Journal of Computer Technology and Software
ISSN:2998-2383
Vol. 4, No. 3, 2025

Risk-Aware Large Language Model Fine-Tuning with Tail-Robust Objectives and Feasible Gradient Updates

Yihang Wang[1*]
[1]University of Southern California, Los Angeles, USA
*Corresponding author: Yihang Wang; yihang.wang.dev@gmail.com

Abstract: This paper addresses the problem of unstable updates and difficulty in constraining behavioral boundaries in large language models during real-world applications due to data noise and distribution perturbations. A robust fine-tuning algorithm framework based on gradient projection and constraint optimization is proposed. The method uniformly models the fine-tuning process as a constrained optimization problem. On the objective side, worst-case robust risk and tail risk are introduced to enhance the resistance to abnormal samples and perturbation inputs. On the constraint side, a feasible region composed of output distribution offset constraints and parameter trust region constraints is constructed to limit distribution drift and parameter drift caused by fine-tuning. To ensure the feasibility of iterative updates, a gradient projection mechanism is designed to decompose and correct the original update direction into an effective descent direction that satisfies the approximation of the local feasible region. Simultaneously, primal-dual updates are used to dynamically allocate constraint pressure, forming a closed-loop synergy between constraint satisfaction and objective optimization. Furthermore, the framework is compatible with efficient parameter fine-tuning. Low-rank incremental parameterization restricts trainable updates to a low-dimensional subspace, reducing training overhead while preserving constraint-driven geometric control capabilities. This research provides a structured, interpretable, and reproducible optimization path for robust fine-tuning and offers a unified technical solution for adapting highly reliable large models to task requirements and constraint boundaries.

Keywords: Gradient projection; constrained optimization; primal dual update; tail risk modeling.

1. Introduction

In recent years, large language models have demonstrated strong generalization capabilities in text understanding and generation tasks, and have been widely applied in scenarios such as knowledge question answering, information extraction, content generation, and decision support[1,2]. However, as application boundaries continue to expand, problems such as data distribution drift, noisy annotation, adversarial input, and security compliance constraints are becoming increasingly prominent in real-world environments, making output stability and controllability key bottlenecks in the implementation process. Compared to offline static evaluation conditions, real-world business applications typically require models to maintain consistent behavior under uncertain inputs and multi-source perturbations, and to have stronger resistance to bias propagation and risk spillover. Therefore, constructing robust and controllable fine-tuning methods has clear practical needs and theoretical value[3].

Efficient parameter fine-tuning, as an important technical path to reduce computational and storage costs, has become the mainstream choice for adapting large language models to downstream tasks. However, common fine-tuning paradigms often rely on empirical loss weighting, heuristic regularization terms, or loose constraint processing, lacking explicit control over the geometric structure of parameter update directions. This can easily lead to gradient bias accumulation and catastrophic shifts when the proportion of noisy gradients and outlier samples increases. Furthermore, when task objectives simultaneously include accuracy, robustness, alignment, and resource constraints, single-objective-driven update strategies struggle to maintain consistency across multiple constraints, leading to performance instability, sensitivity to input perturbations, and unpredictable output behavior. Shifting the fine-tuning process from empirical parameter tuning to interpretable optimization structure design becomes a necessary direction for improving reliability and reproducibility[4].

Gradient projection provides a direct geometric mechanism for constrained parameter updates, limiting the update direction to a feasible region that satisfies the constraints, thereby suppressing harmful drift while maintaining task adaptability. By formulating the fine-tuning objective as a constrained optimization problem, robustness requirements and safety boundaries can be explicitly characterized within a unified framework. For example, structured constraints can be imposed on output distribution shift, parameter norm, update sparsity, and alignment costs, and the gradient can be decomposed into feasible and suppressed components using projection operators. This type of method not only helps reduce the interference of noisy gradients on the update direction but also provides an interpretable optimization basis for multi-objective trade-offs, enabling the model to achieve

more stable convergence behavior and more consistent output characteristics under limited resources and complex constraints[5].

In the context of high-risk and heavily regulated applications, robust fine-tuning is not only crucial for improving task metrics but also for the controllability, auditability, and long-term maintainability of model behavior. Research based on gradient projection and constraint optimization can theoretically establish the correspondence between parameter updates and constraint satisfaction, providing verifiable optimization support for robustness and a scalable, unified modeling approach for different constraint types. At the engineering level, this paradigm facilitates the formation of modular optimization components, reduces reliance on heuristic techniques, and improves stability and consistency during cross-task transfer. This drives the fine-tuning of large language models from experience-driven to structured and interpretable optimization-driven approaches, providing a more solid methodological foundation for the deployment of trustworthy large models.

2. Background

Large language models typically rely on fine-tuning mechanisms to transfer general representations to specific tasks and domain contexts. However, significant differences in target and distribution exist between pre-trained representations and downstream objectives, resulting in strong non-convexity and strong coupling characteristics in parameter updates within a high-dimensional space. In this context, the fine-tuning process not only needs to learn task-related patterns but also to avoid excessive erosion of general capabilities and suppress overfitting and bias amplification under limited sample and weak supervision conditions. Simultaneously, real-world data often contains long-tailed samples, noisy labels, and potential biases, making gradient estimation susceptible to the dominance of outliers. This leads to drastic fluctuations in update direction and structural distortions in the representation space, resulting in decreased output consistency and increased generalization fragility[6].

Constrained optimization provides a systematic perspective on these problems, viewing fine-tuning as an optimization process within a feasible domain. By incorporating safety boundaries, structural priors, and resource constraints into the optimization objective as constraints, explicit control over update behavior is achieved. Unlike soft constraints that rely solely on regularization terms, constrained optimization emphasizes feasibility and boundary satisfaction, allowing model updates to form a clear geometric structure around stability objectives. The gradient projection method plays a crucial role in this framework. Its core lies in performing a geometric transformation on the original gradient, eliminating or correcting components that do not meet the constraints, and preserving the effective descent direction consistent with the feasible region. This ensures learning efficiency while reducing invalid or even harmful parameter drift[7]. By transforming the multi-objective requirements in fine-tuning into a computable set of constraints and combining this with projection operators for iterative updates, a clearer optimization path and a stronger interpretability foundation can be provided for robust fine-tuning.

3. Method

This method formulates robust fine-tuning as a constrained optimization problem, aiming to maintain controllability and feasibility of parameter updates in the presence of uncertain perturbations and noisy gradients. Let the pretrained model parameters be $\boldsymbol{\theta}_0$, the fine-tuned parameters be $\boldsymbol{\theta}$, the training-sample distribution be $\mathcal{D}$, and the task loss be $l(\boldsymbol{\theta}; z)$. The base empirical risk can be written as follows.

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{z\sim\mathcal{D}}[l(\boldsymbol{\theta}; z)]$$

To characterize the worst-case impact induced by input perturbations or distribution shifts, we introduce an uncertainty set $\mathcal{U}$ and adopt the robust risk as the optimization objective.

$$\mathcal{R}(\boldsymbol{\theta}) = \sup_{u\in\mathcal{U}} \mathbb{E}_{z\sim\mathcal{D}}\left[l\left(\boldsymbol{\theta}; \mathcal{T}_u(z)\right)\right]$$

In practice, the $\sup$ -type objective can be smoothly replaced by a risk measure; using a quantile-risk form highlights the influence of tail hard samples on robustness.

$$\mathrm{CVaR}_\alpha(\boldsymbol{\theta}) = \min_{\eta\in\mathbb{R}} \eta + \frac{1}{1-\alpha}\mathbb{E}_{z\sim\mathcal{D}}\left[\left(l(\boldsymbol{\theta}; z) - \eta\right)_+\right]$$

To incorporate behavioral controllability and safety boundaries into a unified optimization framework, we encode key requirements as a constraint set $\mathcal{C}$ that defines the feasible region. Suppose there are m constraint functions $g_i(\boldsymbol{\theta})$; then the feasible set is defined as follows.

$$\mathcal{C} = \left\{\boldsymbol{\theta} \mid g_i(\boldsymbol{\theta}) \le 0,\ i = 1, \ldots, m\right\}$$

To suppress excessive deviation from the reference model distribution during fine-tuning, we introduce an output-distribution distance constraint so that updates remain within a controlled neighborhood.

$$g_{\mathrm{kl}}(\boldsymbol{\theta}) = \mathbb{E}_{x\sim\mathcal{D}_x}\left[\mathrm{KL}\left(p_{\boldsymbol{\theta}}(\cdot\mid x)\ \ p_{\boldsymbol{\theta}_0}(\cdot\mid x)\right)\right] - \epsilon$$

Meanwhile, a trust-region constraint on the parameter distance limits structural drift in the representation space, thereby improving update stability and interpretability.

$$g_{\mathrm{tr}}(\boldsymbol{\theta}) = \|\boldsymbol{\theta} - \boldsymbol{\theta}_0\|_2^2 - \rho^2$$

Under constraints, the core operation is to project the gradient update direction onto the tangent space of the feasible set or an approximate feasible cone, ensuring that each update step satisfies the constraints as much as possible. To facilitate the explanation of the algorithm flow in this paper, the overall model architecture is shown in Figure 1.

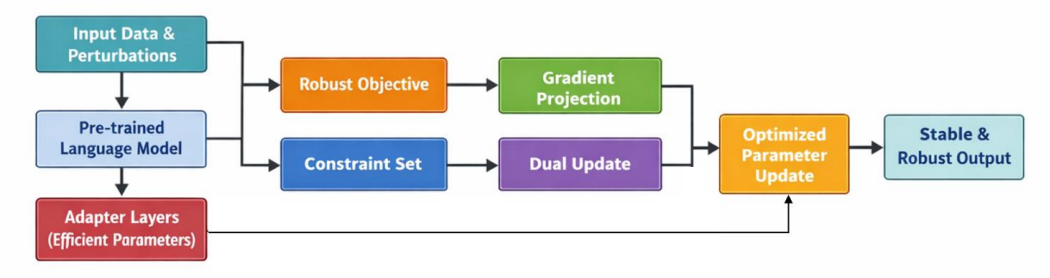

Figure 1. Overall model architecture

Let the robust objective be $\mathcal{J}(\boldsymbol{\theta})$, with gradient $\mathbf{g} = \nabla_{\boldsymbol{\theta}}\mathcal{J}(\boldsymbol{\theta})$. After adopting a local approximation via linearized constraints $\mathbf{A}\mathbf{d} \leq \mathbf{b}$, the projection direction can be obtained by solving a quadratic program.

$$\mathbf{d}^{\star} = \arg\min_{\mathbf{d}} \frac{1}{2}\|\mathbf{d} + \mathbf{g}\|_2^2 \quad \text{s.t. } \mathbf{A}\mathbf{d} \leq \mathbf{b}$$

When the constraints are approximated in equality form as $\mathbf{A}\mathbf{d} = \mathbf{0}$, the projection solution admits a closed-form expression, and the update direction is given by an orthogonal projection operator.

$$\mathbf{d}^{\star} = -\left(\mathbf{I} - \mathbf{A}^{\top}\left(\mathbf{A}\mathbf{A}^{\top}\right)^{-1}\mathbf{A}\right)\mathbf{g}$$

Accordingly, we obtain the parameter iteration rule, where the step size γ_t is determined jointly by stability requirements and the constraint scale.

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \gamma_t \mathbf{d}_t^{\star}$$

To further enhance scalability, we unify feasibility and optimality from a primal-dual perspective and explicitly characterize constraint-driven direction correction via the Lagrangian. Let the dual variables be $\boldsymbol{\lambda} \geqslant \mathbf{0}$; then the formulation is given as follows.

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\lambda}) = \mathcal{J}(\boldsymbol{\theta}) + \sum_{i=1}^{m} \lambda_i \, g_i(\boldsymbol{\theta})$$

Updating the dual variables by projected gradient ascent prevents violations of nonnegativity and continuously suppresses the degree of constraint violation.

$$\boldsymbol{\lambda}_{t+1} = \Pi_{\mathbb{R}_+^m}\left(\boldsymbol{\lambda}_t + \beta_t \mathbf{g}_{\lambda}\left(\boldsymbol{\theta}_t\right)\right)$$

Considering parameter-efficient fine-tuning, we restrict trainable incremental parameters to an adapter subspace to reduce storage and computation overhead. Let the frozen backbone parameters be $\boldsymbol{\theta}_0$ and the trainable increment is $\Delta\boldsymbol{\theta}$; then the parameter decomposition takes the following form.

$$\boldsymbol{\theta} = \boldsymbol{\theta}_0 + \Delta\boldsymbol{\theta}$$

When the increment is modeled with a low-rank structure, the update is constrained to a low-dimensional manifold, facilitating stable and controllable optimization in coordination with the gradient projection mechanism.

$$\Delta\boldsymbol{\theta} = \mathbf{U}\mathbf{V}^{\top}$$

4. Experimental Results and Analysis

4.1 Dataset

This paper selects the OpenAssistant Conversations Dataset OASST1 as the data foundation for fine-tuning and robust modeling. This dataset is an open-source instruction dialogue corpus covering various task types, including question answering, writing, reasoning, code explanation, and knowledge dissemination. The samples are organized in a multi-turn dialogue structure, containing pairing information between user instructions and assistant responses, and providing multiple candidate responses and preference annotations for the same instruction. This facilitates combined modeling of robust goal construction and constraint learning on a unified corpus. The data source and annotation process are open and transparent, and the community maintains a robust system, making it suitable as a standardized carrier for robust fine-tuning methods to support reproducible algorithm research.

To match the robust fine-tuning settings based on gradient projection and constraint optimization, data preprocessing employs a dialogue turn expansion and instruction reconstruction strategy. Each sample is organized into a supervision pair of input prompts and target responses, retaining multiple candidate responses and preference information for constructing training signals related to distribution constraints and risk metrics. The cleaning stage removes excessively short samples, samples with densely packed anomalous symbols, and obviously irrelevant content, and standardizes length pruning and formatting to reduce the unstructured impact of noise gradients. The partitioning method adopts a fixed random partitioning of the training set, validation set, and test set, and sets a random seed to ensure that the optimization process under different constraint strengths and projection strategies has a consistent data benchmark and comparability.

4.2 Experimental Setup

This study completed the model fine-tuning and algorithm verification process in a single-machine, single-GPU environment. The base model used was Qwen7B, and efficient parameter fine-tuning was employed to reduce memory usage and training costs. The software stack consisted of Python on a Linux system combined with mainstream deep learning frameworks. Mixed precision was used during training to improve throughput and control peak memory usage. The optimizer used was AdamW, along with cosine annealing

learning rate scheduling and warm-up strategies for stable convergence. All stochastic components were configured with fixed random seeds, and data loading and parallel operators were kept consistent to reduce fluctuations caused by nondeterminism. Table 1 summarizes the hardware and software environment and core hyperparameter configurations used in the experiment.

During the training phase, a standard input/output concatenation format for instruction fine-tuning was adopted. Constraints were placed on the maximum input and output lengths to adapt to memory and computational budgets. Dynamic padding was used to improve batch processing utilization. Gradient pruning was used to suppress the impact of abnormal gradient peaks on the update direction, and gradient accumulation was used to obtain a more stable equivalent batch size under memory constraints. The parameter efficiency module adopts a low-rank adapter form and sets the rank and scaling factor to ensure that the update scale is comparable under different constraint strengths; the constraint-related threshold and dual update step size are set to a conservative range to ensure that the projection update and dual correction work together and maintain the optimization feature of prioritizing feasibility.

Table 1. Summary of Experimental Environment and Hyperparameter Settings.

Category	Configuration Item	Value
Hardware	GPU	NVIDIA RTX 4090 24GB
Hardware	CPU	Intel Xeon 16 Cores
Hardware	Memory	64 GB
Hardware	Storage	1 TB NVMe SSD
System	Operating System	Ubuntu 22.04 LTS
Software	Python	3.10
Software	Deep Learning Framework	PyTorch 2.2
Software	CUDA	12.1
Software	cuDNN	8.9
Model	Base Model	Qwen7B
Training	Fine-tuning Method	LoRA
Training	LoRA Rank r	8
Training	LoRA Alpha	16
Training	LoRA Dropout	0.05
Training	Mixed Precision	FP16
Training	Random Seed	42
Data	Max Input Length	1024
Data	Max Output Length	512
Data	Dynamic Padding	Enabled
Optimization	Optimizer	AdamW
Optimization	Learning Rate	2e-4
Optimization	Weight Decay	0.01
Optimization	Warm-up Ratio	0.03
Optimization	Scheduler	Cosine
Training	Batch Size	8
Training	Gradient Accumulation	2
Training	Max Gradient Norm	1.0
Training	Training Epochs	3
Constraints	KL Constraint Threshold	0.10
Constraints	Parameter Trust-Region Radius	0.50
Constraints	Dual Step Size	1e-2
Projection	Projection Frequency	Project once per update step

4.3 Experimental Results and Analysis

To evaluate the performance of the proposed robust fine-tuning method based on gradient projection and constraint optimization across different evaluation dimensions, this paper selects representative large-model fine-tuning studies in the same direction as a baseline for comparison and conducts a horizontal comparison under a unified evaluation index system. The experimental results are shown in Table 2. This comparison setting aims to provide comparable quantitative references from the perspectives of generation quality, language consistency, and semantic matching, providing a basis for subsequent methodological discussions.

Table 2. Comparative experimental results

Method	PPL	ROUGE-L	BLEU-4	BERTScore
Tian et al.[8]	28.41	32.55	11.92	0.842
Tian et al.[9]	27.89	33.02	12.10	0.847
Huang et al.[10]	26.77	34.21	12.84	0.853
Ao et al.[11]	25.68	35.90	13.42	0.861
Wang et al.[12]	24.55	36.73	13.88	0.867
Fu et al.[13]	23.92	37.10	14.02	0.872
Xin et al.[14]	23.11	37.68	14.25	0.878
Ours	20.44	40.22	16.31	0.901

Table 2 presents the performance of different methods across four metrics: perplexity, ROUGE-L, BLEU-4, and BERTScore, revealing a clear hierarchical difference. The proposed method achieves optimal performance across all four metrics, demonstrating stronger comprehensive capabilities in terms of generation fluency, text overlap matching, and semantic consistency. This advantage is not limited to a single metric.

From the perspective of perplexity, the proposed method achieves a lower value, indicating a more comprehensive characterization of the target text distribution and output that better aligns with the predictability and coherence expected in language modeling. This result typically corresponds to more stable generation behavior and fewer unnatural fragments, reflecting how the constraints and corrections on parameter updates during fine-tuning effectively reduce unnecessary distribution shifts, thereby maintaining higher controllability and consistency at the expression level.

The advantages in ROUGE-L and BLEU-4 indicate that the generated content is closer to the reference answer in terms of sequence-level and fragment-level matching. Furthermore, the improved BERTscore further validates the higher semantic fit. The simultaneous improvement of the three complementary indicators indicates that the improvement does not come from a simple amplification of surface overlap, but is more likely due to the synchronous enhancement of semantic preservation and expression quality. This makes the output more balanced in terms of information coverage, word choice, and semantic consistency, thus demonstrating the effectiveness of the robust fine-tuning strategy under multi-dimensional quality requirements.

Dual variables are used to characterize the driving strength of constraints on the optimization process, and their evolution

reflects the dynamic distribution of constraint pressure during training. Constraint violation provides a direct characterization of feasibility, describing how well parameter updates fit the constraint boundaries. Visualizing the linkage between the two helps to explain the synergistic relationship between projection and dual updates from the perspective of the optimization mechanism, as shown in Figure 2.

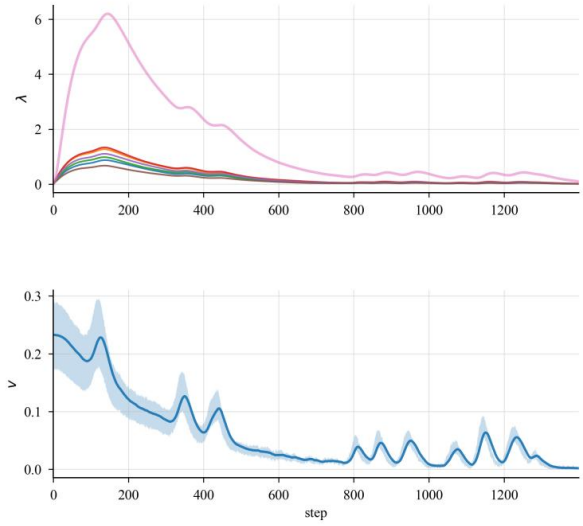

Figure 2. A visualization experiment linking the convergence trajectory of dual variables with the degree of constraint violation.

The visualization consists of two parts. The upper vertical axis represents the magnitude of the dual variable λ, reflecting the pressure and adjustment intensity of constraints during the optimization process. The lower vertical axis represents the constraint violation degree v, depicting the deviation of the current parameter update from the feasible boundary. The proposed method establishes a clear and interpretable linkage during training. When the constraint violation degree is high, the dual variable is activated to enhance the constraint penalty effect. After the violation degree is suppressed, the dual variable gradually returns to a more stable range, demonstrating the constraint-driven adaptive adjustment capability at the mechanism level.

Overall, the constraint violation degree is consistently kept within a low range and quickly returns to a controllable state even when local perturbations occur, indicating that projection updates and dual corrections effectively suppress deviations caused by infeasible directions. Simultaneously, the dual variable curve exhibits a stable and consistent adjustment response in critical stages, without disordered amplification or prolonged stagnation. This reflects that the proposed method achieves a more balanced optimization behavior between robustness and feasibility, thus supporting a more reliable fine-tuning process and more controllable model updates.

Different constraint weights determine the strength of feasibility pressure during the optimization process, thus affecting the behavior of parameter updates near the feasible region. The feasible region boundary coverage visualization aims to observe the reach and distribution density of the updated trajectory in the boundary neighborhood to characterize the impact of constraint adjustment on geometric feasibility. This visualization constructs a simulation process solely based on the gradient projection and constraint optimization mechanism described in this paper, providing intuitive evidence at the mechanistic level.

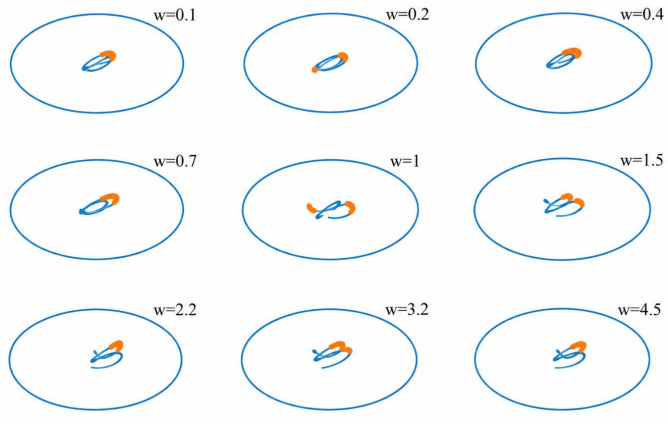

Figure 3. Visualization experiment of feasible domain boundary coverage under different constraint weights.

Figure 3 uses an elliptical boundary to represent the feasible region, showing the activity range and reach distribution of the parameter update trajectory in the neighborhood of the feasible region boundary under different constraint weights w. Here, w represents the weight coefficient of the constraint term in the optimization objective, used to adjust the intensity of the influence of feasibility pressure on the update direction. A larger value means a stronger penalty for violating constraints, and a higher degree of dominance of projection and feasibility correction on the update. As can be seen from the figure, with changes in the setting of w, the distribution pattern of the trajectory within the feasible region and the reach area near the boundary change significantly, reflecting that the constraint strength can explicitly reshape the geometric behavior of the optimization process.

From a mechanistic perspective, a smaller w tends to make the update more influenced by the task-driven terms, with the trajectory more concentrated in local areas within the feasible region and a relatively limited reach to the boundary. As w increases, constraint pressure is more easily activated, and the update will more frequently receive projection corrections or the pull of the boundary neighborhood, thus forming a more sufficient coverage and a more stable feasibility maintenance near the boundary. Overall, the figure provides visual evidence of how gradient projection and constraint optimization work, demonstrating that by adjusting w, a controllable geometric tradeoff can be achieved between the learned degrees of freedom and the feasible region constraints, so that fine-tuning updates maintain effective search while avoiding instability caused by deviations from the feasible region.

5. Conclusion

This paper addresses the core challenge of fine-tuning robust large language models by constructing a unified framework based on gradient projection and constraint optimization. This framework integrates robustness objectives, feasible region constraints, and efficient parameter updates into a single optimization perspective. Through geometric constraints on the update direction and dynamic adjustment of dual variables, this framework explicitly controls distribution shift and parameter drift during training, enabling the model to adapt to task requirements while maintaining stronger controllability and consistency. The methodological contribution lies in transforming the previously experience-based robustness strategy into an interpretable constraint-driven optimization process, providing a clearer theoretical expression and engineering implementation path for the stability sources and constraint satisfaction mechanisms in the fine-tuning phase.

In terms of application, this research provides more actionable technical support for the implementation of trustworthy large models. For high-risk scenarios such as financial risk control, compliance auditing, medical text processing, and government and enterprise knowledge management, model outputs not only need task adaptability but also must meet requirements such as clear behavioral boundaries, controllable risk spillover, and traceable decision-making chains. The constrained robust fine-tuning paradigm proposed in this paper more naturally accommodates such needs, embedding safety boundaries and business constraints into the training process in a computable form. This reduces the probability of inconsistent model behavior under noisy inputs, distribution drift, or long-tail problems, providing a more reliable training foundation for stable deployment and continuous iteration in real-world business scenarios.

From an engineering scalability perspective, this method is highly compatible with efficient parameter fine-tuning mechanisms, enabling the synergy of robust optimization objectives and feasibility constraints in resource-constrained environments. By restricting trainable parameters to a low-rank incremental subspace and performing feasible region projection and dual correction in each update step, the method achieves explicit scheduling of the update geometry while keeping computational overhead under control. This characteristic helps to drive large-scale model fine-tuning from experience-driven to a structured, interpretable, and auditable optimization-driven process, and also provides a transferable implementation paradigm for subsequent applications in larger-scale models and more complex constraint systems.

Future work can be expanded around richer constraint forms and more application-oriented robust objectives. On the one hand, the feasible region can be expanded from a single distribution distance or parameter trust region to a set of complex constraints, including fairness, privacy protection, factual consistency, and security policy consistency. More efficient projection approximation and feasibility maintenance mechanisms can be explored to adapt to ultra-large-scale parameter spaces. On the other hand, in conjunction with online data updates and continuous learning scenarios, constraint adaptive adjustment strategies under time-evolving distributions can be studied, enabling dual variables to respond more precisely to changes in business rules and adjustments in risk preferences. Through advancements in these directions, a robust fine-tuning framework based on gradient projection and constraint optimization is expected to further improve the reliability, controllability, and long-term maintainability of large language models in key industry applications and promote the implementation and standardization of trustworthy artificial intelligence technology systems in a wider range of scenarios.

References

[1] Hu E J, Shen Y, Wallis P, et al. Lora: Low-rank adaptation of large language models[J]. Iclr, 2022, 1(2): 3.

[2] Zhang Q, Chen M, Bukharin A, et al. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning[J]. arXiv preprint arXiv:2303.10512, 2023.

[3] Dettmers T, Pagnoni A, Holtzman A, et al. Qlora: Efficient finetuning of quantized llms[J]. Advances in neural information processing systems, 2023, 36: 10088-10115.

[4] Liu S Y, Wang C Y, Yin H, et al. Dora: Weight-decomposed low-rank adaptation[C]//Forty-first International Conference on Machine Learning. 2024.

[5] Zhang R, Han J, Liu C, et al. Llama-adapter: Efficient fine-tuning of language models with zero-init attention[J]. arXiv preprint arXiv:2303.16199, 2023.

[6] Li K, Xie W, Huang Y, et al. Dual Risk Minimization: Towards Next-Level Robustness in Fine-tuning Zero-Shot Models[J]. Advances in Neural Information Processing Systems, 2024, 37: 66025-66057.

[7] Meng T, Mehrabi N, Goyal P, et al. Attribute controlled fine-tuning for large language models: A case study on detoxification[C]//Findings of the Association for Computational Linguistics: EMNLP 2024. 2024: 13329-13341.

[8] Tian J, He Z, Dai X, et al. Trainable projected gradient method for robust fine-tuning[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023: 7836-7845.

[9] Tian J, Liu Y C, Smith J S, et al. Fast trainable projection for robust fine-tuning[J]. Advances in Neural Information Processing Systems, 2023, 36: 11374-11393.

[10] Huang C, Tian J, Maneechotesuwan B, et al. Directional gradient projection for robust fine-tuning of foundation models[J]. arXiv preprint arXiv:2502.15895, 2025.

[11] Ao S, Dong Y, Hu J, et al. Safe Pruning LoRA: Robust Distance-Guided Pruning for Safety Alignment in Adaptation of LLMs[J]. Transactions of the Association for Computational Linguistics, 2025, 13: 1474-1487.

[12] Wang C, Lyu Y, Sun Z, et al. Continual gradient low-rank projection fine-tuning for LLMs[C]//Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2025: 14815-14829.

[13] Fu Z, Yang H, So A M C, et al. On the effectiveness of parameter-efficient fine-tuning[C]//Proceedings of the AAAI conference on artificial intelligence. 2023, 37(11): 12799-12807.

[14] Xin C, Lu Y, Lin H, et al. Beyond full fine-tuning: Harnessing the power of LoRA for multi-task instruction tuning[C]//Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024). 2024: 2307-2317.