Beyond Gradient Egalitarianism: Dual-Adaptive C-AUM Framework for Few-Shot Fine-Tuning of Large Language Models

Qiaojuan Peng[1,2,3], Xiong Luo[1,2,3,*], Yuqi Yuan[1,2,3], Wenbing Zhao[4]
[1]School of Computer and Communication Engineering,
University of Science and Technology Beijing, Beijing, China
[2]Shunde Innovation School, University of Science and Technology Beijing, Foshan, China
[3]Beijing Key Laboratory of Knowledge Engineering for Materials Science, Beijing, China
[4]Department of Electrical and Computer Engineering,
Cleveland State University, Cleveland, OH, USA
***Correspondence:** xluo@ustb.edu.cn

Abstract

While large language models (LLMs) demonstrate strong text classification capabilities, adapting them to specialized, few-shot industrial domains remains challenging. Vanilla supervised fine-tuning (SFT) applies uniform weighting to training samples and may overfit easy instances while insufficiently optimizing ambiguous boundary samples. Static regularization may further limit adaptation by applying the same constraint throughout training. To address these limitations, we propose cumulative area under the margin (C-AUM), a fine-tuning framework tailored to autoregressive LLM classification. C-AUM tracks the cumulative margin of the final decision token as an instance-level difficulty signal and uses it to coordinate AUM-aware instance weighting with difficulty-aware dynamic LoRA regularization. We also construct and release an expert-verified few-shot text classification benchmark for the steel industry, comprising 1,070 samples across seven fine-grained categories. Evaluations across six representative LLMs show that C-AUM achieves comparable or improved performance over vanilla SFT, with a maximum macro F1 gain of 1.99 percentage points. Our dataset and code are publicly available at https://github.com/pengdada98/C-AUM.

1 Introduction

Large language models (LLMs) have revolutionized the landscape of natural language processing, transforming traditional text classification from discriminative probability estimation to prompt-driven, autoregressive sequence generation (Chae and Davidson, 2026). While LLMs exhibit remarkable zero-shot and in-context learning capabilities on general-domain benchmarks, their direct application to highly specialized industrial domains presents significant challenges (Peng et al., 2025). In fields such as the steel industry, texts are characterized by idiosyncratic vocabularies, subtle semantic overlaps, and an extreme scarcity of expert-annotated data. Given that full-parameter updates on such limited data are highly prone to severe overfitting and catastrophic forgetting, parameter-efficient fine-tuning (PEFT) (Ding et al., 2023), particularly LoRA (Hu et al., 2022), has emerged as the standard paradigm to adapt LLMs to these specialized few-shot scenarios (Nwaiwu, 2025).

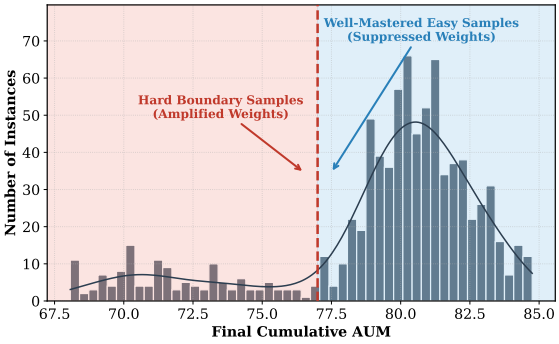


Figure 1: Motivation of C-AUM framework. The empirical distribution of the final cumulative AUM on the steel dataset exhibits a severe bimodal imbalance. Vanilla SFT suffers from gradient egalitarianism, overwhelmingly focusing on the massive cluster of well-mastered easy samples (blue region) while neglecting the crucial hard boundary samples (red region). Our framework aims to adaptively reshape this optimization landscape.

However, adapting LLMs via vanilla supervised fine-tuning (SFT) in few-shot settings exposes a critical optimization flaw rooted in the blind spot of gradient egalitarianism. Vanilla SFT weights training instances uniformly and applies the same token-level objective across all supervised target positions (Sow et al., 2025). In complex domain-specific classification, a dataset typically comprises a majority of straightforward, easy-to-learn samples and a minority of ambiguous, hard boundary samples that dictate the true semantic borders (Swayamdipta et al., 2020). As empirically revealed in Fig. 1, tracking the learning dynamics

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 32641–32653
October 24-29, 2026 ©2026 Association for Computational Linguistics

of an LLM exposes a stark bimodal distribution. Under gradient egalitarianism, the LLM rapidly overfits the massive cluster of easy samples, driving their loss to near zero, while leaving the crucial hard examples severely under-optimized.

Compounding this issue is the rigidity of static global regularization typically employed in standard PEFT paradigms. Applying static, uniform penalties indiscriminately suppresses the model's expressive capacity (Nwaiwu, 2025). In few-shot domain adaptation, this leads to a sub-optimal trade-off where aggressive static constraints hinder the model from capturing subtle semantic nuances in hard boundary examples, while insufficient constraints cannot effectively prevent severe overfitting on easy instances. Consequently, non-adaptive regularization fails to provide precise, capacity-aware optimization tailored to the varying learning difficulty of the training data.

To address this challenge, we introduce cumulative area under the margin (C-AUM), a dual-adaptive fine-tuning framework tailored to autoregressive LLMs. Building on the area under the margin (AUM) metric (Pleiss et al., 2020), C-AUM adapts margin tracking to generative classification by accumulating the decision margin at the final supervised label token as an instance-level measure of learning difficulty. This signal dynamically reshapes sample-level optimization by assigning greater relative weights to hard boundary samples than to well-mastered examples. C-AUM further adjusts LoRA regularization according to batch-level difficulty, mitigating overfitting while preserving sufficient capacity for difficult instances. To evaluate few-shot adaptation in a realistic industrial setting, we also construct and release an expert-verified text classification dataset for the steel industry.

In summary, the main contributions of this paper are three-fold:

- **Dataset and Benchmark:** We construct and release an expert-verified few-shot text classification dataset tailored for the steel industry. Comprising 1,070 samples across seven fine-grained categories, it provides a realistic and challenging benchmark for low-resource industrial LLM adaptation.
- **C-AUM Framework:** We propose C-AUM, a dual-adaptive fine-tuning framework tailored for autoregressive LLMs. By tracking the cumulative margin of the final decision token as an instance-level difficulty signal, C-AUM integrates an AUM-aware instance weighting mechanism to emphasize hard-boundary samples and a difficulty-aware dynamic LoRA regularization strategy for adaptive capacity control. Together, these components alleviate gradient egalitarianism and mitigate overfitting in few-shot domain adaptation.
- **Comprehensive Evaluation:** Experiments across six representative LLMs show that C-AUM achieves comparable or improved performance over vanilla SFT, with a maximum macro F1 gain of 1.99 percentage points. Ablation studies support the contribution of each core component, while experiments under severe data scarcity demonstrate the robustness of C-AUM in low-resource settings.

2 Related Work

In this section, we review the paradigm shift brought by LLMs in text classification and dissect the inherent limitations of current few-shot fine-tuning strategies when confronted with highly ambiguous, domain-specific data.

2.1 LLMs for Text Classification

The field of text classification has undergone a significant paradigm shift, evolving from discriminative models such as BERT (Devlin et al., 2019) and ERNIE (Sun et al., 2019) to generative LLMs represented by DeepSeek (Guo et al., 2025) and Qwen (Yang et al., 2025a). In contrast to traditional approaches that rely on task-specific classification heads to produce probability distributions, LLMs reformulate text classification as an autoregressive sequence-to-sequence generation problem, where categorical labels are directly decoded in a prompt-driven manner (Chae and Davidson, 2026).

While this generative paradigm exhibits impressive zero-shot and in-context learning capabilities on general-domain benchmarks, recent comprehensive evaluations reveal significant limitations in practical classification applications. Recent studies demonstrate that zero-shot LLMs often underperform fully supervised smaller models on complex, specialized classification tasks (Peng et al., 2025; Pecher et al., 2025). These vulnerabilities are dramatically exacerbated in specialized, domain-specific scenarios (e.g., industrial, scientific, or medical texts). Recent literature emphasizes that

general-purpose LLMs frequently falter when confronted with specialized vocabulary, unique grammatical structures, and highly imbalanced data distributions (Du et al., 2025; K. Rostam and Kertész, 2025; Sakai and Lam, 2026). In such domains, texts often share overlapping terminologies but differ subtly in their core semantic focus. Generative LLMs, pre-trained primarily to maximize overall token likelihood and fluency, tend to exploit surface-level textual clues rather than disentangling these fine-grained semantic boundaries. Consequently, to bridge this domain gap and enforce strict categorical discrimination, task-specific PEFT remains an indispensable requirement (Li et al., 2025; Cegin et al., 2025).

2.2 Few-Shot Fine-Tuning for LLMs

To adapt LLMs to downstream tasks with minimal computational overhead, PEFT methods, particularly LoRA (Hu et al., 2022), have become the widely used standard approach (Kumar et al., 2025). However, models trained with vanilla SFT in few-shot settings are highly susceptible to overfitting and catastrophic forgetting of instruction-following alignment (Cegin et al., 2025; Luo et al., 2025). Crucially, current vanilla SFT frameworks suffer from a gradient egalitarianism blind spot. They apply uniform optimization across all instances, treating straightforward samples and highly ambiguous boundary samples (hard examples) equally (Swayamdipta et al., 2020; Ju et al., 2024). In specialized few-shot datasets, where a small fraction of hard examples dictates the true semantic boundaries, vanilla SFT quickly overfits the majority of easy samples while leaving the ambiguous, hard examples under-optimized (Guda et al., 2025; Rahman and Caragea, 2025).

Furthermore, traditional static regularization strategies conventionally employed to combat overfitting are often sub-optimal for autoregressive LLMs in few-shot domain adaptation (Huang et al., 2025; He et al., 2025). These non-adaptive, global constraints impose uniform parameter penalties that indiscriminately suppress the model's expressive capacity. Consequently, they limit the LLM's ability to allocate sufficient representational power to resolve complex semantic nuances and ambiguous boundary examples, motivating the need for dynamic, capacity-aware regularization approaches.

There remains an urgent need for an adaptive, margin-driven fine-tuning mechanism that effectively emphasizes hard-boundary samples while preserving generative alignment. To bridge this gap, we propose C-AUM, a novel dual-adaptive fine-tuning framework tailored for autoregressive LLMs. By isolating and tracking the learning difficulty of the final decision token, C-AUM integrates an AUM-aware instance weighting mechanism to emphasize under-optimized hard-boundary samples, coupled with difficulty-aware dynamic LoRA regularization for adaptive parameter capacity control. Together, these components alleviate gradient egalitarianism and mitigate overfitting in few-shot domain adaptation.

3 Methodology

This section details the core components and mathematical formulation of the proposed C-AUM framework.

3.1 Overall Architecture

We consider few-shot text classification as a sequence generation task for autoregressive LLMs. Given a training set $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^{|\mathcal{S}|}$, where x_i denotes the input text and y_i denotes the target label sequence, the model generates y_i autoregressively conditioned on an instruction template $T(\cdot)$. To address the challenges of gradient egalitarianism and static over-regularization in this setting, C-AUM introduces a dual-adaptive optimization framework, as illustrated in Fig. 2. The framework consists of three coordinated modules: (1) *Autoregressive-Tailored AUM Tracking* for dynamic difficulty estimation, (2) *AUM-Aware Instance Weighting* for adaptive instance-level optimization, and (3) *Difficulty-Aware Dynamic LoRA Regularization* for adaptive parameter capacity control. Together, these modules dynamically adjust both instance-level optimization emphasis and parameter-level regularization according to the evolving learning difficulty of each instance.

3.2 Autoregressive-Tailored AUM Tracking

Unlike discriminative classifiers that derive margin statistics from a dedicated classification head, autoregressive LLMs decode sequences across multiple generative steps. For multi-token label sequences, aggregating margins across all generated positions can introduce irrelevant variation from shared prefixes and non-decisive label tokens. To obtain a cleaner difficulty metric, we isolate the decision margin exclusively at the final supervised label token, corresponding to the final active target

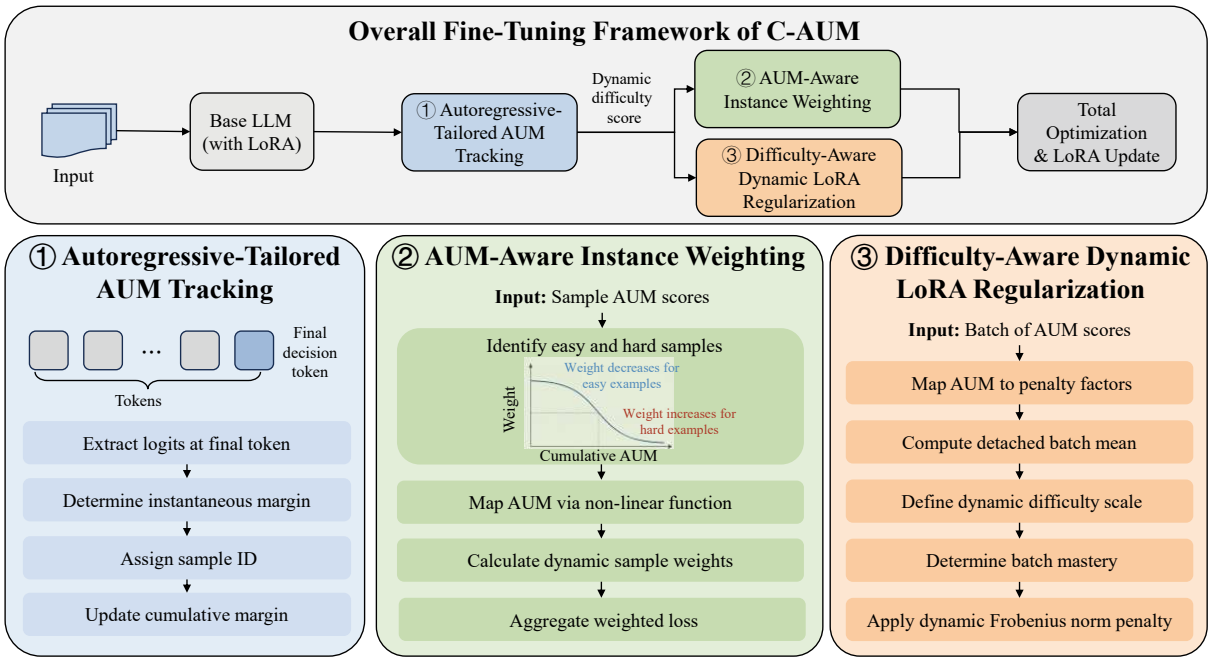


Figure 2: The overall framework of C-AUM.

position t_{last}.

Formally, for a given training instance x_i with a unique identifier s_i, let $z_y^{(t_{\text{last}})}$ denote the logit corresponding to the ground-truth final token, and $\max_{j \neq y} z_j^{(t_{\text{last}})}$ represent the maximum logit among all competing vocabulary tokens at that position. The instantaneous decision margin $m_i^{(t)}$ is defined as:

$$m_i^{(t)} = z_y^{(t_{\text{last}})} - \max_{j \neq y} z_j^{(t_{\text{last}})}. \tag{1}$$

To track the learning dynamics without introducing additional gradient paths, the instantaneous margin is detached from the computation graph and accumulated whenever the corresponding instance is encountered during training:

$$A_i^{(t)} = A_i^{(t-1)} + m_i^{(t)}, \tag{2}$$

where $A_i^{(0)} = 0$. A consistently large positive $A_i^{(t)}$ indicates that an instance is being readily mastered, whereas a negative or near-zero value suggests an ambiguous boundary instance that remains under-optimized.

3.3 AUM-Aware Instance Weighting

Vanilla SFT optimizes all training instances uniformly, leading to rapid overfitting on easy samples while leaving complex boundary instances under-optimized. To resolve this gradient egalitarianism, we introduce an instance-level weighting module driven by the historical margin metric $A_i^{(t)}$.

For the i-th instance in a batch, its adaptive weight w_i is computed via a bounded sigmoid formulation:

$$w_i = 1.0 + \gamma \cdot \sigma\left(-\frac{A_i^{(t)}}{\tau_w}\right), \tag{3}$$

where $\sigma(\cdot)$ is the sigmoid function, τ_w is the weighting temperature controlling transition smoothness, and γ is an amplitude scaling factor (empirically set to $\tau_w = 10.0$ and $\gamma = 0.5$).

The sample-level cross-entropy loss $\mathcal{L}_{\text{CE}}^{(i)}$ is computed as the mean token-level loss across all active target positions. The weighted base objective for a batch of size B is then aggregated as:

$$\mathcal{L}_{\text{batch}} = \frac{1}{B} \sum_{i=1}^{B} w_i \cdot \mathcal{L}_{\text{CE}}^{(i)}. \tag{4}$$

As the historical margin scores evolve during training, w_i automatically assigns greater optimization emphasis to under-optimized boundary instances, providing adaptive instance-level optimization.

3.4 Difficulty-Aware Dynamic LoRA Regularization

To mitigate overfitting in few-shot regimes without overly restricting parameter updates on complex instances, we introduce an adaptive parameter regularization module that scales penalties based on the collective learning state of the batch.

For each instance x_i, its cumulative score $A_i^{(t)}$ is first mapped to a mastery factor f_i:

$$f_i = \sigma\left(\frac{A_i^{(t)}}{\tau_r}\right), \tag{5}$$

where τ_r is the regularization temperature (set to $\tau_r = 5.0$). We define the batch-level mastery scale S as the detached mean of these individual factors:

$$S = \text{sg}\left(\frac{1}{B}\sum_{i=1}^{B} f_i\right), \tag{6}$$

where $\text{sg}(\cdot)$ denotes the stop-gradient operator.

Finally, S dynamically scales the Frobenius norm penalty across all low-rank adapter parameter matrices Θ_{LoRA}. The overall training objective is formulated as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{batch}} + \alpha \cdot S \cdot \left(\frac{1}{|K|}\sum_{W \in \Theta_{\text{LoRA}}} ||W||_F\right), \tag{7}$$

where $\alpha = 10^{-4}$ represents the base penalty coefficient, $|K|$ is the total count of adapted LoRA parameter tensors, and $\|W\|_F = \sqrt{\sum_{j,k} W_{j,k}^2}$ denotes the Frobenius norm.

When a batch comprises predominantly challenging boundary examples, S adaptively decreases to relax parameter constraints and facilitate feature learning. Conversely, when processing well-mastered data, stronger parameter penalties are enforced to prevent over-adaptation on few-shot samples.

4 Experiments

In this section, we conduct experiments to evaluate the effectiveness, component contributions, and robustness of the proposed framework.

4.1 Steel Dataset

The steel dataset is an expert-verified, domain-specific classification benchmark comprising 1,070 samples across seven specialized industrial categories: grade, variety, standard, defect, origin, application, and performance. To alleviate severe data scarcity in specialized industrial sectors, initial candidate texts were generated using advanced LLMs (Gemini and ChatGPT) conditioned on structural seed entities. Because raw LLM outputs often exhibit repetitive syntactic patterns and limited domain realism, three steel-domain experts conducted multiple rounds of manual revision, filtering, and standardization. Consequently, a portion of the raw LLM-generated samples was retained, while the majority underwent substantial refinement to ensure consistency with authentic metallurgical terminology and real-world industrial logic. In accordance with standard few-shot classification conventions, each category contains fewer than 200 instances, establishing a realistic low-resource regime. To facilitate robust evaluation, we adopt a five-fold cross-validation scheme, where each fold strictly maintains an 8:2 split (856 training samples and 214 test samples per fold). The category distribution is illustrated in Fig. 3, and the comprehensive curation pipeline is detailed in Appendix A.

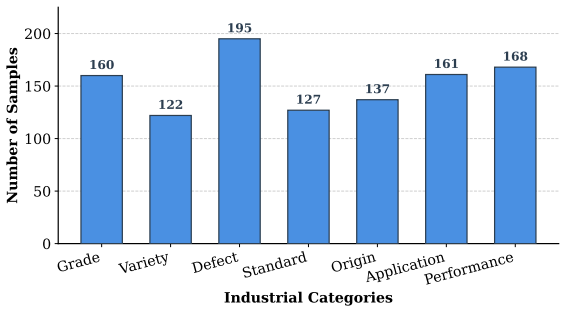


Figure 3: Data distribution of the constructed steel dataset.

4.2 Evaluation Metrics

To comprehensively assess the performance of the text classification model, we employ three macro-level evaluation metrics: macro precision, macro recall, and macro F1. The specific definitions and calculation methods of these three metrics are provided in Appendix B.

4.3 Experimental Setup

All experiments on text classification are conducted using Python 3.10.18, along with Transformers version 4.52.4 and PyTorch version 2.7.1. The code is built upon the LLaMA-Factory framework (Zheng et al., 2024) and runs on a single RTX 6000 GPU featuring 24GB of memory. Furthermore, the detailed hyperparameter configurations and a comprehensive sensitivity analysis of our C-AUM framework are provided in Appendix C.

Model	Macro Precision (%)	Macro Recall (%)	Macro F1 (%)
DeepSeek-R1-Distill-Qwen-1.5B (Guo et al., 2025)	89.75 ± 1.47	90.09 ± 1.37	89.74 ± 1.45
+ C-AUM	$\mathbf{90.90 \pm 1.91}$	$\mathbf{91.17 \pm 1.44}$	$\mathbf{90.86 \pm 1.75}$
Granite-3.3-2B-Instruct (Granite Team, 2025)	90.64 ± 2.07	90.10 ± 2.60	90.11 ± 2.45
+ C-AUM	$\mathbf{91.28 \pm 1.52}$	$\mathbf{91.15 \pm 1.18}$	$\mathbf{91.02 \pm 1.37}$
Gemma-2-2B-it (Riviere et al., 2024)	92.73 ± 1.20	$\mathbf{92.95 \pm 1.75}$	92.62 ± 1.47
+ C-AUM	$\mathbf{92.96 \pm 1.84}$	92.64 ± 2.36	$\mathbf{92.63 \pm 2.19}$
Qwen2.5-3B-Instruct (Yang et al., 2025b)	92.56 ± 1.48	92.64 ± 1.40	92.38 ± 1.47
+ C-AUM	$\mathbf{94.34 \pm 1.12}$	$\mathbf{94.71 \pm 0.57}$	$\mathbf{94.37 \pm 0.81}$
Ministral-3-3B-Instruct (Liu et al., 2026)	93.61 ± 2.17	93.35 ± 1.35	93.31 ± 1.78
+ C-AUM	$\mathbf{94.27 \pm 1.52}$	$\mathbf{94.31 \pm 1.77}$	$\mathbf{94.16 \pm 1.79}$
Qwen3-4B-Instruct (Yang et al., 2025a)	93.93 ± 1.06	93.91 ± 1.13	93.73 ± 1.01
+ C-AUM	$\mathbf{94.17 \pm 1.33}$	$\mathbf{94.02 \pm 1.40}$	$\mathbf{93.86 \pm 1.22}$

Table 1: Main classification performance across various representative LLMs.

Variant	Macro Precision (%)	Macro Recall (%)	Macro F1 (%)
C-AUM	$\mathbf{94.34 \pm 1.12}$	$\mathbf{94.71 \pm 0.57}$	$\mathbf{94.37 \pm 0.81}$
w/o Difficulty-Aware Dynamic LoRA Regularization	92.75 ± 1.05	93.22 ± 1.13	92.73 ± 0.98
w/o AUM-Aware Instance Weighting	93.26 ± 1.47	93.31 ± 1.40	93.09 ± 1.41
w/o Final-token AUM	94.19 ± 0.69	94.57 ± 0.52	94.21 ± 0.61
Baseline (Vanilla SFT)	92.56 ± 1.48	92.64 ± 1.40	92.38 ± 1.47

Table 2: Ablation study results evaluating the core modules of the C-AUM framework.

4.4 Main Results

We evaluate C-AUM against vanilla SFT across six representative publicly available LLMs ranging from 1.5B to 4B parameters. Table 1 reports the five-fold cross-validation results on the steel dataset as mean $\pm$ sample standard deviation. Overall, C-AUM achieves comparable or improved performance across all evaluated backbones, although the gains vary by model. Among the 1.5B–2B models, C-AUM improves macro F1 by 1.12 and 0.91 percentage points for DeepSeek-R1-Distill-Qwen-1.5B and Granite-3.3-2B-Instruct, respectively, while Gemma-2-2B-it remains comparable to vanilla SFT, with a difference of 0.01 percentage points. For Granite-3.3-2B-Instruct, the standard deviation decreases from 2.45 to 1.37, indicating lower performance variation across folds. Among the 3B models, C-AUM improves macro F1 by 1.99 percentage points for Qwen2.5-3B-Instruct and 0.85 percentage points for Ministral-3-3B-Instruct. The improvement on Qwen3-4B-Instruct is smaller at 0.13 percentage points. Qwen2.5-3B-Instruct with C-AUM achieves the highest macro F1 among all evaluated configurations at $94.37 \pm 0.81\%$, exceeding its vanilla SFT counterpart by 1.99 percentage points. On this backbone, C-AUM also improves macro precision and macro recall by 1.78 and 2.07 percentage points, respectively, indicating gains across all three evaluation metrics. The results across different model scales suggest that C-AUM is compatible with diverse autoregressive LLM backbones, although the magnitude of its effect remains model-dependent.

4.5 Ablation Study

To evaluate the contribution of each component of C-AUM, we conduct an ablation study on the steel dataset using five-fold cross-validation with Qwen2.5-3B-Instruct. Table 2 reports macro precision, macro recall, and macro F1 as mean $\pm$ sample standard deviation. Removing difficulty-aware dynamic LoRA regularization reduces macro F1 by 1.64 percentage points, from $94.37 \pm 0.81\%$ to $92.73 \pm 0.98\%$, supporting the contribution of difficulty-aware regularization in the few-shot setting. Removing AUM-aware instance weighting decreases macro F1 by 1.28 percentage points to $93.09 \pm 1.41\%$ and increases cross-fold variation. This result supports the effectiveness of using cumulative AUM to guide sample weighting. Finally, replacing final-token AUM with mean-token AUM slightly reduces macro F1 by 0.16 percentage points to $94.21 \pm 0.61\%$. This result suggests that the final-token margin is a suitable difficulty

Framework Variant	Difficulty Signal	Macro Precision (%)	Macro Recall (%)	Macro F1 (%)
C-AUM	**Final-token AUM**	$\mathbf{94.34 \pm 1.12}$	$\mathbf{94.71 \pm 0.57}$	$\mathbf{94.37 \pm 0.81}$
w/o Final-token AUM	Mean-token AUM	94.19 ± 0.69	94.57 ± 0.52	94.21 ± 0.61
w/o Final-token AUM	Instantaneous Loss	92.57 ± 1.47	92.79 ± 1.09	92.45 ± 1.33
w/o Final-token AUM	Prediction Entropy	93.29 ± 1.53	93.56 ± 1.03	93.22 ± 1.31

Table 3: Ablation study evaluating different difficulty estimation signals.

Method	Macro Precision (%)	Macro Recall (%)	Macro F1 (%)
Loss-based reweighting	93.89 ± 1.25	93.94 ± 1.10	93.71 ± 1.12
Entropy-based reweighting	92.90 ± 0.78	93.11 ± 1.11	92.83 ± 0.90
Focal loss	92.62 ± 1.48	92.63 ± 0.71	92.36 ± 1.01
Static LoRA regularization	92.54 ± 2.24	92.59 ± 1.92	92.27 ± 2.05
C-AUM	$\mathbf{94.34 \pm 1.12}$	$\mathbf{94.71 \pm 0.57}$	$\mathbf{94.37 \pm 0.81}$

Table 4: Performance comparison against representative sample reweighting and regularization baselines.

signal for autoregressive classification. It is also computationally simpler because it avoids margin aggregation across multiple supervised label positions.

4.6 Analysis of Difficulty Estimation Signals

To assess the effectiveness of final-token AUM, Table 3 compares it with alternative difficulty signals, including mean-token AUM, instantaneous loss, and prediction entropy, using five-fold cross-validation. In this controlled ablation, the sample reweighting and dynamic LoRA regularization modules are retained, and only the difficulty signal driving them is replaced. This differs from the standalone loss- and entropy-based reweighting baselines in Table 4, which use these signals only for sample weighting. Replacing final-token AUM with instantaneous loss or prediction entropy reduces macro F1 by 1.92 and 1.15 percentage points, respectively, resulting in scores of $92.45 \pm 1.33\%$ and $93.22 \pm 1.31\%$. Both alternatives also exhibit greater cross-fold variation than final-token AUM. These results suggest that cumulative margin tracking provides a more stable difficulty estimate than single-step signals by capturing learning dynamics over time. Mean-token AUM achieves $94.21 \pm 0.61\%$, representing a smaller decrease of 0.16 percentage points. This result suggests that focusing on the final decision token is at least as effective as aggregating margins across all supervised label tokens, while requiring less margin aggregation. Overall, final-token AUM provides an effective and computationally simple difficulty signal for autoregressive LLM classification.

4.7 Comparison with Representative Baselines

To examine whether C-AUM improves upon stronger alternatives beyond vanilla SFT, Table 4 presents a five-fold cross-validation comparison with representative reweighting and regularization baselines, including focal loss (Lin et al., 2017), loss-based reweighting, entropy-based reweighting, and static LoRA regularization (Hu et al., 2022). C-AUM achieves the highest average performance across all evaluation metrics, with a macro F1 of $94.37 \pm 0.81\%$. Its macro F1 exceeds those of loss-based reweighting, entropy-based reweighting, and focal loss by 0.66, 1.54, and 2.01 percentage points, respectively. Unlike loss- and entropy-based signals that reflect the current model state, cumulative AUM captures training dynamics across optimization steps and may therefore provide a more stable estimate of sample difficulty. Static LoRA regularization achieves a lower macro F1 of $92.27 \pm 2.05\%$ and greater cross-fold variation. These results support the contributions of cumulative AUM-based reweighting and difficulty-aware dynamic regularization.

4.8 Fine-Grained Class-Level Analysis

To examine whether C-AUM benefits individual categories, Table 5 reports per-class F1 scores on fold 1. C-AUM improves six of the seven categories, with the largest gain on variety (4.19 percentage points, from 78.57% to 82.76%), which has the lowest baseline F1. The gains in the remaining categories are 2.97, 2.94, 2.03, 1.88, and 1.80 percentage points for defect, application, origin, standard, and performance, respectively. Perfor-

Category	Vanilla SFT (%)	C-AUM (%)	Δ F1
Performance	95.08	96.88	+1.80
Defect	97.03	100.00	+2.97
Origin	93.62	95.65	+2.03
Variety	78.57	82.76	+4.19
Grade	97.22	95.77	−1.45
Standard	94.12	96.00	+1.88
Application	94.12	97.06	+2.94

Table 5: Per-class F1 comparison between vanilla SFT and C-AUM (fold 1 evaluation).

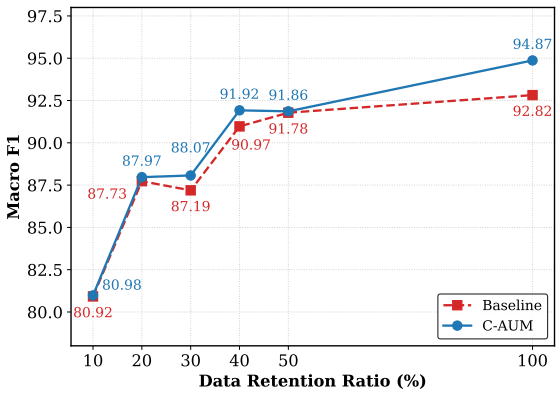


Figure 4: Macro F1 performance across varying data retention ratios (fold 1 evaluation).

mance on grade decreases slightly by 1.45 percentage points, although both methods retain relatively high F1 scores for this category. These gains span multiple types of steel-related information, providing a more fine-grained view of the improvements summarized by macro F1. Despite the modest decrease on grade, the overall results show that C-AUM benefits most categories, with the most pronounced improvement observed for variety.

4.9 Resilience to Extreme Data Scarcity

Vanilla LLM fine-tuning is prone to overfitting when training data is severely constrained. To evaluate C-AUM under severe low-resource conditions, we conduct experiments on fold 1 of the steel dataset and simulate varying levels of data scarcity by randomly retaining 10% to 50% of the original training set. The results are shown in Fig. 4. Across all retention ratios, C-AUM achieves higher macro F1 than vanilla SFT. As the amount of training data decreases, both methods experience varying degrees of performance degradation, particularly when only 10% of the data is retained. Nevertheless, C-AUM maintains a relatively stable advantage across the examined retention ratios, suggesting that its benefits are not limited to training with the complete dataset. Notably, when trained on only 40% of the available data, C-AUM achieves a macro F1 score of 91.92%, compared with 91.78% for vanilla SFT trained on a larger 50% subset. This result suggests that difficulty-aware optimization can improve the use of limited supervision, although it cannot fully offset the performance decline under extreme data scarcity. Overall, C-AUM demonstrates relatively robust adaptation under constrained training-data settings.

4.10 Computational Efficiency Analysis

To quantify the computational overhead of C-AUM, we compare its resource consumption with vanilla SFT using Qwen2.5-3B-Instruct on the steel dataset. The results in terms of training time, inference time, and GPU memory usage are summarized in Fig. 5. C-AUM introduces a small training-time overhead and a moderate increase in training memory, while requiring no additional computation during inference. The training time increases from 1501.84 s for vanilla SFT to 1538.22 s with C-AUM, corresponding to a 2.42% increase. Meanwhile, the inference time remains nearly unchanged, with 45.18 s for vanilla SFT and 45.14 s for C-AUM. This low overhead stems from C-AUM being implemented within the standard training loop without introducing an additional forward pass for difficulty estimation. C-AUM incurs additional GPU memory usage during training due to the extra computations associated with dynamic LoRA regularization, while the AUM tracker maintains only detached scalar statistics. The peak training VRAM increases from 14,434 MB to 16,696 MB, corresponding to a 15.67% increase. In contrast, inference memory remains nearly identical, at 7,098 MB for both vanilla SFT and C-AUM. Overall, C-AUM provides competitive classification performance with modest training overhead and no additional inference computation, demonstrating its practical viability for low-resource industrial deployment.

5 Conclusion

In this paper, we present C-AUM, a dual-adaptive fine-tuning framework designed to address optimization imbalances in few-shot LLM adaptation. By tracking learning difficulty through cumulative margins at the final decision token, C-AUM integrates AUM-aware instance weighting for adaptive instance-level optimization and difficulty-aware dy-

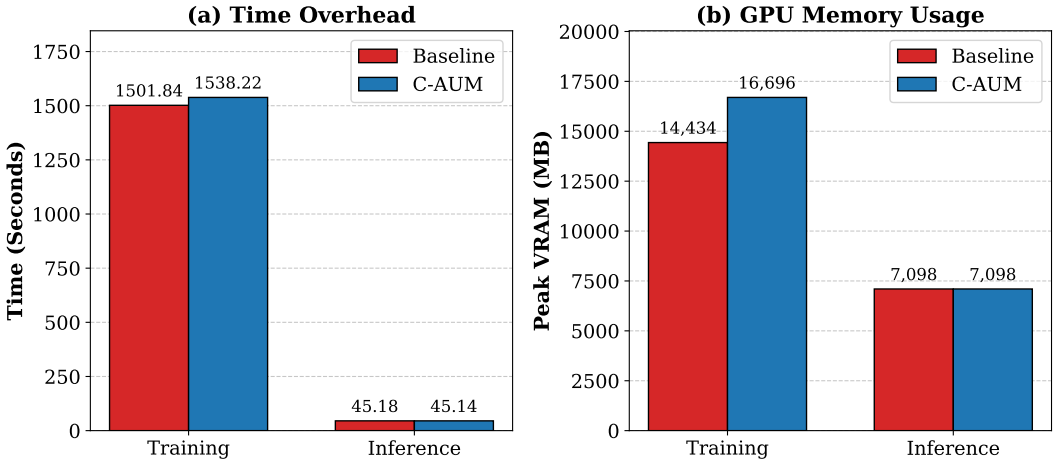


Figure 5: Comparative analysis of computational efficiency between the vanilla SFT baseline and the proposed C-AUM framework (fold 1 evaluation).

namic LoRA regularization for adaptive parameter capacity control. Evaluations on our constructed steel industry benchmark show that C-AUM achieves competitive or improved performance across multiple publicly available LLM backbones, while introducing only modest training overhead and no additional computation during inference. Although the magnitude of improvement varies across base models, our findings demonstrate the potential of adaptive difficulty tracking and capacity control for specialized low-resource text classification.

Limitations

Despite the encouraging performance gains achieved by C-AUM, several limitations warrant further investigation.

First, our evaluation is limited to a single Chinese steel-domain dataset. Therefore, the generalizability of C-AUM across other domains and languages remains unverified. Future work could evaluate the framework on datasets from additional domains and languages to determine whether domain- or language-specific adaptation is required.

Second, the effectiveness of C-AUM varies across LLM backbones. For example, C-AUM improves macro F1 by nearly 2 percentage points on Qwen2.5-3B-Instruct, whereas its performance on Gemma-2-2B-it is comparable to vanilla SFT. This variation may be associated with differences in pre-training data, model alignment, tokenization, or model capacity. Further analysis is needed to clarify these interactions and establish broader guidelines across model families.

Third, C-AUM's performance declines when the available training data becomes extremely limited. Under the 10% retention setting, both C-AUM and vanilla SFT exhibit a noticeable performance decrease, indicating that adaptive optimization cannot fully compensate for insufficient supervision. Future work could incorporate data augmentation, sample selection, or external knowledge to improve performance under extreme low-resource conditions.

Fourth, C-AUM uses persistently low cumulative AUM as a proxy for hard-boundary samples. Although low AUM may reflect informative boundary ambiguity, it can also arise from annotation noise, labeling errors, or intrinsically ambiguous examples. In our steel dataset, multiple rounds of annotation and review by three steel-domain experts reduce the likelihood of severe label noise, while bounded sample weights prevent persistently low-AUM examples from dominating training. Nevertheless, C-AUM does not explicitly distinguish among these sources of persistent model difficulty. Future work could integrate noise-aware methods to better separate informative boundary samples from potentially mislabeled examples.

Acknowledgments

This work was supported in part by the Beijing Natural Science Foundation under Grant L2601018 and in part by the National Key Research and Development Program of China under Grant 2024YFC3017004.

References

Ján Cegin, Jakub Simko, and Peter Brusilovsky. 2025. LLMs vs established text augmentation techniques for classification: When do the benefits outweight the costs? In *Proceedings of the 18th Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 10476–10496, Albuquerque, New Mexico, USA. Association for Computational Linguistics.

Youngjin Chae and Thomas Davidson. 2026. Large language models for text classification: From zero-shot learning to instruction-tuning. *Sociological Methods & Research*, 55(2):501–567.

Jacob Devlin, Mingwei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186, Minneapolis, USA. Association for Computational Linguistics.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, Jing Yi, Weilin Zhao, Xiaozhi Wang, Zhiyuan Liu, HaiTao Zheng, Jianfei Chen, Yang Liu, Jie Tang, Juanzi Li, and Maosong Sun. 2023. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235.

Huaming Du, Lei Yuan, Cancan Feng, Guisong Liu, Gang Kou, and Carl Yang. 2025. Explainable text classification with LLMs: Enhancing performance through dialectical prompting and explanation-guided training. In *Findings of the Association for Computational Linguistics*, pages 12800–12816, Suzhou, China. Association for Computational Linguistics.

Granite Team. 2025. Granite-3.3-2b-instruct.

Blessed Guda, Lawrence Francis, Gabrial Zencha Ashungafac, Carlee Joe-Wong, and Moise Busogi. 2025. Quantifying and mitigating selection bias in LLMs: A transferable LoRA fine-tuning and efficient majority voting approach. 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 2019–2038, Mumbai, India. The Asian Federation of Natural Language Processing and The Association for Computational Linguistics.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z.F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 180 others. 2025. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645:633–638.

Di He, Songjun Tu, Ajay Jaiswal, Li Shen, Ganzhao Yuan, Shiwei Liu, and Lu Yin. 2025. AlphaDecay: Module-wise weight decay for heavy-tailed balancing in LLMs. In *Proceedings of the 39th Annual Conference on Neural Information Processing Systems*, volume 38, pages 29745–29768, San Diego, USA.

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 *Proceedings of the 10th International Conference on Learning Representations*, Online. OpenReview.net.

Jerry Huang, Peng Lu, and Qiuhao Zeng. 2025. Calibrated language models and how to find them with label smoothing. In *Proceedings of the 42nd International Conference on Machine Learning*, pages 25593–25612, Vancouver, Canada. PMLR.

Yiming Ju, Ziyi Ni, Xingrun Xing, Zhixiong Zeng, Hanyu Zhao, Siqi Fan, and Zheng Zhang. 2024. Mitigating training imbalance in LLM fine-tuning via selective parameter merging. In *Proceedings of the 29th Conference on Empirical Methods in Natural Language Processing*, pages 15952–15959, Miami, USA. Association for Computational Linguistics.

Zhyar Rzgar K. Rostam and Gábor Kertész. 2025. Advances in pre-trained language models for domain-specific text classification: A systematic review. *ACM Transactions on Intelligent Systems and Technology*, 16(6):1–41.

Nalin Kumar, Mateusz Lango, and Ondřej Dušek. 2025. Pretraining language models with LoRA and artificial languages. In *Proceedings of the 1st BabyLM Workshop*, pages 525–530, Suzhou, China. Association for Computational Linguistics.

Zhangming Li, Qinghao Hu, Yiqun Chen, Peisong Wang, Yifan Zhang, and Jian Cheng. 2025. LoRaDA: Low-rank direct attention adaptation for efficient LLM fine-tuning. In *Findings of the Association for Computational Linguistics*, pages 12638–12655, Suzhou, China. Association for Computational Linguistics.

Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2999–3007, Venice, Italy. IEEE Computer Society.

Alexander H. Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, Abhinav Rastogi, Adrien Sadé, Alan Jeffares, Albert Jiang, Alexandre Cahill, Alexandre Gavaudan, Alexandre Sablayrolles, Amélie Héliou, Amos You, Andy Ehrenberg, Andy Lo, Anton Eliseev, Antonia Calvi, Avinash Sooriyarachchi, Baptiste Bout, and 101 others. 2026. Ministral 3. *Preprint*, arXiv:2601.08584.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. 2025. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *IEEE Transactions on Audio, Speech and Language Processing*, 33:3776–3786.

Steve Nwaiwu. 2025. Parameter-efficient fine-tuning for low-resource text classification: a comparative study of LoRA, IA^3, and ReFT. *Frontiers in Big Data*, 8:1677331.

Branislav Pecher, Ivan Srba, and Mária Bieliková. 2025. Comparing specialised small and general large language models on text classification: 100 labelled samples to achieve break-even performance. In *Proceedings of the 30th Conference on Empirical Methods in Natural Language Processing*, pages 165–184, Suzhou, China. Association for Computational Linguistics.

Qiaojuan Peng, Xiong Luo, Yuqi Yuan, Fengbo Gu, Hailun Shen, and Ziyang Huang. 2025. Text classification methods in the steel e-commerce industry using the large language model and prompt engineering. *International Journal of Machine Learning and Cybernetics*, 16(11):9399–9412.

Geoff Pleiss, Tianyi Zhang, Ethan R. Elenberg, and Kilian Q. Weinberger. 2020. Identifying mislabeled data using the area under the margin ranking. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 17044–17056, Online.

Md Mezbaur Rahman and Cornelia Caragea. 2025. LLM-guided co-training for text classification. In *Proceedings of the 30th Conference on Empirical Methods in Natural Language Processing*, pages 31104–31121, Suzhou, China. Association for Computational Linguistics.

Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, and 178 others. 2024. Gemma 2: Improving open language models at a practical size. *Preprint*, arXiv:2408.00118.

Hajar Sakai and Sarah S Lam. 2026. Large language models for health care text classification: Systematic review. *JMIR AI*, 5(1):e79202.

Daouda Sow, Herbert Woisetschläger, Saikiran Bulusu, Shiqiang Wang, Hans-Arno Jacobsen, and Yingbin Liang. 2025. Dynamic loss-based sample reweighting for improved large language model pretraining. In *Proceedings of the 13th International Conference on Learning Representations*, Singapore. OpenReview.net.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. ERNIE: Enhanced representation through knowledge integration. *Preprint*, arXiv:1904.09223.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 25th Conference on Empirical Methods in Natural Language Processing*, pages 9275–9293, Online. Association for Computational Linguistics.

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

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, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 23 others. 2025b. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. 2024. LlamaFactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 400–410, Bangkok, Thailand. Association for Computational Linguistics.

A Details of the Steel Dataset Construction

To ensure high transparency and domain realism, this section details the curation and verification pipeline of our steel industry benchmark. The entire construction process was supervised and executed by three steel-domain experts across three sequential phases.

A.1 Taxonomy Definition

The taxonomy was designed by steel industry experts to represent common linguistic patterns observed in industrial transactions, product inquiries, and quality inspections. The resulting taxonomy consists of seven fine-grained categories: grade, variety, standard, defect, origin, application, and performance.

A.2 Seed Collection and LLM-Assisted Candidate Generation

We adopted a hybrid curation strategy to bootstrap candidate samples while addressing the scarcity of open-source industrial corpora:

- **Seed Entity Extraction:** Core domain entities, such as standardized grade designations, execution standards, production defects, and localized mill origins, were collected from authoritative industrial registries and publicly available mill catalogs to provide structured seed information for candidate generation.

- **Prompt-Driven Generation:** These structural entities were incorporated into prompt templates for advanced LLMs (ChatGPT and Gemini) to generate candidate text descriptions. The prompt template is formulated as follows:

> *You are an experienced text classification expert. Please generate n diverse sentences based on the following professional domain information. Ensure that the linguistic expressions are highly varied and accurately reflect real industrial scenarios.*
> *Domain Entity: e*

  where e denotes the input domain entity.

A.3 Expert Verification, Filtering, and Standardization

To ensure high fidelity to real-world industrial interactions, three steel-domain experts conducted iterative rounds of review and revision adhering to three strict criteria:

- **Domain Authenticity and Correctness:** Eliminating hallucinated technical terms, incorrect grade-property associations, and unnatural colloquialisms to ensure strict adherence to industrial standards.

- **Semantic Boundary Clarity:** Resolving ambiguous cases where descriptions spanned multiple candidate categories, ensuring each retained instance possesses an unambiguous ground-truth label.

- **Linguistic Diversity and Realism:** Manually rewriting repetitive syntactic templates into diverse real-world communication styles.

Through this human-in-the-loop pipeline, only a relatively small fraction of raw LLM outputs was kept unmodified. The vast majority of samples underwent extensive expert rewriting, filtering, and standardization. Finally, to preserve the low-resource evaluation setting, each category was limited to fewer than 200 instances, yielding 1,070 curated samples in total.

B Definition of Evaluation Metrics

The following outlines the definitions and computational methods for macro precision, macro recall, and macro F1.

Macro Precision: It measures how well the model correctly predicts each class by calculating the precision for every individual label and then taking the average across all labels. It ensures a fair evaluation for all categories, including those with fewer samples. The computation is defined in (8) and (9).

$$\text{Precision}_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FP}_i} \tag{8}$$

$$\text{Macro Precision} = \frac{\sum_{i=1}^{C} \text{Precision}_i}{C} \tag{9}$$

where C is the total number of classes, TP_i denotes the number of samples correctly classified as label i, FP_i is the count of samples wrongly predicted as label i, and Precision_i represents the precision for class i.

Macro Recall: It assesses the model's ability to identify samples from each class by computing recall for each label and then averaging over all labels. It reflects the model's effectiveness in capturing relevant instances from every category, giving equal weight to rare classes. The calculation is provided in (10) and (11).

$$\text{Recall}_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FN}_i} \tag{10}$$

$$\text{Macro Recall} = \frac{\sum_{i=1}^{C} \text{Recall}_i}{C} \tag{11}$$

where FN_i indicates the count of label i samples that the model fails to recognize, and Recall_i is the recall for class i.

Macro F1: It provides a unified metric that balances precision and recall, giving an overall evaluation of the model's performance across all classes. The formula is given in (12).

$$\text{Macro F1} = \frac{\sum_{i=1}^{C} \frac{2 \times \text{Precision}_i \times \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i}}{C} \tag{12}$$

C Implementation Details and Sensitivity Analysis

To support reproducibility, this section provides the exact instruction template, comprehensive training hyperparameters, and an in-depth sensitivity analysis of our proposed C-AUM framework.

Hyperparameter	Value	Meaning
Training Epoch	5	Number of times the model iterates over the full training dataset.
Learning Rate	2e-4	Controls the step magnitude for updating model parameters during gradient descent.
Batch Size	8	Number of samples fed into the model at once to calculate gradients per iteration.
Gradient Accumulation Step	2	Number of forward/backward passes to accumulate gradients before performing a parameter update, effectively simulating a larger batch size.
Max Input Length	256	Maximum token length allowed for input sequences.
LoRA Rank (r)	8	The rank of the low-rank adaptation matrices, governing the trade-off between parameter efficiency and representation capacity.

Table 6: Key general hyperparameter settings employed for the text classification experiments.

Target Configuration	τ_w	γ	α	τ_r	Macro Precision (%)	Macro Recall (%)	Macro F1 (%)
Baseline (Vanilla SFT)	-	-	-	-	93.20	92.79	92.82
Test τ_w (Extremely Steep)	2.0	0.5	10^{-4}	5.0	92.70	93.67	93.09
Test τ_w (Moderately Steep)	5.0	0.5	10^{-4}	5.0	93.70	94.11	93.80
Test τ_w (Moderately Smooth)	15.0	0.5	10^{-4}	5.0	95.06	94.93	94.93
Test τ_w (Extremely Smooth)	20.0	0.5	10^{-4}	5.0	94.29	93.97	94.01
Test γ (Negligible Amplitude)	10.0	0.1	10^{-4}	5.0	93.96	94.49	94.16
Test γ (Low Amplitude)	10.0	0.3	10^{-4}	5.0	95.43	94.93	95.07
Test γ (High Amplitude)	10.0	0.8	10^{-4}	5.0	93.55	94.77	93.99
Test γ (Extremely High Amplitude)	10.0	1.0	10^{-4}	5.0	94.49	94.55	94.44
Test α (Very Weak Regularization)	10.0	0.5	10^{-5}	5.0	94.73	95.50	95.01
Test α (Weak Regularization)	10.0	0.5	10^{-4}	5.0	94.93	94.97	94.87
Test α (Strong Regularization)	10.0	0.5	5×10^{-4}	5.0	94.91	94.65	94.64
Test α (Very Strong Regularization)	10.0	0.5	10^{-2}	5.0	93.90	94.16	93.87
Test τ_r (Highly Sensitive)	10.0	0.5	10^{-4}	1.0	94.07	94.49	94.24
Test τ_r (Moderately Sensitive)	10.0	0.5	10^{-4}	3.0	93.74	93.86	93.60
Test τ_r (Moderately Forgiving)	10.0	0.5	10^{-4}	7.0	93.55	94.62	93.97
Test τ_r (Highly Forgiving)	10.0	0.5	10^{-4}	10.0	93.36	94.62	93.83

Table 7: Hyperparameter sensitivity analysis (fold 1 evaluation).

C.1 Instruction Template

For all text classification tasks in our experiments, we format the input data using a fixed, concise instruction template to standardize the generative behavior of the LLM. The exact template is as follows:

> *You are an experienced text classification expert. Please classify the given domain text. The text category must be selected from the given options.*
> *Given options:* $\mathcal{Y}$
> *Domain text:* x_i

where $\mathcal{Y}$ denotes the predefined label space comprising all candidate categories, and x_i represents the specific domain text instance to be classified.

C.2 Hyperparameter Configurations

The general training hyperparameters, including learning rate, batch size, and LoRA rank, are detailed in Table 6.

C.3 Sensitivity Analysis

To assess the sensitivity of C-AUM to its hyperparameters, we conduct a one-at-a-time sensitivity analysis on fold 1 using Qwen2.5-3B-Instruct, varying each parameter while keeping the others fixed at their default values. As shown in Table 7, macro F1 ranges from 93.09% to 95.07% across the tested configurations, compared with 92.82% for vanilla SFT. These results suggest that C-AUM maintains relatively stable performance within the examined parameter ranges and is not highly dependent on a single configuration. This stability is useful in few-shot settings where validation data for extensive hyperparameter tuning may be limited.