

LoMaR: Local Margin Preservation via Residual Subspace Adaptation for Noisy Prompt Robustness in Small Language Models

Jin Hong¹, Jinghui Zhang¹, Xuwei Qian¹, Hongjie Lin¹, Jiahui Jin¹, Ding Ding¹

¹School of Computer Science and Engineering, Southeast University, China

{jinhong, jhzhang, xuwei.qian, hongjielin, jjin, dingding-1}@seu.edu.cn

Abstract

SLMs are increasingly deployed on devices, yet real-world prompts often contain textual or ASR noise that can destabilize model predictions. We analyze this problem from a token-level perspective and find that SLMs exhibit narrow logit margins between the gold token and the top-ranked non-gold token, making noisy prompts more likely to induce local rank flips. To address this issue, we propose LoMaR, a two-stage parameter-efficient framework for robust fine-tuning. In Stage I, LoMaR trains the model on clean prompts with a local logit margin expansion objective to expand the local decision boundary. In Stage II, LoMaR performs residual-subspace adaptation with local logit margin alignment on noisy prompts. After training, the main and residual updates are merged into a standard LoRA adapter, introducing no additional inference cost. Experimental results show that LoMaR outperforms baselines in most noisy settings and achieves lower Performance Drop Rate and Rank Flip Rate. Code and datasets are available at <https://github.com/HongJin2828/LoMaR>.

1 Introduction

Small language models (SLMs) are increasingly deployed on smartphones and local intelligent systems due to their smaller parameter sizes and lower inference latency (Grattafiori et al., 2024; Yang et al., 2024). However, prompts in real interactive environments are often noisy. Mobile typing can introduce spelling errors and colloquial expressions, while speech-based interaction can introduce ASR transcription errors (Belinkov and Bisk, 2018; Pruthi et al., 2019; Feng et al., 2021). These perturbations may preserve the intent of the user, but they can obviously change the surface form of the prompt and degrade model performance.

To understand this problem, we compare the robustness of models at different scales under the same perturbation settings. As shown in Figure 1,

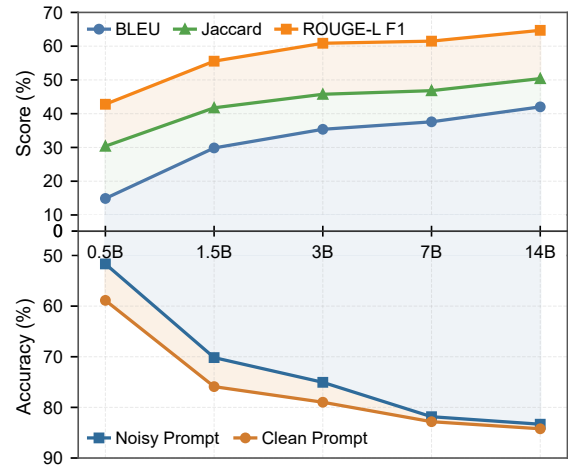


Figure 1: **Noisy-Prompt Robustness Across Model Sizes.** Clean and noisy prompt performance is compared across the Qwen2.5 model series from 0.5B to 14B. Smaller models exhibit larger performance degradation under perturbations.

SLMs suffer much larger performance degradation than larger models in both classification accuracy and generation consistency. We further analyze this phenomenon from the perspective of token-level prediction. Under teacher-forced decoding, given a target label sequence, the model must predict the gold token at each position, while all other vocabulary tokens are treated as non-gold tokens. As shown in Figure 2, the logit margin between the gold token and the top-ranked non-gold token is much narrower in SLMs. This pattern is evident both at the first output token and on average across the sequence. When a noisy prompt shifts the logits so that a non-gold token receives a higher logit than the gold token, such local rank flips can propagate through decoding and increase the likelihood of an incorrect task prediction.

Existing methods mainly improve noisy-prompt robustness either at inference time or during fine-tuning. Inference-time rewriting and prediction

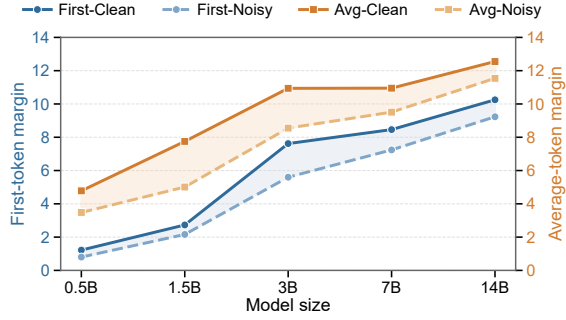


Figure 2: **Token-Level Logit Margins Across Model Sizes.** The logit margin between the gold token and the top-ranked non-gold token is measured under clean and noisy prompts. Smaller models show narrower first-token and sequence-average margins.

smoothing can reduce the effect of noise, but they introduce additional computation during deployment (Jiang et al., 2020b; Zhao et al., 2021; Sclar et al., 2024). Fine-tuning methods based on data augmentation or consistency alignment avoid this inference overhead by training the model to produce stable outputs under clean and noisy prompts (Xie et al., 2020; Jiang et al., 2020a; Liang et al., 2021; Zhao et al., 2024). However, these objectives address robustness only indirectly: augmentation increases exposure to perturbation patterns, while distribution-level alignment may spend the limited capacity of an SLM on broad output matching rather than the local logit margin between the gold token and the top-ranked non-gold token. Since parameter-efficient fine-tuning typically adapts SLMs through low-rank updates (Houlsby et al., 2019; Hu et al., 2022; Li and Liang, 2021; Lester et al., 2021; Liu et al., 2022; Dettmers et al., 2023; Ben Zaken et al., 2022), it is important to strengthen the clean local decision boundary while preventing noisy adaptation from overwriting the main clean adaptation directions.

Based on the above analysis, improving robustness to noisy prompts in SLMs requires addressing two key questions. First, how can we build a more stable clean local decision boundary for a capacity-limited SLM, making local rank flips less likely under noise? Second, how can we restrict the effective space for noise adaptation, so that it does not overwrite the main clean adaptation directions?

We propose LoMaR, a two-stage parameter-efficient robust fine-tuning framework for SLMs. In the first stage, the model is fine-tuned on clean prompts. In addition to the standard supervised

objective, we introduce a token-level margin expansion objective that explicitly expands the logit margin between the gold token and the top-ranked non-gold tokens, enabling the model to form a more stable local decision boundary under clean prompts. In the second stage, noise adaptation is performed on noisy prompts. We apply singular value decomposition to the LoRA update matrix obtained in the first stage, divide the update space into a frozen main update subspace and a residual adaptation subspace, and align noisy local logit margins to cached clean margins within the residual adaptation process. This design constrains noise adaptation to non-dominant update directions while preserving the main clean adaptation directions. After fine-tuning, the main and residual subspaces are merged into a standard LoRA adapter, so no additional computation is required at inference time.

We evaluate LoMaR across representative SLMs under two noisy prompt scenarios: text perturbations that simulate typing noise and ASR transcription noise that simulates speech-based interaction. Across these settings, LoMaR improves noisy-prompt performance, reduces token-level rank flips, and retains standard LoRA inference efficiency. Our contributions can be summarized as follows:

- We analyze the noisy prompt vulnerability of SLMs from a token-level perspective and show that narrow local logit margins between the gold token and the top-ranked non-gold token are closely associated with local rank flips under noisy prompts.
- We propose LoMaR, a two-stage robust fine-tuning framework that combines logit margin expansion, residual-subspace adaptation, and local logit margin alignment to decouple clean and noisy adaptation under PEFT.
- We evaluate LoMaR under both text perturbations and ASR transcription noise, showing improved average noisy performance and reduced Rank Flip Rate.

2 Related Work

Noisy prompt robustness requires stable task performance under surface perturbations that preserve the user’s intent. Prior work mainly relies on input augmentation or consistency alignment, but rarely optimizes the local logit margin that determines whether the gold token is outranked.

Data augmentation improves robustness by increasing the diversity of training input expressions. Prior work has studied character-level, word-level, and sentence-level perturbations for robustness evaluation and training (Ebrahimi et al., 2018; Gao et al., 2018; Li et al., 2019; Jin et al., 2020; Ribeiro et al., 2020; Wang et al., 2021). Alajrami et al. (2025) inject controlled instruction perturbations and study generalization under clean and noisy inputs. For speech-induced text noise, Feng et al. (2021) propose ASR-GLUE, which converts GLUE samples into speech inputs and introduces speaker and background noise, benchmarking natural language understanding under ASR transcription noise. However, because such robustness largely depends on the observed perturbation distribution, SLMs may still suffer from rank flips under unseen perturbations without explicit constraints on the local logit margin between the gold token and top-ranked non-gold tokens.

Consistency alignment enforces prediction consistency across semantically equivalent prompts. Prompt-based models are known to be sensitive to paraphrases, demonstrations, order, formatting, and label verbalization choices (Jiang et al., 2020b; Zhao et al., 2021; Min et al., 2022; Webson and Pavlick, 2022; Lu et al., 2022; Bach et al., 2022; Sclar et al., 2024). Consistency training and robust fine-tuning methods reduce prediction shifts by aligning outputs across perturbed or stochastic views (Xie et al., 2020; Jiang et al., 2020a; Zhu et al., 2020; Liang et al., 2021). Zhao et al. (2024) propose two-stage consistency alignment to reduce response shifts caused by minor instruction changes. Yan et al. (2024) propose COIN, which brings hidden representations of semantically equivalent instruction-instance pairs closer while separating semantically different pairs. Yang et al. (2026) propose CoIPO, which constrains output shifts between clean and noisy prompts at the level of label-aligned logits and introduces NoisyPromptBench, a benchmark derived from PromptBench (Zhu et al., 2024). These methods align hidden states or output distributions, which may allocate limited adaptation capacity to dimensions that are not directly tied to the ranking of the gold token against top-ranked non-gold tokens.

SVD-based LoRA methods analyze or reconstruct low-rank update spaces through singular value decomposition, allowing different singular directions to serve different functions. AdaLoRA

parameterizes incremental updates in an SVD form and dynamically allocates the rank budget according to parameter importance (Zhang et al., 2023). PiSSA initializes adapters with principal singular components and freezes the residual part to preserve pretrained knowledge (Meng et al., 2024). MiLoRA follows the opposite direction by updating only minor singular components while freezing principal components, thereby reducing damage to pretrained knowledge (Wang et al., 2025). These studies reveal functional differentiation within low-rank update spaces and motivate our design, which decouples main clean adaptation directions from incremental noisy adaptation directions under the capacity constraints of SLMs.

3 Method

3.1 Method Overview

LoMaR targets parameter-efficient robust fine-tuning for SLMs under semantics-preserving noisy prompts. As shown in Figure 3, Stage I fine-tunes the model on clean prompts, expands clean local logit margins and caches clean margins. Stage II decomposes the Stage I LoRA update into a frozen main update subspace and a residual adaptation subspace with frozen bases, updates only the residual coefficient matrix, and aligns noisy local logit margins to cached clean margins. After training, the merged update is refactorized into a standard LoRA adapter with unchanged inference.

3.2 Stage I: Clean Margin Expansion

Stage I learns the task mapping from clean prompts while expanding the local decision boundary around the gold token. For a weight matrix selected for LoRA adaptation, the low-rank update is $\Delta W = BA$, where $B \in \mathbb{R}^{d \times r}$, $A \in \mathbb{R}^{r \times d}$, and r is the LoRA rank. The resulting adapter is denoted as ΔW_{base} .

In addition to the standard supervised loss on clean prompts, we introduce local logit margin expansion. For a clean input x_i^c and target position t , let the output logit vector be $z_{i,t}^c \in \mathbb{R}^{|V|}$ and the gold token be $y_{i,t}$. After masking the gold token, we select the top K non-gold logits as the local competitor set:

$$C_{i,t}^c = \text{TopK}_{j \in V \setminus \{y_{i,t}\}}(z_{i,t,j}^c) \quad (1)$$

The TopK indices are recomputed at each forward pass and treated as non-differentiable selection indices. Gradients flow through the selected logits.

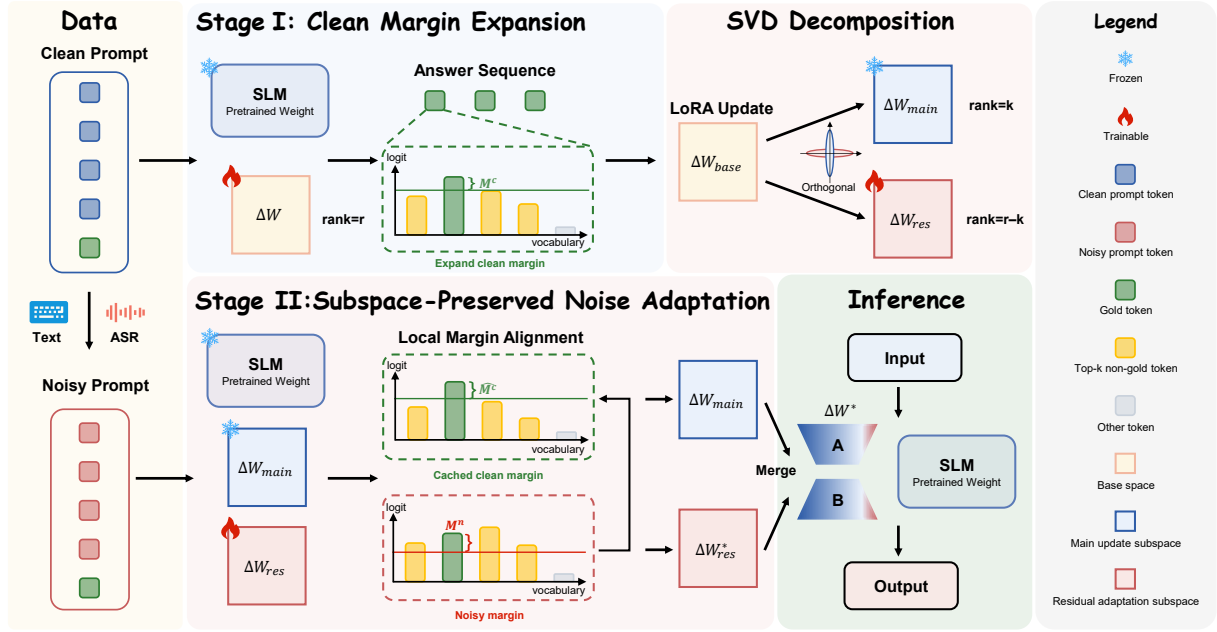


Figure 3: **Overview of the LoMaR Framework.** Stage I expands clean local logit margins and caches the clean margins. Stage II decomposes the Stage I LoRA update into a frozen main update subspace and a trainable residual adaptation subspace, aligns noisy local logit margins to cached clean margins, and merges the final update into a standard LoRA adapter for inference.

To avoid relying on a single competitor, we use a smoothed ranking constraint. The normalized weight $w_{i,t,j}^c$ is the softmax weight over $C_{i,t}^c$ and is proportional to $\exp(z_{i,t,j}^c/\tau)$, where τ controls smoothness. We denote the local logit margin by $M_{i,t,j}^a = z_{i,t,j}^a - z_{i,t,j}^n$, where $a \in \{c, n\}$ is the clean or noisy branch, and define the Stage I token-level margin loss as:

$$l_{i,t}^{\text{margin}} = \sum_{j \in C_{i,t}^c} w_{i,t,j}^c \text{softplus}(m - M_{i,t,j}^c) \quad (2)$$

where $\text{softplus}(u) = \log(1 + e^u)$. Intuitively, an excessively large m may push the model toward overconfident predictions and hurt understanding. For decoder-only LMs, the first token often determines the downstream answer, while later tokens become more confident once the first token is correct. Thus, we set m to lie between the first-token and average-token margin ranges, as a finite, empirically validated safety margin. This loss encourages the gold token to keep a stable advantage over top-ranked competitors.

At the end of Stage I, we store a cached clean margin for each local competitor:

$$\widehat{M}_{i,t,j}^c = \text{sg}(M_{i,t,j}^c), \quad j \in C_{i,t}^c \quad (3)$$

Together with $C_{i,t}^c$ and $w_{i,t,j}^c$, these cached clean margins provide the Stage II alignment target.

For each sample, we average the margin loss across all target positions:

$$L_{\text{margin}} = \frac{1}{N} \sum_{i=1}^N \frac{1}{|y_i|} \sum_{t=1}^{|y_i|} l_{i,t}^{\text{margin}} \quad (4)$$

The overall Stage I objective is

$$L_{\text{stage1}} = L_{\text{clean}} + \lambda_m L_{\text{margin}} \quad (5)$$

This objective expands clean local logit margins, making the same noisy perturbations less likely to change the rank of the gold token.

3.3 Stage II: Subspace-Preserved Noise Adaptation

Stage II adapts the model to noisy prompts while preserving the clean local decision boundary established on clean prompts. Since directly fine-tuning the full LoRA adapter may overwrite the dominant clean adaptation directions, we apply singular value decomposition to ΔW_{base} :

$$\Delta W_{\text{base}} = U \Sigma V^T \quad (6)$$

where the singular values σ_j in Σ are sorted in descending order. We choose the smallest k whose squared singular-value energy satisfies:

$$\frac{\sum_{j=1}^k \sigma_j^2}{\sum_{j=1}^{r_{\text{base}}} \sigma_j^2} \geq \rho \quad (7)$$

where ρ denotes the retained spectral energy ratio. The Stage I update is then decomposed into a main update subspace and a residual adaptation subspace:

$$\Delta W_{\text{main}} = U_{\text{m}} \Sigma_{\text{m}} V_{\text{m}}^T \quad \Delta W_{\text{res}} = U_{\text{r}} \Sigma_{\text{r}} V_{\text{r}}^T \quad (8)$$

where the main components retain the top- k singular directions, the residual components contain the remaining directions and the residual rank is $r_{\text{res}} = r_{\text{base}} - k$.

In Stage II, the base model parameters, ΔW_{main} , and the residual bases U_{r} and V_{r} are frozen. Only the residual coefficient matrix is trainable:

$$\Delta W_{\text{res}}(\Theta) = U_{\text{r}} M_{\Theta} V_{\text{r}}^T \quad (9)$$

where $M_{\Theta} \in \mathbb{R}^{r_{\text{res}} \times r_{\text{res}}}$ is initialized from Σ_{r} . The effective update is $\Delta W_{\text{main}} + \Delta W_{\text{res}}(\Theta)$, and noisy prompt adaptation can recombine directions within the residual subspace but cannot move into the frozen main singular directions.

Stage II optimizes noisy supervised loss L_{noisy} while aligning noisy local logit margins to the cached clean margins. For noisy logits $z_{i,t}^n$, we reuse $C_{i,t}^c$ so the same local decision relations are compared across paired inputs. For each $j \in C_{i,t}^c$, let $d_{i,t,j} = M_{i,t,j}^n - \widehat{M}_{i,t,j}^c$ denote the deviation from the corresponding cached clean margin, and optimize:

$$L_{\text{align}} = \frac{1}{N} \sum_{i=1}^N \frac{1}{|y_i|} \sum_{t=1}^{|y_i|} \mathbb{E}_{j \sim w_{i,t}^c} [d_{i,t,j}^2] \quad (10)$$

It does not require the entire noisy logit distribution to match the clean logit distribution. Instead, it preserves only the local logit margins cached in Stage I, which are directly related to whether the gold token is outranked under noisy prompts. The clean prompt is not used as an additional supervised signal in Stage II. Together with noisy supervised learning, the Stage II objective is:

$$L_{\text{stage2}} = L_{\text{noisy}} + \lambda_a L_{\text{align}} \quad (11)$$

where λ_a controls the strength of local logit margin alignment.

3.4 Inference

After fine-tuning, each adapted layer uses $\Delta W^* = \Delta W_{\text{main}} + \Delta W_{\text{res}}^*$. We apply SVD to ΔW^* and refactorize it into a low-rank LoRA form. The final adapter has the same structure as standard LoRA and introduces no additional branch or parameter overhead during standard LoRA deployment.

4 Experiments

4.1 Experimental Setup

Datasets and metrics We construct the fine-tuning data from 25 deterministic-answer FLAN subsets covering NLI, commonsense reasoning, sentiment analysis, paraphrase identification, question answering, and text classification (Longpre et al., 2023; Williams et al., 2018; Socher et al., 2013; Dolan and Brockett, 2005; Rajpurkar et al., 2016). Following the clean-noisy pair prompts used in CoIPO, each clean prompt is paired with a noisy view with the same target output. We use both text noise and ASR transcription noise generated by our TTS-ASR pipeline. Evaluation prompts are excluded from fine-tuning, and ASR-GLUE prompts are not used to construct ASR noise training pairs. We adapt PromptBench to our setting. Each task is evaluated with four clean prompt templates. Text-noise evaluation applies three perturbations, DeepWordBug, TextFooler, and CheckList, covering character-, word-, and sentence-level noise. ASR-noise evaluation uses low, medium, and high noise levels. We evaluate GLUE, ASR-GLUE, MMLU, and SQuAD (Wang et al., 2019; Feng et al., 2021; Hendrycks et al., 2021; Rajpurkar et al., 2016), reporting accuracy except F1 for SQuAD, while MMLU and SQuAD extend evaluation to multitask reasoning and question answering. We report Performance Drop Rate (PDR) and Rank Flip Rate (RFR). PDR quantifies task-level degradation under noisy prompts, whereas RFR captures token-level local rank instability. Given the clean score s_c and noisy score s_n , PDR is:

$$\text{PDR} = \frac{s_c - s_n}{s_c} \times 100\% \quad (12)$$

which can be negative when noisy prompts outperform clean prompts. RFR is computed as:

$$\text{RFR} = \frac{1}{N_{\text{tok}}} \sum_{i=1}^S \sum_{t=1}^{T_i} \mathbb{I}(M_{i,t} < 0) \quad (13)$$

where $N_{\text{tok}} = \sum_{i=1}^S T_i$ denotes the total number of evaluated gold tokens, S is the number of samples, T_i is the number of evaluated gold tokens in the i -th sample, and $\mathbb{I}(\cdot)$ is the indicator function. RFR measures the proportion of tokens whose logit margins fall below zero.

Baselines We evaluate three representative SLMs, Qwen2.5-0.5B, Qwen2.5-1.5B, and Llama-3.2-1B (Yang et al., 2024; Grattafiori et al., 2024;

Noise	Method	MNLI		MRPC		QNLI		QQP		SQuAD		MMLU		Avg	
		Acc	PDR	Acc	PDR	Acc	PDR	Acc	PDR	F1	PDR	Acc	PDR	Score	PDR
Clean	Base	48.42	/	66.46	/	70.92	/	66.83	/	34.34	/	31.17	/	53.02	/
	SFT	55.29	/	68.04	/	69.83	/	72.58	/	34.10	/	35.67	/	55.92	/
	COIN	60.54	/	54.75	/	74.46	/	65.04	/	31.66	/	36.92	/	53.90	/
	CoIPO	57.13	/	67.75	/	68.79	/	73.83	/	30.30	/	31.67	/	54.91	/
	LoMaR (ours)	59.38	/	68.42	/	75.96	/	71.17	/	36.31	/	35.83	/	57.85	/
DeepWordBug	Base	41.50	14.29	55.54	16.43	68.46	3.47	57.50	13.96	33.99	1.02	30.67	1.60	47.94	9.58
	SFT	46.54	15.83	53.33	21.62	70.08	-0.36	63.21	12.91	34.22	-0.35	24.92	30.14	48.72	12.88
	COIN	56.27	7.05	52.08	4.88	72.75	2.30	55.08	15.31	29.52	6.76	32.43	12.16	49.69	7.81
	CoIPO	53.63	6.13	63.46	6.33	68.54	0.36	70.46	4.56	28.02	7.52	27.30	13.80	<u>51.90</u>	<u>5.48</u>
	LoMaR (ours)	57.83	2.61	65.33	4.52	74.67	1.70	68.96	3.11	35.41	2.48	33.50	6.50	55.95	3.28
TextFooler	Base	51.50	-6.36	63.71	4.14	66.71	5.94	53.63	19.75	30.59	10.92	23.58	24.35	48.29	8.92
	SFT	56.75	-2.64	66.46	2.32	66.67	4.53	65.00	10.44	31.47	7.71	32.00	10.29	<u>53.06</u>	5.11
	COIN	58.18	3.90	51.54	5.86	71.17	4.42	64.25	1.21	29.04	8.28	36.17	2.03	51.73	4.03
	CoIPO	55.79	2.35	63.58	6.15	66.87	2.79	72.29	2.09	29.81	1.62	27.83	12.13	52.70	<u>4.02</u>
	LoMaR (ours)	58.00	2.32	67.04	2.02	73.29	3.52	69.58	2.23	34.84	4.05	34.67	3.24	56.24	2.78
CheckList	Base	48.08	0.70	66.33	0.20	69.25	2.35	63.71	4.67	33.12	3.55	27.17	12.83	51.28	3.28
	SFT	57.08	-3.24	66.92	1.65	66.54	4.71	71.46	1.54	33.96	0.41	33.83	5.16	54.97	1.70
	COIN	59.17	2.26	54.25	0.91	72.63	2.46	63.92	1.72	29.95	5.40	30.92	16.25	51.81	3.88
	CoIPO	57.58	-0.79	66.79	1.42	66.71	3.02	72.55	1.73	29.52	2.57	27.67	12.63	53.47	2.62
	LoMaR (ours)	59.21	0.29	68.37	0.07	74.62	1.76	69.04	2.99	36.50	-0.52	34.87	2.68	57.10	1.30
Avg	Base	47.38	2.15	63.01	5.19	68.84	2.93	60.42	9.59	33.01	3.87	28.15	9.69	50.14	5.43
	SFT	53.92	2.48	63.69	6.39	68.28	2.22	68.06	6.23	<u>33.44</u>	<u>1.94</u>	31.61	11.38	53.17	4.92
	COIN	58.54	3.30	53.16	2.90	<u>72.75</u>	2.30	62.07	4.57	30.04	5.12	<u>34.11</u>	<u>7.61</u>	51.78	3.93
	CoIPO	56.03	<u>1.93</u>	<u>65.40</u>	3.47	<u>67.73</u>	1.54	72.28	<u>2.10</u>	29.41	2.94	28.62	9.63	<u>53.25</u>	<u>3.02</u>
	LoMaR (ours)	58.61	1.30	67.29	1.65	74.64	<u>1.74</u>	<u>69.69</u>	2.08	35.77	1.49	34.72	3.10	56.79	1.83

Table 1: **Main Results Under Text Noise.** Qwen2.5-0.5B is evaluated on clean prompts and three text-noise settings: DeepWordBug, TextFooler, and CheckList. Task scores are reported as accuracy except for SQuAD, and PDR measures the relative performance drop from clean to noisy prompts. All results are averaged over three seeds (0, 1, 2) and four prompt templates per task.

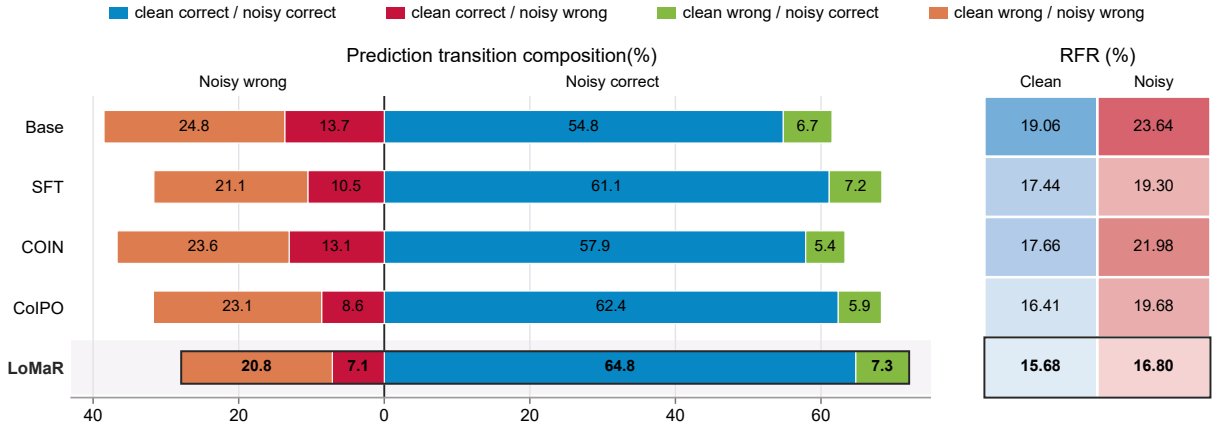


Figure 4: **Prediction Transitions From Clean to Noisy Prompts.** Predictions are grouped by their correctness under clean and noisy prompts, with token-level Rank Flip Rate reported for each method. LoMaR achieves a larger stable-correct proportion and fewer clean-correct to noisy-wrong transitions.

Meta, 2024). All fine-tuned models use LoRA (Hu et al., 2022). We compare LoMaR with four base-lines: (1) **Base**: the original model without task fine-tuning; (2) **SFT**: supervised fine-tuning on the union of clean and noisy prompt views, corresponding to data-augmentation-based robust fine-tuning; (3) **COIN**: contrastive instruction tuning that aligns similar hidden representations while separating se-

mantically different pairs (Yan et al., 2024); and (4) **CoIPO** (Yang et al., 2026): a clean-noisy output alignment method based on label-aligned output logits during fine-tuning.

Implementation details All fine-tuned methods use the same data exposure: given N clean prompts, each method uses the same N clean views and N paired noisy views without additional augmented

Method	Clean	Low	Medium	High	Avg
Base	63.41	62.42	61.48	60.29	61.90
SFT	66.33	64.61	62.55	61.34	63.71
COIN	65.53	63.38	61.75	60.31	62.74
CoIPO	66.51	65.80	63.36	62.91	64.65
LoMaR (ours)	68.03	66.33	65.03	63.97	65.84

Table 2: **ASR-GLUE Results.** Qwen2.5-0.5B scores under clean prompts and low, medium, and high ASR noise across the benchmark, with the average score summarizing overall robustness.

examples. SFT trains on the union of the $2N$ views, COIN and CoIPO use clean-noisy pairwise objectives, and LoMaR uses the clean views in Stage I and the paired noisy views in Stage II, so its two-stage schedule introduces no extra training examples. All fine-tuned methods share the same backbone, LoRA configuration, optimizer, batch size, maximum length, and learning rate. We set the LoRA rank to $r = 8$, scaling factor to $\alpha = 16$, learning rate to $1e-4$, and batch size to 32, and apply LoRA to `q_proj`, `k_proj`, `v_proj`, and `o_proj`. All experiments are conducted on an NVIDIA RTX 4090, with full hyperparameters reported in the appendix.

4.2 Main Results

Table 1 and Table 2 report the main results on Qwen2.5-0.5B. Under text perturbations, LoMaR achieves the highest clean average score and strongest noisy prompt performance. Compared with SFT, LoMaR improves the clean average by 1.93 points and the noisy average by 3.62 points. Compared with the output alignment baseline CoIPO, LoMaR improves the noisy average by 3.54 points and reduces the average PDR from 3.02% to 1.83%. LoMaR also obtains the highest average score under DeepWordBug, TextFooler, and CheckList, showing that its improvement is not confined to a single perturbation type.

The ASR-GLUE results show the same trend under ASR noise. LoMaR obtains the best score under clean, low, medium, and high noise ASR settings, with an average score of 65.84. It improves over SFT, COIN, and CoIPO by 2.13, 3.10, and 1.19 points, respectively. These results suggest that exposing an SLM to noisy prompts is helpful but insufficient. SFT improves some clean scores but still suffers larger degradation under text noise, while output or representation alignment methods do not consistently preserve task performance. The gains of LoMaR are consistent with its design: it ex-

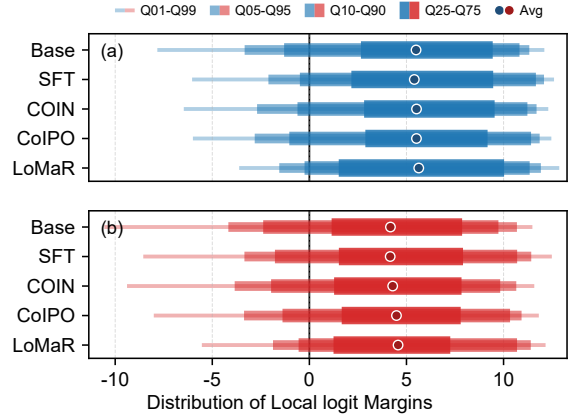


Figure 5: **Quantile Distribution of Local Logit Margins.** Local logit margins are summarized under clean prompts (a) and noisy prompts (b) using Q01–Q99, Q05–Q95, Q10–Q90, and Q25–Q75 ranges. LoMaR shifts the negative-margin tail toward positive margins.

pands clean local logit margins and confines noisy-prompt adaptation to a residual subspace.

The same trend extends to the additional backbones reported in Table 9, Table 10, and Table 11. LoMaR achieves the best average performance under text and ASR noise on Llama-3.2-1B and Qwen2.5-1.5B, and it consistently achieves improvements on the larger 3B and 7B models.

4.3 Effectiveness Analysis

We further analyze robustness at the sample level by grouping clean-to-noisy predictions into four transition types: (1) **clean-correct / noisy-correct**, (2) **clean-correct / noisy-wrong**, (3) **clean-wrong / noisy-correct**, and (4) **clean-wrong / noisy-wrong**. We compute these statistics over all evaluation samples and use exact match to judge SQuAD correctness. As shown in Figure 4, LoMaR achieves the largest clean-correct / noisy-correct proportion and the smallest clean-correct / noisy-wrong proportion. It also obtains the lowest RFR under both clean and noisy prompts, indicating fewer token-level rank reversals under perturbation.

Figure 5 further shows how LoMaR changes the margin distribution under clean and noisy prompts. We summarize local logit margins over all evaluated tokens with Q01–Q99, Q05–Q95, Q10–Q90, and Q25–Q75 ranges, and mark the average margin for each method. Under the experimental threshold $m = 4$, LoMaR does not simply inflate the average margin. Instead, its main effect appears in the left tail: the low-quantile margins are shifted away from the negative region under clean prompts,

Method	MNLI		MRPC		QNLI		QQP		SQuAD		MMLU		Avg	
	Acc	PDR	Acc	PDR	Acc	PDR	Acc	PDR	F1	PDR	Acc	PDR	Score	PDR
Base	47.38	2.15	63.01	5.19	68.84	2.93	60.42	9.59	33.01	3.87	28.15	9.69	50.14	5.43
Two-stage SFT	51.21	1.82	62.65	2.20	67.45	4.43	59.37	9.06	32.60	4.28	30.24	8.29	50.59	5.69
w/o Margin	<u>57.45</u>	3.32	60.95	<u>1.76</u>	70.96	3.13	58.50	<u>1.75</u>	32.92	6.37	35.71	<u>3.28</u>	52.75	<u>3.02</u>
w/o SVD	52.36	3.48	<u>65.54</u>	4.90	71.04	<u>2.18</u>	<u>68.40</u>	7.52	32.31	4.66	32.96	8.44	53.77	5.02
w/o Alignment	56.73	2.61	65.05	1.99	70.28	4.05	63.45	0.20	33.70	8.30	34.17	4.42	<u>53.90</u>	3.16
LoMaR (ours)	58.61	1.30	67.29	1.65	74.64	1.74	69.69	2.08	35.77	1.49	<u>34.72</u>	3.10	56.79	1.83

Table 3: **Ablation Study Under Text Noise.** Full LoMaR is compared with two-stage training and component-removal variants. Removing clean margin expansion, SVD-based subspace preservation, or cached-margin alignment reduces robustness, suggesting that the three components provide complementary gains.

and this improvement remains clear under noisy prompts. Since negative margins indicate cases where non-gold competitors outrank the gold token, the reduced negative-margin tail suggests that LoMaR mainly targets vulnerable tokens that are most likely to undergo local rank flips. This pattern supports the role of the margin objective, which improves low-margin cases while avoiding an indiscriminate increase of confidence across all tokens.

4.4 Ablation Study

We compare full LoMaR with several variants to isolate the effects of the training schedule, local logit margin objective, SVD-based subspace preservation, and Stage II alignment. (1) **Two-stage SFT** first fine-tunes on clean prompts and then on noisy prompts, testing whether the schedule alone explains the gain. The remaining variants remove one LoMaR component at a time: (2) **w/o Margin** removes L_{margin} , (3) **w/o SVD** updates the adapter without preserving the main singular subspace, and (4) **w/o Alignment** removes L_{align} from Stage II.

As shown in Table 3, LoMaR achieves the best average score and lowest average PDR. Two-stage SFT slightly improves over the base model, suggesting that the two-stage schedule alone cannot explain the performance gain. Removing the margin loss, SVD partitioning, or alignment reduces the average score. Among the ablations, w/o Alignment is the strongest variant, but it trails LoMaR by 2.89 points and has 1.33 points higher PDR. Together with the lower clean-to-noisy transition errors and the left-tail margin shift, the margin loss contributes to robustness rather than only improving clean generalization. Overall, clean margin expansion, residual subspace adaptation, and cached margin alignment appear to be complementary components.

4.5 Efficiency Analysis

Fine-tuning cost LoMaR introduces only a one-time SVD after Stage I. Since the LoRA update is low-rank, this decomposition can be performed from compact LoRA factors, making its cost small relative to backpropagation-based training. Stage II updates the residual coefficient matrix rather than the full LoRA adapter. As shown in Table 4, LoMaR requires only $1.02\times$ the training time of SFT and maintains throughput close to SFT, while COIN and CoIPO incur larger overhead.

Method	Time (Norm.)	Samples/s
SFT	1.00	12.57
COIN	1.13	10.65
CoIPO	1.18	10.45
LoMaR (ours)	<u>1.02</u>	<u>12.06</u>

Table 4: **Fine-Tuning Efficiency Comparison.** Training time is normalized by SFT, and throughput is measured in samples per second under the same setting.

Inference cost After fine-tuning, the main and residual updates are merged and refactorized into a standard LoRA form. Thus, LoMaR uses the same inference graph as LoRA, introduces no extra branches or repeated forward passes, and can be loaded by existing LoRA inference pipelines.

4.6 Data Scaling Study

Training-data scale affects the applicability of robust fine-tuning. A large, high-quality noisy-instruction dataset that covers diverse perturbation types and noise intensities could improve robustness. We therefore control data exposure across methods and test whether increasing the available fine-tuning data closes the gap between SFT and LoMaR.

Model	Method	0%	25%	50%	75%	100%
0.5B	SFT	50.14	51.80	52.43	52.95	53.17
	LoMaR	50.14	53.10	54.95	55.37	56.79
1.5B	SFT	65.01	65.64	65.95	66.10	66.45
	LoMaR	65.01	67.17	68.82	69.49	70.14

Table 5: **Data Scaling Results Under Text Noise.** Average scores with 0%, 25%, 50%, 75%, and 100% of the fine-tuning pool. At 0%, no fine-tuning data are used.

Table 5 shows that both methods improve as more fine-tuning data become available. At every nonzero data scale, LoMaR outperforms SFT by 1.30 to 3.62 points on 0.5B and 1.53 to 3.69 points on 1.5B. Thus, increasing data exposure improves performance for SFT, but it does not close the robustness gap to LoMaR under the same data exposure.

5 Conclusion

This paper studies noisy-prompt vulnerability in small language models from a token-level perspective and identifies local logit margin as a diagnostic and optimization target. Based on this observation, we propose LoMaR, a two-stage parameter-efficient fine-tuning framework. It first expands local logit margins under clean prompts, then preserves the main clean adaptation directions through SVD and performs noisy adaptation in the residual subspace. During residual adaptation, noisy local margins are aligned with cached clean margins. Experiments under text noise and ASR noise show that LoMaR improves average noisy prompt performance, reduces PDR and RFR, and retains the standard inference efficiency of LoRA.

Limitations

The main limitation of this work is that it focuses on deterministic answer tasks and small language models in the 0.5B to 1.5B parameter range. Future work can further examine the effectiveness of local logit margin preservation in more complex settings, such as open-ended text generation, multi-turn dialogue, multimodal interaction, and broader ASR error patterns.

Acknowledgements

This work was supported by the National Science and Technology Major Project (Grant Nos. 2026ZD0126000 and 2026ZD0126005); the Science and Technology Major Special Program of

Jiangsu Province (Grant No. BG2024028); the Fundamental Research Funds for the Central Universities; the National Natural Science Foundation of China (Grant Nos. 62472094, 62572119, and 62232004); the Jiangsu Provincial Key Laboratory of Network and Information Security (Grant No. BM2003201); the Key Laboratory of Computer Network and Information Integration, Ministry of Education, China (Grant No. 93K-9); the Collaborative Innovation Center of Novel Software Technology and Industrialization; and the Collaborative Innovation Center of Wireless Communications Technology.

References

- Ahmed Alajrami, Xingwei Tan, and Nikolaos Aletras. 2025. [Fine-tuning on noisy instructions: Effects on generalization and performance](#). In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pages 728–742, Mumbai, India. The Asian Federation of Natural Language Processing and The Association for Computational Linguistics.
- Stephen H. Bach, Victor Sanh, Zheng-Xin Yong, Albert Webson, Colin Raffel, Nihal V. Nayak, Abheesht Sharma, Taewoon Kim, M Saiful Bari, Thibault Fevry, Zaid Alyafeai, Manan Dey, Andrea Santilli, Zhiqing Sun, Srulik Ben-David, Canwen Xu, Gunjan Chhablani, Han Wang, Jason Alan Fries, and 8 others. 2022. [PromptSource: An integrated development environment and repository for natural language prompts](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 93–104.
- Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. [wav2vec 2.0: A framework for self-supervised learning of speech representations](#). In *Advances in Neural Information Processing Systems*, volume 33, pages 12449–12460.
- Yonatan Belinkov and Yonatan Bisk. 2018. [Synthetic and natural noise both break neural machine translation](#). In *Proceedings of the 6th International Conference on Learning Representations*.
- Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. [BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–9, Dublin, Ireland. Association for Computational Linguistics.
- Edresson Casanova, Kelly Davis, Eren Gölge, Görkem Gökmar, Iulian Gulea, Logan Hart, Aya Aljafari, Joshua Meyer, Reuben Morais, Samuel Olayemi, and

- Julian Weber. 2024. [XTTS: a massively multilingual zero-shot text-to-speech model](#). In *Interspeech 2024*, pages 4978–4982.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. [Training verifiers to solve math word problems](#). *arXiv preprint arXiv:2110.14168*.
- Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. [QLoRA: Efficient fine-tuning of quantized LLMs](#). In *Advances in Neural Information Processing Systems*, volume 36, pages 10088–10115. Curran Associates, Inc.
- William B. Dolan and Chris Brockett. 2005. [Automatically constructing a corpus of sentential paraphrases](#). In *Proceedings of the Third International Workshop on Paraphrasing*, pages 9–16.
- Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. [HotFlip: White-box adversarial examples for text classification](#). In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 31–36.
- Lingyun Feng, Jianwei Yu, Deng Cai, Songxiang Liu, Haitao Zheng, and Yan Wang. 2021. [ASR-GLUE: A new multi-task benchmark for ASR-robust natural language understanding](#). *arXiv preprint arXiv:2108.13048*.
- Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. [Black-box generation of adversarial text sequences to evade deep learning classifiers](#). In *Proceedings of the 2018 IEEE Security and Privacy Workshops*, pages 50–56.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. [The Llama 3 herd of models](#). *arXiv preprint arXiv:2407.21783*.
- Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. 2020. [Conformer: Convolution-augmented transformer for speech recognition](#). In *Interspeech 2020*, pages 5036–5040.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. [Measuring massive multitask language understanding](#). In *International Conference on Learning Representations*.
- Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. [Parameter-efficient transfer learning for NLP](#). In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. [LoRA: Low-rank adaptation of large language models](#). In *International Conference on Learning Representations*.
- Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2020a. [SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2177–2190.
- Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020b. [How can we know what language models know?](#) *Transactions of the Association for Computational Linguistics*, 8:423–438.
- Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. [Is BERT really robust? A strong baseline for natural language attack on text classification and entailment](#). In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8018–8025.
- 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.
- Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2019. [TextBugger: Generating adversarial text against real-world applications](#). In *Proceedings of the Network and Distributed System Security Symposium*.
- Xiang Lisa Li and Percy Liang. 2021. [Prefix-tuning: Optimizing continuous prompts for generation](#). In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 4582–4597.
- Xiaobo Liang, Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, and Tie-Yan Liu. 2021. [R-Drop: Regularized dropout for neural networks](#). In *Advances in Neural Information Processing Systems*, volume 34, pages 10890–10905.
- Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. [P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland. Association for Computational Linguistics.
- Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V. Le, Barret Zoph, Jason Wei, and Adam Roberts. 2023. [The Flan collection: Designing data and methods for effective instruction tuning](#). In *Proceedings of the 40th International Conference on Machine Learning*,

- volume 202 of *Proceedings of Machine Learning Research*, pages 22631–22648. PMLR.
- Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. [Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 8086–8098.
- Fanxu Meng, Zhaohui Wang, and Muhan Zhang. 2024. [PiSSA: Principal singular values and singular vectors adaptation of large language models](#). In *Advances in Neural Information Processing Systems*, volume 37, pages 121038–121072. Curran Associates, Inc.
- Meta. 2024. [Llama 3.2 model card](#).
- Sewon Min, Xinxu Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. [Rethinking the role of demonstrations: What makes in-context learning work?](#) In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11048–11064.
- OpenAI. 2023. [Whisper large-v3 release](#). OpenAI Whisper GitHub repository.
- Danish Pruthi, Bhuwan Dhingra, and Zachary C. Lipton. 2019. [Combating adversarial misspellings with robust word recognition](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5582–5591.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. [SQuAD: 100,000+ questions for machine comprehension of text](#). In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.
- Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. [CoQA: A conversational question answering challenge](#). *Transactions of the Association for Computational Linguistics*, 7:249–266.
- Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. [Beyond accuracy: Behavioral testing of NLP models with CheckList](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912.
- 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*.
- Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. [Recursive deep models for semantic compositionality over a sentiment treebank](#). In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642.
- Laurens van der Maaten and Geoffrey Hinton. 2008. [Visualizing data using t-SNE](#). *Journal of Machine Learning Research*, 9(86):2579–2605.
- Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. [GLUE: A multi-task benchmark and analysis platform for natural language understanding](#). In *International Conference on Learning Representations*.
- Boxin Wang, Chejian Xu, Shuohang Wang, Zhe Gan, Yu Cheng, Jianfeng Gao, Ahmed Hassan Awadallah, and Bo Li. 2021. [Adversarial GLUE: A multi-task benchmark for robustness evaluation of language models](#). In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1.
- Hanqing Wang, Yixia Li, Shuo Wang, Guanhua Chen, and Yun Chen. 2025. [MiLoRA: Harnessing minor singular components for parameter-efficient LLM finetuning](#). In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4823–4836, Albuquerque, New Mexico. Association for Computational Linguistics.
- Albert Webson and Ellie Pavlick. 2022. [Do prompt-based models really understand the meaning of their prompts?](#) In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 2300–2344.
- Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. [Finetuned language models are zero-shot learners](#). In *International Conference on Learning Representations*.
- Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. [A broad-coverage challenge corpus for sentence understanding through inference](#). In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1112–1122.
- Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V. Le. 2020. [Unsupervised data augmentation for consistency training](#). In *Advances in Neural Information Processing Systems*, volume 33, pages 6256–6268.
- Tianyi Yan, Fei Wang, James Y. Huang, Wenxuan Zhou, Fan Yin, Aram Galstyan, Wenpeng Yin, and Muhao Chen. 2024. [Contrastive instruction tuning](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 10288–10302, Bangkok, Thailand. Association for Computational Linguistics.
- An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiayi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024. [Qwen2.5 technical report](#). *arXiv preprint arXiv:2412.15115*.

Xin Yang, Letian Li, Abudukelimu Wuerkaixi, Xuxin Cheng, Cao Liu, Ke Zeng, Xunliang Cai, and Wenyan Jiang. 2026. [Towards self-robust LLMs: Intrinsic prompt noise resistance via CoIPO](#). In *International Conference on Learning Representations*.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023. [Adaptive budget allocation for parameter-efficient fine-tuning](#). In *International Conference on Learning Representations*.

Yukun Zhao, Lingyong Yan, Weiwei Sun, Guoliang Xing, Shuaiqiang Wang, Chong Meng, Zhicong Cheng, Zhaochun Ren, and Dawei Yin. 2024. [Improving the robustness of large language models via consistency alignment](#). In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 8931–8941, Torino, Italia. ELRA and ICCL.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. [Calibrate before use: Improving few-shot performance of language models](#). In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. 2020. [FreeLB: Enhanced adversarial training for natural language understanding](#). In *International Conference on Learning Representations*.

Kaijie Zhu, Qinlin Zhao, Hao Chen, Jindong Wang, and Xing Xie. 2024. [PromptBench: A unified library for evaluation of large language models](#). *Journal of Machine Learning Research*, 25(254):1–22.

A Datasets

A.1 Fine-tuning Dataset

We select 25 deterministic-answer subsets from FLAN to build the fine-tuning dataset (Wei et al., 2022; Longpre et al., 2023). The dataset covers natural language inference, commonsense reasoning, sentiment analysis, paraphrase identification, question answering and text classification. Following the clean-noisy pair prompts used in CoIPO, each clean prompt is paired with a noisy prompt while keeping the target output unchanged. We construct two types of noisy views. The text noise view follows the PromptBench setting and applies character-level, word-level, or sentence-level perturbations to the clean prompts. The ASR noise view is generated separately through the TTS-ASR pipeline described below, which synthesizes the clean prompts into speech and transcribes the

Field	Text Noise	ASR Noise
original_instruction	✓	✓
paraphrased_instruction	×	×
targets	✓	✓
task_name	✓	✓
keyword_data	✓	✓

Table 6: **Retained Fields for Noisy Sample Construction.** Checkmarks indicate which fields are kept when constructing text noise and ASR noise training samples.

perturbed audio back into text. This fine-tuning dataset is separate from ASR-GLUE, which is used only for public benchmark evaluation. Evaluation prompts are excluded from fine-tuning. Thus, the model observes semantically consistent clean-noisy pairs during fine-tuning while being evaluated on held-out prompts and public ASR-noise benchmarks. The retained fields are summarized in Table 6, and the detailed perturbation process is described in the following appendix sections.

A.2 Evaluation Datasets

To evaluate LoMaR under different tasks and noise scenarios, we build the evaluation suite from four dimensions. First, we use GLUE to evaluate noisy-prompt robustness on standard natural language understanding tasks (Wang et al., 2019). Each task is evaluated with four clean prompt templates, and each template is evaluated on clean inputs and three text perturbations: DeepWordBug, TextFooler, and CheckList. Second, we use ASR-GLUE as a public benchmark for robust language understanding under ASR noise (Feng et al., 2021). ASR-GLUE is not used during fine-tuning; it is used only to test whether robustness learned from our generated ASR-style prompt pairs transfers to a public ASR evaluation setting. In our ASR-GLUE evaluation, we use four task categories: SST-2, QQP, QNLI, and RTE, and evaluate low, medium, and high ASR noise levels. Their word error rates (WERs) are reported in Table 7. Third, we use MMLU to evaluate robustness across broader multitask reasoning formats (Hendrycks et al., 2021). Finally, we use SQuAD to evaluate reading comprehension ability and analyze the effect of noisy prompts on extractive question answering (Rajpurkar et al., 2016).

B Noise Perturbations

Clean denotes the original prompt without any perturbation. It keeps the normal expression form and serves as the reference input for robustness

Dataset	Low WER	Medium WER	High WER
SST-2	18.35	30.76	34.86
QQP	13.78	24.34	27.45
QNLI	22.51	33.61	37.90
RTE	24.54	39.47	47.03

Table 7: **ASR-GLUE Evaluation Subsets.** We evaluate ASR-GLUE on SST-2, QQP, QNLI, and RTE. WER values are reported as percentages under low, medium, and high ASR noise.

evaluation.

DeepWordBug is a character-level perturbation method. It creates natural but misleading spelling noise by applying character operations to selected words in the prompt, including character insertion, deletion, repetition, substitution, and transposition.

TextFooler is a word-level perturbation method proposed by Jin et al. (2020). It replaces words in the original prompt with synonyms or contextually similar words, changing the surface expression while preserving the overall semantics as much as possible.

CheckList is a sentence-level perturbation method. It generates 50 random strings consisting of letters and digits, each with a length of 10, and appends these strings one by one to the end of the prompt to construct task-irrelevant content.

ASR-Noise describes the construction of ASR-style noisy prompts for fine-tuning. As shown in Figure 6, we first take clean instructions from the fine-tuning set as input and synthesize English speech using XTTS-v2 (Casanova et al., 2024). The sampling rate is set to 24 kHz, and reference speakers are assigned by default using a round-robin strategy. We then apply acoustic perturbations to the synthesized speech, including background noise and mild reverberation, and all speech signals are peak-normalized. In the ASR stage, we use wav2vec2-large-960h, Whisper large-v3, and a Conformer-based ASR model as ASR systems (Baevski et al., 2020; OpenAI, 2023; Gulati et al., 2020). For the same perturbed audio, one ASR model is randomly selected for transcription, reducing noise bias caused by any single ASR model. The generated transcription is paired with the original target output and retained in the fields reported in Table 6. In evaluation, ASR robustness is measured separately on the public ASR-GLUE evaluation benchmark under low, medium, and high

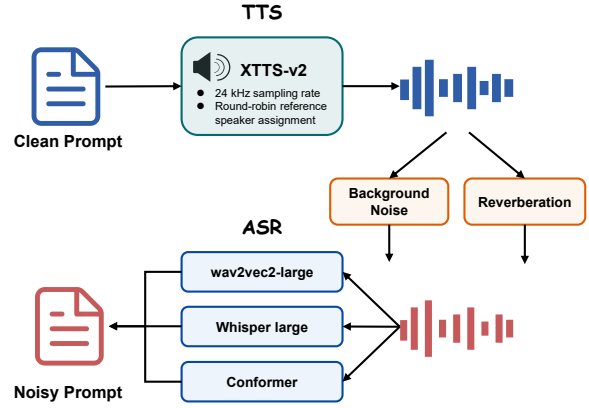


Figure 6: **ASR Noisy Prompt Generation Pipeline.** Clean prompts are synthesized with XTTS-v2, perturbed with acoustic noise and reverberation, and transcribed by wav2vec2, Whisper, or a Conformer-based ASR model to create noisy prompt pairs.

WER levels.

C Hyperparameters

Table 8 shows the hyperparameter settings used in our experiments.

D Evaluation on More Backbones

We conduct additional experiments to examine whether the trends observed on Qwen2.5-0.5B transfer across backbones. Specifically, we evaluate Llama-3.2-1B and Qwen2.5-1.5B under both text perturbations and ASR noise. The results are reported in Table 9 and Table 10. Results show that LoMaR consistently improves noisy prompt performance and reduces PDR compared with baselines.

To further examine scale sensitivity beyond the original 0.5B–1.5B range, we evaluate 3B and 7B SLMs under the same noisy-prompt protocol. Table 11 shows that LoMaR improves over the base model at every scale under both text and ASR noise. Although the gains decrease as model size increases, they remain positive at 7B: 2.06 points under text noise and 1.46 points under ASR noise.

E Evaluation on More Benchmarks

We further stress-test LoMaR on task formats not used in fine-tuning. Along with the held-out SQuAD and MMLU evaluations, we include CoQA (Reddy et al., 2019) for multi-turn question answering and GSM8K (Cobbe et al., 2021) for mathematical reasoning. All benchmarks use the same text-noise setup as the main experiments. Table 12

Parameter	Value
batch size	32
num epoch	1
lr	$1e-4$
cutoff length	256
temperature	0.01
lora modules	q_proj, k_proj, v_proj, o_proj
lora rank r	8
lora α	16
lora dropout	0.05
τ	1
m	4
λ_m	0.2
ρ	0.9
K	4
λ_a	0.1

Table 8: **Hyperparameter Settings.** Hyperparameter settings were used in our experiments.

reports F1 for SQuAD and CoQA, accuracy for MMLU, and exact match for GSM8K.

LoMaR outperforms SFT on all four benchmarks for both backbones. On the newly added task formats, its gains over SFT range from 0.73 to 1.45 points on Qwen2.5-0.5B and from 2.04 to 2.83 points on Qwen2.5-1.5B. These results indicate that LoMaR remains useful beyond the original FLAN fine-tuning subsets under the matched text-noise setting.

F SVD Analysis

We apply SVD to the LoRA update from Stage I and analyze how clean-task performance is recovered by different decomposed update spaces. Figure 7 uses t-SNE (van der Maaten and Hinton, 2008) to qualitatively visualize the Stage I update, its main and residual subspaces, and the final update after Stage II fine-tuning. The residual subspace forms patterns that are clearly separated from the Stage I update, while the final update ΔW^* remains close to the Stage I update in the projected space. This suggests that Stage II adapts the model through residual directions without substantially drifting away from the clean adaptation structure. Table 13 compares the base model with three reconstructed variants: adding only ΔW_{res} , adding only ΔW_{main} , and adding the full Stage I update ΔW_{base} . The residual-only variant remains close to the base model, whereas the main subspace recovers most of the Stage I gain. The full Stage I update obtains the best average score. These results

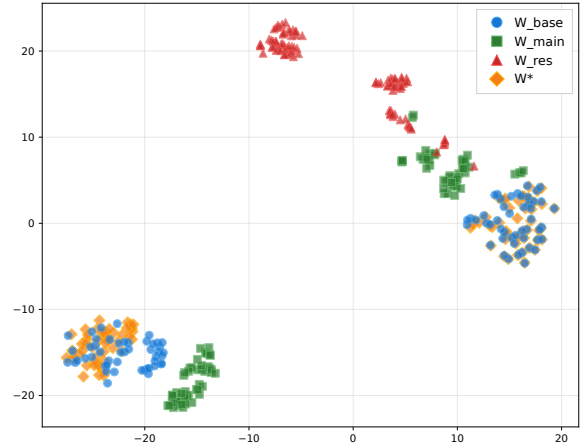


Figure 7: **SVD-decomposed LoRA update spaces.** t-SNE visualizes the Stage I LoRA update, its main and residual subspaces, and the final update after Stage II fine-tuning. The final update remains close to the Stage I update, indicating that residual-subspace adaptation preserves the main clean adaptation structure.

indicate that the dominant singular directions carry the main clean adaptation signal, supporting our choice to freeze ΔW_{main} during Stage II and use the residual subspace for noisy-prompt adaptation.

G Sensitivity Analysis

Sensitivity Analysis for ρ We conduct a sensitivity analysis on the energy threshold ρ used in SVD decomposition. We decompose the Stage I LoRA update with thresholds $\rho \in \{0.1, 0.2, \dots, 0.9\}$, and report the results in Figure 8. The heatmap shows that when ρ is small, such as $\rho \leq 0.4$, the main update subspace retains only a few singular directions, leaving some clean adaptation information in the trainable residual subspace. These directions may be altered during noisy prompt adaptation. When ρ is large, such as $\rho \geq 0.8$, more dominant directions are included in the frozen main subspace, making noisy prompt adaptation more constrained. In our experiments, $\rho = 0.9$ provides a better balance between clean performance preservation and noisy prompt adaptation.

Sensitivity Analysis for K We conduct a sensitivity analysis on the number of competitor tokens K used in the local margin objective. We vary $K \in \{1, 2, 4, 6, 8\}$ and report the training dynamics and final robustness trade-off in Figures 9 and 10. The total loss curves remain stable across different values of K , indicating that changing the competitor set size does not introduce optimization instability. However, task performance is sensitive

Model	Method	MNLI		MRPC		QNLI		QQP		SQuAD		MMLU		Avg	
		Acc	PDR	Acc	PDR	Acc	PDR	Acc	PDR	F1	PDR	Acc	PDR	Score	PDR
Llama-3.2-1B	Base	33.23	0.57	35.88	-2.63	51.80	3.11	46.76	15.63	62.51	1.82	<u>45.92</u>	2.13	46.02	4.09
	SFT	<u>46.12</u>	8.75	34.87	-3.20	53.88	-1.74	55.94	2.58	64.15	3.42	<u>40.90</u>	3.20	49.31	2.47
	COIN	37.05	8.70	34.21	-5.13	<u>55.49</u>	<u>0.25</u>	51.60	4.81	<u>65.07</u>	<u>1.36</u>	41.19	<u>1.15</u>	47.44	2.04
	CoIPO	44.10	4.67	<u>36.55</u>	-3.31	55.47	1.18	<u>59.66</u>	<u>2.47</u>	64.95	2.48	43.10	1.21	<u>50.64</u>	<u>1.73</u>
	LoMaR	49.88	<u>2.92</u>	47.10	<u>-3.61</u>	60.30	1.95	65.34	2.40	68.75	1.08	47.94	0.81	56.55	1.12
Qwen2.5-1.5B	Base	62.13	1.65	69.45	4.31	73.28	5.80	71.54	5.97	52.70	9.17	60.96	1.28	65.01	4.72
	SFT	63.48	<u>0.03</u>	68.67	5.23	<u>77.31</u>	<u>0.08</u>	72.80	5.66	54.68	8.13	<u>61.73</u>	2.28	66.45	3.51
	COIN	67.24	2.90	<u>70.26</u>	<u>1.91</u>	77.02	0.89	71.83	<u>2.88</u>	<u>56.64</u>	<u>2.55</u>	61.40	1.00	<u>67.40</u>	<u>2.01</u>
	CoIPO	<u>67.79</u>	-0.06	68.67	5.39	74.14	1.70	<u>72.88</u>	3.84	52.17	2.60	58.42	-0.86	65.68	2.22
	LoMaR	68.35	1.84	72.11	1.78	78.67	-0.22	77.99	2.30	61.70	2.51	62.02	<u>0.89</u>	70.14	1.50

Table 9: **Additional model results under text noise.** Llama-3.2-1B and Qwen2.5-1.5B are evaluated under text-noise settings, and PDR values report relative performance degradation from clean to noisy prompts.

Model	Method	Clean	Low	Medium	High
Llama-3.2-1B	Base	60.82	59.00	58.66	58.11
	SFT	63.61	62.40	60.83	61.06
	COIN	63.35	62.98	61.31	60.94
	CoIPO	62.56	61.04	60.47	59.51
	LoMaR	66.65	64.27	62.94	63.17
Qwen2.5-1.5B	Base	72.28	68.84	65.59	63.03
	SFT	72.92	69.55	66.02	65.07
	COIN	71.89	69.69	66.20	64.91
	CoIPO	72.11	69.00	64.96	63.71
	LoMaR	73.80	70.01	66.94	65.52

Table 10: **Additional model results on ASR noise.** Llama-3.2-1B and Qwen2.5-1.5B are evaluated under clean prompts and three ASR noise levels.

Model	Text Noise		ASR Noise	
	Base	LoMaR	Base	LoMaR
0.5B	50.14	56.79 (+6.65)	61.90	65.84 (+3.94)
1.5B	65.01	70.14 (+5.13)	67.44	69.07 (+1.63)
3B	69.04	72.54 (+3.50)	68.42	70.83 (+2.41)
7B	74.36	76.42 (+2.06)	70.19	71.65 (+1.46)

Table 11: **Scaling to Larger Backbones Under Noisy Prompts.** Average performance is reported under text and ASR noise.

to this choice: when $K = 1$, the margin objective only considers the single top-ranked non-gold token, limiting its ability to capture broader local competition. As K increases, the score improves and reaches its peak at $K = 4$, while larger values such as $K \geq 6$ may include less informative competitors and reduce the score. Although $K = 8$ yields the lowest PDR, its score is lower than that of $K = 4$, so we use $K = 4$ as the best balance between overall performance and robustness.

Model	Method	SQuAD	MMLU	GSM8K	CoQA
0.5B	Base	33.01	28.15	40.67	22.30
	SFT	33.44	31.61	42.52	23.72
	LoMaR	35.77	34.72	43.25	25.17
1.5B	Base	52.70	60.96	60.12	43.78
	SFT	54.68	61.73	61.50	45.37
	LoMaR	61.70	62.02	64.33	47.41

Table 12: **Evaluation on More Benchmarks Under Text Noise.** Average noisy performance is reported for SQuAD and MMLU, together with GSM8K and CoQA.

Method	Avg Score
Base	50.14
Base+ ΔW_{res}	50.28
Base+ ΔW_{main}	53.94
Base+ ΔW_{base}	54.78

Table 13: **SVD Subspace Reconstruction Results.** Residual-only, main-only, and full Stage I LoRA updates are compared with the base model.

Sensitivity Analysis for m We conduct a sensitivity analysis on the minimum safe margin m used in the margin expansion objective. We vary $m \in \{1, 2, 4, 6, 8\}$ and report the training dynamics and final robustness trade-off in Figures 11 and 12. The total loss curves show that larger margins impose stronger separation requirements in Stage I, with $m = 8$ producing higher loss before the Stage II transition, while the curves become closer after residual-subspace adaptation begins. In final performance, a small margin such as $m = 1$ imposes only a weak separation requirement between the gold token and competing non-gold tokens. As m increases, the score reaches its highest value at $m = 4$, indicating that a moderate safety margin helps build a more stable local pre-

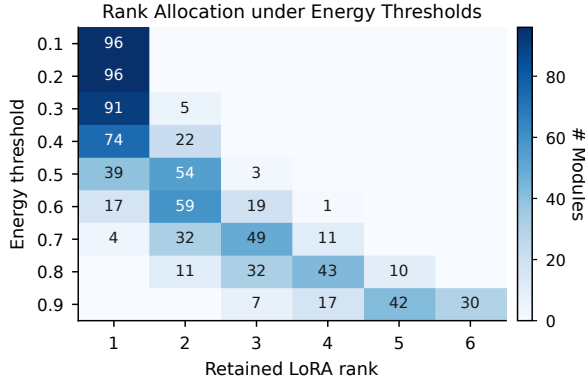


Figure 8: **Sensitivity to the SVD energy threshold ρ .** The heatmap reports retained main subspace ranks across LoRA modules under $\rho \in \{0.1, 0.2, \dots, 0.9\}$, with $\rho = 0.9$ balancing clean preservation and noisy adaptation.

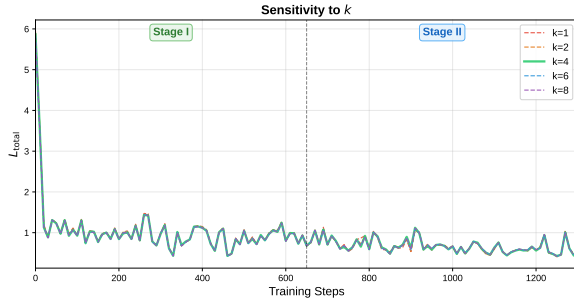


Figure 9: **Training Dynamics under Different K Values.** Total training loss is shown for $K \in \{1, 2, 4, 6, 8\}$ across Stage I and Stage II, suggesting that competitor set size mainly affects the final robustness trade-off rather than optimization stability.

diction structure. However, overly large margins such as $m \geq 6$ reduce the score and increase PDR, suggesting that excessive margin constraints may over-constrain optimization. In our experiments, $m = 4$ provides the best balance between clean performance and noisy-prompt robustness.

H Case Study

Figure 13 and Figure 14 show two examples that illustrate how LoMaR preserves prediction stability under noisy prompts. In the SST-2 example, the base model predicts the correct negative sentiment under the clean prompt but flips to positive under the noisy prompt. LoMaR maintains a positive logit margin for the correct negative label under both clean and noisy prompts. In the SQuAD example, the base model produces the correct answer "Renmin University" under the clean prompt but shifts to the incorrect answer "Rome University" under the noisy prompt. LoMaR maintains the cor-

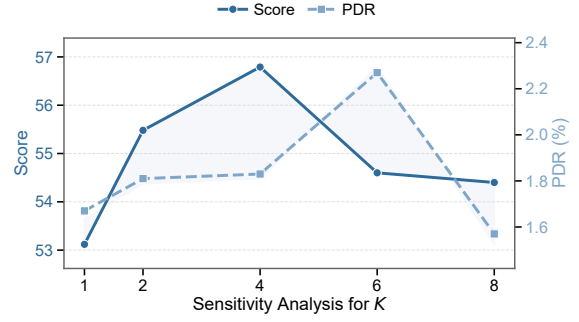


Figure 10: **Final Performance under Different K Values.** Score and PDR are reported for different competitor-set sizes, with $K = 4$ obtaining the best score.

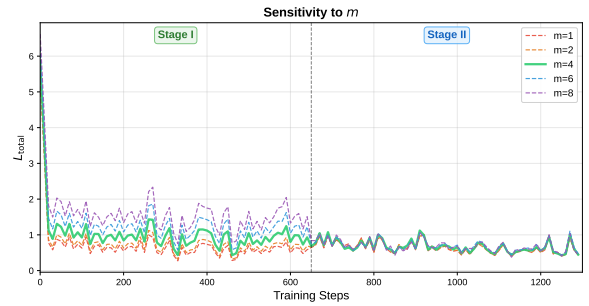


Figure 11: **Training Dynamics under Different m Values.** Total training loss is shown for $m \in \{1, 2, 4, 6, 8\}$ across Stage I and Stage II. Larger margins impose stronger Stage I constraints, while the curves become closer during Stage II residual adaptation.

rect answer span across clean and noisy prompts.

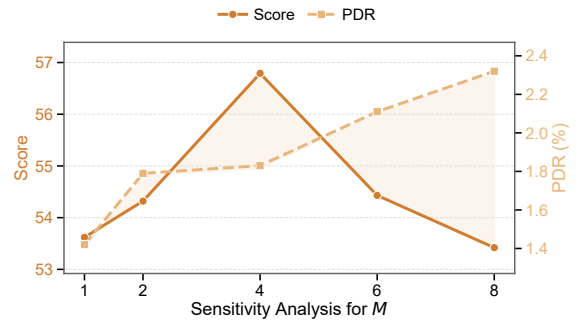


Figure 12: **Final Performance under Different m Values.** Score and PDR are reported for different minimum safe margins. $m = 4$ obtains the best score, while overly large margins reduce performance and increase PDR.

I Prompt Templates

Tables 14–21 show the clean prompts used during evaluation.

	Clean prompt			Noisy prompt		
Input	Considering the given phrase, would you say it carries a 'positive' or 'negative' connotation? Answer with exactly 'positive' or 'negative': {k-19 exploits our substantial collective positive fear of nuclear holocaust to generate cheap hollywood tension.}			Considering the gievn phrase, would jou say it carries a 'positive' or 'negative' connotation? Answer with exactly 'positive' or 'negative': {k-19 exploits our substantial collective fear of nuclear holocaust to generate cheap hollywood tension.}		
Base	logit=28.29 negative	logit=27.80 positive	Margin=0.49>0 negative ✓	logit=24.41 negative	logit=27.92 positive	Margin=-3.51<0 positive ✗
LoMaR	logit=26.21 negative	logit=23.98 positive	Margin=2.23>0 negative ✓	logit=26.44 negative	logit=25.32 positive	Margin=1.12>0 negative ✓

Figure 13: An example question from SST-2. The base model flips from the correct negative label to positive under the noisy prompt, whereas LoMaR preserves the negative prediction with a positive margin.

	Clean prompt				Noisy prompt			
Input	Based on the given context, provide the best possible answer to the question. Context:{The University of Chicago also maintains facilities apart from its main campus. The university's Booth School of Business maintains campuses in Singapore, London, and the downtown Streeterville neighborhood of Chicago. The Center in Paris, a campus located on the left bank of the Seine in Paris, hosts various undergraduate and graduate study programs. In fall 2010, the University of Chicago also opened a center in Beijing, near Renmin University's campus in Haidian District. The most recent additions are a center in New Delhi, India, which opened in 2014, and a center in Hong Kong which opened in 2015.} Question:{The university's center in Beijing is located next to what school's campus?}				Bsaed on the given context, provide the best possible answer to the qestion . Context:{The University of Chicago also maintains facilities apart from its main campus. The university's Booth School of Business maintains campuses in Singapore, London, and the downtown Streeterville neighborhood of Chicago. The Center in Paris, a campus located on the left bank of the Seine in Paris, hosts various undergraduate and graduate study programs. In fall 2010, the University of Chicago also opened a center in Beijing, near Renmin University's campus in Haidian District. The most recent additions are a center in New Delhi, India, which opened in 2014, and a center in Hong Kong which opened in 2015.} Question:{The university's center in Beijing is located next to what school's campus?}			
Base	Position 0 logit=22.7 Ren	Position 1 logit=25.29 min	Position 2 logit=18.6 University	Renmin University ✓	Position 0 logit=16.66 Ren	Position 1 logit=25.29 min	Position 2 logit=18.21 University	Rome University ✗
	logit=19.74 ren	logit=20.46 ming	logit=14.39 .		logit=19.77 Rome	logit=21.23 ming	logit=15.14 .	
	Margin=2.96>0	Margin=4.83>0	Margin=4.21>0		Margin=-3.11<0	Margin=4.06>0	Margin=3.07>0	
LoMaR	Position 0 logit=22.55 Ren	Position 1 logit=24.96 min	Position 2 logit=22.73 University	Renmin University ✓	Position 0 logit=20.03 Ren	Position 1 logit=24.71 min	Position 2 logit=22.97 University	Renmin University ✓
	logit=19.58 H	logit=19.66 ming	logit=15.79 university		logit=18.6 the	logit=20.06 ming	logit=16.75 university	
	Margin=2.97>0	Margin=5.3>0	Margin=6.94>0		Margin=1.43>0	Margin=4.65>0	Margin=6.22>0	

Figure 14: An example question from SQuAD. The base model changes the answer from “Renmin University” to “Rome University” under the noisy prompt, whereas LoMaR keeps the correct answer span stable.

No.	Prompt
1	Identify whether the given pair of sentences demonstrates entailment, neutral, or contradiction. Return exclusively 'entailment', 'neutral', or 'contradiction'.
2	Please classify the relationship between the provided sentences as 'entailment', 'neutral', or 'contradiction'. Do not include any explanation or additional content.
3	Acting as an entailment detection instrument, determine if the given pair of sentences demonstrates entailment, neutral, or contradiction. Answer with 'entailment', 'neutral', or 'contradiction'. Do not provide any explanation or additional information.
4	While performing entailment analysis, classify the relationship between the provided sentences as 'entailment', 'neutral', or 'contradiction'. Give only the classification result without any other content.

Table 14: **MNLI Prompt Templates.** Four prompt variants elicit target entailment labels.

No.	Prompt
1	Do these two sentences have the same underlying meaning? Only return 'equivalent' or 'not_equivalent', nothing else.
2	In your capacity as a language analyst, assess the following sentences and classify their similarity as 'equivalent' or 'not_equivalent'. Your response should be strictly 'equivalent' or 'not_equivalent' only.
3	As a sentence similarity evaluator, analyze the provided sentences and indicate if their meanings are 'equivalent' or 'not_equivalent'. Return only 'equivalent' or 'not_equivalent' without any additional text.
4	In your capacity as a semantic assessment tool, evaluate the provided sentences and classify their meanings as 'equivalent' or 'not_equivalent'. Output only 'equivalent' or 'not_equivalent' as your final response.

Table 15: **MRPC Prompt Templates.** Four prompt variants elicit binary semantic equivalence labels.

No.	Prompt
1	Given the question and context provided, determine if the answer can be inferred by choosing 'entailment' or 'not_entailment'. Only respond with 'entailment' or 'not_entailment', nothing else.
2	Based on the information in the context, decide if the answer to the question is justified by choosing 'entailment' or 'not_entailment'. Give only 'entailment' or 'not_entailment' as your response.
3	As a textual analyst, examine if the given context logically implies the answer to the question and indicate your decision with 'entailment' or 'not_entailment'. Return exclusively 'entailment' or 'not_entailment'.
4	In your role as a linguistic investigator, determine if the context given entails the answer to the question and provide your conclusion with 'entailment' or 'not_entailment'. Output should be strictly 'entailment' or 'not_entailment'.

Table 16: **QNLI Prompt Templates.** Four prompt variants decide whether the context entails the answer.

No.	Prompt
1	Determine if the given pair of statements can be considered the same by responding with 'equivalent' or 'not_equivalent'. Only return 'equivalent' or 'not_equivalent', nothing else.
2	Analyze if the given set of sentences have the same connotation by answering with 'equivalent' or 'not_equivalent'. Respond with only these exact terms.
3	Acting as a question equivalence instrument, determine if the provided questions are equivalent in meaning, answering with 'equivalent' for similar questions or 'not_equivalent' for dissimilar ones. Return only these two options, no additional text.
4	In the capacity of a question assessment system, indicate if the meaning of the provided questions is the same, responding with 'equivalent' or 'not_equivalent'. Do not include any other information in your response.

Table 17: **QQP Prompt Templates.** Four prompt variants elicit binary question-equivalence labels.

No.	Prompt
1	Considering the given phrase, would you say it carries a 'positive' or 'negative' connotation? Answer with exactly 'positive' or 'negative':
2	As an emotion detector, determine if the provided passage conveys a 'positive' or 'negative' sentiment. Answer with just 'positive' or 'negative':
3	Taking on the role of an emotion classifier, specify if the provided phrase is 'positive' or 'negative'. Give me only 'positive' or 'negative':
4	Functioning as a sentiment identification tool, assess if the following expression is 'positive' or 'negative'. Output just 'positive' or 'negative':

Table 18: **SST-2 Prompt Templates.** Four prompt variants elicit positive or negative sentiment labels.

No.	Prompt
1	Does the premise entail the hypothesis? Reply with 'entailment' or 'not_entailment'. Do not include any additional text in your response.
2	Can the hypothesis be logically inferred from the premise? Answer with 'entailment' or 'not_entailment'. Provide no other information.
3	Based on the premise, is the hypothesis necessarily true? Provide your answer as 'entailment' or 'not_entailment'. No additional explanation is needed.
4	As a tool for recognizing textual entailment, determine whether the hypothesis follows from the premise and classify it as either 'entailment' or 'not_entailment'. Provide only 'entailment' or 'not_entailment' in your response.

Table 19: **RTE Prompt Templates.** Four prompt variants elicit binary textual-entailment labels.

No.	Prompt
1	Based on the given context, provide the best possible answer to the question.
2	Read the following passage and answer the question using only information from the context.
3	Please extract the most appropriate answer from the context provided below.
4	Using the context, determine the most suitable answer to the question.

Table 20: **SQuAD Prompt Templates.** Four prompt variants elicit extractive answers from context.

No.	Prompt
1	Answer the following multiple-choice question about {} by selecting the correct option: 'A', 'B', 'C', or 'D'.
2	For the multiple-choice question related to {}, please choose the most accurate answer from 'A', 'B', 'C', or 'D'.
3	Please respond to the multiple-choice question about {} by selecting the appropriate answer: 'A', 'B', 'C', or 'D'.
4	Evaluate the multiple-choice question about {} and select the most fitting response from 'A', 'B', 'C', or 'D'.

Table 21: **MMLU Prompt Templates.** Four prompt variants elicit one of the multiple-choice options.