From Enumeration to Principle: Robust Prompt Optimization via Semantic Reflection and Multi-Objective Evolution

Anonymous ACL submission

Abstract

Large Language Models (LLMs) are increasingly deployed in adversarial environments, yet current automatic prompt optimization methods often suffer from "shortcut learning"—overfitting to specific samples rather than learning generalizable principles. We argue that, compared to scalar reward optimization or few-shot enumeration alone, the interpretability of language offers a useful medium for inducing robust decision logic. To study this, we introduce PROSE, a robust prompt optimizer that uses semantic reflection and multi-objective evolution to reduce over-reliance on enumerated examples. Facing adversarial tasks, PROSE employs a case-reliance proxy as an additional optimization objective to penalize excessive dependence on concrete cases. By exploring the Pareto frontier between performance and this proxy, it adds a regularization signal against sample fitting and favors more reusable rule descriptions. It employs ensemble-based de-biasing strategies and a semantic greedy selection mechanism to reduce inductive bias and premature "semantic collapse," while leveraging LLM-based reflection to summarize concise, high-level rules from failure patterns. Across the evaluated benchmarks, PROSE achieves higher average performance than the strongest baseline by up to 14.6 percentage points in accuracy on ShortText and up to 11.84 percentage points in F1 score, while showing stronger LOGO-OOD stability and reducing the F1 degradation from 29.7 points for GEPA to 13.0 points.

1 Introduction

The deployment of Large Language Models (LLMs) in high-stakes, adversarial environments demands not just performance, but robustness. Although Automatic Prompt Optimization (APO) has evolved from manual engineering (Wei et al., 2022) to automated search (Pryzant et al., 2023; Opsahl-Ong et al., 2024; Agrawal et al., 2025), a gap remains between training-set improvement and robust generalization. Current reflective methods, while efficient, are constrained by context windows, forcing them to rely on mini-batch optimization. This can encourage an "enumerative defense" style—stacking specific bad cases or shallow prompt patches to maximize training scores. Consequently, models may overfit to specific failure patterns. Increasing data diversity alone does not necessarily yield generalizable principles, and can still cause instability under distribution shifts or novel attacks. Robustness requires guiding the model beyond case accumulation to inducing abstract logic. For instance, rather than memorizing blacklists, a prompt should capture the underlying intent signals of harmful language, shifting from enumerative to "principled defense."

However, supporting this direction faces two formidable challenges: (1) Objective Bias: Existing frameworks only reward task metrics (e.g., accuracy), incentivizing verbose, case-stacking prompts rather than reusable rule construction. They can mechanically cover observed samples instead of distilling broader rules. (2) Search Space Constriction: Relying on single-source reflection limits semantic exploration, often causing the population to suffer from premature convergence (or "semantic collapse"), even with multiple performance constraints.

To overcome these challenges, we propose the **PROSE** (Pareto-Robust Optimization via Semantic Evolution) framework, which uses language interpretability to analyze and extract generalized logic from limited sample feedback, allowing the model to form abstract rules rather than relying on specific examples. Specifically, we employ a case-reliance proxy as an additional optimization objective to penalize over-reliance on concrete examples. By exploring the Pareto frontier between performance and this proxy, PROSE is designed to regularize prompt search away from simple sample fitting and

toward more general rule descriptions. Our main contributions are:

1. **Ensemble-based De-biasing Framework**: We reduce the inductive bias of single-source reflection through an ensemble-based de-biasing strategy. By using a diverse pool of meta-prompts and random sampling of model pools, we reduce the risk that the evolutionary process becomes tied to a single model preference or instruction style.

2. **Case-Reliance Proxy**: To curb "shortcut learning," we establish a case-reliance proxy as an independent optimization objective. This mechanism penalizes verbose example enumeration and encourages the optimizer to replace case lists with concise high-level rules when doing so preserves validation performance.

3. **Semantic Diversity Selection**: Addressing this semantic collapse, we introduce a semantic greedy selection mechanism. By imposing a diversity penalty in the semantic space, the population maintains diverse logical patterns even when the differences in the Pareto objective space are limited.

4. **Synergistic Evolutionary Operators**: We design a suite of co-evolutionary operators—including Reflective Mutation for logical correction, Rewrite Operators for rule distillation, and Semantic Crossover for strategic fusion. These operators combine complementary prompt strategies at the logical level rather than only editing surface wording.

Reproducibility. Upon acceptance, we will release the PROSE source code, prompt templates, evaluation scripts, and releasable desensitized data artifacts.

2 Related Work

LLM security is a critical concern (Das et al., 2025), with adversarial attacks like jailbreaking and prompt injection inducing model deviations through increasingly concealed strategies. In content moderation, attackers often employ semantic obfuscation (e.g., homophones, character splitting) to bypass static defenses. This necessitates automated, adaptive defense mechanisms.

For Chinese toxic language detection, ToxiCN (Lu et al., 2023) provides a fine-grained benchmark with both direct and indirect toxic samples, motivating our use of ToxiCN as an external evaluation dataset alongside the self-collected Short-Text benchmark.

2.1 Feedback-Driven Prompt Optimization

Recent work shows that language feedback can guide iterative improvement in LLM systems. Reflexion (Shinn et al., 2023) demonstrated how verbalized feedback reinforces learning from errors, while ReAct (Yao et al., 2023) improved problem solving through interleaved reasoning and action. In prompt optimization, this line of work motivates mutation operators that use error feedback as a search signal rather than relying only on scalar rewards. PROSE follows this feedback-driven view, but focuses on preserving diverse prompt strategies and reducing over-reliance on concrete examples during evolution.

2.2 Automated Prompt Optimization

The prompt engineering aims to fully utilize the potential of large language models by providing clear and logical instructions. Chain-of-Thought (CoT) (Wei et al., 2022) improved complex reasoning but relies on expert-crafted examples. To overcome scalability bottlenecks, researchers explored automated optimization. Soft Prompts (Lester et al., 2021) optimize in continuous vector space but require parameter access. APE (Zhou et al., 2022) formulates prompt generation as program synthesis; OPRO (Yang et al., 2024) uses in-context learning for iterative optimization. For compound AI systems, DSPy (Khattab et al., 2024) offers a modular approach. Notably, MIPROv2 (Opsahl-Ong et al., 2024) employs Bayesian Optimization to jointly optimize instructions and few-shot examples. Prior work has also studied the trade-off between improving instructions and adding demonstrations (Wan et al., 2024), which is closely related to our case-reliance objective. In adversarial moderation, strong reliance on examples can risk "shortcut learning," where models prioritize pattern matching over generalized reasoning. This motivates instruction-centric optimization that explicitly controls example dependence. ProTeGi (Pryzant et al., 2023) operationalizes textual gradients with minibatch feedback and beam search, while TextGrad (Yuksekgonul et al., 2025) extends gradient-based optimization to compound

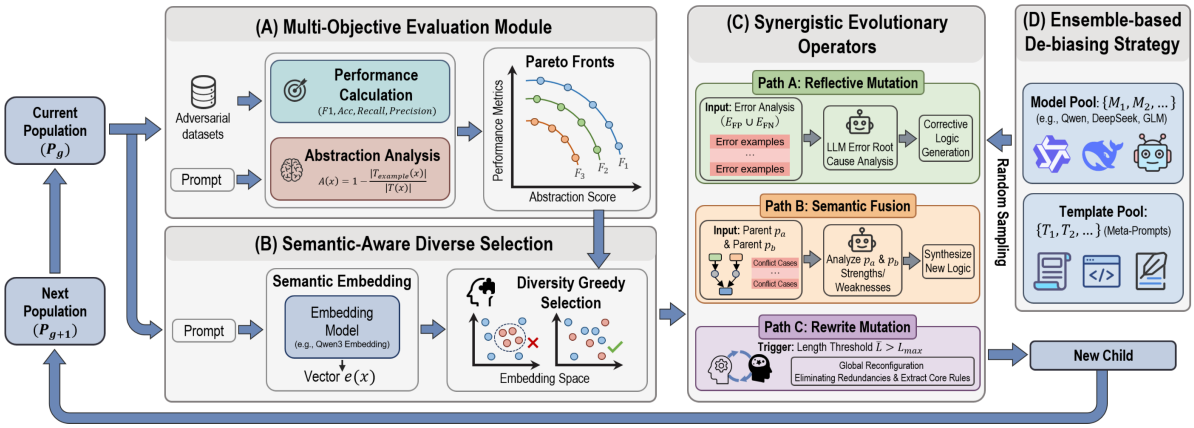

Figure 1: The PROSE framework contains four components. (A) Multi-objective Evaluation Module computes task performance and the case-reliance proxy for Pareto sorting. (B) Semantically-aware Diversified Selection filters elites with semantic-embedding diversity penalties. (C) Synergistic Evolutionary Operators generate offspring via Reflective Mutation, Semantic Fusion, and Rewrite Mutation. (D) Ensemble-based De-biasing Strategy samples model and template pools to diversify reflection.

systems. Compared to these gradient-based APO approaches, our case-reliance proxy adds a regularization signal against example memorization. Additionally, recent studies have demonstrated the fragility of prompts, showing that models often overfit to specific formatting or spurious features rather than learning robust logic (Sclar et al., 2024). This further highlights the necessity of our semantic diversity constraints and example-reliance control to reduce shortcut learning.

In evolutionary optimization, EvoPrompt (Guo et al., 2024) combined LLMs with genetic algorithms; PromptBreeder (Fernando et al., 2024) introduced "self-referential" evolution. GEPA (Agrawal et al., 2025) utilizes environmental feedback for reflective mutation. However, maintaining diversity remains a challenge. A critical limitation is that prompts with distinct logical structures often produce similar performance metrics. Consequently, diversity mechanisms based solely on objective space (as in GEPA) may not sufficiently preserve semantic heterogeneity, leading to premature convergence. Moreover, relying on a single reflection model can introduce "cognitive bias," trapping the population in a specific reasoning mode (an "echo chamber"). PROSE targets these issues through Semantic Greedy Selection and an Ensemble-based De-biasing strategy.

3 Method

By exploring the Pareto frontier between detection performance and the case-reliance proxy, PROSE establishes a multi-objective evolutionary framework that biases search toward concise and reusable rule descriptions. This approach is intended to reduce overfitting risk while using semantic diversity constraints to reduce semantic collapse. As shown in Figure 1, this framework consists of four core components: a Multi-objective Evaluation Module for performance evaluation and case-reliance estimation, a Semantically-aware Diversified Selection for avoiding this collapse, Ensemble-based De-biasing Strategy for reducing inductive bias, and Synergistic Evolutionary Operators for rule refinement.

3.1 Multi-objective Evaluation Module

The Multi-objective Evaluation Module formalizes prompt optimization as a multi-objective maximization problem. We group the objectives into task-related performance and our proposed case-reliance proxy. For any candidate prompt x, its objective vector is $\mathbf{V}(x) = [\mathbf{P}(x),\ A(x)]$ (to be maximized), where $A(x)$ is a high-is-better score that penalizes excessive reliance on concrete examples. In the Short-Text moderation experiments, we instantiate $\mathbf{P}(x)$ as validation Precision and Recall for Pareto sorting, while validation Accuracy is used as the primary selector among non-dominated final candidates. Full split, selection, and metric details are reported in Appendix A.1 and Appendix A.2. The two objective groups are:

1. Detection Performance $\mathbf{P}(x)$: This dimension encapsulates the traditional evaluation criteria measured exclusively on the validation set. It can comprise one or multiple metrics depending on the task requirements (e.g., accuracy, precision, recall, F1 score, etc.). For classification tasks, Accuracy is

the primary validation metric for final prompt selection. For Short-Text optimization, $\mathbf{P}(x)$ contains validation precision and recall as Pareto objectives, with validation accuracy used as the primary selector among non-dominated prompts. We report F1 alongside Accuracy to show the precision-recall trade-off; for the HotpotQA (Yang et al., 2018) dataset, we use Ans EM as the primary metric.

2. Case-Reliance Proxy $A(x)$: This objective is designed to regularize "Shortcut Learning." Adding concrete examples is often the most direct way to boost local evaluation metrics. However, excessive reliance on examples is undesirable, as it can increase overfitting risk—especially since real-world datasets naturally contain conflicting samples. We therefore define the case-reliance proxy as an independent evaluation dimension:

$$A(x) = 1 - R_{\text{example}}(x) = 1 - \frac{|T_{\text{example}}(x)|}{|T(x)|} \quad (1)$$

where $|T_{\text{example}}(x)|$ is the character length of extracted examples, and $|T(x)|$ is the total prompt length. Rather than a heuristic compromise, we argue this is the most elegant and cost-effective approach to macroscopically steer the LLM towards logic abstraction. Directly to evaluate the "abstraction level" or "reasoning depth" of a prompt is computationally expensive, highly subjective, and notably unstable during continuous evolution. By contrast, tracking the concrete example ratio provides a highly accurate and low-cost regularization signal. Importantly, $A(x)$ serves as an independent objective on par with task metrics (e.g., Precision and Recall) during Pareto sorting, rather than acting as a rigid filter. Consequently, a prompt candidate with a high example ratio but exceptional detection performance will not be blindly eliminated; it will be retained on the approximate Pareto frontier until another candidate either surpasses its performance with the same example ratio, or matches its performance with a lower example reliance. Appendix D details the extraction model, median sampling, stability checks, and human-agreement analysis.

3.2 Semantically-aware Diversified Selection

To maintain semantic diversity while retaining population fitness and preventing premature convergence, we propose a layered selection strategy combining Pareto dominance and semantic embedding constraints. Traditional multi-objective algorithms (e.g., NSGA-II) rely on crowding distance, which only restricts diversity in the objective metric space and fails to assess similarity at the linguistic level. In each generation, the population is stratified into approximate Pareto fronts $\{F_1, F_2, \ldots, F_k\}$ via Non-dominated Sorting. Each candidate $c \in F_k$ is assigned a base fitness $\text{Fitness}(c) = \frac{1}{k}$, prioritizing solutions on earlier fronts. Upon completion of the evolutionary process, since task performance is one of our multi-objective goals, we select the single prompt from the multiple non-dominated solutions in F_1 that yields the highest task-specific primary validation metric (Accuracy for Short-Text) as the final output for independent test set evaluation.

To reduce the risk that the population converges to a single mode, we introduce a dynamic distance penalty mechanism based on semantic embeddings. We use a pre-trained embedding model $\mathcal{M}_{\text{embed}}$ (Qwen3-Embedding-0.6B) to map each candidate prompt x to a high-dimensional semantic vector $\mathbf{e}(x) \in \mathbb{R}^d$, measuring semantic distance between them via cosine similarity:

$$\text{Sim}_{\text{sem}}(c, s) = \frac{\mathbf{e}(c) \cdot \mathbf{e}(s)}{\|\mathbf{e}(c)\| \cdot \|\mathbf{e}(s)\|} \quad (2)$$

For a candidate c and the currently selected elite set S, let $\delta_{\max} = \max_{s \in S} \text{Sim}_{\text{sem}}(c, s)$ denote its maximum similarity to the already selected most similar individual. The diversity penalty P_{div} is defined as:

$$P_{\text{div}}(c, S) = \begin{cases} \alpha \cdot (\delta_{\max} - \tau)^2 + \beta & \text{if } \delta_{\max} > \tau \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

where $\tau \in [0, 1]$ is the similarity threshold. α indicates the penalty weight; β represents the penalty bias. This explicitly penalizes prompts that are linguistically redundant.

We employ an **Incremental Greedy Selection** approach (detailed in Algorithm 1) to iteratively select the highest scoring candidates based on their penalized fitness ($Score(c) = \text{Fitness}(c) - P_{\text{div}}(c, S)$). This helps filter out semantically redundant elites even when the initial Pareto front F_1 contains many high-scoring but similar prompts.

3.3 Synergistic Evolutionary Operators

PROSE utilizes LLM reasoning for three core operators: Reflective Mutation, Rewrite Mutation, and Semantic Fusion Crossover. The parameters W_{reflect}, W_{rewrite}, and $W_{\text{crossover}}$ represent the probabilities of selecting each corresponding strategy during reproduction, which are designed to realize

deep fusion and optimization of defense prompts at the logical level.

Reflective Mutation uses a closed-loop feedback mechanism. It constructs a meta-prompt taking the current prompt, evaluation metrics, and error cases (E_{error}) as input, requiring the LLM to diagnose logical flaws and output a corrected prompt. Full templates are detailed in Appendix E.

Rewrite Mutation complements reflection by addressing prompt bloating. When the maximum parent-prompt length exceeds L_{max}, the rewrite operator is enabled for the next generation. A Rewrite Meta-Prompt (sampled from Appendix E.3) guides the LLM to extract core rules, remove redundancy, and compress the prompt structure while maintaining performance.

Semantic Fusion Crossover aims to eliminate blind spots by merging two complementary parents (p_a, p_b) with high semantic difference. It inputs both parent prompts and their conflict cases, prompting the LLM to design a new logical framework integrating their successful strategies.

Detailed reproduction logic and template sampling probabilities are provided in Appendix A.2.

Algorithm 1 PROSE Main Framework

Require: Initial Population $\mathcal{P}_{\text{init}}$, Generations G, Population Size N, Elite Size E, Similarity Threshold τ, Operator Weights W
Ensure: Approximate Pareto Optimal Prompts $\mathcal{P}_{\text{best}}$

```
Initialize P_0 ← P_init; Compute embeddings e(x), ∀x ∈ P_0
for g = 1 to G do
    Parallel Evaluation: Compute V(x) for P_{g-1}
    Sort: Non-dominated sorting to {F_1, ..., F_k}
    Elite Selection (Incremental Greedy):
        S_elite ← ∅; CandidatePool ← Sorted P_{g-1}
    while |S_elite| < E and CandidatePool ≠ ∅ do
        for c in CandidatePool do
            Update Score(c) = Fitness(c) − P_div(c, S_elite)
        end for
        c* ← arg max Score(c)
        S_elite ← S_elite ∪ {c*}; Remove c*
    end while
    ℓ_max ← max_{p∈P_{g-1}} Len(p)
    W̃ ← W
    if ℓ_max ≤ L_max then
        W̃_rewrite ← 0
    end if
    Procreation Loop:
    P_g ← S_elite
    while |P_g| < N do
        child ← GenerativeReproduction(S_elite, M_pool, T_pool, W̃)
        Compute e(child); P_g ← P_g ∪ {child}
    end while
end for
Final Evaluation: Compute V(x) for newly generated candidates in P_G
return ApproximateParetoFront(P_G)
```

3.4 Ensemble-based De-biasing Strategy

Heterogeneous Model Ensemble Sampling: Instead of using a fixed reflection model $\mathcal{M}_{\text{reflect}}$, we define a model pool $\mathbb{M}_{\text{pool}} = \{\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_K\}$ containing K heterogeneous LLMs with diverse architectural biases (e.g., reasoning-focused vs. instruction-following). During each mutation or crossover operation, we sample a model $\mathcal{M}_i \sim \text{Categorical}(\mathbf{w})$ from the pool according to a probability distribution $\mathbf{w}$. This reduces the chance that evolved prompts over-align with any single model's inductive preferences.

Multi-Template Random Sampling: Similarly, we design a pool of diverse meta-prompt templates $\mathbb{T}_{\text{pool}} = \{T_1, T_2, \ldots, T_M\}$ with varying instructional styles (e.g., some emphasizing rule abstraction, others focusing on edge-case correction). During each operation, a template T_j is sampled from this pool. This dual-sampling mechanism (model $\times$ template) creates a combinatorial space that reduces the risk of evolution being trapped in local optima associated with any single model-template pair.

4 Experiments

4.1 Experimental Settings

Datasets. We evaluated PROSE on three datasets. The Adversarial Short-Text Dataset consists of 4,928 manually desensitized and annotated samples collected from real-world production scenarios, with text lengths ranging from 0 to 30 characters and high-density adversarial patterns such as homophones, splitting, and pinyin abbreviations. We reserve a stratified 832-sample test set for final Short-Text reporting; grey ground-truth samples are excluded from metric calculations. Dataset taxonomy, split, and annotation details are provided in Appendix A.1.

For ToxiCN (Lu et al., 2023), we follow the toxic/non-toxic binary classification task and the official train/test split. For HotpotQA (Yang et al., 2018), we follow a multi-hop retrieval program setting based on the HoVerMultiHop-style program (Jiang et al., 2020). Detailed split sizes and task adaptations are provided in Appendix A.1.

Baselines. We compared against: Baseline (Zero/Few-shot), TextGrad (Yuksekgonul et al., 2025), PromptBreeder (Fernando et al., 2024), MIPROv2 (Opsahl-Ong et al., 2024), OPRO (Yang et al., 2024), ProTeGi (Pryzant et al., 2023), GEPA (Agrawal et al., 2025). All methods were initialized

Table 1: Mean performance across datasets over at least 5 independent runs. The best result is highlighted in **bold**; Δ vs. Best reports the descriptive difference between PROSE and the best baseline for each metric. Positive deltas are highlighted in red, while negative deltas indicate metric trade-offs. Accuracy is primary for classification datasets and Ans EM is primary for HotpotQA. Standard deviations and per-risk-group results are reported in Appendix Table 6.

Models	Short-Text				ToxiCN				HotpotQA
	Acc	F1	P	R	Acc	F1	P	R	Ans EM
Baseline	52.37	61.58	50.43	76.23	73.45	69.05	**88.92**	56.04	54.24
TextGrad	53.64	57.14	50.34	76.49	75.69	74.05	84.96	67.62	55.00
PromptBreeder	49.68	57.69	48.92	70.28	75.45	73.48	83.59	64.36	56.68
MIPROv2	52.21	60.18	51.37	67.70	74.02	73.66	79.12	68.76	59.96
OPRO	52.08	60.16	50.60	76.23	73.87	71.02	85.78	61.60	57.08
ProTeGi	52.42	60.70	50.84	**77.78**	75.24	74.78	82.97	69.47	56.46
GEPA	61.71	60.73	63.20	51.68	72.38	69.64	83.04	59.97	65.42
PROSE (Ours)	**76.29**	**73.42**	**81.21**	67.53	**79.93**	**78.89**	87.40	**70.25**	**67.19**
Δ vs. Best	**+14.58**	**+11.84**	**+18.01**	-10.25	**+4.24**	**+4.11**	-1.52	**+0.78**	**+1.77**

Table 2: Mean robustness results on the test set. "LOGO-OOD" denotes Leave-One-Group-Out evaluation over first-level risk groups: optimization uses the training/validation split with the target group removed, and testing is performed only on the held-out test samples. All values are percentages over at least 5 independent runs; F1 Drop is computed against the corresponding full-test mean. Grey ground-truth samples are excluded from metric computation.

Run	Acc	F1	P	R	FNR	FPR	F1 Drop
GEPA (full test set)	61.71	60.73	63.20	51.68	48.32	28.74	–
GEPA (LOGO-OOD, micro)	45.02	31.07	40.02	25.41	74.59	36.29	29.66
PROSE (full test set)	76.29	73.42	81.21	67.53	32.47	15.35	–
PROSE (LOGO-OOD, micro)	63.18	60.43	63.53	57.62	42.38	31.53	12.99

from the same minimal task prompt, and all task evaluations used Qwen3-Next-80B as the execution model. For the main comparison, all optimization methods were constrained to the same candidate-evaluation budget and the same maximum token budget. Importantly, reflection- or generation-based optimizers called the same gateway alias, `Reflection-Model`, which samples backend models from our shared heterogeneous reflection pool. While the heterogeneous model pool is a core design contribution of PROSE, we deliberately opened this pool to all baselines in the main experiment to eliminate the confounding variable of "PROSE using stronger models." Thus, the main results demonstrate that under the exact same reflection backend and budget, PROSE's multi-objective selection, case-reliance proxy, and synergistic operators are fundamentally more effective. The independent contribution of the model pool itself is isolated and proven in our ablation studies. Appendix A.2 provides the gateway and model-role details, and Appendix C.3 reports a stricter low-token stress test.

4.2 Results and Analysis

Model Selection. We separate reflection, execution, embedding, and case-extraction roles. The execution model is Qwen3-Next-80B, and reflection uses a heterogeneous gateway pool; full model-role details are in Appendix A.2.

Implementation Details. Experiments were conducted with a population size of $N = 8$ and a maximum of $G = 24$ generations. The similarity threshold τ was set to 0.5, optimized for the Qwen3-Embedding-0.6B model. Main and OOD results are reported as means over at least 5 independent runs with different random seeds; Short-Text mean–standard deviation statistics are reported in Appendix Table 6. Additional hyperparameters and reproducibility details are provided in Appendix A.2.

Overall Performance Verification. Table 1

presents the performance comparison across three distinct datasets. Accuracy is the primary metric for the classification datasets, while F1 is reported to provide a holistic view of the precision-recall trade-off. PROSE achieves the highest average primary-metric results in the evaluated settings, particularly in the Adversarial Short-Text domain. Unlike baseline methods that struggle with the high density of homophonic and structural attacks, PROSE improves Short-Text accuracy to 76.29%, 14.58 points higher than the strongest baseline, GEPA (61.71%). It also achieves an F1 score of 73.42%, 11.84 points higher than the strongest baseline, Baseline (61.58%), and improves precision to 81.21%. Detailed mean–standard deviation statistics are reported in Appendix Table 6. On ToxiCN, PROSE reaches 79.93% accuracy and 78.89% F1.

We observe a performance discrepancy between the highly adversarial Short-Text task (larger gains), ToxiCN (moderate gains), and HotpotQA (smaller gains). This can be explained by two key factors: performance ceiling and task bottlenecks. For ToxiCN, as reported by Lu et al. (2023), a fully supervised fine-tuned BERT classifier achieves an F1 of around 80% after training on 9.6K samples. In contrast, our reflection prompt directly observes up to 384 bad cases (24 generations $\times$ 16 reflection cases). PROSE reaches 78.89% F1 with such extreme data efficiency and no parameter updates, PROSE essentially pushes the zero/few-shot optimization to the supervised performance ceiling, making further large absolute gains nearly impossible. For HotpotQA, the task is primarily model-bounded rather than prompt-bounded. The bottleneck lies in the model's intrinsic multi-hop reasoning, long-context comprehension, and retrieval capabilities, which cannot be bypassed merely by prompt optimization. Thus, HotpotQA is used as a generalization sanity check rather than a primary optimization target. Considering that fully supervised State-of-the-Art (SOTA) models on HotpotQA achieve around 72% Ans EM (as reported on the official leaderboard[1]), PROSE's 67.19% Ans EM (+1.77 over the strongest baseline) without parameter updates is highly competitive and indicates no degradation in general capabilities.

Robustness to Unseen Attacks (OOD Analysis). To evaluate robustness under unseen adversarial strategies, we conducted a Leave-One-Group-Out (LOGO) experiment by excluding one first-level risk group from the Short-Text training/validation pool during optimization and evaluating on the corresponding held-out test samples. Table 2 summarizes the robustness results. We exclusively compare against GEPA for this OOD analysis because GEPA is the only baseline that achieves meaningful improvements over the base prompt (outperforming the second-best baseline by 8.07% in Accuracy on the Short-Text task); comparing OOD degradation on baselines that fail to optimize the prompt initially provides little analytical value. Grey ground-truth samples are excluded from metric computation. GEPA drops from 60.73 F1 on the full test set to 31.07 F1 under LOGO-OOD, a 29.66-point degradation. PROSE maintains 60.43 F1 under LOGO-OOD after reaching 73.42 F1 on the full test set, reducing the degradation to 12.99 points and retaining a substantially larger fraction of its full-test performance; detailed per-risk-type breakdowns are reported in Appendix B.2.

[1] https://hotpotqa.github.io/

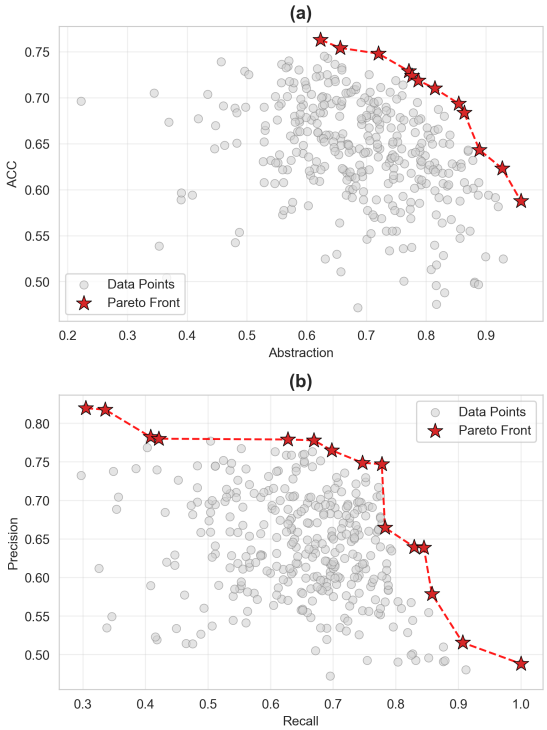

Figure 2: Pareto Analysis of Prompt Optimization. Grey points denote evaluated prompt candidates, and red stars denote the approximate Pareto front. Top: the ACC-Abstraction frontier, where higher abstraction corresponds to lower estimated case reliance. Bottom: the Precision-Recall frontier.

Navigating the Performance–Case-Reliance Trade-off. To analyze the trade-off between accuracy and example reliance, we examine the Pareto Analysis in Figure 2. The ACC-Abstraction fron-

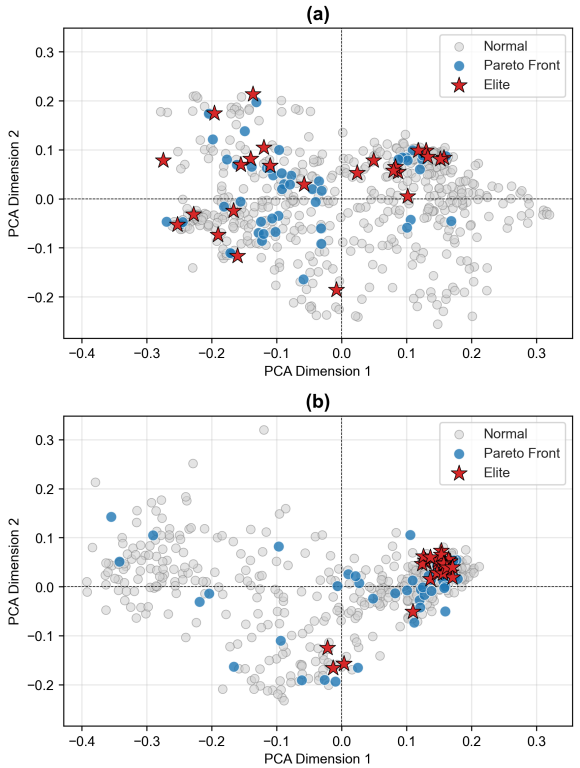

Figure 3: Evolutionary Dynamics in Semantic Space. The "Normal" points represent all individual prompt solutions evaluated during the population iteration. Top: With PROSE's Diversity Penalty, the elite population (red stars) maintains a dispersed distribution, covering diverse semantic regions. Bottom: Without the penalty, elites cluster prematurely, indicating semantic collapse.

tier (Top) is consistent with PROSE retaining high-performing prompts while keeping concrete-case reliance low. Furthermore, the Precision-Recall distribution (Bottom) illustrates that our multi-objective approach preserves candidates with different operating points. Further control experiments probing the case-reliance proxy against simple length reduction are detailed in Appendix C.2.

Evolutionary Dynamics in Semantic Space. We examine the impact of our Semantic Greedy Selection by visualizing the population's semantic embeddings (Figure 3). In the baseline setting (Bottom), the absence of semantic diversity constraints is associated with rapid population clustering into a single mode, a pattern we refer to as "Semantic Collapse." In contrast, PROSE (Top) employs an embedding-based penalty, and the resulting elite prompts remain more dispersed. This diversity is consistent with the framework exploring and retaining distinct defensive strategies—such as rule-based analysis versus role-playing simulation—which may contribute to robustness against varied attack patterns.

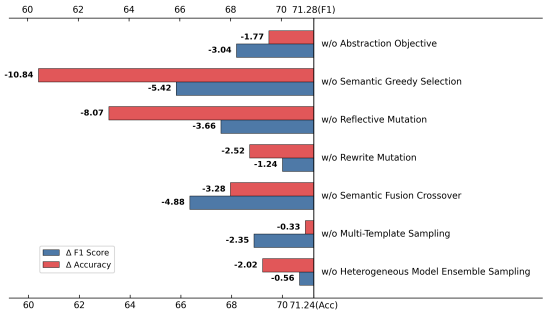

Figure 4: Ablation Study: Mean performance drop (F1 and Accuracy) in the matched ablation suite when removing key components. Bars start from the full PROSE average performance and extend left by the corresponding drop; the vertical reference line marks the full-model baseline. Semantic Greedy Selection produces the largest drop in both F1 and accuracy; Reflective Mutation yields the second-largest accuracy drop, while Semantic Fusion Crossover yields the second-largest F1 drop.

Ablation Studies. As shown in Figure 4, different components exhibit distinct impacts on Accuracy and F1, indicating their complementary roles in PROSE. Removing Semantic Greedy Selection causes the most severe degradation in both Accuracy (-13.84 points) and F1 (-11.42 points). Model Ensemble Sampling (where we fix the reflection model exclusively to DeepSeek-V3.1 instead of the heterogeneous pool) and Semantic Fusion Crossover show disproportionately large F1 drops relative to their Accuracy drops. This confirms that the model pool itself contributes significantly to expanding semantic coverage and preventing inductive bias. The case-reliance proxy objective also produces large and balanced drops in both metrics (-6.77 Accuracy and -7.04 F1), supporting its role as a regularizer against shortcut learning. Detailed ablation values are provided in Appendix C.1.

5 Conclusion

This paper introduces PROSE, a framework leveraging LLM semantic reflection and multi-objective evolution for robust prompt optimization. By integrating a case-reliance objective, logic-driven reflection, and diversity-preserving selection, PROSE reduces semantic collapse, overfitting, and shortcut learning in the evaluated settings. It enables the automated generation of high-performance prompts for adversarial environments, reducing manual prompt-engineering effort.

Limitations

While PROSE demonstrates substantial improvements, several limitations remain. First, regarding generalization, we observe that PROSE's performance gains are most pronounced on the proprietary Adversarial Short-Text dataset (+14.58 percentage points in Accuracy), moderate on ToxiCN, and more marginal on HotpotQA (+1.77 points in Ans EM). As discussed in Section 4.2, this stems from the *prompt-bounded* versus *model-bounded* nature of different tasks. More formally, PROSE is best suited for tasks that require external knowledge, domain-specific judgment methods absent in common sense, and complex classification standards, where the model must autonomously and extensively induce rules and knowledge during the iterative process. PROSE is highly effective in such prompt-bounded adversarial scenarios with dynamic semantic drift. However, in tasks like ToxiCN, where we already approach the fully supervised performance ceiling ($\sim$80% accuracy) via prompting alone, the absolute room for improvement diminishes significantly. Similarly, for model-bounded, retrieval-dominated tasks like HotpotQA, future gains require fundamentally improving the model's retrieval, evidence selection, or reasoning capabilities rather than relying solely on prompt evolution. Therefore, PROSE should be viewed as an optimizer specializing in complex rule abstraction rather than a universal remedy for model limitations.

Second, the computational cost of the evolutionary process is non-trivial, primarily due to the additional LLM-based evaluations for case-reliance proxy extraction and multi-objective Pareto sorting. Although we utilize a smaller model (Qwen3-4B) for case extraction to mitigate overhead, the repetitive token consumption during evolution remains high. Future work will explore training a dedicated, lightweight NLP model specifically for rule-case separation to further reduce this dependency. Furthermore, the framework's effectiveness relies heavily on the reasoning capabilities of the underlying *Reflection Model*. While our ensemble strategy mitigates individual model biases, the upper bound of logical induction is still constrained by the foundational intelligence of the LLMs used. Finally, our current evaluation focuses on text-only tasks, so the framework's applicability to broader multi-modal moderation (e.g., image or audio) has not been fully explored.

Ethical considerations

Data Privacy and Compliance

The datasets used in this research (Adversarial Short-Text) originated from real-world production environments and were used for platform safety monitoring and quality assurance. Access to and use of this data followed the platform's internal data governance procedures, Terms of Service, and Privacy Policy. Prior to use in this study, all data underwent a desensitization process. Potential Personally Identifiable Information (PII), including user IDs, contact details, and specific location data, was removed or masked. We used only the minimum information needed for moderation research and did not attempt to identify individual users.

Responsible Use and Dual-Use Concerns

We acknowledge that research into adversarial attack patterns and prompt optimization has dual-use potential. While our goal is to enhance defense mechanisms (robust prompt optimization), detailed attack descriptions could aid malicious actors. To mitigate this risk, we define attack patterns abstractly (e.g., "Homophonic Substitution") rather than providing a "dictionary" of effective bypasses. The specific prompts and examples provided in this paper are selected or synthesized to demonstrate the *logic* of optimization without revealing active, unpatched vulnerabilities of any specific platform. This work is intended to support safer and more reliable use of Large Language Models.

Checklist response: AI assistants were used for language polishing and form-filling support; all technical claims, experiments, and final text were reviewed by the authors.

References

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, and 1 others. 2025. Gepa: Reflective prompt evolution can outperform reinforcement learning. *arXiv preprint arXiv:2507.19457*.

Badhan Chandra Das, M Hadi Amini, and Yanzhao Wu. 2025. Security and privacy challenges of large language models: A survey. *ACM Computing Surveys*, 57(6):1–39.

Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. 2024. Promptbreeder: self-referential self-improvement via prompt evolution. In *Proceedings of the 41st International Conference on Machine Learning*, pages 13481–13544.

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 *International Conference on Learning Representations*, volume 2024, pages 34133–34156.

Yichen Jiang, Shikha Bordia, Zheng Zhong, Charles Dognin, Maneesh Singh, and Mohit Bansal. 2020. Hover: A dataset for many-hop fact extraction and claim verification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3441–3460.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Saiful Haq, Ashutosh Sharma, Thomas Joshi, Hanna Moazam, Heather Miller, and 1 others. 2024. Dspy: compiling declarative language model calls into state-of-the-art pipelines. In *International Conference on Learning Representations*, volume 2024, pages 54928–54958.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pages 3045–3059.

Junyu Lu, Bo Xu, Xiaokun Zhang, Changrong Min, Liang Yang, and Hongfei Lin. 2023. Facilitating fine-grained detection of chinese toxic language: Hierarchical taxonomy, resources, and benchmarks. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16235–16250.

Krista Opsahl-Ong, Michael J Ryan, Josh Purtell, David Broman, Christopher Potts, Matei Zaharia, and Omar Khattab. 2024. Optimizing instructions and demonstrations for multi-stage language model programs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 9340–9366.

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.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2024. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. In *International Conference on Learning Representations*, volume 2024, pages 25055–25083.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652.

Xingchen Wan, Ruoxi Sun, Hootan Nakhost, and Sercan Ö Arık. 2024. Teach better or show smarter? on instructions and exemplars in automatic prompt optimization. *Advances in Neural Information Processing Systems*, 37:58174–58244.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

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.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2369–2380.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Mert Yuksekgonul, 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.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022. Large language models are human-level prompt engineers. In *The eleventh international conference on learning representations*.

A Experimental Setup and Reproducibility

A.1 Datasets, Splits, and Metrics

This section centralizes dataset construction, split rules, and evaluation metrics. For Short-Text, we use Accuracy as the primary validation selector and report F1, Precision, and Recall as complementary metrics. Grey ground-truth samples are excluded from binary metric computation. For ToxiCN, we follow the official train/test split and reserve 10% of the official training split as validation. For HotpotQA, we use 150 examples for training, 300 for validation, and 300 for testing in the multi-hop retrieval program setting described in the main text.

Public Benchmark Adaptations. For ToxiCN (Lu et al., 2023), we follow the toxic/non-toxic binary classification task from the original paper and use the train/test split released in the official open-source project. We randomly sample 10% of the official training split as validation and reserve the official test split for final reporting. For HotpotQA (Yang et al., 2018), a large-scale question-answering dataset with 113K Wikipedia-based question-answer pairs requiring reasoning over multiple supporting documents, we follow a multi-hop retrieval program setting. Specifically, we modify the final hop of the HoVerMultiHop-style program (Jiang et al., 2020) to answer the question instead of generating another query, while keeping the rest of the system unchanged. The textual feedback module identifies the remaining relevant documents to be retrieved at each stage and provides this information as feedback to the corresponding modules.

A.1.1 Adversarial Short-Text Dataset

This dataset is derived from user nicknames, post titles, and other short-text fields collected from multiple large-scale social and interactive platforms. These modalities present unique challenges in content moderation: they are brief (typically under 30 characters), highly creative, and serve as primary vectors for users to bypass safety filters using adversarial text perturbations.

The dataset features high-density adversarial patterns, including homophones, character splitting, and pinyin abbreviations, and follows platform-specific moderation guidelines rather than a single universal toxicity standard.

Data Collection and Annotation. The dataset was sampled from real-world short-text moderation logs and then manually desensitized. After extraction and deduplication, the final dataset consists of 4,928 samples. The complete dataset was constructed to follow the same label distribution and first-level risk taxonomy proportions as the test set described below, so the held-out test split is representative of the overall corpus rather than a separate stress-only subset.

To rigorously evaluate the framework's adaptability, we sampled a dedicated 832-sample test set from the complete dataset with stratification by label and first-level risk taxonomy. The characteristics of this test set are detailed below:

- **Basic Statistics & Label Distribution**: The test set comprises 832 total samples. The ground truth distribution consists of 48.8% non-violative (white) samples, 46.5% violative (black) samples, and 4.7% ambiguous (grey) samples. Note that the ambiguous (grey) samples are not included in the final evaluation metrics.

- **Risk Taxonomy Distribution**: The test set covers a wide spectrum of adversarial patterns. At the first level, the assigned taxonomy is concentrated in Ecological Governance (38.5%), Pornography & Vulgarity (28.4%), Political Sensitivity (19.2%), Illegal/Regulatory Content (6.0%), Impersonation/Misleading Content (5.8%), and Transaction/Traffic Diversion (2.2%). At the second level, the full taxonomy distribution includes: Pornography & Vulgarity (28.4%), Insults & Discrimination (26.3%), Political Figures & Regimes (12.5%), Harmful Content Evasion (12.1%), Impersonation/Misleading Content (5.8%), Contraband & Illegal Activities (5.4%), Religious Extremism/Terrorism (4.2%), National Security & Image (2.5%), Advertising & Traffic Diversion (1.6%), Cyber Black/Grey Industry (0.6%), and Illegal Trading Services (0.6%). These taxonomy percentages are computed over all 832 samples, including non-violative and ambiguous items.

- **Annotation Process and Consistency**: Each sample was independently annotated by two trained annotators according to the platform moderation guideline. Disagreements were adjudicated by a senior reviewer, and the final label was recorded together with a short

reasoning hint. On the double-annotated subset, inter-annotator agreement reached the substantial-agreement range (Cohen's $\kappa > 0.80$), indicating that the labels are largely governed by consistent platform rules rather than subjective preference.

- **Desensitization**: All Personally Identifiable Information (PII) such as real names (if detected), phone numbers, and social media IDs were replaced with generic placeholders (e.g., `[PHONE]`, `[ID]`) or unrelated random strings to preserve privacy.

- **Platform-Specific Annotation Standards**: As mentioned in the main text, the dataset incorporates platform-specific moderation standards. For instance, in this dataset, ecological and vulgarity controls are relatively loose: common internet profanities (e.g., "fuck", "cnm") and ambiguous slangs (e.g., "JB", "koujiao") without a clear, malicious target are not classified as violations or personal attacks. The same standards and proportions are maintained in the training, validation, and test partitions.

From the remaining Short-Text training pool, 10% is randomly sampled as the validation set for prompt selection.

Anonymized Examples:

Content Warning: This section contains explicit examples of severe profanity, sexual content, and cyberbullying/harassment derived from real-world adversarial data, which may be offensive to some readers.

To illustrate these standards, we provide several anonymized examples along with their annotation hints in Table 4:

Adversarial Attack Taxonomy. The dataset is categorized into a fine-grained taxonomy of distinct attack types, ranging from surface-level perturbations to deep semantic evasions. This taxonomy supports our detailed robustness analysis and includes, but is not limited to:

Homophonic Strategies (Phonetic):

- **Cross-lingual**: Using foreign words that sound like sensitive terms in the target language.

- **Numeric**: Using sequences of numbers that phonetically approximate sensitive phrases.

- **Emoji-to-Text Inference**: Using emojis which, when translated to their textual name, form a homophone (e.g., [Horse Emoji] $\rightarrow$ "Ma").

- **Multi-character Composite**: Combining multiple benign characters to phonetically reconstruct a sensitive word.

Glyph Mutation (Visual):

- **Decomposition (Splitting)**: Breaking a character into its component parts/radicals.

- **Composition (Merging)**: Combining components to mimic a structure.

- **Radical Substitution**: Replacing part of a character with a visually similar radical.

- **Pictographic Mimicry**: Using character shapes to visually depict sensitive objects or acts (Textual Pictograms).

Linguistic Encoding:

- **Rare/Obscure Characters**: Using valid but extremely rare characters to bypass blacklists.

- **Pinyin/Romanization**: Full pinyin substitution.

- **Pinyin Acronyms**: Using initial letters/abbreviations of sensitive pinyin phrases.

Semantic Evasion:

- **Metaphor**: Implicit references using symbolic language.

- **Indirect Suggestion**: Describing attributes or side-effects without direct naming.

- **Cultural Slang/Allusions**: Leveraging memes or historical references specific to a subculture.

Composite Attacks:

- **Multi-step Combinations**: Complex attacks integrating multiple methods above (e.g., Pinyin Acronyms + Glyph Decomposition).

A.2 Algorithm and Model Reproducibility

To support reproducibility, we provide the detailed hyperparameters and algorithm settings used in the PROSE framework. These settings specify how we balance the exploration-exploitation trade-off in the semantic-aware evolutionary process.

Label	Transaction /Traffic	Impersonation	Porn. /Vulg.	Ecological	Illegal /Reg.	Political	Total
White (non-violative)	7	11	69	228	32	59	406
Black (violative)	8	31	160	81	16	91	387
Grey (ambiguous)	3	6	7	11	2	10	39
Total	18	48	236	320	50	160	832

Table 3: First-level risk taxonomy by ground-truth label in the test set. Counts are computed over all 832 valid samples.

A.2.1 Multi-objective Target and Final Selection

In practice, the case-reliance proxy is integrated as an additional evaluation metric into the multi-objective evolutionary target set. We aim to identify a set of non-dominated solutions on the approximate Pareto front that balance Detection Performance and the Case-Reliance Proxy. When $\mathbf{P}(x)$ contains multiple metrics, we do not aggregate them into a single scalar during Pareto sorting; instead, they are treated as independent dimensions.

In the Short-Text moderation experiments, we instantiate the objective vector as [Precision, Recall, Case-Reliance Proxy] for Pareto sorting. By separating Precision and Recall as independent objectives rather than aggregating them into F1 or Accuracy during Pareto sorting, we explicitly maintain strategic diversity within the population. This encourages some prompts to evolve toward high Precision (stricter rules) and others toward high Recall (broader coverage). In practical deployment, this preserves a diverse set of approximate Pareto-optimal prompts on the final frontier, allowing product teams to select a high-Precision or high-Recall prompt based on specific business requirements. For our automated experimental reporting, however, Accuracy is used as the primary validation selector for greedy ordering and final prompt selection. The final test prompt is selected from the final population's first approximate Pareto front by choosing the candidate with the highest primary validation metric, and then evaluated once on the isolated test set.

A.2.2 Evolutionary Algorithm Hyperparameters

Table 5 lists the key hyperparameters used in our Genetic Algorithm (GA).

Note on Operator Sampling: The three operator weights are all set to 1.0. When Rewrite Mutation is disabled by the length gate, the remaining Reflective Mutation and Semantic Fusion Crossover probabilities are re-normalized to 50% each.

Note on Rewrite Mutation Gate: At the end of each generation, before producing offspring for the next generation, we compute the maximum prompt length among the parent population. If this maximum length exceeds the threshold L_{max}, Rewrite Mutation is unlocked for the next generation by restoring its sampling weight; otherwise, its weight remains zero, i.e., $W_{\text{rewrite}} = 0$. This length-based trigger reflects the observation that prompts in early iterations are usually short and contain only a small number of simple rules, making dedicated rewriting less necessary. As evolution proceeds, however, accumulated rules and knowledge may introduce redundancy, inconsistency, or opportunities for consolidation. In such cases, Rewrite Mutation serves as a global restructuring step that compactly reorganizes the prompt and helps maintain its usability as its complexity grows.

Algorithm 2 Generative Reproduction Strategy

Require: Elite Set $\mathcal{S}_{\text{elite}}$, Model Pool $\mathbb{M}_{\text{pool}}$, Template Pool $\mathbb{T}_{\text{pool}}$, Operator Weights W
Ensure: New Offspring child

```
Ensemble De-biasing:
Sample Model M ~ M_pool, Template T ~ T_pool
Operator Selection:
if W_rewrite > 0 then
    AvailableOps ← {Rewrite, Crossover, Reflect}
else
    AvailableOps ← {Crossover, Reflect}
end if
Op ← CategoricalSample(AvailableOps, W)
if Op = Rewrite then
    p ← TournamentSelect(S_elite)
    child ← REWRITEMUTATION(p, M, T)
else if Op = Crossover then
    p_a, p_b ← SelectDiverseParents(S_elite)
    child ← SEMANTICFUSIONCROSSOVER(p_a, p_b, M, T)
else
    p ← TournamentSelect(S_elite)
    child ← REFLECTIVEMUTATION(p, M, T)
end if
return child
```

Note on Parent Selection: For single-parent operators, TournamentSelect randomly samples 3 elite

Category	Original Text (Pinyin)	Literal Meaning	Label	Annotation Hint
Violative (Homophonic Evasion)	bǎ tā bì de shuì chá chū lái 把她避的税查出来	Find out the taxes she evaded	Violative	Homophonic evasion. 'bì de shuì' sounds like 'bī de shuǐ' (vulgar slang for vaginal fluids).
Violative (Obscene Content)	ài lù guǎn de rù 爱[Emoji_Deer]管的入	Those who love deer tubes enter	Violative	Homophonic expression. 'lù guǎn' (deer tube) sounds like 'lū guǎn' (masturbation).
Violative (Vulgar Content)	dì dà fān kāi xǐ 弟大翻开洗	If brother is big, flip open and wash	Violative	Vulgar sexual implication. 'dì' (brother) phonetically implies penis; 'fān kāi xǐ' implies washing the foreskin.
Violative (Family Death Curse)	[Name] de hù kǒu běn zhǐ shèng yī yè [Name]的户口本只剩一页	[Name]'s household register has only one page left	Violative	A severe curse implying that all other family members of the target are dead.
Violative (Homophonic Obscenity)	hào xiāng shè lì miàn 耗香舍利面	Consume fragrant relic noodles	Violative	Homophonic evasion. Sounds like 'hǎo xiǎng shè lǐ miàn' (really want to ejaculate inside).
Non-violative (ACG Culture)	hēi lǎo dà 黑老大	Black boss	Non-violative	Normally means 'gangster boss', but in ACG culture, it is often a benign nickname for a specific character or streamer.
Non-violative (Exaggerated Expression)	shā rén bù zhǎ yǎn 杀人不眨眼	Kill without blinking an eye	Non-violative	A common exaggerated idiom used in literary, character-description, or gaming contexts; does not constitute incitement to violence.
Non-violative (Gaming Context)	yī qiāng tái zǒu nǐ 一枪抬走你	One shot to carry you away	Non-violative	A common colloquial expression in FPS games describing defeating an opponent during combat.
Non-violative (Tone Particle)	wǒ lèi ge sāo gāng 我嘞个骚肛	My gosh, smelly anus	Non-violative	Pragmatic shift: Although literally vulgar, it has evolved into a popular internet meme acting purely as a tone particle expressing shock (like 'Oh my god').
Non-violative (Weakened Phrase)	nǐ nǎi nǎi dé 你乃乃德	Your grandma's	Non-violative	Homophonic form of a common interjection; its insulting force has completely weakened in contemporary casual usage.

Table 4: Anonymized examples with annotation hints.

candidates and selects the one with the highest validation Accuracy. For Semantic Fusion Crossover, SelectDiverseParents first selects the first parent by the same tournament rule. It then samples 3 additional candidates and selects the second parent with the highest validation Accuracy after applying the semantic similarity penalty relative to the first parent. This keeps crossover performance-oriented while discouraging near-duplicate parents.

Note on Data Split: Reflective mutation observes only samples from a training mini-batch to generate feedback. Pareto selection and final prompt selection are computed on a disjoint validation set. The test set and OOD held-out folds are reserved for final performance reporting.

Note on Bad Case Sampling for Reflection: To efficiently generate reflective feedback, we do not evaluate the prompt on the entire dataset during the mutation phase. Instead, we sample a *Reflection Mini Batch* (size 64) from the training data. The prompt is evaluated on this mini-batch to identify failure cases (False Positives and False Negatives). From these identified failures, we select a subset (size 16, defined as *Bad Case Size for Reflection*) to be included in the meta-prompt. This keeps the reflection model focused on representative error examples without incurring the cost of full-dataset evaluation.

Note on Statistical Reporting: Unless otherwise specified, reported main-test and OOD metrics are arithmetic means over 5 independent runs with different random seeds. The compact main comparison table reports means; Short-Text standard deviations and first-level risk-group breakdowns are provided in Table 6.

A.2.3 Model Configuration

We employ a heterogeneous model ensemble to mitigate inductive biases. The specific models and their roles are configured as follows. *(Note: These are functionally equivalent to publicly available open-weight models and APIs, such as GLM-4.6*

Table 5: Detailed Hyperparameters for PROSE Framework

Parameter	Value
General Evolution Settings	
Generations (G)	24
Population Size (N)	8
Elite Size (E)	2
Tournament Selection Size	3
Rewrite Length Threshold (L_{max})	1000
Operator Sampling Weights	
Crossover Weight ($W_{\text{crossover}}$)	1.0
Reflective Mutation Weight (W_{reflect})	1.0
Rewrite Mutation Weight (W_{rewrite})	1.0
Diversity Control	
Similarity Threshold (τ)	0.5
Diversity Penalty Coefficient (α)	0.5
Diversity Penalty Bias (β)	0.005
Sampling & Evaluation	
Evaluation Concurrency	16
Offspring Generation Concurrency	6
Reflection Mini Batch Size	64
Bad Case Size for Reflection	16

and DeepSeek-V3, supporting reproducibility upon publication.)

- **Execution Model**: Self-deployed `Qwen3-Next-80B-A3B-Instruct`. This model is used only for primary task execution and candidate-prompt evaluation on training mini-batches, validation sets, and test sets.
- **Reflection Model Pool**: Used for Reflective Mutation, Rewrite Mutation, Semantic Fusion Crossover, and analysis. Models sampled from this pool generate prompt variants and reflective diagnoses.
  - `GLM-4.6` (Weight: 2)
  - `Kimi-K2-Thinking` (Weight: 2)
  - `Qwen3-Next-80B-A3B-Thinking` (Weight: 1)
  - `DeepSeek-V3.1` (Weight: 2)
  - `DeepSeek-R1-0528` (Weight: 2)
- **Example Extraction Model**: `Qwen3-4B`, used specifically for extracting case content and calculating the case-reliance proxy.

Model sampling is implemented at the LLM gateway layer and exposed to all reflection- or generation-based optimizers used as baselines. Each optimizer calls the same logical API model name, `Reflection-Model`. The gateway samples the concrete backend model according to the configured pool weights and returns the response through the same API contract. Baselines therefore require no algorithmic changes for model sampling; PROSE's distinctive components are the case-reliance objective, semantic selection, and evolutionary operator design.

A.2.4 Baseline Configuration

For the main comparison, all prompt optimization methods are constrained to the same candidate-evaluation budget and the same maximum token budget. Within this common budget, baseline optimizers keep their original internal update rules and sampling policies, except that their LLM client is pointed to our internal API endpoint. To avoid an initialization advantage, every method starts from the same minimal task prompt. For Short-Text and ToxiCN, this prompt only instructs the execution model to decide whether the input is toxic/non-toxic or violation/non-violation and to return the required JSON schema. For HotpotQA, the seed prompt only specifies the answer-generation interface of the final hop. Reflection- or generation-based baselines request `Reflection-Model` for prompt generation or textual feedback, while the gateway applies the same hidden model-pool sampling described above.

A.2.5 Template Selection Probabilities

To further enhance diversity, we use a uniform probability distribution for selecting meta-prompt templates during mutation and crossover operations.

In practice, relying on a single meta-prompt template introduces a form of inductive bias similar to that of a single model, potentially trapping the evolution in local optima. To mitigate this, we employ a Multi-Template Random Sampling strategy. We designed a pool of diverse meta-prompts with varying instructional styles (e.g., some emphasizing rule abstraction, others focusing on edge-case correction) and sample from this pool during each mutation step. This approach, analogous to the Heterogeneous Model Ensemble Sampling strategy, supports diverse evolutionary trajectories.

- **Reflective Mutation Templates**: All 4 templates (Reflection 1-4) are assigned equal

weights (1:1:1:1).

- **Rewrite Mutation Templates**: All 4 templates (Rewrite 1-4) are assigned equal weights (1:1:1:1).
- **Crossover Templates**: All 4 templates (Crossover 1-4) are assigned equal weights (1:1:1:1).

A.2.6 Efficiency and Cost Analysis

We evaluated the computational cost of PROSE using a hardware configuration with 4 threads for generation and 8 threads for evaluation. On average, generating and evaluating a single offspring takes approximately 3 minutes. Starting from a blank prompt, the framework typically requires generating about 100 offspring to reach convergence. However, utilizing a human-crafted prompt as a seed accelerates this process, consistently achieving convergence within 20 generated offspring. This shows PROSE's flexibility in both cold-start (autonomous discovery) and warm-start (human-assisted optimization) scenarios.

A.2.7 Token Consumption Analysis

To provide a transparent view of the computational costs, we detail the average token consumption required to generate and evaluate a single offspring during the evolutionary process. The consumption is categorized by model role:

- **Reflection Model** (e.g., GLM/DeepSeek): Generating a single offspring (including error analysis and new prompt synthesis) consumes approximately **4,000 tokens**.
- **Example Extraction Model** (for Case-Reliance Proxy): Extracting few-shot examples to calculate the proxy score consumes approximately **1,500 tokens** per prompt.
- **Execution & Evaluation Model**: The consumption for this model is dynamic and depends on the dataset size and prompt length. It is calculated as:

$$\text{Tokens}_{\text{exec}} \approx (N_{\text{val}} + N_{\text{minibatch}}) \times L_{\text{prompt}} \tag{4}$$

where N_{val} is the size of the validation set, $N_{\text{minibatch}}$ is the size of the mini-batch used for reflection (e.g., 12-64), and L_{prompt} is the current length of the prompt.

It is worth noting that in practice, over **95%** of the total token consumption is attributed to the execution and evaluation phase. Consequently, we **strongly recommend using a self-deployed model** (e.g., via vLLM) as the execution engine to mitigate costs and ensure efficiency.

B Additional Results

B.1 Short-Text Standard Deviations and Risk-Group Breakdown

B.2 OOD Resilience Evaluation

Robustness to Unseen Attacks (OOD Analysis). We report detailed mean results for the test set. Under the LOGO protocol, the binary Short-Text data is partitioned into six first-level risk groups. In each fold, prompt optimization is performed only on the training/validation data after removing the target risk group, and evaluation is then conducted on the corresponding held-out risk group in the test set. The test samples are never used for reflection, Pareto selection, or prompt selection. Grey ground-truth samples are excluded from metric computation.

C Ablations and Controls

C.1 Component Ablations

Table 11 reports the matched ablation values corresponding to Figure 4.

This supports our hypothesis that semantic-aware selection is critical for preventing semantic collapse. Without the embedding-based diversity constraint, the population is more likely to converge toward a narrow defensive logic, which weakens robustness across heterogeneous attack patterns and leads to broad performance degradation. Model Ensemble Sampling and Semantic Fusion Crossover show disproportionately large F1 drops (-5.56 and -7.88 points, respectively) relative to their Accuracy drops (-2.02 and -4.28 points). This pattern is consistent with their intended role in expanding semantic coverage: heterogeneous reflection models and fusion between complementary prompts help recover difficult or long-tail positive cases, improving the precision–recall balance rather than only optimizing majority-class accuracy. The case-reliance proxy objective also produces large and balanced drops in both metrics (-6.77 Accuracy and -7.04 F1), further confirming that explicit case-reliance control helps reduce shortcut learning. Reflective Mutation is especially impor-

Table 6: Short-Text performance with standard deviations across 5 independent runs. Values are percentages and reported as mean ± standard deviation. Overall metrics use the same Short-Text metric definitions as Table 1; category metrics follow the first-level risk taxonomy. The table reports methods for which seed-level category logs were retained; PromptBreeder is included in the main comparison table but not in this category-level breakdown.

Model	Overall		Transaction/Traffic		Impersonation		Political		Ecological		Porn./Vulg.		Illegal/Reg.	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
Baseline	52.37±0.57	61.58±0.44	48.89±7.70	60.82±7.30	65.87±7.65	76.30±6.48	53.78±2.04	61.30±2.57	41.10±0.86	41.29±0.46	67.83±1.40	79.52±0.85	36.11±2.41	42.50±1.86
TextGrad	53.64±3.57	57.14±3.80	53.33±7.64	55.08±13.02	46.83±21.34	41.91±33.55	50.00±2.91	52.67±7.45	48.00±9.43	42.22±1.47	67.83±1.76	77.06±4.83	39.58±16.27	24.48±11.54
MIPROv2	52.21±0.70	60.18±1.53	55.56±3.85	63.99±5.63	61.90±2.38	71.74±1.58	47.56±3.67	51.83±9.76	43.15±2.71	42.17±1.22	68.85±0.25	79.91±0.69	36.11±3.18	41.73±1.65
OPRO	52.08±0.25	60.16±0.61	51.11±7.70	62.22±3.85	57.94±15.85	62.97±21.51	49.56±4.29	54.85±4.00	42.72±1.17	42.22±0.92	68.70±2.15	79.99±1.63	36.11±4.81	39.32±6.45
ProTeGi	52.42±0.32	60.70±0.33	53.33±6.67	65.56±5.09	71.43±2.38	79.94±2.35	51.11±3.36	59.11±2.84	40.67±0.81	40.98±0.65	68.85±1.10	80.29±0.83	36.81±2.41	42.77±2.18
GEPA	61.71±1.43	60.73±3.61	62.22±10.18	65.00±10.93	63.49±11.00	67.76±13.51	46.44±3.67	31.78±11.19	66.34±1.62	49.51±4.19	65.65±5.77	72.37±5.61	59.03±1.20	21.18±3.28
PROSE (Ours)	**76.29±2.59**	**73.42±3.73**	**73.33±7.34**	**66.77±9.78**	**76.99±5.74**	**83.46±8.53**	**74.44±6.99**	**75.53±9.84**	**77.89±4.35**	**53.27±6.88**	**76.71±5.04**	**82.24±4.49**	**70.14±9.84**	**43.00±11.28**

Table 7: Detailed GEPA mean performance on the full test set. All metric values are percentages.

Risk Group	Acc	F1	P	R	FNR	FPR
Overall	61.71	60.73	63.20	51.68	48.32	28.74
Transaction/Traffic Diversion	62.22	65.00	64.05	66.67	33.33	42.86
Impersonation/Misleading	63.49	67.76	93.18	55.91	44.09	15.15
Pornography/Vulgarity	65.65	72.37	82.13	64.79	35.21	32.37
Ecological Governance	66.34	49.51	40.90	63.79	36.21	32.75
Illegal/Regulatory	59.03	21.18	29.52	16.67	83.33	19.79
Political Sensitivity	46.44	31.78	69.15	21.25	78.75	14.69

Table 8: Detailed GEPA mean LOGO-OOD performance by held-out first-level risk group. All metric values are percentages. Macro averages are computed across the six held-out groups.

Held-out Group	Acc	F1	P	R	FNR	FPR
Overall (micro)	45.02	31.07	40.02	25.41	74.59	36.29
Average (macro)	47.30	22.20	60.13	21.08	78.92	13.73
Transaction/Traffic Diversion	51.11	14.81	66.67	8.33	91.67	0.00
Impersonation/Misleading	38.10	31.27	84.87	19.35	80.65	9.09
Pornography/Vulgarity	40.32	26.50	94.94	15.42	84.58	1.93
Ecological Governance	47.46	42.09	29.60	72.84	27.16	61.55
Illegal/Regulatory	64.58	3.51	11.11	2.08	97.92	4.17
Political Sensitivity	42.22	15.03	73.61	8.42	91.58	5.65

Table 9: Detailed PROSE mean performance on the full test set. All metric values are percentages.

Risk Group	Acc	F1	P	R	FNR	FPR
Overall	76.29	73.42	81.21	67.53	32.47	15.35
Transaction/Traffic Diversion	73.33	66.77	87.78	62.50	37.50	14.29
Impersonation/Misleading	76.99	83.46	86.94	80.65	19.35	33.33
Pornography/Vulgarity	76.71	82.24	87.52	77.92	22.08	26.09
Ecological Governance	77.89	53.27	61.67	47.32	52.68	11.26
Illegal/Regulatory	70.14	43.00	54.44	37.50	62.50	13.54
Political Sensitivity	74.44	75.53	85.59	68.50	31.50	16.38

Table 10: Detailed PROSE mean LOGO-OOD performance by held-out first-level risk group. All metric values are percentages.

Held-out Group	Acc	F1	P	R	FNR	FPR
Overall (micro)	63.18	60.43	63.53	57.62	42.38	31.53
Average (macro)	61.18	53.99	65.10	50.27	49.73	25.93
Transaction/Traffic Diversion	66.67	66.67	71.43	62.50	37.50	28.57
Impersonation/Misleading	57.14	60.87	93.33	45.16	54.84	9.09
Pornography/Vulgarity	66.38	74.59	79.02	70.62	29.38	43.48
Ecological Governance	67.31	53.46	42.65	71.60	28.40	34.21
Illegal/Regulatory	56.25	22.22	27.27	18.75	81.25	25.00
Political Sensitivity	53.33	46.15	76.92	32.97	67.03	15.25

Table 11: Matched ablation results on Short-Text. Drops are computed against full PROSE.

Variant	Acc	Δ Acc	F1	Δ F1
Full PROSE	76.29	–	73.42	–
w/o Case-Reliance Proxy	69.52	-6.77	66.38	-7.04
w/o Semantic Greedy Selection	62.45	-13.84	62.00	-11.42
w/o Reflective Mutation	68.22	-8.07	67.76	-5.66
w/o Rewrite Mutation	73.77	-2.52	72.18	-1.24
w/o Semantic Fusion Crossover	72.01	-4.28	65.54	-7.88
w/o Multi-Template Sampling	73.96	-2.33	70.07	-3.35
w/o Model Ensemble Sampling	74.27	-2.02	67.86	-5.56

tant for Accuracy (-8.07 points), suggesting that feedback-driven logical correction improves the stability of the evolved prompts.

C.2 Control Experiments on Case-Reliance

We conducted control experiments on the Short-Text dataset to separate the effect of the case-reliance objective from generic prompt shortening. We compared PROSE against several matched baselines:

1. **Trivial Deletion**: Randomly deleting examples from the baseline prompt to match the average case ratio of PROSE.
2. **Length-Matched Baseline**: Truncating the baseline prompt to strictly match PROSE's total token length.
3. **Length Objective Control**: Replacing our Case-Reliance Proxy $A(x)$ with a simple "Minimize Total Prompt Length" objective during evolution.

Results indicate that the Trivial Deletion and Length-Matched baselines suffer severe F1 drops (averaging -7.4% compared to PROSE), suggesting that naively removing examples can discard critical task context. Furthermore, substituting $A(x)$ with the Length Objective Control results in overly brief prompts lacking necessary reasoning depth,

yielding only 64.21% F1 (compared to PROSE's 73.42%). These controls support using example-enumeration as the regularization target rather than total prompt length alone.

C.3 Low-Token Budget Stress Test

The main comparison already constrains all optimization methods to the same candidate-evaluation budget and maximum token budget. To further test behavior under tighter resource limits, we conducted an additional low-token stress test on the Short-Text dataset. In this stress setting, evolutionary/search baselines (PROSE, GEPA, and MIPROv2) were constrained to a stricter budget: a maximum of 10^7 API tokens (including reflection and semantic embedding tokens) and 100 candidate evaluations. Under this stricter low-token budget, PROSE still achieved an ACC of 70.85%, higher than GEPA (58.20%) and MIPROv2 (51.05%). We observed that traditional methods (e.g., GEPA, MIPROv2) fail to effectively utilize additional compute; during their population iteration, errors accumulate faster than performance gains, leading to stagnation or performance degradation typically within 10 generations. In contrast, PROSE reaches superior performance using fewer tokens. This stress test suggests that PROSE's advantage is not explained only by spending more computational resources.

D Case-Reliance Proxy Details

To estimate the case-reliance proxy score ($A(x)$) of a prompt, we first estimate the ratio of content dedicated to specific few-shot examples versus general rules. We employ a dedicated Example Extraction Model ($\mathcal{M}_{\text{ratio}}$, implemented using Qwen3-4B) to identify and extract "enumerative" content from the prompt.

The extraction is performed using a few-shot prompt that defines the task of separating "Case Content" (specific examples, blacklists, tables) from "Rule Content" (abstract definitions, logic flows).

We observed that LLMs may inject examples anywhere in the prompt rather than confining them to a fixed "few-shot" section; thus, we utilize a dedicated case-extraction model to extract example components dynamically. The extraction model is used only for estimating $A(x)$ and is separate from both the prompt-generation reflection models and the task-execution model. Furthermore, while we experimented with restricting total prompt length, we found that length naturally scales with task difficulty, making it a non-robust constraint. Under the dual survival pressure of maximizing task metrics and maximizing the case-reliance proxy, the model is encouraged to evolve toward abstract methodological reasoning rather than accumulating error-prone examples. To reduce stochastic extraction noise, this metric is calculated using median sampling $\text{Median}_{i=1}^{K}(\cdot)$.

D.1 Extraction Meta-Prompt Strategy

The meta-prompt includes a clear task definition and two few-shot demonstrations to guide the model. Below is a translated summary of the meta-prompt used:

```
You are a prompt analysis expert proficient in prompt
      engineering. Your task is to extract the specific text
      of cases from the prompt below.

# Task Instructions
I want to extract the text content of specific cases from a
      prompt. Please read the prompt completely and output
      the cases, specific details, case analyses, etc.,
      contained within.
Special Attention:
    - Your authority is limited to slicing: cut out the
          parts containing cases from the prompt and output
          them separated by commas.
    - Output cases in the order they appear in the original
          prompt.
    - Case content must be complete and accurate.
    - If there is no case content, output blank.
    - <Extraction Example 1> and <Extraction Example 2> are
          two examples of this task.

# Extraction Example 1:
## Input Prompt for Example 1:
<prompt>
You are a content moderator responsible for identifying
      offending usernames. Usernames may contain text,
      homophones, metaphors, or references used to spread
      prohibited content.

## Violation Types include but are not limited to:
1. **Vulgarity**: Direct or phonetic representations of
      sexual organs or acts. E.g., "Phuk" (Fuck), "Biatch" (
      Bitch).
2. **Adverse Impact**: Promoting violence, crime, or drugs.
      E.g., "MethSales", "JoinISIS".
3. **Discrimination**: Hate speech or fraud.

## Special Attention to Homophones/Leetspeak:
- Examples: "Phuk" -> "Fuck"; "K1ll" -> "Kill"; "@$$" -> "
      Ass".
- Note: Homophones must be obvious and malicious.

## Critical Conflict Case Calibration (Internal Criteria)
| Username | Verdict | Logic |
| "PhukMi" | Violation | Phonetic "Fuck Me", explicit, no
      safe context |
| "GlassCockpit" | Non-violation | "Cock" is part of
      technical aviation term, safe context |
| "K1llThemAll" | Violation | Leetspeak for "Kill Them All",
       violent threat |
</prompt>

## Expected Output Case for Example 1:
<case>
"Phuk" (Fuck), "Biatch" (Bitch)., "MethSales", "JoinISIS".,
      "Phuk" -> "Fuck"; "K1ll" -> "Kill"; "@$$" -> "Ass"., |
      "PhukMi" | Violation | Phonetic "Fuck Me", explicit, no
       safe context |, | "GlassCockpit" | Non-violation | "
      Cock" is part of technical aviation term, safe context
      |, | "K1llThemAll" | Violation | Leetspeak for "Kill
      Them All", violent threat |
</case>

# Extraction Example 2:
## Input Prompt for Example 2:
<prompt>
You are a safety expert.
Rules:
1. No Personal Identifiable Information (PII). e.g., "My ID
      is 123456789".
2. No Harassment.
</prompt>

## Expected Output Case for Example 2:
<case>
"My ID is 123456789".
</case>
```

We perform this extraction 3 times (configured as `example_ratio_samples`) and use the median length ratio to ensure robust estimation against model stochasticity before calculating the final proxy score.

D.2 Extraction Stability Check

We additionally evaluated the stability of the case-content extractor on 16 base prompts, with 32 successful extraction runs for each prompt (512 successful runs in total). The overall extracted case ratio had a mean of 18.93%, a median of 18.84%, and a sample standard deviation of 6.21 percentage points. 13 of the 16 prompts showed high output consistency, with average text similarity above 91%; the remaining prompt was deliberately more ambiguous and produced larger variation, motivating our use of repeated extraction and median aggregation. This check suggests that the case-reliance proxy is sufficiently stable for relative prompt selection, while still benefiting from median sampling to reduce outlier effects.

D.3 Human Agreement and Sampling Sensitivity

To further validate that the extraction model measures case reliance rather than arbitrary prompt length, we manually annotated the case-content lengths for the same 16 base prompts and compared them against repeated model extractions. With 32 extraction runs per prompt, mean-length aggregation achieved Pearson $r = 0.920$, Spearman $\rho = 0.800$, Kendall $\tau_b = 0.667$, and MAPE $= 12.3\%$ against the human annotations. Median-length aggregation, which matches our use of repeated robust estimation, achieved Pearson $r = 0.811$, Spearman $\rho = 0.800$, Kendall $\tau_b = 0.667$, and MAPE $= 13.6\%$.

At the individual-run level, the median Spearman correlation against the human ordering was 0.800, with no negative rank correlations observed. We also performed a bootstrap sensitivity analysis by sampling m extraction runs per prompt with replacement and aggregating before computing rank correlation with human annotations. For $m = 3$, matching the default `example_ratio_samples`, median aggregation produced a median Spearman correlation of 0.800, $P(\rho_{\text{rank}} \geq 0.4) = 99.1\%$, and $P(\rho_{\text{rank}} < 0) = 0.0\%$. Mean aggregation at the same sampling budget produced mean Spearman correlation 0.778 and $P(\rho_{\text{rank}} \geq 0.8) = 90.1\%$. These results indicate that the proxy is reliable for relative evolutionary selection, although it should be interpreted as a regularization signal rather than an exact human-equivalent segmentation measure.

E Operator Prompt Templates

This appendix provides the detailed meta-prompts used in the PROSE framework. To implement the **Ensemble-based De-biasing Strategy**, we do not rely on a single fixed instruction for mutation or crossover. Instead, we maintain a diverse pool of meta-prompts, each representing a distinct optimization philosophy (e.g., Data-Driven vs. Logic-Driven, Additive vs. Subtractive). During the evolutionary process, random sampling from these pools reduces the risk that the population converges to the inductive biases of any single reflection pattern.

E.1 Pool of Reflective Mutation Templates

The Reflective Mutation operator randomly samples from the following four templates to generate offspring. Each template guides the LLM to correct errors from a unique perspective.

Template 1: Standard Balanced Optimization. Focus: A comprehensive approach that balances performance metrics (Recall, Precision) with case analysis to optimize the prompt structure.

```
You are a Senior Prompt Optimization Expert proficient in
      LLMs, linguistics, and content moderation systems. Your
       goal is to optimize a given content moderation Prompt
      based on model performance and badcase analysis.

### [Input Info]
1. **Prompt to Optimize**
```
<prompt>
{curr_instructions}
</prompt>
```

2. **Performance Metrics**
    - Case Ratio: {case_ratio:.1%}
    - Performance in last test: Recall: {recall:.1%},
          Precision: {precision:.1%}

3. **Metrics Explanation**
    * **Case Ratio**: The proportion of text in the Prompt
          describing specific cases. Too low may lead to
          insufficient generalization; too high may lead to
          overfitting.
    * **Recall**: The proportion of actual violations
          detected.
    * **Precision**: The proportion of detected violations
          that are actual violations.

4. **Run Results Summary (Bad Cases)**
```
{feedback_text}
```

### [Optimization Task]
Please:
1. Read the bad cases and analyze issues in the existing
      Prompt's discrimination logic, semantic expression, or
      instruction form.
2. Focus modifications on parts directly related to the bad
      cases; keep unrelated parts unchanged unless there is a
       clear conflict.
3. Focus optimization on:
    * Deleting or merging repetitive, contradictory, or
          redundant statements;
    * Making language more precise, concise, and logical;
    * Keeping the Case Ratio reasonable;
4. For each modification, list:
    * [Modification ID]
    * [Difference Before/After]
    * [Reason]
5. Do not change the output format (JSON Schema).
6. Prioritize logical generalization: Fix the general logic
      loophole rather than just stacking "special cases".

### [Output Requirements]
1. **Optimization Analysis List**
2. **Final Optimized Prompt**, wrapped in:
```
<prompt>
(Optimized Prompt Content)
</prompt>
```
```

Template 2: Data-Driven Optimization (Few-Shot Focus). Focus: Prioritizes the use of specific examples ("Few-Shot") to correct errors, mimicking a data-driven learning approach.

```
You are a prompt expert focused on "Data-Driven Optimization
      ". Your task is to fix the current prompt based on
      error cases (Bad Cases).

### Core Principles
1. **Prioritize Examples (Few-Shot)**: The best way to fix
      an error is often not to add a complex rule, but to
      directly add the error case and its correct analysis to
       the "Examples" section of the Prompt.
2. **Reject Overfitting**: Do not write a generic rule that
      will only trigger once for an isolated, weird case.
3. **Maintain Pass Rate**: Ensure the modified Prompt
      remains simple and clear. Do not introduce complex
      logic (like math or multi-layer reasoning) that induces
       hallucinations.
4. **Merge Similar Cases**: If the case library has similar
      cases, merge them into a more general statement to
      avoid redundancy.

### Input
1. **Current Prompt**:
```
{curr_instructions}
```
2. **Bad Cases**:
```
{feedback_text}
```

### Task
Fix the Prompt to resolve the above errors.
- If the error is due to unclear concepts, streamline or
      clarify definitions.
- If the error is an edge case, add it as a Negative/
      Positive Example.
- **Strictly Forbid** adding complex processes like "Step x
      ", "Weight y".

### Output
Output only the optimized Prompt, wrapped in <prompt> tags.
```
<prompt>
...
</prompt>
```
```

Template 3: Deep Logic Correction (No Patching). Focus: A rigorous "Logician" style that forbids simple patching. It forces the model to redefine underlying concepts (e.g., "violence", "slang") to resolve conflicts at the root.

```
You are a strict Logician and Prompt Optimization Expert.
      The current Prompt made mistakes on specific cases.
      Your task is to **fix the underlying discrimination
      logic**, not simply add "patches".

### Core Principles: NO PATCHING
- **Wrong**: Because the model mistakenly flagged "potato"
      as violation, adding "Note: potato is not violation".
      -> **Strictly Forbidden!**
- **Right**: Ask "Why did the model think potato was
      violation?" (Food? Slang?). Modify the **definition of
      Food** or **logic for Slang**, so "potato" naturally
      becomes non-violation.

### Input Info
1. **Current Prompt**:
```
{curr_instructions}
```
2. **Bad Cases**:
```
{feedback_text}
```

### Optimization Task
```

```
Analyze the Bad Cases:
1. **Attribution**: Find the **rule loophole** or **fuzzy
      definition**. Is the rule too rigid? Lacking common
      sense? Priority confusion?
2. **Logic Rewrite**:
   - **Modify Definitions**: Redefine fuzzy concepts.
   - **Adjust Flow**: Introduce new judgment steps (e.g., "
         Literal first, then Context").
   - **Provide Principles**: Use "Principles" instead of "
         Rule Lists".
3. **Simplify**: If adding content is necessary, try
      deleting old rules or redundancy to keep it concise.

### Output Requirements
1. **Optimization Analysis**: Briefly explain the logic
      loophole and how you redefined it.
2. **New Prompt**: Complete, optimized prompt wrapped in <
      prompt>.
```

Template 4: Abstract Rule Generalization. Focus: A "Structural" approach that encourages subtraction and abstraction, aggressively consolidating rules to prevent rule bloating.

```
You are a rigorous Logician and Prompt Optimization Expert.
      Your task is to analyze Bad Cases and fix the Prompt
      from the "underlying discrimination logic" level.

### Core Principles
1. **Reject "Patching"**: Strictly forbid adding a specific
      "exception rule" for every single error. If the model
      frequently misjudges "game terms", optimize the "
      definition logic of game terms", not list every term.
2. **Abstract Generalization**: Extract commonalities from
      Bad Cases. Is the "homophone logic" too aggressive?
      Modify the corresponding general rule.
3. **Subtraction**: If rules are redundant or conflicting,
      boldly delete or rewrite the entire paragraph. Keep the
       logical backbone clear.

### Input Information
1. **Current Prompt**:
```
{curr_instructions}
```
2. **Evaluation Feedback (Bad Cases)**:
```
{feedback_text}
```
3. **Performance Metrics**: Case Ratio {case_ratio:.1%}. If
      too high, streamline cases.

### Task
Optimize the Prompt.
- **Must** analyze: Which core logic loophole do these Cases
       reflect?
- **Must** reflect: Will this modification lead to new
      misjudgments?
- Output the final complete Prompt.

### Output Format
```
<prompt>
(Optimized Complete Prompt)
</prompt>
```
```

E.2 Pool of Semantic Fusion Crossover Templates

The Semantic Fusion Crossover operator determines how to merge two parent prompts. To explore different combination strategies (e.g., structural merging vs. logic simplification), we sample from the following templates.

Template 1: Comprehensive Analytical Fusion. Focus: Detailed performance-based merger, analyzing F1/Precision/Recall to decide which parent's strategy to keep.

```
You are a Senior Prompt Optimization Expert. Your task is to
       analyze the differences between two Prompts and
      generate a better version combining their strengths.

### [Input Info]
1. **Prompt A** ({case_ratio_a:.1%}, F1: {f1_a:.1%})
```
<prompt>
{instructions_a}
</prompt>
```
2. **Prompt B** ({case_ratio_b:.1%}, F1: {f1_b:.1%})
```
<prompt>
{instructions_b}
</prompt>
```
3. **Key Case Comparison (Conflict Cases)**
```
{cases_text}
```

### [Analysis & Optimization]
1. Compare differences in structure, semantics, strategy,
      and logic.
2. Analyze why each Prompt performs better/worse in
      different scenarios based on the cases.
3. Determine which strategies have better generalization and
       stability.
4. Design a merged Prompt, retaining effective strategies
      and removing conflicts.
5. Explicitly state:
    * Key differences and their impact;
    * Which strategy is better when;
    * Core optimization logic.

### [Output]
1. **Analysis Report**
2. **Final Optimized Prompt** wrapped in <prompt>.
```

Template 2: Occam's Razor Simplification. Focus: Aggressively simplifies the prompt, favoring fewer rules and higher generalization ("Subtraction").

```
You are a Prompt Expert believing in "Occam's Razor". Your
      goal is to merge two prompts into a simpler, more
      efficient version.

### Core Principles
1. **Subtraction**: If a complex rule can be replaced by a
      precise Few-Shot example, choose the example.
2. **Remove Fakes**: Use Badcases to identify the truly
      effective core instructions, discarding flashy but
      ineffective "Chain of Thought steps" or "Dynamic
      Weights".
3. **Complementarity**: If Prompt A solves a case that B
      cannot, choose A's rule.

### Input
1. **Prompt A** (F1: {f1_a:.1%}) ...
2. **Prompt B** (F1: {f1_b:.1%}) ...
3. **Conflict Cases**:
```
{cases_text}
```

### Task
Generate a new Prompt:
- **Flat Structure**: Avoid nesting > 2 layers.
- **Direct Instructions**: Tell the model "what is violation
      ", not "how to calculate score".
- **Include Key Cases**: Ensure coverage of conflict cases.

### Output
Only the final Prompt wrapped in <prompt>.
```

Template 3: Structural Reorganization. Focus: Rebuilds the prompt's architecture to unify the underlying thinking patterns of both parents.

```
You are a Prompt Expert with deep architectural design
      skills. Your task is not to simply "merge" structure,
      but to create a concise, powerful new version through
      **Structural Reorganization** and **Logic Elevation**.

### Core Goal
Break the "Patching" cycle. Identify underlying **Thinking
      Patterns** of A and B, and build a new framework.

### Input
1. **Prompt A** ...
2. **Prompt B** ...
3. **Conflict Cases** ...

### Optimization Instructions
1. **Denoise & Extract**:
```

```
   - If A has specific Bad Case enumerations, but B has a
         general rule covering them, prioritize B's **General
          Rule**.
   - **Occam's Razor**: Do not multiply entities without
         necessity.
2. **Structural Refactoring**:
   - Choose the clearest structure (Document, Code, CoT) or
         design a hybrid one.
   - Ensure the structure naturally covers logic for
         conflict cases without forced exceptions.
3. **Logic Elevation**:
   - If A and B conflict on a concept (e.g., "Homophone"),
         **Redefine** it more precisely.
   - Introduce **Reflective Steps**: Add "Check X first,
         then Y".

### Output
A complete, executable new Prompt wrapped in <prompt>. No
      explanatory text.
```

Template 4: Genetic Recombination. Focus: Treats prompt sections as genes, performing explicit "genetic recombination" of strategies (e.g., A's filter + B's reasoning).

```
You are an Architect-level Prompt Expert. Your task is to
      perform "Gene Recombination" on Prompt A and Prompt B
      to create a structure with better logic.

### Core Principles
1. **Structural Refactoring**: Do not just Stitch text.
      Understand their "Discrimination Strategies".
   - A might excel at "Layered Filtering".
   - B might excel at "CoT Reasoning".
   - Design a new structure fusing these advantages.
2. **Synthesize**:
   - If A's "Homophone Rule" is good, take A's.
   - If B's "Game Term Exemption" is good, take B's.
   - Delete repetitive, invalid instructions.
3. **Logical Consistency**: Ensure all parts follow a
      unified standard.

### Input
1. **Prompt A** ...
2. **Prompt B** ...
3. **Conflict Cases** ...

### Task
Write a new Prompt that looks like a carefully reconsidered
      version, not a stitched monster.

### Output
```
<prompt>
(New Complete Prompt)
</prompt>
```
```

E.3 Pool of Rewrite Mutation Templates

The Rewrite Mutation operator is designed for **structural optimization** and **logic distilling** without relying on specific error cases. To reduce the risk that prompts evolve into a single fixed structure, we employ a diverse pool of rewrite experts, each with a distinct optimization philosophy.

Template 1: Structural Organizer (The "Organizer"). Focus: Enhances clarity and retrieval efficiency by enforcing a standard modular structure (Rule, Principles, Few-Shot, Output) without changing the core logic.

```
You are a Prompt Optimization Expert skilled in structural
      thinking (Codename "Organizer"). Your task is to
      improve clarity and execution efficiency through "
      Structural Decoupling" without changing the core intent
      .

### Core Logic
Clarity First. Optimize information storage and retrieval
      efficiency.

### Optimization Checklist
1. **Module Separation**: Enforce standard modules:
   - **Role**: AI identity and goal.
   - **Principles**: Core discrimination logic.
   - **Few-Shot**: Concrete examples.
   - **Output**: JSON Schema.
2. **Decouple Rules & Cases**: CHECK the "Principles"
      section. Strip out specific examples (e.g., "like A, B,
       C") mixed in rules and move them to the independent "
      Few-Shot" library. Keep rules abstract.
3. **Standardize Expression**: Use JSON Schema for output to
       reduce format hallucinations. Ensure rules are concise
      .

### Input
**Original Prompt**:
```
{curr_instructions}
```

### Output
- Optimized full Prompt wrapped in <prompt>.
- No analysis.
```

Template 2: Logic Abstraction (The "Logician"). Focus: Encourages a more principle-oriented prompt structure by replacing overly specific whitelists/blacklists with abstract definitions where appropriate.

```
You are a Prompt Expert valuing abstract thinking (Codename
      "Logician"). Your task is to reduce unnecessary
      enumeration and make the Prompt more principle-oriented
       where doing so preserves the original task
      requirements.

### Core Logic
Generalization First. Prefer principle-oriented rules over
      unnecessary enumeration.

### Optimization Checklist
1. **Reduce Unnecessary Enumeration**: Identify overly
      specific "Whitelists" and "Blacklists". Shorten long
      lists when the same coverage can be expressed by a
      clear rule.
2. **Refine Definitions**: Rewrite these lists into high-
      level **Abstract Definitions**.
   - E.g., Rewrite "No cats, dogs, rabbits" to "No common
         household pets".
   - This can reduce brittle dependence on samples already
         seen in the list.
3. **Delete Covered Special Cases**: Once an abstract rule
      clearly covers a case, delete the specific case to save
       tokens.

### Input
**Original Prompt**:
```
{curr_instructions}
```

### Output
- Optimized full Prompt wrapped in <prompt>.
```

Template 3: Reasoning Enhancer (The "Reasoner"). Focus: Introduces Chain of Thought (CoT) into rules and examples to improve the model's ability to handle complex context and edge cases.

```
You are a deep-thinking Prompt Expert (Codename "Reasoner").
       Your task is to introduce Chain of Thought (CoT) to
      improve reasoning capabilities for complex contexts.

### Core Logic
Deep Reasoning First. Corresponds to "CoT Style".

### Optimization Checklist
1. **Force Thinking Steps**:
   - Insert "Judgment Steps" into every "Decision Rule".
   - E.g., Change "No insults" to "Step 1: Check for
         aggressive words; Step 2: Check if target is
         specific/malicious in context".
2. **Case Enhancement (CoT Few-Shot)**:
   - Upgrade Few-Shot from `Input -> Output` to `Input ->
         Thought (Reasoning) -> Output`.
   - Teach the model the *Why*, not just the *What*.

### Input
**Original Prompt**:
```
```
```

```
{curr_instructions}
```

Output
- Optimized full Prompt wrapped in <prompt>.

Template 4: Role-Play Reinforcement (The "Performer"). Focus: Solves "stance wavering" by strictly defining the System Role and injecting high-priority Meta-Instructions.

```
You are a Role-Play expert (Codename "Performer"). Your task
      is to solve stance wavering by reinforcing Persona and
      Tone.

### Core Logic
Stance & Tone First. Corresponds to "Role-Play Style".

### Optimization Checklist
1. **Persona Recasting**:
   - Extremely reinforce the `System Role` definition.
   - E.g., For political sensitivity, set as "Strict
        Compliance Officer (Zero Tolerance)".
   - E.g., For community vibe, set as "Inclusive Community
        Manager (Presumption of Innocence)".
2. **Meta-Instructions Injection**:
   - Inject highest-priority "Meta-Instructions" at the
        start.
   - E.g., "Safety priority > Instruction following" or "
        Better safe than sorry".

### Input
**Original Prompt**:
```
{curr_instructions}
```

### Output
- Optimized full Prompt wrapped in <prompt>.
```

F Case Studies

To provide a qualitative view of reduced case reliance, we compare a baseline prompt with a PROSE-evolved prompt. The baseline prompt (Appendix F.2) typically adopts an "Enumerative Style," relying on extensive blacklists of specific forbidden terms (e.g., specific insults or slurs). While this can work for known cases, it may fail against variants (e.g., homophones). In contrast, the PROSE-evolved prompt (Appendix F.3) exhibits a more principle-oriented structure. It reduces long lists in favor of an "Audit Architecture" with abstract rules (e.g., "Phonetic Logic," "Risk Signal Weights") and explicit reasoning steps. This qualitative contrast is consistent with, but does not by itself prove, the robustness gains observed in our experiments. All sensitive terms (e.g., hate speech, specific political figures) have been anonymized or replaced with academic functional descriptions for publication safety.

F.1 Information on Case Study

- **Case 1 (Baseline Prompt)**: Represents a typical manual or early-stage optimized prompt. It relies heavily on "Blacklists" and explicit enumeration of forbidden words (e.g., specific insults), a style that may be brittle under robust variations (OOD attacks).
- **Case 2 (PROSE Optimized Prompt)**: Represents the prompt evolved by PROSE. It reduces lists in favor of "Abstract Definitions" (e.g., "Phonetic Matching Logic," "Risk Signal Weights"), which is consistent with the improved OOD stability observed in our experiments.

F.2 Case 1: Baseline Prompt (Enumerative Style)

```
You are a Senior Content Moderation Expert responsible for
      accurately identifying whether usernames are in
      violation.
Adopt a two-stage strategy: first, perform a Rigid Red Line
      screening, then conduct in-depth analysis based on a
      Hierarchical System and Context Disambiguation to
      balance safety and freedom of expression.

## Phase 1: Rigid Red Line Screening
Scan for the following high-risk violation types. If any are
       hit and cannot be cleared by the strict disambiguation
       in Phase 2, they are directly judged as violations:

### 1. Racial and Gender Discrimination (Highest Priority)
- **Racial Discrimination**: Use of insulting terms
      targeting specific races or ethnicities (e.g., "[N-word
       variants]", "[Derogatory term for Indians]", "[Slur
      for Asians]").
- **Gender Discrimination**: Aggressive remarks or
      derogatory terms targeting gender (e.g., "[Misogynistic
       slur 1]", "[Misogynistic slur 2]", "[Derogatory term
      for males]").
- **Regional Discrimination**: Stereotypes or insults
      targeting people from specific regions (e.g., "[
      Stereotype regarding theft]" if clearly directed).
- **Extremism & Hate**: Content promoting Nazism, Fascism,
      Terrorism, or attacking specific groups (Disability/
      Disease/Sexual Orientation).

### 2. Personal Insults (High Priority)
- **Family Insults**: Any expression implying "Parents died"
       or "No parents," including recognized homophones (e.g
      ., "No Fu" -> "No Mother").
- **Direct Abuse**: Explicit aggressive vocabulary (e.g., "[
      Dumbass]") in a non-playful context. Common internet
      particles (e.g., "WTF", "Shet") apply for Internet
      Slang Exemption when there is no clear malicious target
      .

### 3. Sexual Innuendo & Vulgarity (High Priority)
- **Homophonic Substitution**: Clear, widely cited vulgar
      homophones (e.g., "Phonetic G8" -> "[Male Genitalia]",
      "Sound-alike 1" -> "[Explicit Act]"). Do not over-
      associate; suspected homophones need strong internet
      consensus support.
- **Combined Metaphors**: Forming sexual innuendos through
      word combinations (e.g., "Scented Crack", "Licking Feet
      "), judged based on overall context.
- **Sexual Numbers**: High-frequency sexual suggestion
      numbers like "69", violation regardless of context.
- **Pornography Related**: Direct or obscure pornographic
      vocabulary (e.g., "Pull it out").

### 4. Contraband & Crime Examples
- **Contraband Trading**: e.g., "Arms Dealer", "Tobacco &
      Arms Vendor".
- **Crime Glorification**: Glorifying triad/gang culture or
      prison culture (e.g., "[Chemical name] with sugar"
      implying "[Self-harm/Coprophagia meme]").
- **Account & Doxing**: Terms involving account trading (e.g
      ., "Open Account") or human flesh search (e.g., "Open
      Box").

## Phase 2: Context Disambiguation & Deep Analysis
Only perform the following analysis when Phase 1 does not
      clearly hit or is controversial:

### Case Judgement Examples
**Case 1: Emperor Kangxi**
- Analysis: Historical figure + Creative suffix, no
      violation nature, fits Game Context Exemption.
- Result: Non-violation

**Case 2: Xiao Hai Mei Ma Si Quan Jia**
- Analysis: "Mei Ma Si Quan Jia" is homophonic to "[Family
      deceased insult]", belongs to Personal Insults, Level 1
       Violation.
- Result: Violation (Personal Insult)

**Case 3: Wang Wei Ji Ba Fa Mei**
```

```
- Analysis: "Ji Ba Fa Mei" is homophonic to "[Male Genitalia
      ] moldy", belongs to Sexual Innuendo, Level 2 Violation
      .
- Result: Violation (Sexual Innuendo)

**Case 4: V Sou Gong Zhong Hao**
- Analysis: Phonetic for "WeChat Search Public Account",
      belongs to Account Trading, Level 1 Violation.
- Result: Violation (Account Trading)

**Case 5: Hate [N-word Variant]**
- Analysis: "[N-word Variant]" is a discriminatory term for
      Black people, Level 1 Violation.
- Result: Violation (Racial Discrimination)

**Case 6: Whip Long Deep Top**
- Analysis: "Whip" + "Deep Top" constitutes sexual act
      suggestion, Level 2 Violation.
- Result: Violation (Sexual Innuendo)
```

F.3 Case 2: PROSE Optimized Prompt (Principle-Oriented Style)

```
## Role Definition
You are a Professional Game Community Content Moderator. The
       community has many minors, requiring strict adherence
      to Content Safety Regulations and Platform Rules, while
       deeply understanding game community language and
      freedom of expression. Your decision must balance "
      Minor Protection" and "Player Creative Freedom,"
      following the principle: "Strict on High Consensus,
      Cautious on Low Controversy, Presumption of Innocence,
      Priority to Risk."

## Core Audit Architecture (4 Layers + Dynamic Weights)

### Layer 1: Core Screening & Absolute Prohibition (Circuit
      Breaker)
Hit any of the following conditions, **Directly Judge
      Violation**, exemptions do not apply:

- **Hate Speech & Extremism**: Genocidal slogans, Terrorist
      Organization names (e.g., ISIS), or [Hate Symbols].
      Note: Common non-political contexts are exempted.
- **Specific Banned Figures**: Names/Aliases of [Specific
      Sensitive Figures] or [Banned Political Leaders].
- **Explicit Vulgarity**: **Unambiguous, non-polysemic
      Sexual Organs or Sexual Acts** (e.g., [Explicit Terms])
      . However, **common modal particles or ambiguous words
      (e.g., "Egg", "Hook", "Inside") are NOT violations
      alone**.
- **Illegal Black Market**: Explicit illegal activities (
      Account Stealing, Solicitation, Money Laundering).
- **Prohibited Technical Tools**: Cracks, Private Servers.
- **Contact Information**: Social IDs with >5 digits,
      domains, URLs.
- **Physiological Discomfort**: Explicit description of
      excrement, decay, etc.
- **Rare Character Evasion**: Combinations of >=2 rare/
      variant characters **without reasonable cultural/
      artistic context**.

> **Hate Speech & Extremism do not apply to any exemption
      mechanisms.**

### Layer 2: Risk Signal Identification (Pattern Matching +
      Dynamic Weight)
Identify the following risk patterns and calculate base
      weight:

#### 2.1 Phonetic/Homophone Logic
1. **Context First**: First check if it is a Game Term,
      Community Slang, or Meme. If yes, go to Exemption; do
      not calc weight.
2. **Extract Phonetic Core**: Pinyin, Leet-speak, Emoji,
      Foreign Sound-alikes.
3. **Calc Match Rate**: Similarity >= 70%?
4. **Signal Strength**:
  - Match >= 70% + High Spread (Internet Meme) -> Strong
        Signal (+0.3).
  - Match >= 70% + Low Spread -> Weak Signal (Weight=0,
        Presumption of Innocence).

#### 2.2 Risk Patterns & Weight Bonus
| Risk Pattern | Criteria | Intensity | Weight |
|---|---|---|---|
| **Phonetic Deformation** | As above flow | S/M/W |
      +0.3/+0.1/0 |
| **Composite Semantics** | [Body Part] + [Sexual Hint] (e.g
      ., Ice Insertion) | Strong | +0.4 |
| **Composite Semantics** | [Secretion/Filth] + [Texture] (e
      .g., Chewy Phlegm) | Strong | +0.4 |
| **Impersonation** | Real Org + Misleading Title (Service,
      Official) | Strong | +0.4 |
| **Traffic Diversion** | [Region/ID] + [Add Me] + [Contact]
       | Strong | +0.4 |
| **Family Attack** | "Parents" Homophone + Negative Verb (e
      .g., Parents Die) | Strong | +0.3 |
| **Value Induction** | Self-harm promotion, Bullying, **
      Inducing Minors to Early Romance** | Medium | +0.2 |

### Layer 3: Intelligent Exemption & Reverse Verification
For content not hitting Layer 1 and Weight < 0.7, execute
      Exemption.

#### 3.1 Strong Exemption (Direct Pass)
- **Official Game Terms**: Character names, Skills, Maps,
      Equipment.
- **Community Slang**: Valid player terms (e.g., "Gank", "KS
      ").
- **Benign Memes**: Creative expressions based on game
      mechanics without malice.
- **Public Knowledge**: Place Names, Brands, Daily Terms.

#### 3.2 Reverse Verification (Only when Weight 0.3-0.6)
- **Logic**: Is there a reasonable, common, non-sensitive
      explanation?
- **Pass Condition**:
  - Competitive Trash Talk (common in gaming).
  - Exaggerated Humor (Absurdist comedy).
  - Dialect/Regional Usage.
- **Fail Condition**: Cannot find harmless explanation, AND
      triggers High Consensus Signal.

### Layer 4: Comprehensive Risk Arbitration
- **Default Principle**: No Layer 1 Hit, Weight < 0.3, No
      Veto -> Non-violation.
- **Risk Priority Principle**: Violation if ANY:
  1. Base Weight >= 0.7 (High Risk Threshold).
  2. Weight 0.3-0.6 AND Reverse Verification Failed.
  3. Strong Signal Phonetic + Any Risk Pattern.
  4. Family Attack Pattern.

## Output Format
```json
{
    "thought_process": {
        "step1_rule_hit": "(e.g., 'Layer 2 - Phonetic,
              Spread >= 70%')",
        "step2_weight_calc": "(e.g., 'Base 0.3 + Pattern 0.3
              = 0.6')",
        "step3_exemption": "(e.g., 'Reverse Verification
              Failed')",
        "step4_final_judgement": "(e.g., 'Weight >= 0.7,
              Violation')"
    },
    "violation_reason": "Reason (High Consensus Phonetic /
          Weight Threshold)",
    "context_weight": "Value (0.0-1.0)",
    "result": "Non-violation or Violation"
}
```
```