1938) between two rankings of metrics according to local metric accuracy under two contexts. The Kendall's-τ_{AP} values help us quan-

tify how the metrics' total ordering changes as the context changes. To emphasize the metric selection task, we adopt a widely-used variant of Kendall's-τ_{AP} known as τ_{AP} proposed by Yilmaz et al. (2008), such that changes at the top of the ranked list are weighted higher than the ones that are much lower in the list. We will also compare the rankings of metrics according to local metric accuracy within one context and the rankings of metrics according to their global metric accuracy.

5 Results

5.1 Machine Translation

We visualize the metric accuracies for the machine translation metrics under the different SYSTEM contexts, as shown in Figure 1a. We observe that the line for each metric changes as we change the context, as indicated by the varying slopes of the lines. The results of the χ^2 test in Appendix A.4 show that each metric has a p-value less than 0.05, indicating that the difference in the metric accuracies across the different contexts is statistically significant, supporting H1 for MT.

Figure 1a also contains intersections between lines corresponding to different metrics, indicating that there is a change in the relative position of each metric in the different contexts, which also shows a change in the total ordering of metrics by local accuracy across contexts, supporting H2. This is further reinforced in Figure 1b, where the τ_{AP} values show that the correspondence between the pairs of metric accuracy rankings varies considerably for each pair of SYSTEMS.

From Figure 4, we can see that local metric accuracies exhibit a non-linear trend across the different translation quality contexts: they are low for poor translation quality, rise in the mid-range, and drop again for high-quality translations. This pattern suggests that current machine translation metrics struggle to effectively distinguish between very low and very high translation quality, indicating limitations at both ends of the quality range. We are able to divide the data into different translation quality contexts as the WMT 2022 dataset (Freitag et al., 2022) comes with MQM scores (Freitag et al., 2021), which reflect the human judgment of the system outputs. Because MQM scores are continuous, we used a quantile-based discretization function to discretize the scores into equal-sized buckets based on $q = 250$. This is done instead of discretizing the MQM scores into equal ranges because some ranges have zero density.

5.2 Automatic Speech Recognition

For ASR, we report the local metric accuracy under different SPEAKER IDs which come from different dataset QUALITY contexts (CLEAN/RAW). We plot the local metric accuracy for contexts associated with different contexts shown in Figure 2a. We observe that the lines corresponding to each metric are not straight, which indicates that the absolute local accuracy for each metric changes with context, supporting H1. Our χ^2 test results in Appendix A.4 show that each metric has consistent p-values less than 0.05, confirming that the difference in local metric accuracies in the different contexts is statistically significant, providing evidence supporting H1 for ASR.

Interestingly, we do not observe the same level of overlaps between the lines corresponding to the different metrics as we did for MT. This, along with the consistently high values of τ_{AP} for the majority of SPEAKERS in Figure 2b, indicates that there is no evidence to support H2 for ASR. We will discuss this observation in more detail in 6.2.

5.3 Ranking

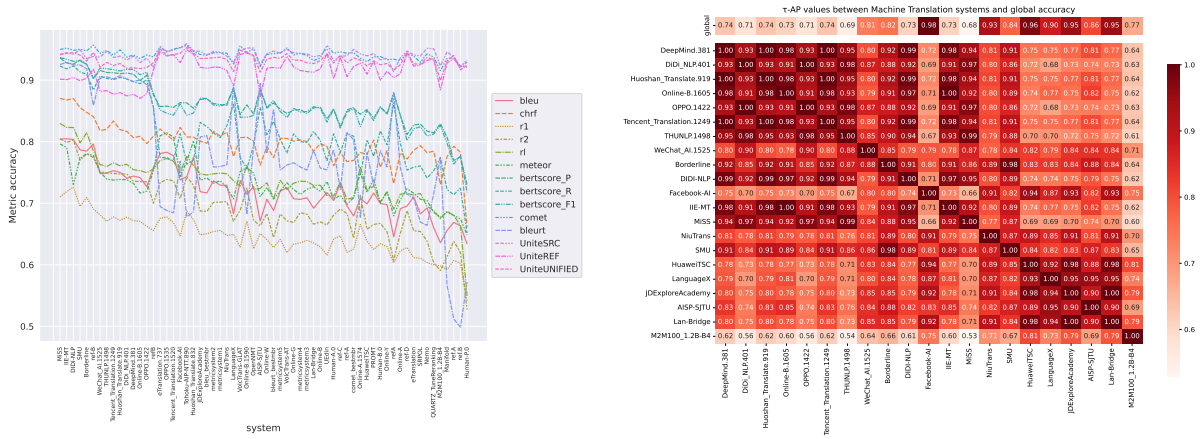
Putting the metric accuracies for the ranking metrics for the different ALGORITHMS in Figure 3a, we can first observe that none of the lines corresponding to the different metrics are straight lines, which supports H1. The large fluctuations within each line suggest that the changes in the absolute local accuracies for each metric are rather significant. The χ^2 test in Appendix A.4 shows that each metric has a consistent p-value less than 0.05, which means a statistically significant change, providing evidence to support H1.

Furthermore, we can see that overlaps exist between the different lines corresponding to the different metrics, similar to the observation we made in the MT case. This indicates that the total ordering of metric accuracies changes as the context changes, supporting H2. The τ_{AP} results (Figure 3b) show clustering by algorithm, as with MT.

6 Discussion

6.1 H1: Absolute Local Accuracies

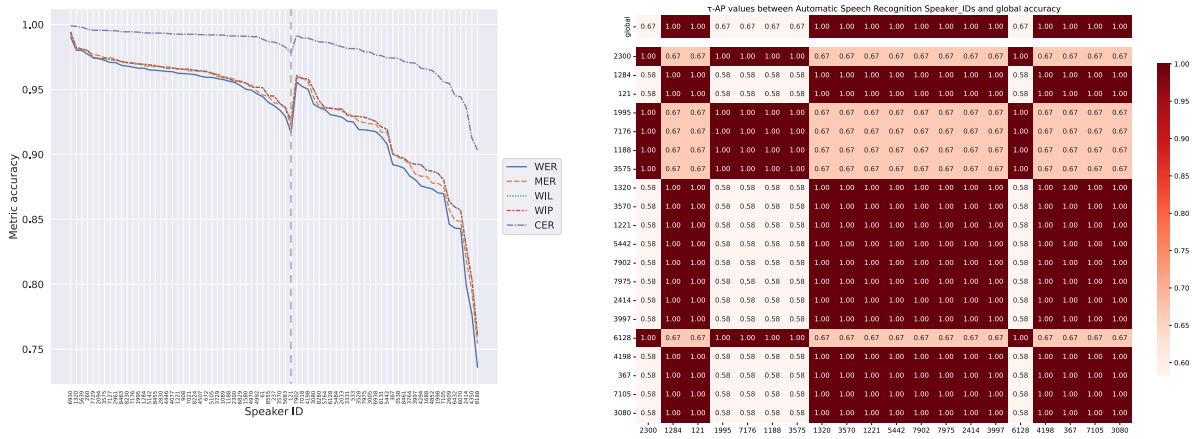
The results in Section 5 generally provide evidence supporting H1, as our experiments consistently show that the local metric accuracy changes as the context changes.



(a) Local metric accuracy across the different contexts

(b) τ_{AP} values of metrics ordered by local accuracy between different contexts and comparison with global ranking

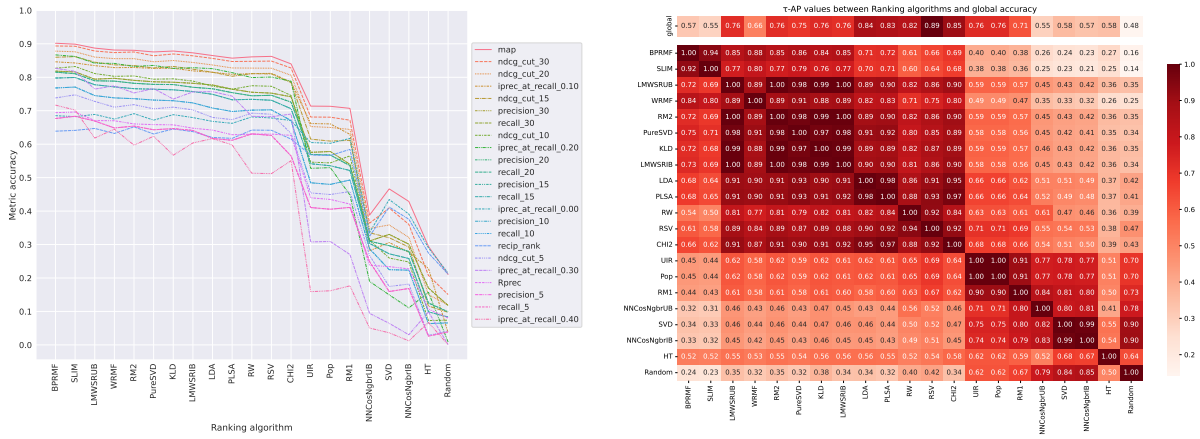
Figure 1: Machine Translation. Metric accuracy for Machine Translation metrics across the different systems.



(a) Local metric accuracy across the different contexts

(b) τ_{AP} values of metrics ordered by local accuracy between different contexts and comparison with global ranking

Figure 2: Automatic Speech Recognition. Metric accuracy for Automatic Speech Recognition metrics across the different Speaker IDs. (a) Speaker IDs to the left of the gray line come from the QUALITY=CLEAN LibriSpeech-100 dataset, while the Speaker IDs to the right come from the QUALITY=OTHER LibriSpeech-100 dataset.



(a) Local metric accuracy across the different contexts

(b) τ_{AP} values of metrics ordered by local accuracy between different contexts and comparison with global ranking

Figure 3: Ranking. Metric accuracy for Ranking metrics across the different systems.

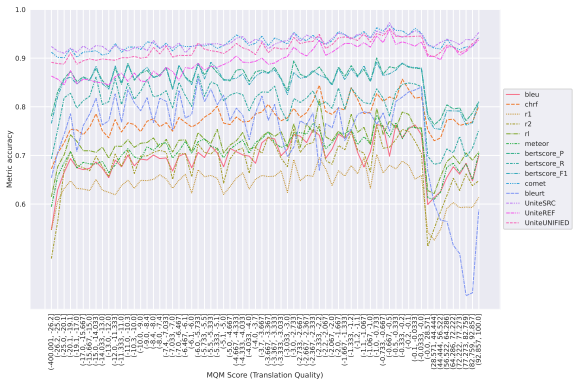


Figure 4: Local metric accuracy across the different MQM scores (human-labeled translation quality scores)

Our perturbation method catastrophically degrades good outputs, and bad outputs are already poor and difficult to make demonstrably worse. In Figure 3a, we can observe that the local metric accuracy for a context is related to the average quality of outputs in that context. For example, in the ranking setting, the most effective system according to MAP is BPRMF, which is also the context whose perturbed outputs are easiest to distinguish. In contrast, perturbed outputs in the random ranker are more difficult for all metrics to distinguish. We will return to this in Section 6.3.

The results corresponding to how the local metric accuracy changes as the context changes suggest that evaluators may be interested in the stability of local metric accuracy when selecting a metric. A more stable metric with respect to local metric accuracy is more predictable when deployed under a new context, and the probability of selecting the wrong system is consistent. This is especially important if we consider a new evaluation context where poor local metric accuracy puts users—or a vulnerable subgroup of users—at risk. Work in robust machine learning provides existing methods for designing metrics stable across context changes (Yuan et al., 2024).

In addition to stability, we can organize metrics according to systematic behavior in the local metric accuracy. For example, in Figure 1a, more complex embedding-based and model-based evaluation metrics generally perform better than the simpler lexical-based metrics (Zhang et al., 2019; Freitag et al., 2022). More complex metrics cluster at the top of the figure, while simpler metrics occupy the bottom regions; any overlap occurs mostly within a specific subset. Such an analysis allows evaluators to understand the empirical relationships between metric ensembles. Although picking the best metric might involve selecting a metric that occupies the top of the figure, there may be contexts in which local metric accuracies are close enough to allow flexibility in selecting metrics with lower local metric accuracy.

More generally, we can consider multi-objective metric development. For example, since embedding- and model-based methods are more time-intensive and computationally costly compared to the lexical-based methods, adopting simpler and cheaper metrics when local metric accuracies are comparable (e.g., early in model development) would result in cost savings and faster iteration. Beyond cost and local metric accuracy, one can imagine local versions of metric interpretability, metric engineering overhead, metric optimizability, and other criteria when conducting for contextual meta-evaluation.

6.2 H2: Relative Local Accuracies

Although the observations in Section 5 err toward accepting H2, the evidence from our experiments on the ASR task in Section 5.2 suggests that it strongly depends on the nature of the task and the metrics. In ASR, the ambiguity of the correct answers is low, unlike in MT or ranking, where two outputs (e.g., translations or permutations) can be equally good (Wieting et al., 2019). ASR outputs are typically direct transcriptions of spoken language into text, and there is often a single 'correct' or 'expected' output for a given input. The main evaluation objective is to check whether the transcribed text accurately matches the spoken words without considering other criteria such as semantic similarity, grammatical structure, or fluency. Furthermore, ASR tasks primarily rely on phonetic accuracy, and the goal is to replicate spoken words as text as accurately as possible. Although there may be additional layers of complexity, such as accents, homophones, and background noise, the task does not require deep semantic and contextual understanding beyond recognizing the correct phonetic and lexical forms, unlike machine translation. Hence, the metrics commonly used to benchmark ASR systems only slightly vary in the construct they are trying to measure, and they are all operationalized following similar statistical methods.

Figures 1b and 3b indicate that there are groups of contexts where the relative reliability of the metrics is similar (i.e., the relative ranking of metrics in these groups of contexts do not change signifi-

cantly). When contexts can be structured according to metric accuracy, one can adopt a fixed evaluation metric. This has practical implications in terms of engineering and development overhead or, in the case of model-based metrics, model development cost. *Predicting* the similarity in local metric accuracy ordering (i.e., cells in Figures 1b and 3b) is an important task because it allows evaluators to confidently adopt an evaluation metric without conducting contextual meta-evaluation. Predictive features include any metadata we have about the contexts. For example, in ranking, Valcarce et al. (2018) categorize ALGORITHMs into different families of techniques: matrix factorization (SVD, PURESVD, BPRMF, WRMF), neighborhood-based (CHI2, KLD, RSV, ROCCHIO'S WEIGHTS).

Additionally, the top rows on Figures 1b and 3b show that the ranking of the metrics in different contexts varies considerably from the ranking of the metrics based on the global accuracy. This observation aligns with the scenario illustrated by the toy example in Table 1, where we show that selecting the most appropriate metric may not be as straightforward as selecting the metric with the highest global precision and further motivates the need to shift towards context-specific metric evaluations.

6.3 Methodology

Although our results demonstrate that local metric accuracy analysis can provide insight into metric behavior, there are several opportunities to improve the methodology. First, our perturbations, while reliable in generating output degradations, may result in outputs that are easily detected by metrics, especially for highly effective systems. Moreover, perturbed outputs may be sufficiently different as to be unlikely to occur in a specific context. For example, if we are evaluating in the context of highly effective MT systems, a translation with a missing word is very unlikely by any highly effective MT system, even though we know that it is of lower quality. In order to address this, developing perturbation methods that reliably degrade performance *and* are likely to occur within a context will be important for future local metric accuracy development. This is related to synthesizing hard negative examples in the contrastive learning literature (Kalantidis et al., 2020). Alternatively, we can consider non-perturbation data, perhaps from human annotators, although this compromises the cost-effectiveness of output perturbation.

Finally, in order to help with clarity, we focused on contexts that were interpretable, and which contexts are relevant depends on the broader model evaluation environment. Focusing on models, as we did for MT and ranking, emphasizes contexts that reflect iterative model development and refinement within a narrow set of constraints (i.e., the particular model being evaluated). If we benchmark a diverse set of systems, we are interested in comparing a broader set of possible outputs than those from a single system. In cases where we are designing a metric agnostic to a particular context, we may be interested in robust performance across arbitrary contexts. Although this is similar to global analysis, a more rigorous and formal approach to context selection, such as found in the distributionally robust machine learning literature (Duchi et al., 2018), may be more appropriate.

7 Conclusion

We introduce the notion of local metric accuracy and demonstrate how to use it to conduct contextual metric meta-evaluation. Our results show that both the absolute and relative local accuracy of a metric vary as we vary the context, though this depends on the nature of the task. Based on our results, we believe that by moving beyond global metric meta-evaluation, we can achieve a more accurate understanding of metric performance, which in turn increases the reliability of the evaluations and provide actionable insights for improving NLP systems.

8 Limitations

As mentioned in Section 4.2 and Appendix A.2.2, our experiments adopt relatively simple rule-based and LLM-based perturbation methods to cover several tasks. However, given the wide range of system output qualities in our dataset, it is rather difficult to guarantee that applying a perturbation function to y will always produce a y' that is worse in utility.

We also compute local accuracy by uniformly weighting all output-perturbation pairs. In reality, different outputs have different probabilities of occurring in a specific context. These probabilities should be incorporated into the accuracy calculation to provide a more reliable estimate of local metric accuracy. Estimating the distribution over outputs for a specific context itself is a difficult research question that we plan to address in future work.

References

AI@Meta. 2024. Llama 3 model card.

Chris Buckley and Ellen M Voorhees. 2004. Retrieval evaluation with incomplete information. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 25–32.

Chris Callison-Burch, Philipp Koehn, Christof Monz, Josh Schroeder, and Cameron Shaw Fordyce. 2008. Proceedings of the third workshop on statistical machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*.

Yanran Chen and Steffen Eger. 2023. MENLI: Robust evaluation metrics from natural language inference. *Transactions of the Association for Computational Linguistics*, 11:804–825.

J. C. Duchi, P. W. Glynn, and H. Namkoong. 2018. Statistics of robust optimization: A generalized empirical likelihood approach. *CoRR*, abs/1610.03425.

Alexander R Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Marina Fomicheva and Lucia Specia. 2019. Taking MT evaluation metrics to extremes: Beyond correlation with human judgments. *Computational Linguistics*, 45(3):515–558.

Markus Freitag, George Foster, David Grangier, Viresh Ratnakar, Qijun Tan, and Wolfgang Macherey. 2021. Experts, errors, and context: A large-scale study of human evaluation for machine translation. *Transactions of the Association for Computational Linguistics*, 9:1460–1474.

Markus Freitag, Ricardo Rei, Nitika Mathur, Chi-kiu Lo, Craig Stewart, Eleftherios Avramidis, Tom Kocmi, George Foster, Alon Lavie, and André F. T. Martins. 2022. Results of WMT22 metrics shared task: Stop using BLEU – neural metrics are better and more robust. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 46–68, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

F. Maxwell Harper and Joseph A. Konstan. 2015. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4).

Tianxing He, Jingyu Zhang, Tianle Wang, Sachin Kumar, Kyunghyun Cho, James Glass, and Yulia Tsvetkov. 2023. On the blind spots of model-based evaluation metrics for text generation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12067–12097, Toronto, Canada. Association for Computational Linguistics.

Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. 2020. Hard negative mixing for contrastive learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 21798–21809. Curran Associates, Inc.

Maurice G Kendall. 1938. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93.

Tom Kocmi, Christian Federmann, Roman Grundkiewicz, Marcin Junczys-Dowmunt, Hitokazu Matsushita, and Arul Menezes. 2021. To ship or not to ship: An extensive evaluation of automatic metrics for machine translation. In *Proceedings of the Sixth Conference on Machine Translation*, pages 478–494, Online. Association for Computational Linguistics.

Chia-Wei Liu, Ryan Lowe, Iulian Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2122–2132, Austin, Texas. Association for Computational Linguistics.

Nitika Mathur, Johnny Wei, Markus Freitag, Qingsong Ma, and Ondřej Bojar. 2020. Results of the WMT20 metrics shared task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 688–725, Online. Association for Computational Linguistics.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252, Copenhagen, Denmark. Association for Computational Linguistics.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: An asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210.

Karl Pearson. 1900. X. on the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 50(302):157–175.

Karl Pearson. 1904. *On the theory of contingency and its relation to association and normal correlation*. Drapers' Company research memoirs. Cambridge University Press.

Mark Przybocki, Kay Peterson, Sébastien Bronsart, and Gregory Sanders. 2009. The nist 2008 metrics for machine translation challenge—overview, methodology, metrics, and results. *Machine Translation*, 23:71–103.

Ehud Reiter and Anja Belz. 2009. An investigation into the validity of some metrics for automatically evaluating natural language generation systems. *Computational Linguistics*, 35(4):529–558.

Ananya B. Sai, Tanay Dixit, Dev Yashpal Sheth, Sreyas Mohan, and Mitesh M. Khapra. 2021. Perturbation CheckLists for evaluating NLG evaluation metrics. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7219–7234, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Miloš Stanojević, Amir Kamran, Philipp Koehn, and Ondřej Bojar. 2015. Results of the WMT15 metrics shared task. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 256–273, Lisbon, Portugal. Association for Computational Linguistics.

Amanda Stent, Matthew Marge, and Mohit Singhai. 2005. Evaluating evaluation methods for generation in the presence of variation. In *International conference on intelligent text processing and computational linguistics*, pages 341–351. Springer.

Daniel Valcarce, Alejandro Bellogín, Javier Parapar, and Pablo Castells. 2018. On the robustness and discriminative power of information retrieval metrics for top-n recommendation. In *Proceedings of the 12th ACM conference on recommender systems*, pages 260–268.

Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai. 2018. ESPnet: End-to-end speech processing toolkit. In *Proceedings of Interspeech*, pages 2207–2211.

John Wieting, Taylor Berg-Kirkpatrick, Kevin Gimpel, and Graham Neubig. 2019. Beyond BLEU: Training neural machine translation with semantic similarity. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4344–4355, Florence, Italy. Association for Computational Linguistics.

Ziang Xiao, Susu Zhang, Vivian Lai, and Q. Vera Liao. 2023. Evaluating evaluation metrics: A framework for analyzing NLG evaluation metrics using measurement theory. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10967–10982, Singapore. Association for Computational Linguistics.

Emine Yilmaz, Javed A Aslam, and Stephen Robertson. 2008. A new rank correlation coefficient for information retrieval. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 587–594.

Lifan Yuan, Yangyi Chen, Ganqu Cui, Hongcheng Gao, Fangyuan Zou, Xingyi Cheng, Heng Ji, Zhiyuan Liu, and Maosong Sun. 2024. Revisiting out-of-distribution robustness in nlp: Benchmarks, analysis, and llms evaluations. *Advances in Neural Information Processing Systems*, 36.

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

Kaitlyn Zhou, Su Lin Blodgett, Adam Trischler, Hal Daumé III, Kaheer Suleman, and Alexandra Olteanu. 2022. Deconstructing NLG evaluation: Evaluation practices, assumptions, and their implications. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 314–324, Seattle, United States. Association for Computational Linguistics.

A Appendix

A.1 Details for Tasks, Datasets, and Metrics

Table 3 shows the dataset and metrics that we used in our experiments. For each task, we used readily available system outputs to improve reproducibility. For each metric, we employ its respective official implementations or, when unavailable, the most widely used implementation with default parameters. For any neural metric computation, we used an NVIDIA RTX A6000 GPU. For BLEU, we used `sentencebleu` implementation from `nltk`[1]. We also used `nltk`'s implementation for METEOR. For ROUGE[2] and BERTSCORE[3], we have used the implementation released by their respective authors. For BLEURT[4], COMET[5], CHRF[6] and UNITE[7], we have used their official implementations via the `evaluate` library on HuggingFace. We used the `jiwer`[8] Python package to compute the ASR metrics. We used the `trec_eval`[9] to calculate the ranking metrics.

A.2 Perturbations

A.2.1 List of perturbations

The complete list of perturbations used in our experiments and their descriptions are in Table 4.

[1] https://www.nltk.org/index.html
[2] https://github.com/google-research/google-research/tree/master/rouge
[3] https://github.com/Tiiiger/bert_score
[4] https://github.com/google-research/bleurt#readme
[5] https://unbabel.github.io/COMET/html/models.html
[6] https://github.com/mjpost/sacreBLEU#chrf--chrf
[7] https://huggingface.co/Unbabel/unite-mup
[8] https://github.com/jitsi/jiwer
[9] https://github.com/usnistgov/trec_eval

Task	Dataset	Metrics
MT	Over 150,000 system outputs and reference translation from 62 different MT systems submitted to the WMT metrics task from the year 2022 and prior (Freitag et al., 2022) with MQM annotations (Freitag et al., 2021) for the source-target language pairs English-Russian (en-ru), English-German (en-de), Chinese-English (zh-en). The subsets that are available are YEAR, DOMAIN, SYSTEM, and MQM SCORE.	BLEU, ROUGE-1, ROUGE-2, ROUGE-L, METEOR, BERTSCOREP, BERTSCORER, BERTSCOREF1, COMET, BLEURT, CHRF, UNITESRC, UNITEREF, UNITEUNIFIED
ASR	Over 33,000 system outputs from six different ASR models on ESPnet (Watanabe et al., 2018) on the LibriSpeech 100 dataset (Panayotov et al., 2015). The subsets that are available are SYSTEM, SPEAKER ID, and QUALITY.	Word Error Rate (WER), Match Error Rate (MER), Word Information Lost (WIL), Word Information Preserved (WIP), Character Error Rate (CER)
Ranking	Ranked list of top-100 items retrieved by 21 recommender algorithms provided by Valcarce et al. (2018) on the MovieLens1M dataset (Harper and Konstan, 2015) submitted to TREC (Buckley and Voorhees, 2004). We were able to segment the outputs by ALGORITHM.	Mean Average Precision (MAP), Precision@R, where R is the number of relevant documents (RPREC), Reciprocal Rank (RECIP_RANK), Interpolated Precision at Recall Level X (for $X = \{0.0, 0.1, 0.2, 0.3, 0.4\}$) (IPREC_AT_RECALL_X), Precision@K (P_K), Recall@K (RECALL_K), nDCG@K (NDCG_CUT_K) (where $K = 5, 10, 15, 20, 30$)

Table 3: Datasets and metrics used for different tasks

A.2.2 Implementation of perturbations

For the system output corresponding to machine translation and automatic speech recognition tasks, we first preprocess the target output by performing tokenization, part-of-speech tagging, named entity recognition, negation detection, etc. Additionally, the WMT dataset (Freitag et al., 2022) contains target outputs in three different languages - English, Russian, and German - we were careful to use libraries that are tailored to each language. We then use simple string manipulation functions to perform rule-based insertion, deletion, substation, and more. To introduce an additional layer of complexity, we also prompted an instruction-tuned LLM to perturb the system and generate a degraded output.

We employ simple rule-based perturbation techniques with the addition of an LLM-based perturbation. We noted that not all perturbation techniques are applicable for each task, and they are also not applicable for each system output in each task. The applicability here is defined by whether or not applying the perturbation function $f(\cdot)$ to y will produce a y' that is worse with high probability.

Swapping. We apply this perturbation method to the outputs of all three tasks. We tokenize lowercased sentences for machine translation and automatic speech recognition using `nltk`'s `word_tokenize` function. We apply this preprocessing method to all the rule-based perturbation functions. Then, we randomly pick a pair of words and swap their positions. Similarly, for ranking, we pick a pair of items within the ranked list and swap them. To offset the fact that we have a limited number of perturbations for ranking, we swap several items within the list, while in machine translation and automatic speech recognition, we only switch the position of a pair of words.

Removal. We apply this perturbation method to outputs from machine translation and automatic speech recognition. After preprocessing the output, we randomly select a word from the sentence and remove it.

Insertion. We apply this perturbation method to outputs from machine translation and automatic speech recognition. After preprocessing the output, we insert a randomly selected word from a list of words corresponding to the language of the system output. To reduce computation time, we store a corpus of English, Russian, and German words from the dataset and randomly select one from the list.

Synonym substitution. We apply this perturbation method to the output of machine translation. We have to take a language-specific approach to obtain synonyms: for English, we used `nltk` `wordnet`'s `synsets`; for Russian, we used `synsets` from `RuWordNet`[10]; for German, we performed an API call using the `PyMultiDictionary`[11] li-

[10] https://www.ruwordnet.ru/ru

[11] https://github.com/ppizarror/PyMultiDictionary

brary. Not every word has a synonym, so we iterate through the sentence and substitute the first word with a synonym.

Pronoun substitution. We apply this perturbation method to the output of machine translation. We adopted a strictly rule-based approach, defining a set of pronouns for each language (English, Russian, and German). We loop through the words in the sentence, look at all available pronouns, and replace one with a random pronoun from the predefined list.

Named entity substitution. We apply this perturbation method to the output of machine translation. Similar to pronoun substitution, we adopted a strictly rule-based approach, where for each language, we collect a list of named entities by performing named entity recognition on all the sentences using `spacy`[12]. Then, for each sentence from the system output, we look for named entities using the same approach and replace them with a randomly named entity from our cached list of named entities corresponding to the language.

Negation. We apply this perturbation method to the output of machine translation. For English outputs, we perform POS tagging using `nltk`. Whenever we see a verb, adjective, or auxiliary verb, we insert a 'not' in the appropriate position (before the word in the first two cases and after the word in the latter). For German, instead of performing POS tagging, we defined a list of common auxiliary verbs, adjectives, and verbs and then added a 'nicht' in the appropriate position. We do the same thing for the Russian outputs.

Negation removal. We apply this perturbation method to the output of machine translation. Semantically speaking, negating a sentence is the same as removing a negation (for example, removing the word 'not' from a sentence results from adding a double negation to a sentence). However, we operationalize this differently in the implementation. We adopt a strictly rule-based approach where, for each language, we look for instances of negation like "not", "didn't", "never", and replace them with their negated counterpart (or for the case of words like "never", just removed them entirely).

Phonetic substitution. We apply this perturbation method to outputs from automatic speech recognition. Like several methods described above, we adopted a rule-based method to replace consonants and vowels (or consonant and vowel groups) with something phonetically similar. For example, we can perform a `t -> d` substitution to `water -> wader`. We do this using `regex`.

LLM-based perturbation. We apply this perturbation method to outputs from machine translation and automatic speech recognition. We employ a `Llama-3-8B-instruct` model (AI@Meta, 2024) to perturb a system output. We use the following hyperparameters: `temperature=0.3`, `max_response_tokens=200`, and `top_p=1.0`. We run inference on a single NVIDIA A6000 GPU. Below is an example prompt that we use to perturb a machine translation system output:

> "I have the following source text (SOURCE_ENGLISH), reference translation (REFERENCE_GERMAN), and machine translation system output (MT_OUTPUT_GERMAN). Your task is to make MT_OUTPUT_GERMAN slightly worse by introducing minor machine translation errors while keeping the output generally understandable.
>
> SOURCE_ENGLISH: "Fiddes also addressed allegations of child sexual abuse against Jackson by Wade Robson and James Safechuck, which were aired in the controversial, Emmy Award-winning documentary Leaving Neverland."
>
> REFERENCE_GERMAN: "Fiddes hat auch die Anschuldigungen des sexuellen Kindermissbrauchs von Wade Robson und James Safechuck durch Jackson angesprochen, die in der kontroversen Dokumentation „Leaving Neverland" die auch einen Emmy gewonnen hat, dargelegt wurden."
>
> MT_OUTPUT_GERMAN: "Fiddes befasste sich auch mit Vorwürfen des sexuellen Missbrauchs von Kindern gegen Jackson durch Wade Robson und James Safechuck, die in dem umstrittenen, Emmy-prämierten Dokumentarfilm Leaving Neverland ausgestrahlt wurden."
>
> Please modify MT_OUTPUT_GERMAN to introduce minor typical machine translation errors, making the output only slightly worse. Do not include any explanations or

[12] https://spacy.io/

> additional text. Only return the perturbed output."

The prompt above yields the following output:

> "Fiddes befasste sich auch mit Vorwürfen des sexuellen Missbrauchs von Kindern gegen Jackson von Wade Robson und James Safechuck, die in dem umstrittenen, Emmy-premierten Dokumentarfilm Leaving Neverland ausgestrahlt wurden."

Similarly, we present an example prompt that we use to perturb an automatic speech recognition system output:

> I have the following reference text (ASR_REFERENCE_TEXT) and an ASR system output (ASR_SYSTEM_OUTPUT). Your task is to make ASR_SYSTEM_OUTPUT slightly worse by introducing minor ASR errors.
>
> ASR_REFERENCE_TEXT: "concord returned to its place amidst the tents" ASR_SYSTEM_OUTPUT: "concord returned to its place amidst the tents"
>
> Please modify ASR_SYSTEM_OUTPUT to introduce minor typical ASR errors, making the output only slightly worse. Do not include any explanations or additional text. Only return the perturbed output.

The prompt above yields the following output:

> "concord returned to its pace amids the tents"

We use the same prompt template for all system outputs used for the meta-evaluation. In both cases, we do not guide the model by using examples such as "perturb the model by dropping some words" or other detailed instructions. This will allow the LLM to 'explore' different ways to perturb an output beyond our rule-based methods.

Task	Perturbation	Description
MT	Removal	Drop a random word from the string, e.g. "John said he did not like the cake baked."
	Insertion	Add a random word to the string at a random position, e.g. "John said he did not really like the cake she baked."
	Swapping	Switch the position of a random pair of words from the string, e.g."John he said did not like the cake she baked."
	Synonym sub.	Substitute a random word with its synonym, e.g. "John said he did not enjoy the cake she baked."
	Pronoun sub.	Substitute a random pronoun in the string with another random pronoun, e.g. "John said she did not like the cake she baked"
	Named entity sub.	Substitute a random named entity in the string with another named entity, e.g. " Mike said he did not like the cake she baked."
	Negation	Negate a random word in the string, e.g. "John said he did not like the cake she did not baked."
	Negation removal	Remove instances of negation from the string, e.g. "John said he liked the cake she baked."
	LLM	Prompt an LLM to perturb the string, e.g. "John mentioned he didn't enjoy the cake she made."
ASR	Removal	Drop a random word from the string, e.g. "John said he did not like cake she baked."
	Insertion	Add a random word to the string at a random position, e.g. "John said he did not like the cake that she baked."
	Swapping	Switch the position of a random pair of words from the string, e.g. "John said he did like not the cake she baked."
	Phonetic sub.	Replace a random letter (or a letter group) with phonetically similar ones, e.g. "John sed he did not like the cake she bakked."
	LLM	Prompt an LLM to perturb the string, e.g. "John sed he did not like cake she baked."
Ranking	Swapping	Switch the position of an item with another item within the ranked list, e.g. Original ranking: [Cake, Pie, Cookie] Perturbed: [Cookie, Pie, Cake]

Table 4: Full list of perturbations for MT, ASR, and Ranking tasks. The original unperturbed sentence for the MT and ASR examples is "John said he did not like the cake she baked." We outline how each perturbation method is implemented in Appendix A.2.2.

A.2.3 The effect of adding more perturbations

In this section, we will discuss how we can reduce the effects of potential confounding between the choice of perturbation and the different metrics being evaluated. We can detect whether a metric confounds with a specific perturbation method by observing its variance across the different perturbation methods, as initially highlighted in Section 4.2.

We can think about how an adversarial metric can take advantage of such confounders and how introducing more perturbations would help. Let us say we only adopt the removal perturbation function in our framework. Let μ_{adv} be the adversarial metric that assigns a score based on the length of the system output y, such that $\mu_{adv}(x, y) = len(y)$. Given that y' will always be shorter than y due to the perturbation function that we chose, it is always true that $\mu_{\text{adv}}(x, y')$ is always lower than $\mu_{\text{adv}}(x, y)$, hence we will have $ACC_{\mu}(Q_x) = 1$ for μ_{adv} regardless of the context. It is likely that μ_{adv} is not correlated with μ^* in measuring the true utility of the output.

Now, let us say we include two other versions of y' obtained by applying the insertion and swapping perturbation function. In this case, $len(y) \leq len(y')$ will always be true; therefore, the value of $\mu_{adv}(x, y)$ will always be less than $\mu_{adv}(x, y')$. Hence, the adversarial metric will always achieve $ACC_{\mu}(Q_x) = 0$. If we aggregate the values from the different perturbations, we will get a $ACC_{\mu}(Q_x) = 0.33$.

We can observe this phenomenon from empirical examples of perturbing machine translation outputs using nine different techniques in Figure 6. As discussed in Section 4.2, the swapping perturbation function penalizes disproportionately ROUGE-1, as the metric measures the position-agnostic lexical alignment between two outputs. Consequently, we can see that in Figure 6b, the metric accuracy values for ROUGE-1 are consistently 0. However, when we observe its metric accuracy values under the other perturbation functions, they fluctuate at values above 0. If we compare this with other metrics, such as the family of UNITE metrics, their metric accuracy values have a smaller variance between the different perturbation functions.

From the examples above, we can observe that if a metric confounds with specific perturbation methods, its local metric accuracy scores across different contexts would have a high variance when we change the perturbation method. This follows from the definition of variance: when a variable (in our case, the local metric accuracy under different perturbations) has a high variance, it means that its values are spread out over a wide range, indicating that under some perturbations the local metric accuracy is high, while it is low in others, instead of being close to the average value. Given that we keep the contexts the same when measuring the variances, we can also say that if the local metric accuracy of a particular metric across different contexts shows a high variance when changing the perturbation method, then it is likely caused by the presence of confounding between the perturbation and the metric design.

We will show empirically that this variance will be reduced if we combine different perturbation methods for machine translation. First, we obtain all the possible combinations of perturbations: each $k = [1, 2, 3, 4, 5, 6, 7, 8]$ denoting the number of perturbations in each combination would have $\binom{9}{k}$ instances each. We omit $k = 9$ because we would just have one combination of 9 different perturbations. For example, a combination of one perturbation method would be $\binom{9}{1} = 9$ different single perturbations, while a combination of two perturbation methods would be $\binom{9}{2} = 36$ different pairs, where each y would have two different y's, and so on. Let us denote the different groups of combinations c_k, where each c_k contains all the possible perturbation combinations of size k. We randomly chose $\min(\binom{9}{k}, 10)$ for each c_k to speed up the calculation. For each combination in $c_{(k,i)}$, we compute the means of the local metric accuracies across the different machine translation systems. For each metric μ, we compute how it varies with the local metric accuracies of the other possible perturbation combinations of size k.

We show the above result in Figure 5. We can interpret it as follows: the variance between groups of two perturbation combinations (for example, between $\langle f_{\text{removal}}, f_{\text{swap}} \rangle$ and $\langle f_{\text{removal}}, f_{\text{insertion}} \rangle$) is higher than between groups of three perturbation combinations (for example, between $\langle f_{\text{removal}}, f_{\text{swap}}, f_{\text{insertion}} \rangle$ and $\langle f_{\text{removal}}, f_{\text{LLM}}, f_{\text{negation}} \rangle$) for metrics such as BLEURT and ROUGE-1. Therefore, we can see that by increasing the number of perturbations for a single instance of y, we will see reduced effects on confounders.

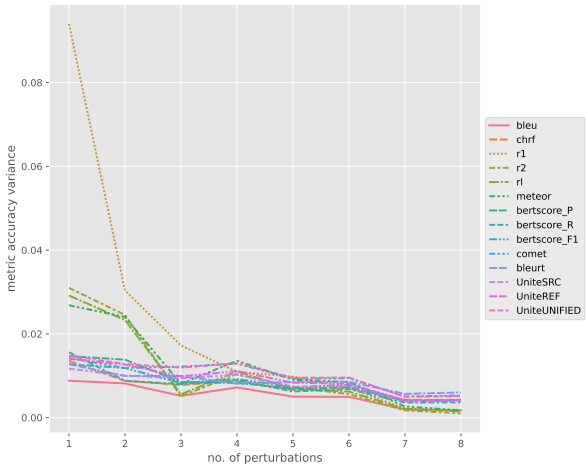
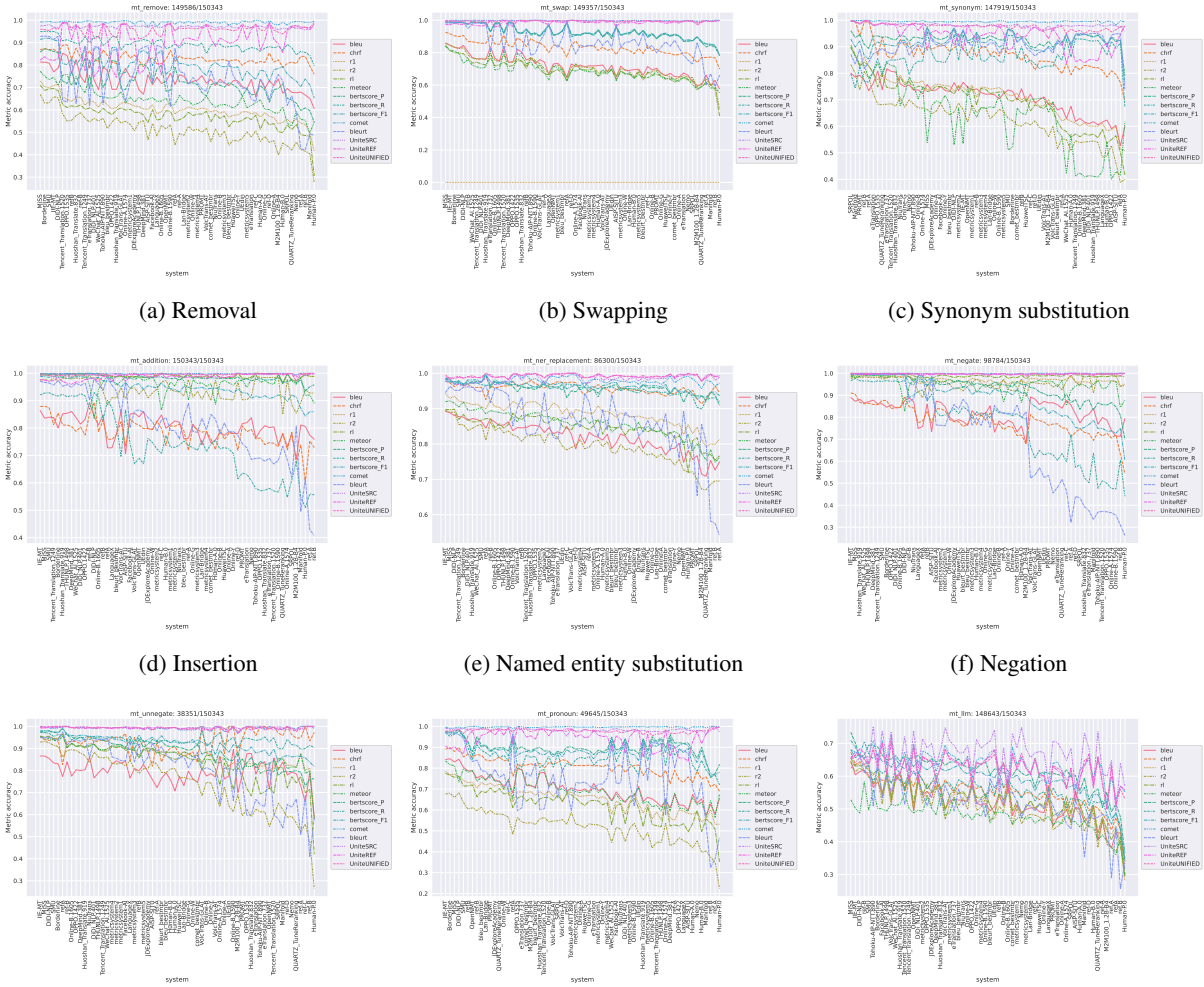


Figure 5: Local metric accuracy variance between the different number of perturbation combinations.



(a) Removal (b) Swapping (c) Synonym substitution

(d) Insertion (e) Named entity substitution (f) Negation

(g) Negation removal (h) Pronoun substitution (i) LLM-based perturbation

Figure 6: Metric accuracy graphs from using a single perturbation technique on each instance.

A.3 Metric accuracy graphs

A.3.1 Machine translation

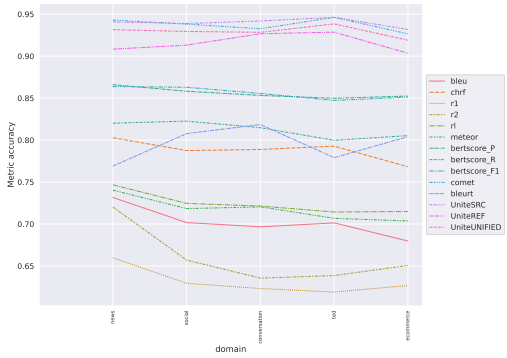


Figure 7: Metric accuracy for Machine Translation metrics across the different DOMAINS.

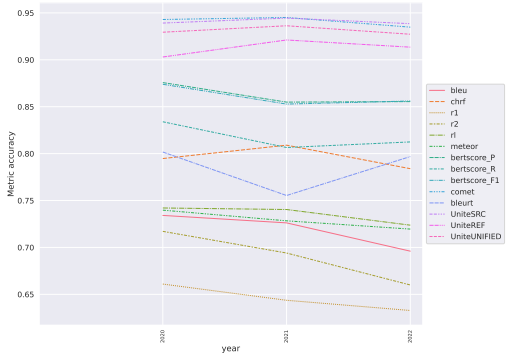


Figure 8: Metric accuracy for Machine Translation metrics across the different YEARS.

A.3.2 Automatic speech recognition

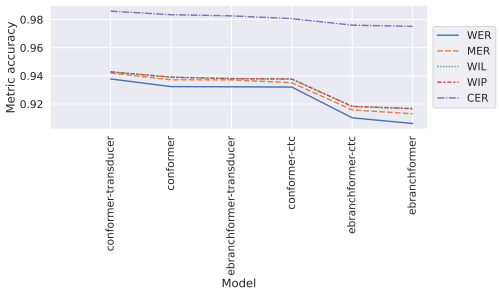


Figure 9: Metric accuracy for Automatic Speech Recognition metrics across the different SYSTEMS.

A.4 Statistical Significance Testing results

The statistical significance test results referred to for Machine Translation (Section 5.1), Automatic Speech Recognition (Section 5.2), and Ranking (Section 5.3), are shown in Tables 5, 6, 7, respectively.

Metric	χ^2 stat	p-value	D.O.F.
bleu	3528.548807	0.0	61
chrf	2821.829504	0.0	61
r1	1918.071678	0.0	61
r2	5709.352541	0.0	61
rl	3475.847682	0.0	61
meteor	2328.442112	0.0	61
bertscore_P	4587.738541	0.0	61
bertscore_R	9192.162703	0.0	61
bertscore_F1	5485.386924	0.0	61
comet	702.717102	7.264963e-110	61
bleurt	19874.815221	0.0	61
UniteSRC	699.907240	2.631570e-109	61
UniteREF	2527.075118	0.0	61
UniteUNIFIED	834.976241	2.211392e-136	61

Table 5: χ^2 Contingency Test Results for Machine Translation Metrics

Metric	χ^2 stat	p-value	D.O.F.
WER	677.507568	2.884115e-99	72
MER	561.242007	7.168449e-77	72
WIL	1328.151350	2.412446e-230	72
WIP	3956.386096	0.0	72
CER	1085.985573	8.197217e-181	72

Table 6: χ^2 Contingency Test Results for Automatic Speech Recognition Metrics

Metric	χ^2 stat	p-value	D.O.F.
map	154124.458407	0.0	20
Rprec	105742.720203	0.0	20
recip_rank	47484.273357	0.0	20
iprec_at_recall_0.00	57663.526828	0.0	20
iprec_at_recall_0.10	178593.017387	0.0	20
iprec_at_recall_0.20	237843.573156	0.0	20
iprec_at_recall_0.30	234700.982451	0.0	20
iprec_at_recall_0.40	179188.961383	0.0	20
P_5	122190.649832	0.0	20
P_10	143434.121966	0.0	20
P_15	147076.902489	0.0	20
P_20	146921.819515	0.0	20
P_30	140459.426693	0.0	20
recall_5	122190.649832	0.0	20
recall_10	143434.121966	0.0	20
recall_15	147076.902489	0.0	20
recall_20	146921.819515	0.0	20
recall_30	140459.426693	0.0	20
ndcg_cut_5	148042.996719	0.0	20
ndcg_cut_10	173763.267978	0.0	20
ndcg_cut_15	179134.760304	0.0	20
ndcg_cut_20	179938.561533	0.0	20
ndcg_cut_30	175786.549438	0.0	20

Table 7: χ^2 Contingency Test Results for Ranking Metrics