When Gradients Collide: Failure Modes of Multi-Objective Prompt Optimization for LLM Judges

Parth Darshan[♢*]
[♢]IIT Jodhpur
b22cs040@iitj.ac.in

Abhishek Divekar[♠†*]
[♠]Amazon
adivekar@amazon.com

Abstract

Customizing an LLM judge to a specific problem or domain often involves optimizing its prompt across multiple evaluation criteria simultaneously. Textual gradient methods automate this for a single judge criterion, however they produce natural-language critiques, not numerical vectors. Thus, the conflict-resolution toolkit of multi-task learning (PCGrad, MGDA) does not apply to this multi-objective textual gradient setting. We extend TextGrad to the multi-objective setting and test four decomposition modes of textual gradient optimizers by varying how much cross-objective information the loss, gradient and optimizer LLMs share. We find the gradient's task-focus drops by 59% (9.0 to 3.7 out of 10) when the gradient LLM must provide feedback on multiple criteria jointly. Separately, we observe that naively combining single-objective optimized instructions into a single prompt degrades Spearman ρ from 0.305 to 0.220 (-0.085). These results identify two separable failure modes: optimization-time *gradient dilution* and inference-time *instruction interference*, which together constrain the design space for multi-objective judge optimization using textual feedback.[1]

1 Introduction

Modern judges, whether human or LLMs, are used to evaluate text along multiple quality dimensions at once. SUMMEVAL (Fabbri et al., 2021) scores summaries on four dimensions simultaneously; MT-Bench (Zheng et al., 2023) spans eight categories from coding to roleplay. Kim et al. (2024) developed Prometheus 2 specifically because existing evaluators "do not possess the ability to evaluate based on *custom evaluation criteria*". Prompt optimization methods like TextGrad (Yüksekgönül et al., 2025), OPRO (Yang et al., 2024), and GEPA (Agrawal et al., 2026) can improve prompts for LLM judges automatically, but they optimize a single objective. Whether they extend to multi-objective judges, where one prompt must score several evaluation dimensions at once, remains less studied.

The core challenge is structural. When multi-task deep learning models encounter conflicting task gradients, methods like PCGrad (Yu et al., 2020) and MGDA (Sener and Koltun, 2018) resolve conflicts via projection or constrained optimization. The conflict-resolution tools of numerical multi-task learning do not apply directly, because textual gradients lack the vector-space structure these methods require. As textual gradients are natural-language strings, they do not have equivalent concepts of magnitude, inner products, or linear subspaces. The instruction "Make the coherence rubric more specific" cannot be numerically projected against "simplify the fluency criteria".

Prior work addresses related problems but not this intersection. For example, intra-task feedback quality has been studied: recently Chu et al. (2026) identify *rule dilution*, when heterogeneous error modes are aggregated within a single optimization step, and Melcer et al. (2025) show that the gradient analogy does not accurately explain why Automatic Prompt Optimization (APO) methods succeed. Multi-objective prompt optimization also exists: Jafari et al. (2024) find that MGDA underperforms volume-based alternatives for discrete prompt tokens. Multi-task judge *training* is studied extensively (Whitehouse et al., 2026; Yang et al., 2026; Wang et al., 2025), but these methods update weights, not prompts. To our knowledge, no prior work studies multi-objective judge *prompt* optimization using textual gradients.

We propose and test four plausible decomposition modes of textual gradient optimization for

[*]Equal contribution. [†]Research lead and corresponding author. Author was employed at Amazon when this work was conducted, but the work was performed independently and did not use Amazon data, resources or confidential information.

[1]Code, data and prompts available at https://github.com/adivekar-utexas/when-gradients-collide

Proceedings of the Second Workshop on Customizable NLP: Progress and Challenges in Customizing NLP for a Domain, Application, Group, or Individual (CustomNLP4U), pages 216–228

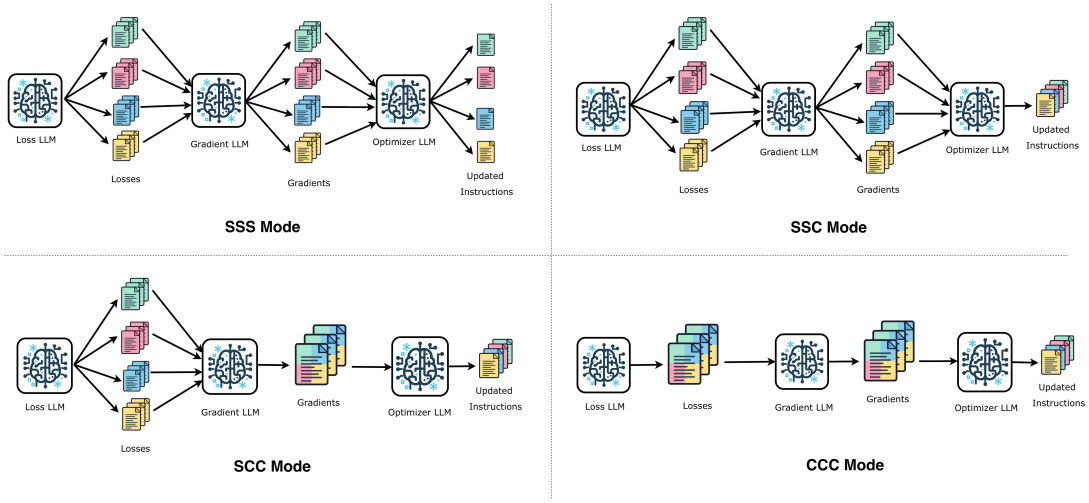

Figure 1: Overview of the optimization pipeline. Each step consists of four stages: (1) the task model predicts scores, (2) the loss LLM critiques predictions against ground truth, (3) the gradient LLM converts critiques into instruction edits, and (4) the optimizer LLM rewrites the prompt. The decomposition mode determines whether each stage operates per-task (Separate) or over all tasks jointly (Combined). Only the per-task instruction text is updated; the prompt skeleton and output format remain frozen throughout optimization.

multi-objective judges: SSS, SSC, SCC, CCC, where the naming encodes whether each pipeline stage (loss, gradient, optimizer) processes tasks separately (S) or combined (C). We also introduce two process-level diagnostics: *gradient specificity* (how targeted each gradient is to a single task) and *feedback adherence* (whether the optimizer follows the gradient). We evaluate on SUMMEVAL (Fabbri et al., 2021) with four criteria, using two validation gate settings (MAE gating and no gating).

Our results reveal a consistent structure in how and why multi-criteria prompt optimization stagnates. In 6 of 10 configurations with Qwen3 judges, optimization never exceeds the initial generic prompt (§5, Table 1); only SINGLE-TASK with `val=mae` improves (+0.031 Spearman at step 2). Our diagnostics localize this effect: gradient specificity drops by 59% (9.0 to 3.7) when the gradient LLM processes all tasks jointly §6.1, Table 4). Feedback adherence remains high (7.8 to 8.8), indicating that the optimizer incorporates the gradient's suggestions regardless of their specificity. A separate oracle experiment shows that even independently optimal per-task instructions *degrade* from 0.305 to 0.220 (−0.085 Spearman) when combined into one prompt (§6.3, Table 3). These results identify two separable failure modes: optimization-time *gradient dilution* and inference-time *instruction interference*.

Our contributions are as follows: (1) we present an empirical study of multi-criteria textual gradient optimization for LLM judges across four decomposition modes (§4, §5, Table 1); (2) we propose two process-level diagnostics (*gradient specificity* and *feedback adherence*) that localize issues in optimization to gradient quality, not optimizer compliance (§6.1, Figure 3, Table 2). (3) we conduct an oracle-instruction experiment which isolates inference-time *instruction interference* as a failure mode distinct from optimization-time dilution (§6.3, Table 3).

2 Related Work

Textual Gradient Methods. Early methods in Prompt optimization used scalar signals: OPRO (Yang et al., 2024) rewrites prompts by conditioning on (prompt, score) histories, and APE (Zhou et al., 2023) generates candidates from demonstrations and selects the highest-scoring variant. PROTEGI (Pryzant et al., 2023) replaced scalar signals with *textual gradients*: natural-language critiques that guide prompt rewrites. TextGrad (Yüksekgönül et al., 2025) extended this to multi-component computation graphs, propagating critiques through LLM pipelines in analogy to backpropagation.

Subsequent work has questioned the gradient analogy. GPO (Tang et al., 2025) decomposed textual optimization into two factors (update direction and update method), drawing analogies to gradient, momentum, and learning rate, but found that adding a reflection step hurts performance. Melcer

et al. (2025) showed empirically that the gradient decomposition (the chain-rule structure central to the analogy) does not consistently improve prompt optimization, and that the gradient metaphor does not accurately explain why APO methods succeed. GEPA (Agrawal et al., 2026) evolves language programs via reflective mutation with Pareto selection. To our knowledge, all methods above optimize a single objective and none provide a mechanism to observe or control how per-task feedback interacts during optimization.

Multi-Objective Prompt Optimization. Multi-objective prompt optimization is a nascent area with two distinct approaches. Population-based methods maintain a Pareto front of candidate prompts: MOPO (Menchaca Resendiz and Klinger, 2025) applies NSGA-II (Deb et al., 2002) with LLM-based mutation to affective text generation, and ParetoPrompt (Zhao et al., 2025) decomposes objectives into scalarized subproblems. MORL-Prompt (Jafari et al., 2024) adapts multi-objective reinforcement learning to discrete prompt tokens but found that MGDA (Sener and Koltun, 2018) underperforms the simpler product-of-rewards at balancing competing objectives. Evolutionary approaches include EvoPrompt (Guo et al., 2024) for single-objective and Baumann and Kramer (2024) for multi-objective sentiment balancing. These methods operate at the *candidate-selection* level: they choose which prompts to keep from a population. To our knowledge, none of the above studies how per-task feedback interacts *within* a single textual gradient trajectory, which is the setting we investigate.

LLM-as-a-Judge Evaluation. MT-Bench and Chatbot Arena (Zheng et al., 2023) established the paradigm of LLM-based evaluation where human annotation is expensive. Subsequent works use a fixed prompt to improve judge quality through tuning weights on rubric-grounded data (Kim et al., 2024), reinforcement learning (Whitehouse et al., 2026), debiasing (Yang et al., 2026), and preference optimization (Wang et al., 2025). A parallel line of work optimizes the evaluation prompt itself. CARO (Chu et al., 2026) identifies *rule dilution* when heterogeneous error modes are aggregated into a single optimization step. RRD (Shen et al., 2026) recursively decomposes rubrics to improve coverage and remove redundancy. MPO (Sharma and Henley, 2026) applies section-local textual gradients to individual prompt components (role, context, constraints) independently. MAPGD (Han et al., 2025) coordinates multiple gradient agents, using semantic similarity to resolve conflicting edits. All these operate on a single evaluation objective, or decompose along axes other than evaluation criteria. None jointly optimizes a judge prompt across multiple criteria while preserving per-task gradient observability (i.e., the ability to trace how each criterion's feedback shaped each rewrite), the setting we study.

3 Method: Multi-objective TextGrad

TextGrad pipeline. We implement a 4-stage optimization loop based on TextGrad (Yüksekgönül et al., 2025). At each step, a **task LLM** predicts objective-scores for a minibatch of examples using the current prompt. A **loss LLM** critiques the predictions by comparing them to ground-truth annotations and produces a natural-language "loss" for each example (called a *reflection* by some authors). A **gradient LLM** aggregates the per-example losses into a "textual gradient": a structured set of instruction-level edit suggestions. We restrict gradients to 3 paragraphs. An **optimizer LLM** rewrites the prompt based on the gradient. Only the per-objective instructions are modified; the prompt skeleton (role preamble, output format, few-shot examples) is frozen throughout (see Appendix E for the full initial prompt and an example of optimized instructions). This design isolates the effect of objective-level instruction modifications from possible confounds introduced by structural prompt changes.

Decomposition modes. We extend TextGrad to the multi-objective setting stagewise. As shown in Figure 1, we parameterize the multi-objective interaction via a 3-letter *decomposition code*. Each letter denotes whether a stage operates in **S**eparate (per-task) or **C**ombined (all-tasks-joint) mode, applied to the loss, gradient, and optimizer stages respectively.

Four modes span the design space. **SSS** operates all three stages independently per task. **SSC** computes loss and gradient per task, but the optimizer receives all four gradients simultaneously and rewrites the prompt in a single call. **SCC** computes only the loss per task; the gradient LLM receives critiques from all four tasks and produces a unified set of edits. **CCC** operates all stages jointly. We also include a **Single-Task** baseline in which each objective is optimized in a completely inde-

	MAE validation				No validation			
Mode	**Initial**	**Best (step)**	**Δ**	**HVI**	**Initial**	**Best (step)**	**Δ**	**HVI**
SINGLE-TASK	0.274	**0.305** (2)	**+0.031**	—	0.269	0.284 (5)	**+0.015**	—
SSS	0.284	0.284 (0)	+0.000	2.749	0.283	0.283 (0)	+0.000	2.867
SSC	0.289	0.289 (0)	+0.000	2.832	0.283	**0.291** (2)	+0.007	2.845
SCC	0.282	0.282 (0)	+0.000	2.801	0.282	0.282 (0)	+0.000	2.779
CCC	0.285	0.296 (9)	+0.012	**2.900**	0.287	0.287 (0)	+0.000	**2.983**

Table 1: Main results on SUMMEVAL using Qwen3. We measure the task-averaged Spearman correlation (mean over $N = 3$ runs). Δ is the improvement from the initial prompt to the best step. HVI is the hypervolume indicator at step 12 (higher = more Pareto-diverse prompts). We evaluate four decomposition modes (SSS through CCC) plus a single-task optimization baseline (SINGLE-TASK), covering the full spectrum from separate to joint optimization. In 6 of 10 configurations, the initial generic prompt ("*Rate from 1 to 5*") is never exceeded.

pendent run, and evaluated through its own forward pass; i.e., the reported Spearman is the average of four per-criterion evaluations, each using a prompt specialized to a single objective.

The key architectural boundary lies between SSC and SCC. In SSC, the gradient LLM sees one task at a time. In SCC, it must reconcile feedback from all four tasks into a coherent edit plan.

Validation Gating. We evaluate two validation strategies. Under **val=mae**, a candidate prompt is accepted only if its mean absolute error on a held-out validation set does not exceed that of the current prompt. This acts as a monotonic filter that prevents prompt-regression. Under no validation gate, every candidate is accepted unconditionally; we can observe the optimization trajectory without gating. For each configuration, we run 3 independent trials with different random seeds, each for 12 optimization steps. We report the mean and standard deviation of task-averaged Spearman $\bar{\rho}$ across the 3 runs.

4 Experimental Setup

Datasets. We evaluate on SUMMEVAL (Fabbri et al., 2021), a summarization meta-evaluation benchmark. The original dataset contains source news articles, each paired with summaries from 16 different summarization systems, with expert annotations from multiple annotators per pair. We randomly subsample this into 160 pairs for training (used as optimization batches) and 480 pairs for held-out evaluation. Each pair is scored on four dimensions (fluency, relevance, coherence, consistency) on a 1-5 scale.

These four dimensions are the tasks in our multi-task optimization setting: the judge prompt must produce accurate scores across all four simultaneously. We report Spearman rank correlation (ρ) between predicted and human scores as the primary metric, following prior work on LLM-based evaluation (Liu et al., 2023). Unless stated otherwise, all reported results are *task-averaged* Spearman, the arithmetic mean of per-task ρ values across the four dimensions.

Models. Our main results are evaluated on the Qwen3 family (Yang et al., 2025) using Qwen3-8B as the task LLM and Qwen3-235B-A22B as others. We use a higher optimizer temperature ($T = 0.7$) to encourage diverse rewrites, and a lower loss and gradient temperatures ($T = 0.3$) promote consistent critiques. For the task LLM we set $T = 1.0$ to allow resampling in case of JSON formatting errors. In Appendix D, we show results on DeepSeek v4 Flash and Pro (DeepSeek-AI, 2026).

5 Results

Table 1 reports task-averaged Spearman ρ for each decomposition mode and validation configuration, averaged over $N = 3$ runs. In 6 of 10 configurations, the best prompt is the initial generic prompt ("*Rate from 1 to 5*"): optimization either fails to improve or actively degrades performance. The only multi-task mode that improves is CCC with MAE validation and SSC without validation, which achieve modest gain over 12 steps. Only the single-task baseline with MAE validation achieves meaningful improvement: $+0.031$ Spearman at step 2 (Table 1, **bold**). This confirms that the TextGrad pipeline can improve individual-task prompts when gradient signal is not contaminated by cross-task information.

We plot optimization trajectories in Figure 2, revealing the dynamics behind these aggregates. Without a validation gate (`val=none`, bottom row), SSC drops from 0.283 to 0.184 by step 7; SCC

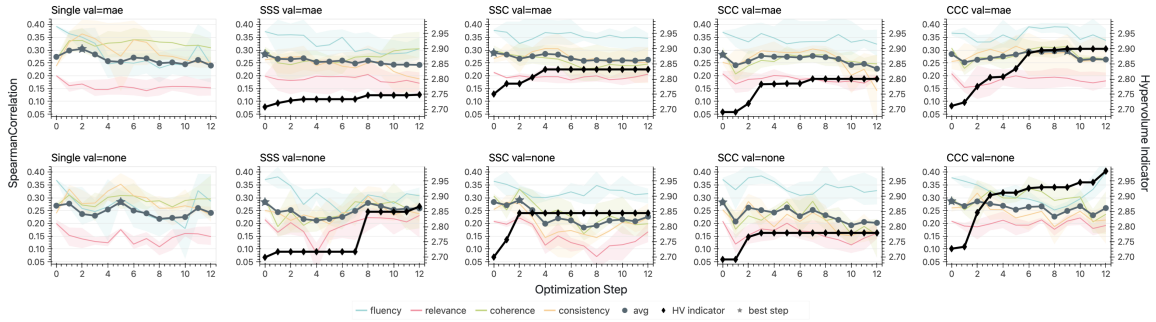

Figure 2: Per-task Spearman ρ for each optimization steps on SUMMEVAL with Qwen3. We average over $N=3$ runs (shaded bands show min to max). Each column shows one of the five decomposition modes. On the top row we apply validation-MAE to gate prompts at each step, while bottom row has no gating. Gray line indicates task-averaged ρ; stars mark best step. Black diamonds (right axis) denote the hypervolume indicator for the prompt-candidates accumulated over steps. Without validation gating, multi-task modes degrade, and with it, they plateau.

shows comparable degradation. The optimizer proposes changes that improve the training-batch loss but harm held-out generalization. MAE validation partially mitigates this by rejecting spurious updates. With the validation gate active (top row), trajectories flatten rather than decline: the gate prevents catastrophic degradation but cannot compensate for uninformative gradients.

Performance degrades roughly as Single-Task > SSS, SSC, SCC, CCC without validation gating. This is consistent with the hypothesis that increasing cross-task coupling amplifies gradient dilution. The pattern is non-monotonic under MAE validation, however: CCC slightly outperforms SSC. Full coupling may occasionally produce complementary gradients that survive the validation gate. The process-level diagnostics in §6 disentangle these effects.

Despite stagnation in Spearman, the hypervolume indicator (HVI) shows an increasing trend. For CCC, HVI grows continuously; the optimizer discovers diverse specialist prompts that expand the Pareto front, even when no single prompt dominates the initialization on all four tasks. Menchaca Resendiz and Klinger (2025) report a similar pattern: multi-objective prompt optimization expands the Pareto front at modest per-objective cost. In our setting, however, the per-task degradation is substantially larger when coupling is high.

We rerun with DeepSeek v4 under MAE validation in Appendix D, where we observe better absolute improvements in Spearman through optimization due to stronger models, but a similar trend: the *improvement* through prompt optimization is most effective for single-objective optimizations.

6 Analysis

The results in §5 establish *that* multi-objective textual gradient prompt optimization lags compared to single-task; this section investigates *why*. We identify two separable failure modes: optimization-time *gradient dilution* (§6.1) and inference-time *instruction interference* (§6.3). We report the evaluation prompts in Appendix F for reproducibility.

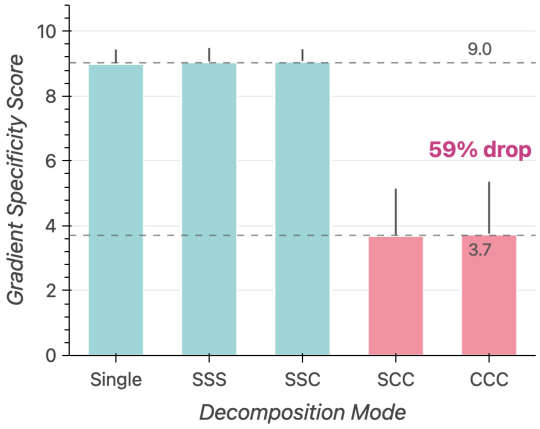

Figure 3: Gradient specificity (1 to 10 scale, higher is more task-focused) by decomposition mode.

6.1 Gradient Specificity and Dilution

To measure gradient quality directly, we evaluate each textual gradient for *task-focus*: the degree to which its improvement suggestions target a single evaluation criterion rather than offering generic advice. An LLM evaluator (Claude Sonnet 4.6, Anthropic (2026)) rates each gradient on a 1 to 10 scale. A score of 10 means the gradient addresses exactly one task's rubric; a score of 1 means it's so generic it could apply to any task (see Appendix F for the evaluation prompt). This diagnostic tests

Mode	MAE validation	No validation
Single-Task	8.70 ± 0.47	8.53 ± 0.60
SSS	8.84 ± 0.44	8.72 ± 0.54
SSC	7.94 ± 0.80	7.83 ± 0.78
SCC	8.08 ± 1.03	7.76 ± 1.49
CCC	7.90 ± 1.62	8.01 ± 1.28

Table 2: Feedback adherence scores (1 to 10 scale, mean $\pm$ std over all objectives, runs, and steps). All modes achieve high adherence, ruling out optimizer non-compliance as an explanation for multi-task optimization failure.

the cross-objective analogue of Chu et al.'s rule-dilution hypothesis. Concretely, we ask: does combining multiple evaluation criteria in a single gradient call dilute the criteria-specific signal?

We evaluate all gradients from steps 1 to 12 across all four modes. A sharp transition is present (Figure 3). Per-task modes (Single, SSS, SSC), where each gradient LLM call processes exactly one task, achieve a mean specificity of 9.0 ($\pm$0.3; Table 4, top rows). Cross-task modes (SCC, CCC), where the gradient LLM processes all four tasks in a single call, drop to 3.7 ($\pm$0.5; Table 4, bottom rows), a 59% reduction with zero overlap between the two groups.

Our gradient dilution results extend Chu et al.'s within-criterion finding to the cross-criterion setting. Chu et al. (2026) show that aggregating heterogeneous error modes in a single optimization step degrades rubric accuracy; we observe an analogous effect when multiple task gradients are combined in a single gradient call, degrading the per-task optimization signal. A per-task breakdown (Table 4) reveals that consistency is most diluted, while coherence retains moderate focus. This possibly occurs because coherence rubrics share surface-level vocabulary with the generic "writing quality" feedback the gradient LLM defaults to under multi-task load.

To confirm this cliff is structural when moving from per-task to cross-task gradients, in Appendix C we switch *only* the gradient LLM to a stronger LLM (`DeepSeek-V4-Pro`) while continuing to use Qwen3 for the other steps. We observe the same SSC to SCC specificity drop. Appendix D.2 replicates the overall gradient-specificity analysis using DeepSeek for all modes, showing the same per-task vs. cross-task gradient dilution pattern.

Method	Fl.	Rel.	Coh.	Con.	Avg ρ
Initial	**0.366**	0.208	0.308	0.256	0.284
Single-Task	0.350	0.168	**0.338**	**0.363**	**0.305**
Cherry-pick	0.303	**0.257**	0.215	0.105	0.220

Table 3: Oracle cherry-pick experiment (MAE validation). For each task, the single-task instruction with the highest test-set metric is selected and combined into one multi-task prompt. Despite oracle selection, combined instructions degrade below the individually optimized single-task performance, demonstrating inference-time instruction interference. We report both task-level and average ρ for the multi-task prompt.

6.2 Feedback Adherence

We also measure *feedback adherence*: the degree to which the optimizer LLM incorporates the textual gradient into its instruction edits (Table 2). We prompt Claude Sonnet 4.6 evaluator to output an adherence score on a 10-point scale (refer Appendix F for the prompt).

Across all modes and validation settings, adherence is uniformly high (7.8 to 8.8 on a 10-point scale), confirming that the optimizer faithfully implements whatever gradient it receives, even when those suggestions are generic rather than criterion-specific. This indicates the ceiling to multi-objective judge prompt optimization is gradient quality rather than optimizer compliance.

6.3 Inference-Time Instruction Interference

Gradient dilution explains why SCC and CCC fail to optimize effectively, but it does not explain why SSS and SSC optimizations also stagnate. These per-task modes produce highly specific gradients and high-adherence prompt updates, yet still fail to improve over the initialization. To diagnose this, we design an oracle cherry-pick experiment that isolates the *inference-time* component of multi-task failure. For each task, we select the single best instruction across all single-task optimization runs (the instruction with the highest held-out Spearman ρ for that task), then combine these four oracle-optimal instructions into one multi-task prompt and evaluate on the full test set.

We see that oracle-optimal instructions *degrade* when combined (Table 3), achieving 0.22 average Spearman, a drop of 0.085 from the single-task optimized performance of 0.305. In Appendix B we explore more variants by using different metrics to select the best instruction per task, and observe the same effect. These instructions each individually outperform the baseline on their respective tasks,

but combining them produces performance strictly worse than the generic initial prompt.

The primary mechanism appears to be instruction-length asymmetry. Optimization produces over-specified rubrics for some tasks (the fluency instruction expands to ∼800 tokens with detailed scoring anchors) while leaving others under-specified (the relevance instruction remains at ∼4 tokens of the initial prompt). When packed into a single prompt, verbose instructions receive disproportionate attention relative to brief ones at inference time.

This finding strengthens a result from Shen et al. (2026), who observe that naive rubric construction degrades GPT-4o preference-judgment accuracy by 13 points on JudgeBench. The degradation we observe is larger and occurs with oracle-selected rather than naively constructed instructions. Shen et al.'s result shows that *bad* rubrics hurt. Ours shows that individually *good* rubrics can hurt when combined. This implies that instruction interference is not resolvable by improving per-task optimization alone.

7 Conclusion

Multi-criteria textual gradient optimization for LLM judges exhibits two failure points that our decomposition study and process-level diagnostics expose. These are systematic failures, corresponding to distinct pipeline stages and affect different decomposition modes: (i) Gradient dilution operates at optimization time. When the gradient LLM must reconcile feedback from multiple criteria in a single call, its suggestions lose task-specificity (59% drop; Appendix Table 4). The optimizer propagates the low-specificity signal through to the final prompt; (ii) Instruction interference operates at inference time. Independently optimized per-task instructions degrade when combined into one prompt (Table 3) because instruction-length asymmetry causes verbose rubrics to receive disproportionate attention relative to brief ones.

For practitioners customizing judges to domain-specific criteria (Kim et al., 2024), these results indicate architectural changes are required before the multi-objective setting can work reliably. Addressing either failure mode alone is insufficient. Separate judge calls per criterion eliminates interference but multiplies inference cost. Conflict-aware gradient resolution adapted from numerical multi-task learning (Yu et al., 2020; Liu et al., 2021) could address dilution if textual gradients can be meaningfully embedded and projected. Our proposed diagnostics (gradient specificity and feedback adherence) provide the measurement tools to evaluate either approach.

8 Future Work

Our findings open several concrete directions to broader research on customized LLM evaluation.

Statistically reliable LLM diagnostics with PPI. Our diagnostics are LLM-judged, introducing an evaluator-bias. Prediction-Powered Inference (Angelopoulos et al., 2023; Boyeau et al., 2025) combines a small human-judged set with a large LLM-judged set to produce provably unbiased estimates; the hierarchical PPI extension of Divekar and Majumder (2026) is directly applicable here, since annotations are per-gradient but the quantity of interest is the per-mode mean. This would allow us to report the specificity gap as a confidence interval and scale diagnostics to hundreds of runs without scaling human annotation linearly.

Synthetic task generation for aligned criteria. A new direction is to *synthesize the criteria themselves* rather than treat them as fixed. Persona-driven synthesis at billion-person scale (Chan et al., 2024; Yu et al., 2023) and instruction-data generation with controlled diversity (Divekar and Durrett, 2024; Kowshik et al., 2024) can produce synthetic objectives that are complementary for optimization. If the synthesized tasks are mutually aligned by construction, the combined gradient should suffer less semantic drift, raising gradient focus and potentially mitigating both failure modes at their source.

Multi-objective critics for agentic workflows. Multi-objective judge prompts could serve as critics in agentic LLM systems, where a critic must track several quality dimensions of tool-use trajectories at once (Ding, 2026; Chuang et al., 2026; Rudman et al., 2026). Applying our optimization pipeline to such critics raises an open question: do gradient dilution and instruction interference still emerge when criteria are tool-grounded and partially verifiable, or does verifiability yield more robust gradients? Our diagnostics provide a principled way to measure this.

Mitigations. Our diagnostics motivate concrete mitigations. For *gradient dilution*, a specificity-aware router could fallback to per-task gradient LLM when a multi-task LLM specificity drops

below threshold, capturing CCC's hypervolume without losing task-focus. For *instruction interference*, we propose two avenues: (i) next-token attention masking for per-criterion output generation (eliminating interference as no cost), and (ii) length-aware instruction synthesis that normalizes rubric length during optimization so no single criterion dominates the attention budget.

Limitations

Our experiments are scoped to SUMMEVAL as it provides expert human annotations on four clearly separable criteria. Other benchmarks would validate our identified failure modes: BRIGHTER (Muhammad et al., 2025) tests whether dilution scales with task count and crosses language boundaries; ASAP++ (Mathias and Bhattacharyya, 2018) (per-trait essay-grading rubrics) tests whether the SCC to CCC gradient specificity cliff transfers beyond summarization; EMSCAD (Vidros et al., 2017), GitBugs (Patil et al., 2026), test heterogeneous classification criteria rather than ordinal quality scales.

Other prompt optimization paradigms may exhibit different multi-task dynamics, though our diagnostics are algorithm-agnostic and can be applied to any textual gradient approach.

Our sample size ($N = 3$ runs) limits statistical power, and we restrict our claims to effect sizes that are robust at this sample size (e.g., the 59% specificity drop and -0.085 cherry-pick degradation). Finally, gradient specificity and feedback adherence are scored by an LLM evaluator, which introduces a potential confound.

Ethics Statement

Optimized judge prompts inherit biases present in the underlying LLMs and in the human annotations used for evaluation; practitioners should audit optimized prompts before deploying them in sensitive evaluation contexts (e.g., content moderation or hiring). Our code and diagnostics will be released under an open-source license to support reproducibility.

References

Lakshya A. Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J. Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alexandros G. Dimakis, Ion Stoica, Daniel Klein, Matei Zaharia, and Omar Khattab. 2026. GEPA: Reflective prompt evolution can outperform reinforcement learning. In *The Fourteenth International Conference on Learning Representations*.

Anastasios N. Angelopoulos, Stephen Bates, Clara Fannjiang, Michael I. Jordan, and Tijana Zrnic. 2023. Prediction-powered inference. *Science*, 382(6671):669–674.

Anthropic. 2026. Introducing claude sonnet 4.6. Anthropic Blog.

Jill Baumann and Oliver Kramer. 2024. Evolutionary multi-objective optimization of large language model prompts for balancing sentiments. In *Applications of Evolutionary Computation (EvoApplications)*, pages 212–224. Springer.

Pierre Boyeau, Anastasios Nikolas Angelopoulos, Tianle Li, Nir Yosef, Jitendra Malik, and Michael I. Jordan. 2025. Autoeval done right: Using synthetic data for model evaluation. In *Forty-second International Conference on Machine Learning*.

Xin Chan, Xiaoyang Wang, Dian Yu, Haitao Mi, and Dong Yu. 2024. Scaling synthetic data creation with 1,000,000,000 personas. *arXiv preprint arXiv:2406.20094*.

Yucheng Chu, Hang Li, Kaiqi Yang, Yasemin Copur-Gencturk, Joseph Krajcik, Namsoo Shin, and Jiliang Tang. 2026. Confusion-aware rubric optimization for LLM-based automated grading. *arXiv preprint arXiv:2603.00451*.

Yun-Shiuan Chuang, Chaitanya Kulkarni, Alec Chiu, Avinash Thangali, Zijie Pan, Shivani Shekhar, Yirou Ge, Yixi Li, Uma Kona, Linsey Pang, and Prakhar Mehrotra. 2026. Toward scalable verifiable reward: Proxy state-based evaluation for multi-turn tool-calling llm agents. *Preprint*, arXiv:2602.16246.

Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, 6(2):182–197.

DeepSeek-AI. 2026. Deepseek-v4: Towards highly efficient million-token context intelligence.

Liang Ding. 2026. Adarubric: Task-adaptive rubrics for reliable llm agent evaluation and reward learning. *Preprint*, arXiv:2603.21362.

Abhishek Divekar and Greg Durrett. 2024. Synthesizrr: Generating diverse datasets with retrieval augmentation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages 19200–19227. Association for Computational Linguistics.

Abhishek Divekar and Anirban Majumder. 2026. PRECISE: Reducing the bias of LLM evaluations using prediction-powered ranking estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence (IAAI Track)*, volume 40, pages 39929–39938.

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.

Qingyan Guo, Rui Wang, Junliang Guo, Bei Li, Kaitao Song, Xu Tan, Guoqing Liu, Jiang Bian, and Yujiu Yang. 2024. Connecting large language models with evolutionary algorithms yields powerful prompt optimizers. In *The Twelfth International Conference on Learning Representations*.

Yichen Han, Bojun Liu, Zhengpeng Zhou, Guanyu Liu, Zeng Zhang, Yang Yang, Wenli Wang, Isaac Shi, Yunyan, Lewei He, and Tianyu Shi. 2025. MAPGD: Multi-agent prompt gradient descent for collaborative prompt optimization. In *NeurIPS 2025 Workshop on Scaling Environments for Agents*.

Yasaman Jafari, Dheeraj Mekala, Rose Yu, and Taylor Berg-Kirkpatrick. 2024. MORL-prompt: An empirical analysis of multi-objective reinforcement learning for discrete prompt optimization. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 9878–9889, Miami, Florida, USA. Association for Computational Linguistics.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2024. Prometheus 2: An open source language model specialized in evaluating other language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4334–4353.

Suhas S. Kowshik, Abhishek Divekar, and Vijit Malik. 2024. CorrSynth: A correlated sampling method for diverse dataset generation from LLMs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages 16076–16095. Association for Computational Linguistics.

Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. 2021. Conflict-averse gradient descent for multi-task learning. In *Advances in Neural Information Processing Systems*, pages 18878–18890.

Yang Liu, Dan Iter, Yichong Xu, Shuo Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: Nlg evaluation using gpt-4 with better human alignment. *ArXiv*, abs/2303.16634.

Sandeep Mathias and Pushpak Bhattacharyya. 2018. ASAP++: enriching the ASAP automated essay grading dataset with essay attribute scores. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*. European Language Resources Association (ELRA).

Daniel Melcer, Qi Chen, Wen-Hao Chiang, Shweta Garg, Pranav Garg, and Christian Bock. 2025. Textual gradients are a flawed metaphor for automatic prompt optimization. *arXiv preprint arXiv:2512.13598*.

Yarik Menchaca Resendiz and Roman Klinger. 2025. MOPO: Multi-objective prompt optimization for affective text generation. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 5588–5606, Abu Dhabi, UAE. Association for Computational Linguistics.

Shamsuddeen Hassan Muhammad, Nedjma Ousidhoum, Idris Abdulmumin, Jan Philip Wahle, Terry Ruas, Meriem Beloucif, Christine de Kock, Nirmal Surange, Daniela Teodorescu, Ibrahim Said Ahmad, David Ifeoluwa Adelani, Alham Fikri Aji, Felermino D. M. A. Ali, Ilseyar Alimova, Vladimir Araujo, Nikolay Babakov, Naomi Baes, Ana-Maria Bucur, Andiswa Bukula, and 29 others. 2025. BRIGHTER: BRIdging the gap in human-annotated textual emotion recognition datasets for 28 languages. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8895–8916, Vienna, Austria. Association for Computational Linguistics.

Avinash Patil, Siru Tao, and Aryan Jadon. 2026. Gitbugs: Bug reports for duplicate detection, retrieval augmented generation, triage, and more. *Preprint*, arXiv:2504.09651.

Reid Pryzant, Dan Iter, Jerry Li, Yin Lee, Chenguang Zhu, and Michael Zeng. 2023. Automatic prompt optimization with "gradient descent" and beam search. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7957–7968, Singapore. Association for Computational Linguistics.

William Rudman, Abhishek Divekar, Kanishk Jain, Sebastian Joseph, Stella S. R. Offner, Matthew Lease, Kyle Mahowald, Greg Durrett, and Junyi Jessy Li. 2026. Vesta: Visual exploration with statistical tool agents. *Preprint*, arXiv:2606.00384.

Ozan Sener and Vladlen Koltun. 2018. Multi-task learning as multi-objective optimization. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Prith Sharma and Austin Z. Henley. 2026. Modular prompt optimization: Optimizing structured prompts with section-local textual gradients. *arXiv preprint arXiv:2601.04055*.

William F. Shen, Xinchi Qiu, Chenxi Whitehouse, Lisa Alazraki, Shashwat Goel, Francesco Barbieri, Timon Willi, Akhil Mathur, and Ilias Leontiadis. 2026. Rethinking rubric generation for improving LLM judge and reward modeling for open-ended tasks. *arXiv preprint arXiv:2602.05125*.

Xinyu Tang, Xiaolei Wang, Wayne Xin Zhao, Siyuan Lu, Yaliang Li, and Ji-Rong Wen. 2025. Unleashing the potential of large language models as prompt optimizers: Analogical analysis with gradient-based

model optimizers. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence*.

Sokratis Vidros, Constantinos Kolias, Georgios Kambourakis, and Leman Akoglu. 2017. Automatic detection of online recruitment frauds: Characteristics, methods, and a public dataset. *Future Internet*, 9(1):6.

Peifeng Wang, Austin Xu, Yilun Zhou, Caiming Xiong, and Shafiq Joty. 2025. Direct judgement preference optimization. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 1979–2009.

Chenxi Whitehouse, Tianlu Wang, Ping Yu, Xian Li, Jason Weston, Ilia Kulikov, and Swarnadeep Saha. 2026. J1: Incentivizing thinking in LLM-as-a-judge via reinforcement learning. In *The Fourteenth International Conference on Learning Representations*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Bo Yang, Lanfei Feng, Yunkui Chen, Yu Zhang, Xiao Xu, and Shijian Li. 2026. FairJudge: An adaptive, debiased, and consistent LLM-as-a-judge. *arXiv preprint arXiv:2602.06625*.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. 2024. Large language models as optimizers. In *International Conference on Learning Representations*, volume 2024, pages 12028–12068.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 5824–5836. Curran Associates, Inc.

Yue Yu, Yuchen Zhuang, Jieyu Zhang, Yu Meng, Alexander J. Ratner, Ranjay Krishna, Jiaming Shen, and Chao Zhang. 2023. Large language model as attributed training data generator: A tale of diversity and bias. *arXiv preprint arXiv:2306.15895*.

Mert Yüksekgönül, Federico Bianchi, Joseph Boen, Sheng Liu, Pan Lu, Zhi Huang, Carlos Guestrin, and James Zou. 2025. Optimizing generative AI by backpropagating language model feedback. *Nature*, 639(8055):609–616.

Guang Zhao, Byung-Jun Yoon, Gilchan Park, Shantenu Jha, Shinjae Yoo, and Xiaoning Qian. 2025. Pareto prompt optimization. In *The Thirteenth 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 Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. Large language models are human-level prompt engineers. In *The Eleventh International Conference on Learning Representations*.

A Per-Task Gradient Specificity

Table 4 shows gradient specificity broken down by task for the combined-gradient modes (SCC, CCC). Averaged across SCC and CCC, consistency is the most diluted dimension (specificity 2.5), while coherence retains moderate focus (5.0). This suggests that the gradient LLM's attention is not uniformly distributed across tasks when processing them simultaneously.

Mode	Fl.	Rel.	Coh.	Con.	Avg
PER-TASK GRADIENT MODES					
Single	8.9	8.9	9.1	9.0	9.0
SSS	9.0	9.0	9.1	9.0	9.0
SSC	9.0	9.1	9.1	9.0	9.0
CROSS-TASK GRADIENT MODES					
SCC	3.0	4.3	4.8	2.6	3.7
CCC	3.2	4.3	5.1	2.4	3.8

Table 4: Gradient specificity by task (mean over both val settings, all runs and steps). Per-task modes maintain uniformly high specificity. In all-task modes, consistency is most diluted and coherence retains moderate focus.

B Cherry-Pick Experiment: All Variants

Table 5 shows all six cherry-pick variants (three selection metrics and two validation settings). All variants degrade below the initial generic baseline, confirming that inference-time instruction interference is robust across metric choices.

C Gradient Specificity Under Gradient-Model Swap

We swap the gradient LLM backbone to `DeepSeek-V4-Pro` (DeepSeek-AI, 2026) while keeping the task, loss, and optimizer LLMs in the Qwen3 family and holding the rest of the pipeline fixed. As shown in Figure 4, SSC remains high (8.82 ± 0.10) while SCC drops sharply (4.22 ± 0.26,

Validation	Selection metric	Avg ρ
Initial prompt (generic)		0.284
MAE	Spearman	0.220
MAE	Off-by-one	0.232
MAE	MAE	0.231
None	Spearman	0.120
None	Off-by-one	0.200
None	MAE	0.172

Table 5: All cherry-pick variants. Every oracle combination degrades below the initial generic baseline. The worst case (val=none, Spearman selection) additionally shows zero fluency correlation.

a 52% reduction), indicating that gradient dilution persists under a cross-family gradient model.

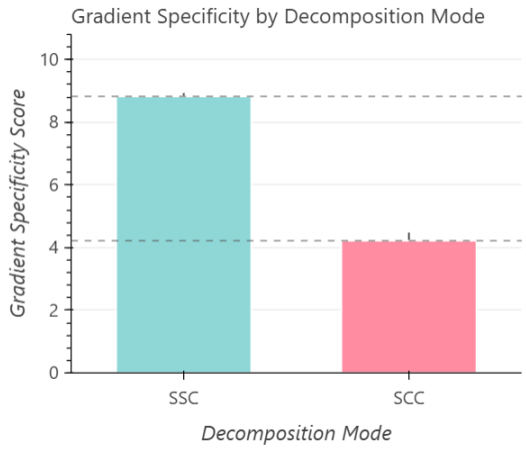

Figure 4: Gradient specificity for SSC vs. SCC after swapping the gradient LLM to DeepSeek-V4-Pro while keeping the other stages on Qwen3. SSC remains high (8.82 ± 0.10) while SCC drops to 4.22 ± 0.26, a 52% reduction.

Mode	Initial	Best (step)	Δ	HVI
SINGLE-TASK	0.287	0.403 (9)	**+0.117**	—
SSS	0.384	0.421 (9)	+0.037	4.264
SSC	**0.390**	**0.429** (7)	+0.039	**4.436**
SCC	0.379	0.418 (11)	+0.039	4.331
CCC	0.383	0.409 (7)	+0.026	4.282

Table 6: Main results under the MAE validation gate for the DeepSeek backbone. We measure the task-averaged Spearman correlation (mean over $N = 3$ runs). Notation same as Table 1.

D DeepSeek v4 Results

We repeat our analysis on DeepSeek v4 models (`DeepSeek-v4-Flash` as the task LLM; `DeepSeek-V4-Pro` for loss, gradient, and optimizer LLMs). These provide a stronger starting point for the task-following and optimization steps. We use the MAE validation gate and run $N = 3$ seeds.

D.1 Overall Performance

Table 6 summarizes performance at step 12. We observe a stronger backbone for the task and optimization LLMs result in a superior increase from the initial prompt. Multi-objective modes start from a stronger initialization ($\rho \approx 0.38$) and reach best avg ρ between 0.409 and 0.429. SSC attains the best avg ρ and the highest HVI. However, we observe that the trend sustains as before, with the *improvement* through prompt optimization being highest for single-objective optimizations.

Figure 5 plots the per-task optimization trajectories. Unlike the Qwen3 setting, where multi-objective modes stagnated at or below the initial prompt (Figure 2), the DeepSeek backbone yields consistent improvement from initialization.

The pattern of improvement, however, mirrors the main results: Single-Task achieves the largest gain across steps, while the multi-objective modes show flatter trajectories. The stronger optimizer LLM lifts all modes substantially, but it does not close the gap between single-objective and multi-objective optimization.

Per-task traces remain bounded within narrower bands than their Qwen3 counterparts, consistent with the observation that a stronger task model produces more stable evaluation dynamics.

D.2 Gradient Specificity

With DeepSeek, the gradient dilution cliff persists: per-task gradient modes (Single/SSS/SSC) remain highly task-focused (mean 8.78 specificity), while joint-gradient modes (SCC/CCC) drop to mean 4.25 (a 52% reduction), indicating that combining criteria in a single gradient call yields substantially more generic feedback. As shown in Figure 6, gradient specificity remains high for per-task gradient modes, while joint-gradient modes exhibit a sharp drop under the DeepSeek configuration.

E Multi-Objective Judge Prompt Template

Our multi-criteria prompt has two parts: a **frozen skeleton** (evaluation directive, output format) and **mutable per-task instructions** (the "Instructions" section).

Only the per-task instructions are updated during optimization; the skeleton remains fixed through-

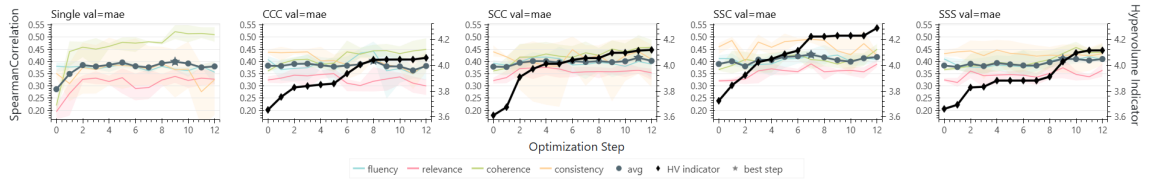

Figure 5: Per-task Spearman ρ for each optimization steps on SUMMEVAL with DeepSeek v4. Notation same as Figure 2.

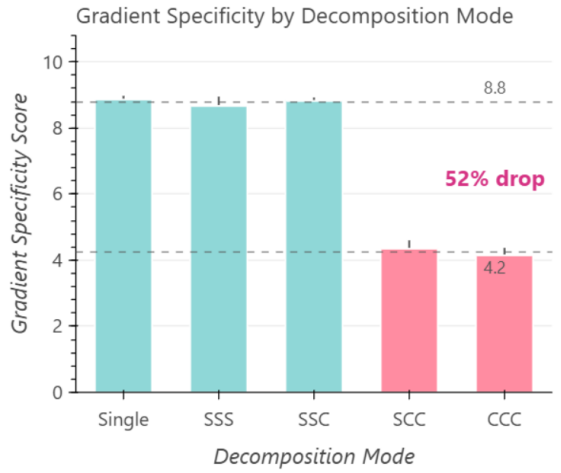

Figure 6: Gradient specificity by decomposition mode under DeepSeek.

out all 12 steps. Our skeleton is specific to SUMMEVAL but may be easily adapted to other problems.

Initial prompt template

```
You are a careful, calibrated evaluator. Your goal is to
produce an accurate evaluation by following the
Instructions below.

## Task
Evaluate the Summary given the Source Text using the
Instructions below.
1. Consider every strength and flaw you find when making
your evaluation.
2. Based on the number and severity of the strengths and
flaws, assign a value.
Use the Instructions below to perform your evaluation.
Output a JSON with the requested scores. Do NOT include
reasoning or explanations.

## Output format (follow this EXACTLY):
{
  "fluency": 1|2|3|4|5,
  "relevance": 1|2|3|4|5,
  "coherence": 1|2|3|4|5,
  "consistency": 1|2|3|4|5
}

## Instructions:
- fluency: Rate from 1 to 5.
- relevance: Rate from 1 to 5.
- coherence: Rate from 1 to 5.
- consistency: Rate from 1 to 5.


## Sample:
Summary: <summary text>
Source Text: <source text>
```

After optimization, the per-task instructions expand substantially. Below is an example from a SSS run (MAE validation, step 1), showing how the optimizer rewrites the generic one-line instructions into detailed, criterion-specific rubrics.

Example optimized instructions (step 1)

```
## Instructions:
- fluency: Fluency: Rate 1-5 based on
  grammatical correctness, sentence flow,
  and ease of parsing. A score of 5 reflects
  natural, effortless readability even with
  minor syntactic informality. Do not penalize
  for stylistic choices that do not impede
  comprehension.
- relevance: Relevance: Rate 1-5 based on how
  completely the summary addresses the core
  informational needs implied by the source
  text. A score of 5 means all key facts,
  outcomes, and implications are included with
  no deviation. Relevance penalizes both
  omission (missing key facts) and commission
  (adding unsupported claims).
- coherence: Coherence: Rate 1-5 based on
  logical flow, temporal sequence, and
  referential stability. A score of 5 means
  events unfold in a cause-effect or
  chronological order, with clear antecedents
  for all pronouns and noun phrases.
- consistency: Consistency: Rate 1-5 based on
  whether the summary contradicts any claim
  in the source text. A score of 5 requires
  no factual or inferential contradictions.
  A single major inconsistency reduces the
  score to 1.
```

F Diagnostic Evaluation Prompts

Below we include the prompts used for task-level diagnostics. Both diagnostics are evaluated post-hoc by Claude Sonnet 4.6.

F.1 Gradient Specificity Evaluator

The following prompt rates each textual gradient on a 1 to 10 scale for task-specificity:

Gradient specificity evaluator prompt

```
You are measuring how focused a piece of
textual feedback (called a "gradient") is on
a specific evaluation task, versus being
diluted with generic advice or advice that
belongs to other tasks.

The target task is "{task}". The possible
tasks are: fluency, relevance, coherence,
consistency.

Rate from 1 to 10 how well this gradient
focuses on the "{task}" task.
1 = completely generic or mostly addresses
    other tasks.
10 = laser-focused on "{task}" with concrete,
     task-specific fixes.

## The Gradient
{gradient_text}

Respond with ONLY a single integer from 1
to 10. No explanation.
```

F.2 Feedback Adherence Evaluator

The following prompt measures how well the optimizer incorporated each gradient into its instruction edit:

Feedback adherence evaluator prompt

```
You are evaluating whether revisions to
task-specific instructions correctly
addressed the gradient (suggested changes).

The instructions are for an LLM judge that
evaluates the "{task}" task. The Gradient
may contain suggestions about multiple
tasks; consider only suggestions pertaining
to "{task}".

Rate from 1 to 10 how well the New
Instructions address the Gradient for
"{task}".
1 = completely ignores/contradicts.
10 = precisely addresses every point while
     preserving what worked.

## Old Instructions
{old_instruction}

## New Instructions
{new_instruction}

## Gradient (Suggested Changes)
{gradient_text}

Respond with ONLY a single integer from 1
to 10. No explanation.
```