

# DRNash: Efficient and Robust Multi-Task Learning for Financial Sentiment Analysis

Hongcheng Ding<sup>1,\*</sup>, Xuanhuang Liu<sup>2,\*</sup>, Deshinta Arrova Dewi<sup>3</sup>,  
Shamsul Nahar Abdullah<sup>3</sup>, Qingyu Liu<sup>2</sup>, Xuanze Zhao<sup>1,†</sup>

<sup>1</sup>Dongfang College, Zhejiang University of Finance and Economics, China

<sup>2</sup>SEGi University, Malaysia    <sup>3</sup>INTI International University, Malaysia

\*Equal contribution.    †Correspondence: 20070032@zufedfc.edu.cn

## Abstract

Fine-tuning large language models (LLMs) for sentiment polarity regression often suffers from severe data distribution imbalance. Financial texts typically concentrate in the neutral range, leaving extreme sentiments underrepresented, yet traditional techniques offer limited remedy since such extremes are inherently rare. We introduce an auxiliary classification task to provide complementary supervision, but this multi-task formulation faces two challenges: (1) sentiment group construction for the classification task, and (2) gradient conflicts between tasks that fixed weights cannot resolve. These challenges are coupled: improper group boundaries yield biased gradients, which amplify task conflicts. We propose the **DRNash** framework as a unified solution to jointly address both challenges and the underlying data distribution imbalance, including **D**istributionally **R**obust **O**ptimization (DRO) and **N**ash Task Coordinator (NTC). DRO partitions the sentiment space into quantile-based groups and performs worst-case aggregation over group losses, automatically assigning higher weights to underperforming sentiment regions. NTC formulates task weighting as a Nash bargaining game, deriving closed-form weights that achieve scale-invariant balancing. Experiments demonstrate that DRNash achieves average improvements of 14.1% in MSE and 1.92% in accuracy (ACC) across multiple financial datasets, with cross-domain experiments confirming generalizability beyond financial texts.

## 1 Introduction

Sentiment analysis is crucial across domains from business intelligence to financial forecasting (Saravanos and Kanavos, 2025; Ding et al., 2024c; Bouteska et al., 2024). Fine-tuning LLMs for sentiment polarity regression has become a dominant paradigm in financial text analysis (Figure 1A). However, we observed severe performance degradation when fine-tuning LLMs on financial datasets

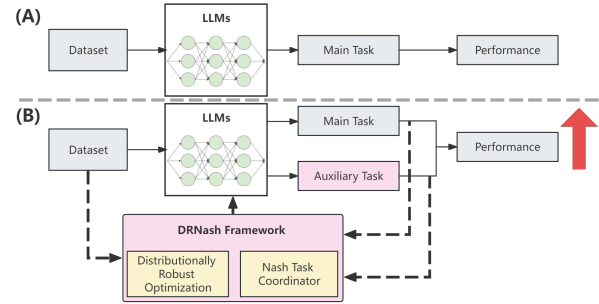


Figure 1: Comparison between traditional single-task approach (A) and our DRNash framework for multi-task learning (B). Our framework introduces an auxiliary classification task and achieves improved performance.

due to data distribution imbalance. Financial texts typically exhibit heavy concentration in the neutral range while extreme sentiments remain underrepresented (Du et al., 2024; Davidovic and McCleary, 2025), causing models to develop strong bias toward over-represented regions (Mujahid et al., 2024; Nassr et al., 2025). This is particularly problematic as extreme sentiments often signal market turning points and carry actionable information for risk management (Song, 2024; Nguyen and Nguyen, 2024; Zou and Herremans, 2023).

Traditional approaches to address data distribution imbalance include data augmentation techniques (Wang et al., 2026; Assunção et al., 2024) and cost-sensitive learning methods (Araf et al., 2024; Xiao et al., 2025). However, these methods face fundamental limitations in sentiment regression. Data augmentation may introduce artificial patterns that deviate from authentic financial discourse (Gao et al., 2025; Schaudt et al., 2023), and more critically, extreme sentiments are inherently rare in real-world distributions, so artificially replicating them risks overfitting rather than genuine generalization. Cost-sensitive methods require extensive hyperparameter tuning and lack adaptability to the continuous nature of sentiment scores (Khan et al., 2018; Edsa et al., 2025).

We introduce an auxiliary classification task that provides complementary supervision (Cai et al., 2025; Yao et al., 2023), helping models better capture sentiment boundaries across the spectrum. Prior work has shown that jointly modeling related tasks improves representation learning (Lai et al., 2026; Yang et al., 2021). However, this multi-task formulation introduces two coupled challenges (1) how to partition the continuous sentiment space into meaningful groups for classification, and (2) inherent gradient conflicts between regression and classification objectives that fixed task weights cannot resolve (Chen and Er, 2025; Wang et al., 2021). These challenges are coupled: inappropriate group boundaries cause biased gradient estimates, which in turn exacerbates task conflicts.

We propose the DRNash framework as a unified solution to jointly address both challenges and the underlying data distribution imbalance (Figure 1B). DRNash comprises two components: DRO constructs quantile-based sentiment groups and performs worst-case aggregation to emphasize underperforming regions, while NTC derives dynamic task weights via the Nash bargaining solution. The key insight is that DRO ensures balanced gradient contributions from each sentiment region, while NTC’s scale-invariant weighting resolves magnitude discrepancies via Nash bargaining (Navon et al., 2022). The bargaining formulation admits a closed-form solution under which the two tasks contribute gradients of equal magnitude, so the balance holds even when regression MSE and classification cross-entropy differ by orders of magnitude. Together, they break the coupling cycle that undermines conventional approaches. The main contributions of this paper are as follows:

- We identify two coupled challenges when fine-tuning LLMs for sentiment polarity regression: (1) data distribution imbalance biases models toward over-represented neutral regions, and (2) gradient conflicts between regression and classification objectives that fixed task weights cannot resolve. These challenges are coupled: improper group construction yields biased gradients, which amplify task conflicts.
- We propose the DRNash framework as a unified solution to jointly address both challenges and the underlying data distribution imbalance, comprising DRO for balanced optimization via worst-case aggregation and NTC for

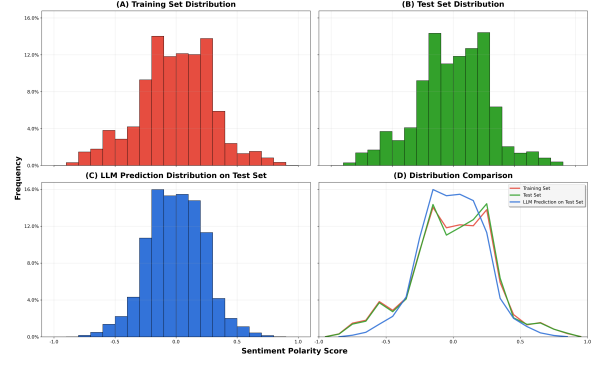


Figure 2: Data distribution analysis: (A) Training set, (B) Test set, (C) Fine-tuned LLM predictions on the test set, and (D) Comparative analysis. More Details are provided in Appendix A.1.

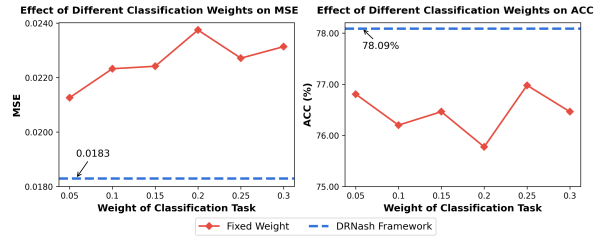


Figure 3: Effect of different task weights on MSE and ACC in multi-task learning, compared with the performance of our proposed DRNash framework.

scale-invariant task balancing via Nash bargaining.

- Experiments on multiple financial datasets demonstrate that DRNash achieves average improvements of 14.1% in MSE and 1.92% in ACC compared with previous work. Cross-domain experiments further confirm generalizability beyond financial texts.

## 2 Motivation

In this section, we introduce data-level and task-level challenges, and the coupling between them.

**Data-Level Challenge: Distribution Bias Amplification.** Figure 2 reveals a critical phenomenon: fine-tuned LLMs do not merely inherit the training distribution bias, but amplify it. Comparing panels (A)-(C), while training data concentrates around neutral sentiments, the model’s predictions (C) exhibit an even sharper peak near zero with severely attenuated tails. Panel (D) quantifies this amplification: the training and test distributions nearly coincide, while the prediction distribution departs from both. This bias amplification is particularly problematic for financial applications where extreme sentiments often carry the most actionable information.

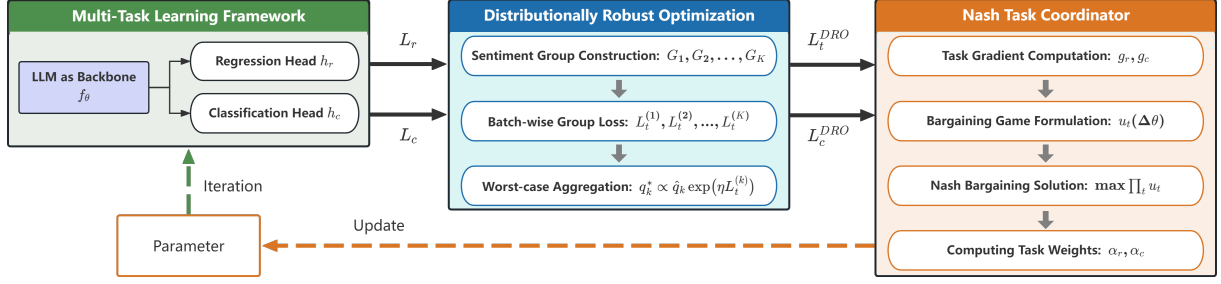


Figure 4: DRNash framework, comprising (Middle) distributionally robust optimization and (Right) Nash task coordinator, can be seamlessly integrated into (Left) multi-task learning architectures (Details in Appendix A.4).

### Task-Level Challenge: Inherent Task Conflict.

To address distribution bias, a natural solution is multi-task learning with an auxiliary classification task. However, Figure 3 reveals a fundamental obstacle: no fixed weight configuration can simultaneously optimize both tasks. The performance surface is non-convex with respect to task weights, and the optimal weight for MSE differs from that for ACC. This is not merely a hyperparameter tuning problem but reflects an inherent conflict between regression and classification gradients during optimization (More details in Appendix A.2).

**Coupling Between Challenges.** These challenges are coupled: distribution bias causes neutral samples to dominate gradient estimates, making gradient magnitudes unreliable for task balancing and further amplifying conflicts. More detailed analysis is provided in Appendix A.3. This motivates us to model task weighting as a Nash bargaining game (Navon et al., 2022), where the two tasks negotiate over a shared parameter update rather than competing through fixed coefficients, enabling principled scale-invariant balancing without manual calibration.

## 3 Method

In this section, we present problem formulation and DRNash framework, which comprises distributionally robust optimization, Nash task coordinator and end-to-end integration with LoRA.

### 3.1 Problem Formulation

To address the challenges identified in Section 2, we formulate a multi-task optimization problem:

$$\min_{\theta} \mathcal{L}(\theta) = \sum_{t \in \{r, c\}} \underbrace{\alpha_t}_{\text{how to balance (task-level)}} \cdot \underbrace{L_t(f_{\theta})}_{\text{what to optimize (data-level)}}, \quad (1)$$

where  $f_{\theta}$  denotes the LLM backbone with trainable parameters  $\theta$ , and  $t \in \{r, c\}$  indexes the regression

and classification tasks.  $L_t(f_{\theta})$  ensures balanced optimization across the sentiment spectrum, addressing the data-level challenge;  $\alpha_t$  dynamically balances task contributions based on gradient information, addressing the task-level challenge.

### 3.2 DRNash Framework

To solve Eq. (1), we propose DRNash as illustrated in Figure 4, a unified framework that jointly addresses both challenges: DRO for the data-level distribution bias and NTC for the task-level gradient conflicts. DRNash builds upon a multi-task architecture (More details are provided in Appendix A.4.), consisting of three components: a LLM backbone  $f_{\theta}$ , a regression head  $h_r$  predicting continuous sentiment scores  $\hat{y} \in [-1, 1]$ , and a classification head  $h_c$  predicting discrete categories  $\hat{c} \in \{1, \dots, K\}$ . The regression head serves as the primary task, while the classification head provides complementary supervision that helps capture sentiment boundaries across the spectrum.

### 3.3 Distributionally Robust Optimization

DRO addresses the data-level challenge identified in Section 2 by instantiating  $L_t(f_{\theta})$  in Eq. (1) to ensure balanced optimization across the sentiment spectrum. As illustrated in Figure 4 (Middle), sentiment groups are first constructed to partition the label space, then batch-wise group losses are computed during training, and finally worst-case aggregation is performed to emphasize underperforming regions (More details in Appendix A.5).

**Sentiment Group Construction.** Let  $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$  denote a sentiment dataset, where  $x_i$  is the input text and  $y_i \in [-1, 1]$  is the continuous sentiment polarity. Before training, the sentiment space is partitioned into  $K$  discrete groups based on quantile boundaries. Let  $\tau_k = Q_{k/K}(\{y_i\}_{i=1}^N)$  denote the  $k/K$ -quantile of training sentiment scores.

The  $k$ -th sentiment group is defined as:

$$G_k = \{(x_i, y_i) : \tau_{k-1} < y_i \leq \tau_k\}, \quad (2)$$

where  $k = 1, \dots, K$ ,  $\tau_0 = -1$ , and  $\tau_K = 1$  (with  $G_1$  extended to include  $y_i = \tau_0$ , so every sample receives a group). Each sample is assigned a group label  $c_i \in \{1, \dots, K\}$  indicating which group it belongs to. This quantile-based construction ensures approximately equal sample counts per group ( $|G_k| \approx N/K$ ). Equal group sizes follow from quantile partitioning itself rather than from any assumption about the data: since the  $k/K$ -quantile is the value below which a  $k/K$  fraction of the scores falls,  $|G_k| \approx N/K$  holds however skewed the distribution is, and what adapts to the skew is the *width* of each interval. On INV, roughly 50% of samples lie in  $[-0.2, 0.2]$  while the extreme regions ( $|y| > 0.5$ ) account for only 11.5%, so the boundaries assign narrow intervals to the dense neutral region and wide ones to the sparse tails. Fixed-threshold grouping instead places most samples in a single neutral group and leaves the extreme groups nearly empty. For the classification task, the group label  $c_i$  serves as the ground truth, transforming sentiment classification into predicting the intensity region of the input text. More details are provided in [Appendix A.6](#).

**Batch-wise Group Loss.** During training, for each mini-batch  $\mathcal{B}$ , DRO computes the average loss within each sentiment group. Let  $\mathcal{B}_k = \{(x, y, c) \in \mathcal{B} : c = k\}$  denote samples in batch  $\mathcal{B}$  belonging to group  $G_k$ , and let  $\mathcal{K}_{\mathcal{B}} = \{k : |\mathcal{B}_k| > 0\}$  denote the set of groups present in the current batch. For regression, the group loss is the mean squared error (MSE):

$$L_r^{(k)}(\theta; \mathcal{B}) = \frac{1}{|\mathcal{B}_k|} \sum_{(x, y, c) \in \mathcal{B}_k} (h_r(f_\theta(x)) - y)^2. \quad (3)$$

For classification, the group loss is the cross-entropy (CE):

$$L_c^{(k)}(\theta; \mathcal{B}) = \frac{1}{|\mathcal{B}_k|} \sum_{(x, y, c) \in \mathcal{B}_k} \text{CE}(h_c(f_\theta(x)), c). \quad (4)$$

**Worst-case Aggregation.** Instead of uniform averaging, group losses are aggregated using worst-case weighting. Let  $\hat{q}_k = |G_k|/N$  denote the proportion of group  $k$  in the full training set. Since quantile-based boundaries ensure  $|G_k| \approx N/K$ ,

this yields  $\hat{q}_k \approx 1/K$ . The DRO loss is defined as:

$$L_t(f_\theta; \mathcal{B}) = \frac{1}{\eta} \log \left( \sum_{k \in \mathcal{K}_{\mathcal{B}}} \hat{q}_k \exp(\eta L_t^{(k)}(\theta; \mathcal{B})) \right), \quad (5)$$

where  $\eta > 0$  controls the robustness level. This formulation corresponds to the worst-case expectation over distributions within a KL-divergence ball around the empirical distribution (See [Appendix A.7](#)). The optimal worst-case weights take the form:

$$q_k^* \propto \hat{q}_k \exp(\eta L_t^{(k)}(\theta; \mathcal{B})), \quad (6)$$

which assigns exponentially larger weights to groups with higher losses, automatically emphasizing underperforming sentiment regions. When  $\eta \rightarrow 0$ , Eq. (5) recovers the standard weighted average; when  $\eta \rightarrow \infty$ , it approaches the maximum group loss (See [Appendix A.7](#)). The summation is restricted to groups present in the current batch ( $\mathcal{K}_{\mathcal{B}}$ ), which is standard practice in group DRO as absent groups do not contribute to the current gradient ([Sagawa et al., 2020](#)).

Eq. (5) instantiates  $L_t(f_\theta)$  in Eq. (1) for both regression ( $t = r$ ) and classification ( $t = c$ ) tasks, ensuring balanced optimization across the sentiment spectrum without manual threshold tuning.

### 3.4 Nash Task Coordinator

NTC addresses the task-level challenge identified in Section 2 by computing  $\alpha_t$  in Eq. (1) to dynamically balance task contributions. As illustrated in Figure 4 (Right), task gradients are first computed from DRO losses, then a bargaining game is formulated where tasks negotiate over parameter updates, the Nash bargaining solution is solved, and finally closed-form task weights are derived. More details are provided in [Appendix A.8](#).

**Task Gradient Computation.** At each step, after computing DRO losses  $L_r(f_\theta; \mathcal{B})$  and  $L_c(f_\theta; \mathcal{B})$  via Eq. (5), we obtain task-specific gradients:

$$g_t = \nabla_\theta L_t(f_\theta; \mathcal{B}) \in \mathbb{R}^{|\theta|}, \quad t \in \{r, c\}, \quad (7)$$

where  $|\theta|$  is the number of trainable parameters.

**Bargaining Game Formulation.** A naive approach would be to sum task gradients with fixed or heuristically-tuned weights. However, such approaches suffer from scale sensitivity, as regression MSE and classification CE have different magnitudes. NTC instead models tasks as rational players

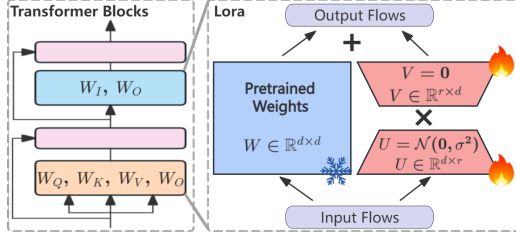


Figure 5: LoRA integration in Transformer blocks.

negotiating over the shared parameter update  $\Delta\theta$ . Each task’s utility is the resulting loss reduction:

$$u_t(\Delta\theta) \approx -g_t^\top \Delta\theta. \quad (8)$$

Positive utility indicates improvement while negative indicates degradation. The disagreement point  $(0, 0)$  represents no update, which both tasks prefer to avoid.

**Nash Bargaining Solution.** The Nash bargaining solution maximizes the product of utility gains:

$$\Delta\theta^* = \arg \max_{\|\Delta\theta\| \leq 1} \prod_{t \in \{r, c\}} (-g_t^\top \Delta\theta), \quad (9)$$

where  $\|\Delta\theta\| \leq 1$  constrains the update magnitude. A key property is scale invariance (Navon et al., 2022): rescaling either loss leaves the two weighted gradient contributions equal in magnitude, so the balance between regression MSE and classification CE is preserved without manual calibration (Details in Appendix A.9).

**Computing Task Weights.** The solution takes the form  $\Delta\theta^* \propto -\sum_t \alpha_t^* g_t$ . Following Navon et al. (2022), in high-dimensional parameter spaces where task gradients are approximately orthogonal, the Nash solution simplifies to inverse gradient magnitude weighting. For regression:

$$\alpha_r = \frac{\|g_c\| + \epsilon}{\|g_r\| + \|g_c\| + 2\epsilon}, \quad (10)$$

and similarly for classification:

$$\alpha_c = \frac{\|g_r\| + \epsilon}{\|g_r\| + \|g_c\| + 2\epsilon}, \quad (11)$$

where  $\epsilon > 0$  ensures numerical stability. This assigns smaller weights to tasks with larger gradients, balancing their contributions.

Eq. (10) and Eq. (11) instantiate  $\alpha_t$  in Eq. (1), dynamically adapting task weights based on gradient information at each training step.

| Benchmark      | Model                                         | Task                         |
|----------------|-----------------------------------------------|------------------------------|
| B1             | RoBERTa-Large (Liu et al., 2019)              | Regression                   |
| B2             | Twitter-RoBERTa-Large (Loureiro et al., 2023) | Regression                   |
| B3             | Twitter-RoBERTa-Large (Loureiro et al., 2023) | Multi-task (Constant Weight) |
| B4-DRNash      | Twitter-RoBERTa-Large (Loureiro et al., 2023) | Multi-task (DRO + NTC)       |
| B5-DRNash      | TinyLlama-1.1B (Zhang et al., 2024)           | Multi-task (DRO + NTC)       |
| B6-DRNash      | Qwen2-0.5B (Yang et al., 2024)                | Multi-task (DRO + NTC)       |
| B7-DRNash+LoRA | Twitter-RoBERTa-Large (Loureiro et al., 2023) | Multi-task (DRO + NTC)       |
| B8-DRNash+LoRA | TinyLlama-1.1B (Zhang et al., 2024)           | Multi-task (DRO + NTC)       |
| B9-DRNash+LoRA | Qwen2-0.5B (Yang et al., 2024)                | Multi-task (DRO + NTC)       |

Table 1: Benchmark models and experimental settings.

### 3.5 End-to-End Integration with LoRA

**End-to-End Training.** DRNash integrates DRO and NTC into a unified end-to-end training pipeline, as illustrated in Figure 4. At each iteration, input text is processed by the language model backbone and task heads to obtain predictions. DRO computes robust task losses via worst-case aggregation over sentiment groups, and NTC derives adaptive task weights from gradient magnitudes. The weighted gradients then update the backbone parameters, forming a closed loop that iteratively addresses both data-level and task-level challenges.

**LoRA for Efficient Fine-tuning.** LoRA (Hu et al., 2022) is integrated for parameter-efficient fine-tuning as shown in Figure 5. LoRA introduces rank- $r$  matrices  $U \in \mathbb{R}^{d \times r}$  and  $V \in \mathbb{R}^{r \times d}$  into transformer blocks:

$$W' = W + UV, \quad (12)$$

where pretrained weights  $W$  remain frozen, significantly reducing the number of trainable parameters (Complexity analysis in Appendix A.10).

## 4 Experiments

### 4.1 Implementation Details

**Software and Hardware.** We use Ubuntu 22.04, Python 3.9.19, PyTorch 2.3.1, PEFT 0.12.0, and CUDA 12.4, running on a system with 64GB RAM and NVIDIA A100 GPUs with 80GB VRAM.

**Benchmark.** Table 1 summarizes our benchmarks: three baselines (B1-B3), three DRNash configurations (B4-B6), and three DRNash+LoRA variants (B7-B9). Details are provided in Appendix A.11.

**Evaluation Metrics.** Regression metrics include MSE, MAE, RMSE and  $R^2$ ; classification metrics include ACC, weighted Precision (Prec.) and F1. We also measure Time Usage (relative training time to reach full fine-tuning’s best MSE, where  $<100\%$  indicates acceleration).  $\downarrow$  denotes lower is better;  $\uparrow$  denotes higher is better. Details in Appendix A.12.

**Dataset.** We use three financial text datasets: NEU, FXE, and INV (Details in Appendix A.13), containing news and analysis on EUR/USD exchange

rates. For cross-domain evaluation, we additionally use SemEval-2017 Task 5 (Cortis et al., 2017) for stock sentiment, SemEval-2018 Task 1 V-reg (Mohammad et al., 2018) for Twitter valence, and EmoBank (Buechel and Hahn, 2017) for multi-genre text (Details in Appendix A.19). These datasets are randomly split into training and validation sets with a 9:1 ratio.

**Hyperparameters.** We use batch size 32, max sequence length 512, and train for 100 epochs with AdamW optimizer (learning rate  $1 \times 10^{-5}$ ), cosine decay and 100 warmup steps. For DRO,  $K = 5$  groups and  $\eta = 1.0$ ; for NTC,  $\epsilon = 10^{-8}$ . LoRA ranks range from 128 to 512 with alpha equal to rank and dropout 0.05. Random seeds are fixed at 42 for reproducibility. More details are provided in Appendix A.14.

## 4.2 Main Results

**Unified Framework Effectiveness.** Table 2 compares constant weight multi-task learning with our DRNash framework across three architectures spanning different paradigms (encoder-only vs. decoder-only) and scales (355M to 1.1B parameters). Across all configurations, DRNash achieves consistent improvements: 14.1% average MSE reduction and 1.92% ACC gain. Twitter-RoBERTa-Large shows the largest MSE reduction (17.2% to 19.0%), while TinyLlama 1.1B achieves the best absolute performance.

These results validate that DRNash effectively decouples the data-level and task-level challenges. By jointly optimizing group-wise robustness and task balancing, DRNash breaks the vicious cycle where distribution bias amplifies gradient conflicts. The consistent gains across architecturally diverse models, from encoder-based RoBERTa to decoder-based LLMs, confirm that our framework addresses fundamental optimization challenges inherent to imbalanced sentiment regression. More details are provided in Appendix A.15.

**Effectiveness of Auxiliary Classification Task.** The upper section of Table 3 (gray rows) validates the effectiveness of auxiliary classification by comparing single-task regression with constant weight MTL. The auxiliary task partitions the sentiment space  $[-1, 1]$  into five equal-width intervals, and jointly optimizes with regression using fixed weights ( $w_r = 0.9, w_c = 0.1$ ).

As shown in Table 3, introducing auxiliary classification achieves consistent MSE reductions of

| Model                    | Dataset | Constant Weight |       | DRNash |       |
|--------------------------|---------|-----------------|-------|--------|-------|
|                          |         | MSE↓            | ACC↑  | MSE↓   | ACC↑  |
| Twitter<br>RoBERTa-Large | NEU     | 0.0231          | 75.89 | 0.0187 | 77.34 |
|                          | FXE     | 0.0244          | 75.12 | 0.0202 | 76.78 |
|                          | INV     | 0.0222          | 76.56 | 0.0183 | 78.09 |
| Qwen 0.5B                | NEU     | 0.0247          | 74.89 | 0.0214 | 76.58 |
|                          | FXE     | 0.0252          | 74.67 | 0.0227 | 75.86 |
|                          | INV     | 0.0241          | 75.18 | 0.0209 | 76.81 |
| TinyLlama<br>1.1B        | NEU     | 0.0204          | 76.89 | 0.0179 | 78.21 |
|                          | FXE     | 0.0215          | 76.23 | 0.0187 | 77.69 |
|                          | INV     | 0.0196          | 77.34 | 0.0174 | 78.49 |

Table 2: Performance comparison between constant weight and DRNash framework. **Pink**: best results.

3.0% to 4.0% and  $R^2$  gains of 0.7 to 1.1 absolute points across all configurations. Although these gains are modest, they confirm that auxiliary classification provides complementary supervision that enhances sentiment boundary learning, establishing multi-task formulation as an effective foundation for our DRNash framework. More details can be found in Appendix A.16.

**Comparison with Traditional Imbalanced Learning Methods.** The middle section of Table 3 evaluates whether traditional imbalanced learning methods can effectively address data distribution imbalance. We compare four representative approaches: LDS applies kernel density smoothing to re-weight samples; Balanced MSE uses a contrastive loss formulation; Focal-R adapts Focal Loss by down-weighting easy samples; and SMOGN generates synthetic samples for data augmentation. Blue cells indicate cases where traditional methods outperform constant weight MTL. More details can be found in Appendix A.17.

The results reveal two critical limitations of traditional methods. First, they exhibit severe dataset-dependent variance: LDS improves on NEU but degrades on FXE, whereas Balanced MSE degrades on both and improves only on INV. Second, several configurations perform worse than Single Task baseline, indicating that these methods can be harmful rather than helpful. SMOGN produces the worst results with 13.3% to 35.7% MSE increase, confirming that synthetic oversampling introduces artificial patterns. These findings suggest that addressing data distribution imbalance through sample re-weighting or augmentation alone is insufficient for sentiment regression tasks.

**Efficient Fine-tuning with LoRA Integration.** Table 4 evaluates LoRA integration with our DR-

| Method                                           | Model                 | NEU    |        |                  | FXE    |        |                  | INV    |        |                  |
|--------------------------------------------------|-----------------------|--------|--------|------------------|--------|--------|------------------|--------|--------|------------------|
|                                                  |                       | MSE↓   | MAE↓   | R <sup>2</sup> ↑ | MSE↓   | MAE↓   | R <sup>2</sup> ↑ | MSE↓   | MAE↓   | R <sup>2</sup> ↑ |
| Single Task                                      | Twitter-RoBERTa-Large | 0.0239 | 0.1103 | 72.68            | 0.0253 | 0.1136 | 72.26            | 0.0230 | 0.1082 | 73.66            |
|                                                  | Qwen 0.5B             | 0.0256 | 0.1141 | 70.80            | 0.0262 | 0.1155 | 70.42            | 0.0249 | 0.1125 | 71.83            |
|                                                  | TinyLlama 1.1B        | 0.0211 | 0.1036 | 75.91            | 0.0224 | 0.1067 | 75.54            | 0.0202 | 0.1014 | 76.82            |
| Constant Weight MTL                              | Twitter-RoBERTa-Large | 0.0231 | 0.1085 | 73.56            | 0.0244 | 0.1114 | 73.32            | 0.0222 | 0.1063 | 74.50            |
|                                                  | Qwen 0.5B             | 0.0247 | 0.1121 | 71.84            | 0.0252 | 0.1133 | 71.50            | 0.0241 | 0.1107 | 72.67            |
|                                                  | TinyLlama 1.1B        | 0.0204 | 0.1019 | 76.72            | 0.0215 | 0.1046 | 76.47            | 0.0196 | 0.0998 | 77.53            |
| LDS<br>(Yang et al., 2021)                       | Twitter-RoBERTa-Large | 0.0218 | 0.1053 | 75.07            | 0.0261 | 0.1152 | 71.42            | 0.0221 | 0.1059 | 74.69            |
|                                                  | Qwen 0.5B             | 0.0236 | 0.1095 | 73.04            | 0.0279 | 0.1189 | 68.52            | 0.0242 | 0.1108 | 72.53            |
|                                                  | TinyLlama 1.1B        | 0.0191 | 0.0985 | 78.21            | 0.0234 | 0.1091 | 74.38            | 0.0195 | 0.0994 | 77.68            |
| Balanced MSE<br>(Ren et al., 2022)               | Twitter-RoBERTa-Large | 0.0253 | 0.1134 | 71.06            | 0.0248 | 0.1122 | 72.87            | 0.0219 | 0.1055 | 74.90            |
|                                                  | Qwen 0.5B             | 0.0274 | 0.1180 | 68.71            | 0.0259 | 0.1148 | 70.76            | 0.0238 | 0.1100 | 72.96            |
|                                                  | TinyLlama 1.1B        | 0.0225 | 0.1069 | 74.31            | 0.0218 | 0.1053 | 76.12            | 0.0192 | 0.0988 | 78.04            |
| Focal-R<br>(Lin et al., 2017; Yang et al., 2021) | Twitter-RoBERTa-Large | 0.0242 | 0.1109 | 72.34            | 0.0291 | 0.1216 | 68.14            | 0.0228 | 0.1076 | 73.85            |
|                                                  | Qwen 0.5B             | 0.0259 | 0.1147 | 70.41            | 0.0312 | 0.1259 | 64.79            | 0.0248 | 0.1122 | 71.67            |
|                                                  | TinyLlama 1.1B        | 0.0213 | 0.1041 | 75.68            | 0.0263 | 0.1156 | 71.21            | 0.0199 | 0.1006 | 77.18            |
| SMOGLN<br>(Branco et al., 2017)                  | Twitter-RoBERTa-Large | 0.0285 | 0.1204 | 67.42            | 0.0317 | 0.1269 | 65.28            | 0.0252 | 0.1132 | 71.08            |
|                                                  | Qwen 0.5B             | 0.0311 | 0.1257 | 64.47            | 0.0342 | 0.1318 | 61.41            | 0.0273 | 0.1177 | 68.87            |
|                                                  | TinyLlama 1.1B        | 0.0256 | 0.1141 | 70.74            | 0.0286 | 0.1206 | 68.68            | 0.0223 | 0.1065 | 74.38            |
| <b>DRNash (Ours)</b>                             | Twitter-RoBERTa-Large | 0.0187 | 0.0978 | 78.50            | 0.0202 | 0.1015 | 77.82            | 0.0183 | 0.0965 | 79.00            |
|                                                  | Qwen 0.5B             | 0.0214 | 0.1044 | 75.53            | 0.0227 | 0.1074 | 74.45            | 0.0209 | 0.1031 | 76.14            |
|                                                  | TinyLlama 1.1B        | 0.0179 | 0.0954 | 79.55            | 0.0187 | 0.0976 | 79.60            | 0.0174 | 0.0941 | 80.05            |

Table 3: Performance comparison across single-task regression, constant weight MTL, traditional imbalanced learning methods, and DRNash. **Gray**: baselines; **Blue**: imbalanced-learning methods beating constant weight MTL; **Pink**: best overall.

| Model                 | Dataset | Rank = 128 |       |         | Rank = 256 |       |         | Rank = 384 |       |         | Rank = 512 |       |        |
|-----------------------|---------|------------|-------|---------|------------|-------|---------|------------|-------|---------|------------|-------|--------|
|                       |         | MSE↓       | ACC↑  | Time↓   | MSE↓       | ACC↑  | Time↓   | MSE↓       | ACC↑  | Time↓   | MSE↓       | ACC↑  | Time↓  |
| Twitter RoBERTa-Large | NEU     | 0.0190     | 77.12 | 95.62%  | 0.0187     | 77.45 | 78.12%  | 0.0186     | 77.48 | 50.38%  | 0.0185     | 77.64 | 34.16% |
|                       | FXE     | 0.0206     | 76.45 | 119.24% | 0.0202     | 76.82 | 103.46% | 0.0200     | 76.95 | 67.20%  | 0.0197     | 77.25 | 45.59% |
|                       | INV     | 0.0186     | 77.82 | 97.46%  | 0.0184     | 78.08 | 79.08%  | 0.0182     | 78.31 | 51.83%  | 0.0180     | 78.46 | 34.53% |
| Qwen 0.5B             | NEU     | 0.0218     | 76.28 | 94.37%  | 0.0214     | 76.78 | 75.88%  | 0.0213     | 76.92 | 67.95%  | 0.0212     | 77.11 | 42.60% |
|                       | FXE     | 0.0231     | 75.52 | 124.42% | 0.0227     | 75.91 | 107.04% | 0.0225     | 76.08 | 91.62%  | 0.0223     | 76.35 | 60.90% |
|                       | INV     | 0.0213     | 76.48 | 90.24%  | 0.0209     | 77.02 | 74.26%  | 0.0208     | 77.15 | 66.95%  | 0.0206     | 77.42 | 42.13% |
| TinyLlama 1.1B        | NEU     | 0.0182     | 77.89 | 104.53% | 0.0180     | 78.25 | 85.06%  | 0.0178     | 78.48 | 80.81%  | 0.0176     | 78.71 | 56.76% |
|                       | FXE     | 0.0191     | 77.35 | 130.19% | 0.0189     | 77.72 | 114.70% | 0.0187     | 78.02 | 108.30% | 0.0186     | 78.28 | 76.68% |
|                       | INV     | 0.0177     | 78.18 | 112.12% | 0.0175     | 78.52 | 90.93%  | 0.0173     | 78.72 | 87.92%  | 0.0171     | 78.97 | 61.32% |

Table 4: LoRA integration with our DRNash framework: performance across different ranks. Time (%) denotes the relative training time to reach equivalent performance compared to full fine-tuning (DRNash without LoRA), where <100% indicates faster convergence. More detailed analysis is provided in Appendix A.18.

Nash framework across four rank configurations. The results demonstrate that LoRA can be seamlessly integrated into DRNash as a parameter-efficient fine-tuning (PEFT) strategy. As rank increases from 128 to 512, MSE decreases by 2.6% to 4.4%, ACC improves by 0.5 to 0.9 absolute points, and training time reduces to 34.16% to 76.68% of full fine-tuning. Higher ranks enable LoRA to capture more task-relevant information while maintaining computational efficiency. Notably, at Rank=512, DRNash+LoRA surpasses full fine-tuning performance while reducing training time by approximately 50%.

**Scaling to Larger Backbones.** The backbones above range from 355M to 1.1B parameters. To verify that DRNash remains effective as backbone capacity grows, Table 5 reports full fine-tuning with three larger LLMs: Gemma-2-2B (Gemma Team,

| Backbone     | Data | MSE ↓         | RMSE ↓        | ACC ↑        |
|--------------|------|---------------|---------------|--------------|
| Gemma-2-2B   | NEU  | 0.0174        | 0.1319        | 78.41        |
|              | FXE  | 0.0183        | 0.1353        | 77.84        |
|              | INV  | 0.0168        | 0.1296        | 78.71        |
| Llama-3.2-3B | NEU  | 0.0170        | 0.1304        | 78.56        |
|              | FXE  | 0.0178        | 0.1334        | 78.06        |
|              | INV  | 0.0165        | 0.1285        | 78.74        |
| Qwen2.5-7B   | NEU  | <b>0.0163</b> | <b>0.1277</b> | <b>78.86</b> |
|              | FXE  | <b>0.0171</b> | <b>0.1308</b> | <b>78.21</b> |
|              | INV  | <b>0.0158</b> | <b>0.1257</b> | <b>79.07</b> |

Table 5: DRNash with larger backbones under full fine-tuning. All three surpass the 1.1B configuration (B5) on all nine model-dataset pairs, with MSE decreasing monotonically as capacity grows.

2024), Llama-3.2-3B (Grattafiori et al., 2024) and Qwen2.5-7B (Qwen Team, 2024). All three surpass

the 1.1B configuration B5 on every dataset, and MSE decreases monotonically as capacity grows (0.0174  $\rightarrow$  0.0163 on NEU). DRNash therefore continues to benefit from additional backbone capacity.

**Comparison with an Ordinal Baseline.** Because sentiment intensity is ordinal, we also compare against CORAL (Cao et al., 2020), replacing the cross-entropy auxiliary head with a rank-consistent ordinal loss while keeping all else identical to Constant Weight MTL. CORAL yields a consistent 2.0–3.0% MSE reduction over Constant Weight MTL across all nine configurations (e.g. 0.0224 on NEU with Twitter-RoBERTa-Large, 0.0199 with TinyLlama-1.1B), confirming that exploiting ordinal structure helps. It nevertheless remains well behind DRNash, indicating that our gains come from group-wise worst-case aggregation and scale-invariant task balancing rather than ordinal supervision alone. Label distribution learning (Geng, 2016) does not apply here, as it requires a full label distribution per instance whereas our datasets provide single-point scores.

**Cross-Domain Generalization.** To address whether DRNash generalizes beyond our specific financial datasets, we evaluate on three additional benchmarks: SemEval-2017 Task 5 for stock sentiment, SemEval-2018 Task 1 V-reg for Twitter valence intensity, and EmoBank for multi-genre text. As shown in Appendix A.19 (due to page limits), DRNash achieves consistent performance across all domains, with average MSE of 0.0199 and ACC of 76.94%. These results confirm that our framework is domain-agnostic and applicable to general sentiment polarity regression tasks.

**Pre-training Bias Analysis.** The comparison between RoBERTa-Large and Twitter-RoBERTa-Large (Table shown in Appendix A.20 due to page limits) reveals that models inherit distributional biases from their pre-training corpora. Twitter-RoBERTa-Large achieves lower MSE across all datasets due to its exposure to social media text during pre-training. This observation motivates our use of DRO to mitigate the impact of such biases on sentiment regression performance.

### 4.3 Ablation Studies

**Component Ablation Study.** To validate the contributions of each component in DRNash, we conduct ablation experiments with four configurations:

Single Task regression, Constant Weight MTL (fixed  $w_r = 0.9$ ,  $w_c = 0.1$ ), MTL with DRO only (adding worst-case aggregation with re-tuned fixed weights to account for DRO’s transformed loss landscape), and MTL with NTC only (replacing fixed weights with Nash bargaining while using standard loss averaging). As shown in Figure 6(A), both DRO and NTC independently improve performance; averaged over the three datasets, DRO provides 8.8% MSE reduction and NTC 5.6%. DRO contributes more because it addresses the root cause (data-level distributional bias) rather than its downstream effect (task-level gradient conflicts). Notably, DRNash achieves 17.9% average MSE reduction, exceeding the sum of individual contributions (14.4%) and demonstrating synergistic effects where DRO’s balanced gradients enable more effective Nash bargaining. Details are provided in Appendix A.21.

**Sensitivity to Group Number  $K$ .** Figure 6(B) investigates how the number of sentiment groups  $K$  affects DRO performance. Both  $K = 3$  and  $K = 7$  degrade performance compared to the optimal  $K = 5$ , but through different mechanisms:  $K = 3$  causes 8.2% to 10.4% MSE degradation due to insufficient granularity, while  $K = 7$  causes 3.8% to 8.4% degradation due to sample sparsity. Notably, all configurations remain superior to Constant Weight MTL, confirming that DRO provides value even with suboptimal group settings. More details are provided in Appendix A.22.

**Sensitivity to Batch Size.** Figure 6(C) reveals that small batch sizes catastrophically degrade DRO performance due to unreliable group coverage and noisy loss estimates. At  $B = 8$ , performance degrades by 41.5% to 60.4% relative to optimal and becomes significantly worse than Single Task baseline (12.6% to 28.1% degradation), demonstrating that DRO with unreliable estimates is more harmful than no robustness at all. Even at  $B = 12$ , all datasets remain below Single Task performance. Performance stabilizes around  $B = 32$ , with marginal gains ( $<1.5\%$ ) beyond. More details are provided in Appendix A.23.

**Sensitivity to Robustness Parameter  $\eta$ .** Figure 6(D) reveals a U-shaped performance curve with notable asymmetry. When  $\eta$  is too small ( $\eta \leq 0.5$ ), DRO approaches standard averaging with 6.3% to 14.4% average MSE degradation. When  $\eta$  is too large, over-compensation

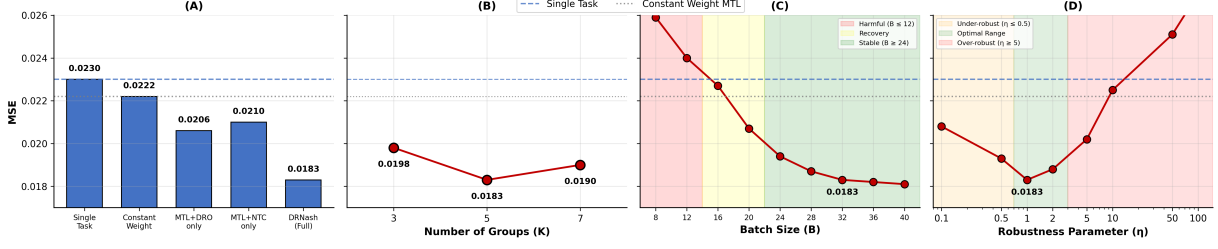


Figure 6: Ablation studies using Twitter-RoBERTa-Large on INV dataset. (A) Component Ablation. (B) Sensitivity to  $K$ . (C) Sensitivity to Batch Size. (D) Sensitivity to Robustness Parameter  $\eta$ . Percentages in the text are aggregated over the three datasets; per-panel values are in Appendix A.21–A.25.

| Region         | Range            | Freq. (%) | CW MSE | DRNash MSE | $\Delta$ MSE (%) |
|----------------|------------------|-----------|--------|------------|------------------|
| R1 Strong Neg. | $[-1.00, -0.30]$ | 14.5      | 0.0356 | 0.0268     | -24.7            |
| R2 Negative    | $[-0.30, -0.10]$ | 23.3      | 0.0184 | 0.0160     | -13.0            |
| R3 Neutral     | $[-0.10, 0.10]$  | 24.0      | 0.0158 | 0.0142     | -10.1            |
| R4 Positive    | $[0.10, 0.30]$   | 25.8      | 0.0170 | 0.0149     | -12.4            |
| R5 Strong Pos. | $[0.30, 1.00]$   | 12.4      | 0.0375 | 0.0278     | -25.9            |
| Overall        | $[-1.00, 1.00]$  | 100       | 0.0222 | 0.0183     | -17.6            |

Table 6: Error analysis by sentiment region on INV (Twitter-RoBERTa-Large). CW: Constant Weight MTL.  $R_1$ – $R_5$  are fixed intervals, not quantile training groups.

causes more severe degradation (28.9% to 57.5% at  $\eta \geq 10$ ), and at  $\eta \geq 50$ , performance becomes worse than Single Task baseline as DRO actively misallocates optimization effort. More details are provided in Appendix A.24.

**Where the Gains Come From.** Table 6 decomposes the overall MSE reduction by sentiment region. The improvement is not uniform: DRNash reduces error by 24.7% and 25.9% in the extreme regions  $R_1$  and  $R_5$ , which together hold only 26.9% of the data, versus 10.1% in the neutral region  $R_3$ . Since worst-case aggregation upweights the groups with the highest current loss, and the extreme regions are exactly those the model fits worst under uniform averaging, this asymmetry is the intended behaviour. The aggregate 17.6% reduction thus comes from the underrepresented tails that carry the most actionable financial signal, not from the majority neutral region.

## 5 Related Work

**Sentiment Analysis.** Sentiment analysis has been widely used across domains, especially in finance where accurate sentiment extraction can improve forecasting. Saravanos and Kanavos (2025) and Agrawal et al. (2025) report gains in crude oil and stock prediction with sentiment-aware models. Xing et al. (2025) and Baklanova et al. (2024) further link NFT-related tweet sentiment to market

trends. Beyond finance, Wen et al. (2024) evaluates product competitiveness from online reviews, and Li et al. (2025) studies drivers of travel happiness. Despite their effectiveness in leveraging unstructured data, most existing work centers on classification and leaves the distribution imbalance in continuous sentiment regression underexplored.

**Multi-task Learning.** Multi-task learning can alleviate imbalance by adding complementary supervision. Cai et al. (2025) and Abdelsamie et al. (2026) show that pairing classification with regression better captures sentiment tendencies, reduces noise, and mitigates sparsity. Yet it raises two coupled issues: defining group boundaries for the auxiliary task and resolving gradient conflicts between objectives (Chen et al., 2024; Chen and Er, 2025; Fontana et al., 2024). Prior methods often use fixed task weights and thus fail to adapt to changing optimization dynamics during training.

**PEFT.** Fine-tuning LLMs for sentiment regression is resource-intensive, motivating parameter-efficient adaptation. Hu et al. (2023) propose LLM-Adapters, achieving competitive performance with fewer trainable parameters, while Lei et al. (2023) introduce Conditional Adapters for efficient transfer and faster inference; Han et al. (2024) and Wang et al. (2025) survey this area. Hu et al. (2022) presents LoRA, which updates weights through low-rank matrices with minimal memory and latency overhead.

## 6 Conclusion

We presented DRNash, which couples quantile-based distributionally robust optimization with a Nash bargaining task coordinator to jointly address data imbalance and gradient conflict in financial sentiment regression. It improves MSE by 14.1% and ACC by 1.92%, with gains on INV concentrated in the underrepresented extreme regions, and transfers beyond finance.

## Limitations

This work has several limitations that should be acknowledged:

- **Language Scope.** Our experiments are conducted exclusively on English datasets. The effectiveness of DRNash on other languages, particularly those with different syntactic structures or limited pre-trained model support, has not been validated.
- **Deployment Considerations.** This work focuses on offline training and evaluation. Aspects such as real-time inference latency, on-line adaptation to distribution drift, and integration with production trading systems are beyond the scope of this study.
- **Interpretability.** While DRNash demonstrates consistent performance improvements, fine-grained analysis of which linguistic features contribute to better predictions in specific sentiment regions is not addressed in this work.

## Ethical Considerations

**Data Usage.** The financial datasets (NEU, FXE, INV) are sourced from publicly accessible financial news websites (investing.com and forexempire.com), collected following the methodology described in prior published research with proper citation, while cross-domain datasets (SemEval-2017, SemEval-2018, EmoBank) are publicly available benchmarks. All datasets are used consistent with their intended purposes.

**Privacy.** All experiments use financial news articles and general text data that do not contain personally identifiable information.

**Potential Risks.** Financial sentiment analysis tools could potentially be misused for market manipulation. We emphasize responsible deployment and adherence to financial regulations in real-world applications.

**Environmental Impact.** Our LoRA integration reduces training costs by approximately 50% compared to full fine-tuning, contributing to more efficient and environmentally conscious model development.

## Acknowledgments

This work was supported by the Zhejiang Provincial Education Science Planning Project (No. 2025SCG214). We thank the anonymous reviewers for their constructive comments, which substantially improved this paper.

## References

- Mahmoud Mohamed Abdelsamie, Shahira Shaaban Azab, and Hesham A Hefny. 2026. Deep multi-task learning: a review of concepts, methods, and cross-domain applications. *International Journal of Data Science and Analytics*, 21(1):77.
- Shubham Agrawal, Nitin Kumar, Geetanjali Rathee, Chaker Abdelaziz Kerrache, Carlos T Calafate, and Muhammad Bilal. 2025. Improving stock market prediction accuracy using sentiment and technical analysis. *Electronic Commerce Research*, 25(5).
- Imane Araf, Ali Idri, and Ikram Chairi. 2024. [Cost-sensitive learning for imbalanced medical data: a review](#). *Artificial Intelligence Review*, 57(4):80.
- Gabriel O Assunção, Rafael Izbicki, and Marcos O Prates. 2024. [Is augmentation effective in improving prediction in imbalanced datasets?](#) *Journal of Data Science*.
- Valeriia Baklanova, Aleksei Kurkin, and Tamara Teplova. 2024. [Investor sentiment and the nft hype index: to buy or not to buy?](#) *China Finance Review International*, 14(3):522–548.
- Ahmed Bouteska, Mohammad Zoynul Abedin, Petr Hajek, and Kunpeng Yuan. 2024. [Cryptocurrency price forecasting – a comparative analysis of ensemble learning and deep learning methods](#). *International Review of Financial Analysis*, 92:103055.
- Paula Branco, Luís Torgo, and Rita P Ribeiro. 2017. Smogn: a pre-processing approach for imbalanced regression. In *First international workshop on learning with imbalanced domains: Theory and applications*, pages 36–50. PMLR.
- Sven Buechel and Udo Hahn. 2017. [Emobank: Studying the impact of annotation perspective and representation format on dimensional emotion analysis](#). In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 578–585.
- Yujian Cai, Xingguang Li, Yingyu Zhang, Jinsong Li, Fazheng Zhu, and Lin Rao. 2025. [Multimodal sentiment analysis based on multi-layer feature fusion and multi-task learning](#). *Scientific Reports*, 15(1):2126.
- Wenzhi Cao, Vahid Mirjalili, and Sebastian Raschka. 2020. [Rank consistent ordinal regression for neural networks with application to age estimation](#). *Pattern Recognition Letters*, 140:325–331.

- Jie Chen and Meng Joo Er. 2025. [Mitigating gradient conflicts via expert squads in multi-task learning](#). *Neurocomputing*, 614:128832.
- Shijie Chen, Yu Zhang, and Qiang Yang. 2024. [Multi-task learning in natural language processing: An overview](#). *ACM Computing Surveys*, 56(12):1–32.
- Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. 2018. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International Conference on Machine Learning*, pages 794–803. PMLR.
- Keith Cortis, André Freitas, Tobias Daudert, Manuela Huerlimann, Manel Zarrouk, Siegfried Handschuh, and Brian Davis. 2017. [Semeval-2017 task 5: Fine-grained sentiment analysis on financial microblogs and news](#). In *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)*, pages 519–535.
- Milivoje Davidovic and Jacqueline McCleary. 2025. [News sentiment and stock market dynamics: A machine learning investigation](#). *Journal of Risk and Financial Management*, 18(8):412.
- Jean-Antoine Désidéri. 2012. [Multiple-gradient descent algorithm \(mgda\) for multiobjective optimization](#). *Comptes Rendus Mathématique*, 350(5-6):313–318.
- Hongcheng Ding, Fuzhen Hu, Ruiting Deng, Xuanze Zhao, Shamsul Nahar Abdullah, and Deshinta Arrova Dewi. 2024a. [Dynamic adaptive rank space exploration for efficient sentiment analysis with large language models](#). *arXiv preprint arXiv:2410.16589*.
- Hongcheng Ding, Xiangyu Shi, Ruiting Deng, Salaar Farooq, Deshinta Arrova Dewi, Shamsul Nahar Abdullah, and Bahiah A Malek. 2024b. [Eur/usd exchange rate forecasting incorporating text mining based on pre-trained language models and deep learning methods](#). *arXiv preprint arXiv:2411.07560*.
- Hongcheng Ding, Xuanze Zhao, Ruiting Deng, Shamsul Nahar Abdullah, and Deshinta Arrova Dewi. 2024c. [Eur/usd exchange rate forecasting based on information fusion with large language models and deep learning methods](#). *arXiv preprint arXiv:2408.13214*.
- Kelvin Du, Frank Xing, Rui Mao, and Erik Cambria. 2024. [Financial sentiment analysis: Techniques and applications](#). *ACM Computing Surveys*, 56(9):1–42.
- John C Duchi and Hongseok Namkoong. 2021. [Learning models with uniform performance via distributionally robust optimization](#). *The Annals of Statistics*, 49(3):1378–1406.
- Said Al Afghani Edsa, Khamron Sunat, and Haocheng Guo. 2025. [Cost-sensitive neural network: A grey wolf optimizer-based approach for breast cancer prediction](#). *Expert Systems with Applications*, 266:126176.
- Maxime Fontana, Michael Spratling, and Miaoqing Shi. 2024. [When multitask learning meets partial supervision: A computer vision review](#). *Proceedings of the IEEE*, 112(6):516–543.
- Emanuele Francazi, Marco Baity-Jesi, and Aurelien Lucchi. 2023. A theoretical analysis of the learning dynamics under class imbalance. In *International Conference on Machine Learning*, pages 10285–10322. PMLR.
- Zijun Gao, Haibao Liu, and Lingbo Li. 2025. [Data augmentation for time-series classification: An extensive empirical study and comprehensive survey](#). *Journal of Artificial Intelligence Research*, 83.
- Gemma Team. 2024. [Gemma 2: Improving open language models at a practical size](#). *arXiv preprint arXiv:2408.00118*.
- Xin Geng. 2016. [Label distribution learning](#). *IEEE Transactions on Knowledge and Data Engineering*, 28(7):1734–1748.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, et al. 2024. [The llama 3 herd of models](#). *arXiv preprint arXiv:2407.21783*.
- Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. 2024. [Parameter-efficient fine-tuning for large models: A comprehensive survey](#). *Transactions on Machine Learning Research*.
- 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*.
- Zhaolin Hu and L. Jeff Hong. 2012. [Kullback-leibler divergence constrained distributionally robust optimization](#). Technical report, Optimization Online.
- Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Ka-Wei Lee. 2023. [Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*.
- Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7482–7491.
- Salman H Khan, Munawar Hayat, Mohammed Benamoun, Ferdous A Sohel, and Roberto Togneri. 2018. Cost-sensitive learning of deep feature representations from imbalanced data. *IEEE transactions on neural networks and learning systems*, 29(8):3573–3587.

- Jihie Kim, Jaebong Yoo, Ho Lim, Huida Qiu, Zornitsa Kozareva, and Aram Galstyan. 2013. Sentiment prediction using collaborative filtering. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 7, pages 685–688.
- Wenna Lai, Haoran Xie, Guandong Xu, and Qing Li. 2026. Multi-task learning with llms for implicit sentiment analysis: Data-level and task-level automatic weight learning. *IEEE Transactions on Knowledge and Data Engineering*, 38(1):506–517.
- Tao Lei, Junwen Bai, Siddhartha Brahma, Joshua Ainslie, Kenton Lee, Yanqi Zhou, Nan Du, Vincent Zhao, Yuexin Wu, Bo Li, et al. 2023. Conditional adapters: Parameter-efficient transfer learning with fast inference. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, pages 8152–8172.
- Xuerong Li, Wei Shang, and Shouyang Wang. 2019. Text-based crude oil price forecasting: A deep learning approach. *International Journal of Forecasting*, 35(4):1548–1560.
- Yong-Quan Li, Rui Li, Wen-Qi Ruan, and Shu-Ning Zhang. 2025. Releasing happiness: Influencing factors and configuration paths of shared accommodation customers’ positive emotions. *Tourism Management*, 108:105113.
- Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.
- Bing Liu. 2022. *Sentiment analysis and opinion mining*. Springer Nature.
- Bo Liu, Yihao Feng, Peter Stone, and Qiang Liu. 2023. Famo: Fast adaptive multitask optimization. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, pages 57226–57243.
- Liyang Liu, Yi Li, Zhanghui Kuang, Jing-Hao Xue, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. 2021. Towards impartial multi-task learning. In *International conference on learning representations*.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.
- Daniel Loureiro, Kiamehr Rezaee, Talayeh Riahi, Francesco Barbieri, Leonardo Neves, Luis Espinosa Anke, and Jose Camacho-Collados. 2023. Tweet insights: A visualization platform to extract temporal insights from twitter. *arXiv preprint arXiv:2308.02142*.
- Saif Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018. Semeval-2018 task 1: Affect in tweets. In *Proceedings of the 12th international workshop on semantic evaluation*, pages 1–17.
- M. Mujahid, E. Kina, F. Rustam, et al. 2024. Data oversampling and imbalanced datasets: an investigation of performance for machine learning and feature engineering. *Journal of Big Data*, 11:87.
- John F. Nash. 1950. The bargaining problem. *Econometrica*, 18(2):155–162.
- Zakaria Nassr, Fatima Benabbou, Nouredine Sael, and Taher Hamim. 2025. Improving sentiment analysis performance on imbalanced moroccan dialect datasets using resample and feature extraction techniques. *Information*, 16(1):39.
- Aviv Navon, Aviv Shamsian, Idan Achituve, Haggai Maron, Kenji Kawaguchi, Gal Chechik, and Ethan Fetaya. 2022. Multi-task learning as a bargaining game. In *International Conference on Machine Learning*.
- An Tuan Nguyen and Nhung Thi Nguyen. 2024. How does investor sentiment affect stock market crash risk? evidence from asia-pacific markets. *Cogent Economics & Finance*, 12(1):2422959.
- Qwen Team. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.
- Jiawei Ren, Mingyuan Zhang, Cunjun Yu, and Ziwei Liu. 2022. Balanced mse for imbalanced visual regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7926–7935.
- Sara Rosenthal, Noura Farra, and Preslav Nakov. 2017. Semeval-2017 task 4: Sentiment analysis in twitter. In *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)*, pages 502–518.
- Shiori Sagawa, Pang Wei Koh, Tatsunori B. Hashimoto, and Percy Liang. 2020. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. In *International Conference on Learning Representations*.
- Hassan Saif, Yulan He, and Harith Alani. 2012. Alleviating data sparsity for twitter sentiment analysis. In *Proceedings of the 2nd Workshop on Making Sense of Microposts (#MSM2012)*, pages 2–9. CEUR Workshop Proceedings (CEUR-WS. org).
- Christina Saravanos and Andreas Kanavos. 2025. Forecasting stock market volatility using social media sentiment analysis. *Neural Computing and Applications*, 37(17):10771–10794.
- David Schaudt, Rico von Schwerin, Andreas Hafner, et al. 2023. Augmentation strategies for an imbalanced learning problem on a novel covid-19 severity dataset. *Scientific Reports*, 13(1):18299.

- Ozan Sener and Vladlen Koltun. 2018. Multi-task learning as multi-objective optimization. In *Advances in neural information processing systems*, volume 31.
- 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.
- Yoojeong Song. 2024. [Enhancing stock market trend reversal prediction using feature-enriched neural networks](#). *Heliyon*, 10(2).
- Jingru Tan, Xin Lu, Gang Zhang, Changqing Yin, and Quanquan Li. 2021. [Equalization loss v2: A new gradient balance approach for long-tailed object detection](#). In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1685–1694.
- Luping Wang, Sheng Chen, Linnan Jiang, Shu Pan, Runze Cai, Sen Yang, and Fei Yang. 2025. [Parameter-efficient fine-tuning in large language models: a survey of methodologies](#). *Artificial Intelligence Review*, 58(8):227.
- Zaitian Wang, Pengfei Wang, Kunpeng Liu, Pengyang Wang, Yanjie Fu, Chang-Tien Lu, Charu C. Aggarwal, Jian Pei, and Yuanchun Zhou. 2026. [A comprehensive survey on data augmentation](#). *IEEE Transactions on Knowledge and Data Engineering*, 38(1):47–66.
- Zirui Wang, Yulia Tsvetkov, Orhan Firat, and Yuan Cao. 2021. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. In *International Conference on Learning Representations*.
- Zhanming Wen, Yanjun Chen, Hongwei Liu, and Zhouyang Liang. 2024. [Text mining based approach for customer sentiment and product competitiveness using composite online review data](#). *Journal of Theoretical and Applied Electronic Commerce Research*, 19(3):1776–1792.
- Jin Xiao, Sihan Li, Yuhang Tian, Jing Huang, Xiaoyi Jiang, and Shouyang Wang. 2025. [Example dependent cost sensitive learning based selective deep ensemble model for customer credit scoring](#). *Scientific Reports*, 15(1):6000.
- Yunfei Xing, Justin Z Zhang, Yuming He, and Yueqi Li. 2025. [Toward an ecosystem of non-fungible tokens from mapping public opinions on social media](#). *Financial Innovation*, 11(1):6.
- An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. [Qwen2 technical report](#). *arXiv preprint arXiv:2407.10671*.
- Yuzhe Yang, Kaiwen Zha, Yingcong Chen, Hao Wang, and Dina Katabi. 2021. Delving into deep imbalanced regression. In *International conference on machine learning*, pages 11842–11851. PMLR.
- Yuanzhou Yao, Zhao Zhang, Kaijia Yang, Huasheng Liang, Qiang Yan, and Yongjun Xu. 2023. [An auxiliary task boosted multi-task learning method for service account retrieval with limited human annotation](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 522–531.
- Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. [Gradient surgery for multi-task learning](#). In *Advances in Neural Information Processing Systems*, volume 33, pages 5824–5836.
- Yi Zeng, Xuelin Yang, Li Chen, Cristian Canton Ferrer, Ming Jin, Michael I. Jordan, and Ruoxi Jia. 2024. [Fairness-aware meta-learning via nash bargaining](#). In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*, pages 83235–83267.
- Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. 2024. [Tinyllama: An open-source small language model](#). *arXiv preprint arXiv:2401.02385*.
- Yanzhao Zou and Dorien Herremans. 2023. [Prebit—a multimodal model with twitter finbert embeddings for extreme price movement prediction of bitcoin](#). *Expert Systems with Applications*, 233:120838.

## Table of Contents

|                                                            |           |
|------------------------------------------------------------|-----------|
| <b>The Use of Large Language Models (LLMs)</b>             | <b>14</b> |
| <b>A.1 Imbalanced Data Distribution</b>                    | <b>14</b> |
| <b>A.2 Inherent Task Conflict</b>                          | <b>15</b> |
| <b>A.3 Coupling Between Challenges</b>                     | <b>16</b> |
| <b>A.4 Multi-task Learning Framework</b>                   | <b>17</b> |
| <b>A.5 Distributionally Robust Optimization</b>            | <b>18</b> |
| <b>A.6 Sentiment Group Configuration</b>                   | <b>19</b> |
| <b>A.7 Theoretical Foundation of DRO</b>                   | <b>21</b> |
| <b>A.8 Nash Task Coordinator</b>                           | <b>22</b> |
| <b>A.9 NTC Derivations</b>                                 | <b>23</b> |
| <b>A.10 Complexity Analysis</b>                            | <b>25</b> |
| <b>A.11 Benchmark</b>                                      | <b>26</b> |
| <b>A.12 Evaluation Metrics</b>                             | <b>26</b> |
| <b>A.13 Dataset</b>                                        | <b>28</b> |
| <b>A.14 Hyperparameters</b>                                | <b>28</b> |
| <b>A.15 DRNash vs Constant Weight</b>                      | <b>29</b> |
| <b>A.16 Effectiveness of Auxiliary Classification Task</b> | <b>29</b> |
| <b>A.17 Traditional Imbalanced Learning Methods</b>        | <b>31</b> |
| <b>A.18 LoRA Integration</b>                               | <b>35</b> |
| <b>A.19 Cross-Domain Generalization</b>                    | <b>37</b> |
| <b>A.20 Pre-training Bias Analysis</b>                     | <b>38</b> |
| <b>A.21 Component Ablation Study</b>                       | <b>40</b> |
| <b>A.22 Sensitivity: Group Number <math>K</math></b>       | <b>42</b> |
| <b>A.23 Sensitivity: Batch Size</b>                        | <b>44</b> |
| <b>A.24 Sensitivity: Parameter <math>\eta</math></b>       | <b>48</b> |
| <b>A.25 Ablation Study Visualizations</b>                  | <b>51</b> |

## The Use of Large Language Models (LLMs)

During the preparation of this work, the authors used large language models to translate an initial draft into English, to check grammar, to suggest figure layouts, and to help identify bugs in the implementation.

### Appendix A.1: Imbalanced Data Distribution

In the sentiment polarity analysis task for exchange rate texts, we observe that the performance of

LLMs is unsatisfactory after fine-tuning on the standard and customized financial text dataset (Ding et al., 2024b). This can be attributed to the model’s lack of exposure to news domain-specific texts containing specialized jargon, implicit sentiments, and subtle variations. The financial domain presents unique challenges including economic terminology, market-specific expressions, and complex sentiment patterns that differ significantly from general text corpora.

Subsequently, we employ an alternative LLM, Twitter-RoBERTa-Large (Loureiro et al., 2023) pre-trained on a large-scale tweet corpus, and the model’s performance shows a slight improvement. This domain adaptation demonstrates the importance of pre-training data alignment with target applications.

Figure 7 provides comprehensive empirical evidence for the severe data distribution imbalance challenge that fundamentally underlies these performance issues. The training set distribution shown in Figure 7(A) concentrates almost entirely in the moderate sentiment band, with peak frequencies of roughly 14% near  $-0.2$  to  $-0.1$  and again near  $0.2$  to  $0.3$ . The extreme regions ( $|y| > 0.5$ ) together account for only about 11.5% of the training data, and the most extreme bins ( $|y| > 0.7$ ) for under 2% each, even though these are precisely the regions that carry the most actionable information in financial applications.

Figure 7(B) shows that the test set follows the same profile: the two distributions agree closely across the entire polarity range, including the sparsity of both tails. The imbalance is therefore not an artifact of how the data were partitioned but a property of the corpus itself, and it is reproduced at evaluation time rather than corrected by it.

The most critical finding appears in Figure 7(C), which reveals that LLM predictions exhibit severe bias toward the neutral sentiment region. The model predictions show extreme concentration around the neutral region from approximately  $-0.2$  to  $0.2$ , with peak frequencies of about 16% in the near-neutral range. The model demonstrates virtually no capability to predict extreme sentiments, with predictions rarely extending beyond  $\pm 0.4$  on the sentiment scale. This prediction pattern suggests that the model has learned to default to safe, neutral predictions rather than risk extreme sentiment classifications, indicating over-regularization toward the training set’s most frequent regions.

Figure 7(D) overlays the three distributions and

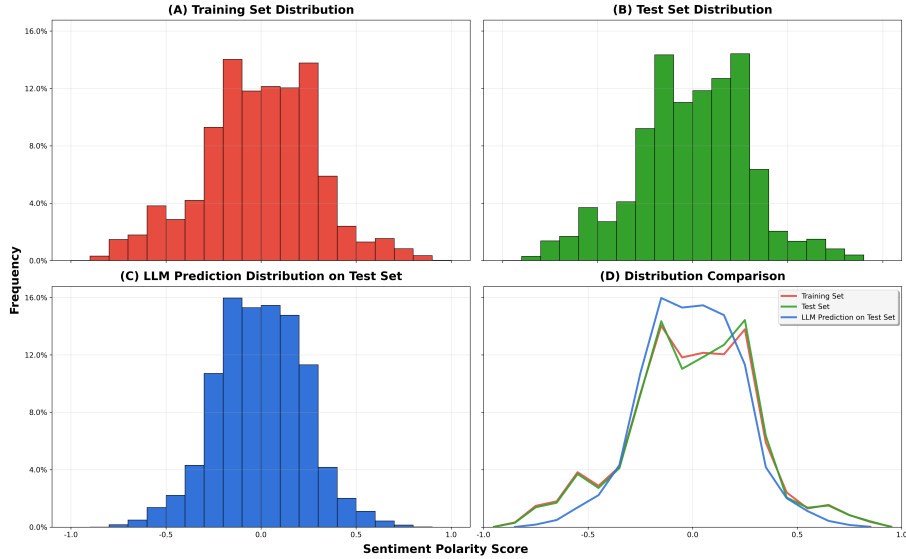


Figure 7: Data distribution analysis revealing imbalance challenges in sentiment analysis, including: (A) Training set distribution, (B) Test set distribution, (C) Fine-tuned LLM predictions on the test set, and (D) Comparative analysis.

isolates where the discrepancy actually lies. The training and test curves are nearly indistinguishable, whereas the prediction curve departs from both in the same direction: it is higher across the central range and markedly lower in both tails. The model therefore does not merely inherit the imbalance already present in the data, it amplifies it, compressing its outputs toward the region that dominates the training distribution and vacating the sparse extremes altogether. This is the failure mode that motivates distributionally robust optimization: reweighting the underrepresented sentiment regions during training, rather than correcting a mismatch between the training and evaluation splits.

## Appendix A.2: Inherent Task Conflict

To address the data imbalance issues identified in the previous section, we incorporate a classification task alongside the regression task to better capture sentiment tendencies embedded in the text through multi-task learning. However, our multi-task learning experiments reveal a fundamental challenge: the model’s performance exhibits significant sensitivity to task weight configurations, indicating an inherent conflict between the two task objectives that cannot be resolved through simple weight adjustment strategies.

Figure 8 demonstrates the critical impact of task weight allocation on model performance across both regression and classification objectives. The

left panel shows that as the weight of the classification task increases from 0.05 to 0.3, the MSE exhibits a non-monotonic trajectory, starting at approximately 0.0212, reaching a peak of 0.0237 at weight 0.2, with 0.0231 at weight 0.3. The right panel reveals an even more complex relationship, with accuracy ranging from a minimum of approximately 75.8% at weight 0.2 to a peak of around 77.0% at weight 0.25.

The experimental results reveal that the regression and classification tasks operate at fundamentally different scales and exhibit varying learning dynamics that cannot be effectively balanced through static weight allocation. The regression task, which predicts continuous sentiment scores, operates on a different loss magnitude scale compared to the classification task, while the tasks also exhibit different convergence rates. The sensitivity analysis demonstrates that small changes in weight allocation can lead to substantial performance degradation, with shifting the classification weight from the optimal value of 0.25 to 0.2 increasing MSE and reducing accuracy by 1.2 points.

**Gradient Conflict Analysis.** Beyond the weight sensitivity issue, we observe that the performance surface is non-convex with respect to task weights, and the optimal weight for MSE differs from that for ACC. This reflects an inherent conflict between regression and classification gradients during optimization, rather than merely a hyperparameter tuning problem. The gradients from different tasks

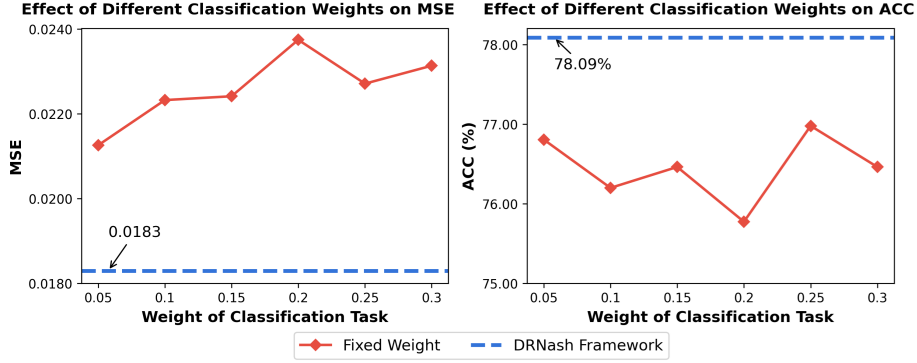


Figure 8: Effect of different task weights on MSE and ACC in multi-task learning, compared with the performance of our proposed DRNash framework.

may point in conflicting directions, making it impossible for any fixed weight configuration to simultaneously optimize both objectives.

In contrast, our proposed DRNash framework, represented by the blue dashed lines in Figure 8, maintains consistently superior performance with a stable MSE of 0.0183 and accuracy of 78.09%, demonstrating substantial improvements over the best fixed weight configuration and validating the effectiveness of Nash bargaining-based task weighting in addressing the fundamental limitations of static multi-task learning approaches.

### Appendix A.3: Coupling Between Challenges

The coupling between data-level imbalance and task-level conflicts in multi-task learning has been established through theoretical analysis and empirical validation across multiple research communities, providing a solid foundation for our unified framework design.

**Gradient Domination by Over-Represented Samples.** Theoretical analysis has established that gradient estimates under class imbalance are systematically biased toward majority samples. Francazi et al. (2023) provide formal convergence analysis proving that gradient descent dynamics are dominated by majority-class gradients, with minority class performance initially deteriorating because reducing majority-class loss outweighs the loss increase from minority samples. Their theoretical framework demonstrates that the angle between per-class gradients must satisfy increasingly restrictive conditions as imbalance grows, fundamentally constraining optimization. This gradient domination phenomenon was first empirically observed in object detection by Lin et al. (2017), who showed

that easy negative samples (analogous to neutral sentiment in our setting) overwhelm training gradients, motivating their focal loss formulation that down-weights well-classified examples. Tan et al. (2021) further established that the gradient ratio between over-represented and under-represented categories serves as a reliable indicator of training imbalance in long-tailed recognition, directly supporting our use of distributionally robust optimization to reweight gradient contributions across sentiment groups.

**Empirical Evidence for the Coupling Between Imbalance and Task Conflicts.** The interaction between sample-level imbalance and task-level gradient conflicts has been documented across diverse multi-task settings. Yu et al. (2020) identify a “tragic triad” of optimization failures in multi-task learning: conflicting gradient directions, high curvature, and large differences in gradient magnitudes, with the last factor directly resulting from data distribution imbalance. Their experiments demonstrate that when gradient magnitudes differ significantly due to imbalanced task difficulty or data distribution, the larger gradient dominates parameter updates, causing systematic performance degradation on other tasks. Chen et al. (2018) show that task imbalance manifests as imbalances between backpropagated gradients, with their NYUv2 experiments revealing that without gradient normalization, tasks with larger loss scales overfit quickly while others underperform. Liu et al. (2021) document the “partial training” phenomenon where gradient imbalance causes specific tasks to be systematically under-optimized, demonstrating that gradient balance and loss balance address complementary aspects of the multi-task optimization problem. These findings collectively support our

hypothesis that data distribution bias in sentiment regression propagates to task-level conflicts when combined with auxiliary classification.

**Unreliability of Gradient Magnitudes as Task Difficulty Indicators.** Gradient magnitudes confound loss scale effects with learning difficulty, rendering them unreliable for task balancing under data imbalance. [Chen et al. \(2018\)](#) explicitly identify this disconnect, noting that tasks with high output variance backpropagate larger gradients regardless of their actual learning difficulty, creating a paradox where such tasks dominate optimization even when they are easier to learn. Recent theoretical analysis ([Navon et al., 2022](#)) establishes that while gradient magnitude correlates with learning gains within a single task, this correlation breaks down across tasks because the smoothness constant differs, meaning large-gradient tasks can achieve similar or lower learning gains than small-gradient ones. [Yu et al. \(2020\)](#) provide empirical evidence through ablation studies showing that magnitude-only modifications underperform methods that also address gradient direction, proving that magnitude-based weighting is fundamentally limited. [Kendall et al. \(2018\)](#) demonstrate that uncertainty-based weighting, which uses output variance as task weights, produces suboptimal results under imbalance because variance estimates are confounded by sample frequency effects rather than reflecting true task uncertainty. These findings motivate our Nash-based task coordinator, which uses scale-invariant bargaining rather than gradient magnitude to determine task weights.

In summary, the theoretical and empirical evidence from the multi-task learning literature establishes a causal chain from data imbalance to gradient domination to unreliable task weighting, indicating that these two challenges must be addressed jointly rather than in isolation. This motivates our unified DRNash framework that simultaneously handles group construction and task balancing.

#### Appendix A.4: Multi-task Learning Framework

This subsection provides detailed technical specifications of our multi-task learning framework that serves as the foundation for the DRNash framework integration. The framework is designed to be model-agnostic and has been validated across multiple LLM architectures including RoBERTa-Large, TinyLlama-1.1B, and Qwen2-0.5B.

**Backbone: LLMs.** In sentiment analysis, LLMs serve as text data embedding generators, concatenating various task-specific heads for various sentiment analysis tasks. This work employs different LLM architectures as the embedding generator for text data, as illustrated in Figure 9 (A). The training set is defined as  $D = \{(x_i, y_i)\}_{i=1}^N$ , where  $N$  denotes the total number of texts,  $x_i$  represents each text in the dataset, and  $y_i \in [-1, 1]$  corresponds to the annotated sentiment polarity score.

The pre-trained LLM tokenizer  $\text{Tokenizer}(\cdot)$  processes each text  $x_i$  in the dataset, and then the tokenized text passes through multiple Transformer blocks to generate the final hidden state  $H_i$ :

$$H_i = \text{LLM}(\text{Tokenizer}(x_i)), \quad (13)$$

where  $H_i \in \mathbb{R}^{1 \times d_{\text{model}}}$  and  $d_{\text{model}}$  represents the hidden dimension of the specific LLM architecture (1024 for RoBERTa-Large, 2048 for TinyLlama-1.1B, and 896 for Qwen2-0.5B). This hidden state serves as input to task-specific heads. For sentiment polarity analysis, we implement a regression head (Figure 9 (B)) comprising two linear layers and a sigmoid activation function:

$$S_i = \text{LL}_2(\sigma(\text{LL}_1(H_i))), \quad (14)$$

where  $\text{LL}_1 : \mathbb{R}^{d_{\text{model}}} \rightarrow \mathbb{R}^{128}$  is the first linear layer,  $\sigma(\cdot)$  denotes the sigmoid function, and  $\text{LL}_2 : \mathbb{R}^{128} \rightarrow \mathbb{R}$  produces the final sentiment polarity score  $S_i \in [-1, 1]$ .

For the regression task, we use Mean Squared Error (MSE) as the loss function, denoted as  $L_r$ :

$$L_r = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2, \quad (15)$$

where  $n$  denotes the number of texts in the training set,  $\hat{y}_i$  represents the predicted polarity score, and  $y_i$  is the ground truth sentiment polarity score.

**Multi-task Learning Architecture.** In sentiment analysis, particularly in data-limited scenarios, enhancing model performance is crucial. We introduce a multi-task learning approach that operates consistently across different LLM architectures. Specifically, we incorporate a classification task alongside the regression task to better capture the sentiment tendencies embedded in the text. This approach mitigates the impact of noise and small fluctuations while effectively alleviating the data sparsity problem often encountered in sentiment analysis ([Saif et al., 2012](#); [Kim et al., 2013](#)).

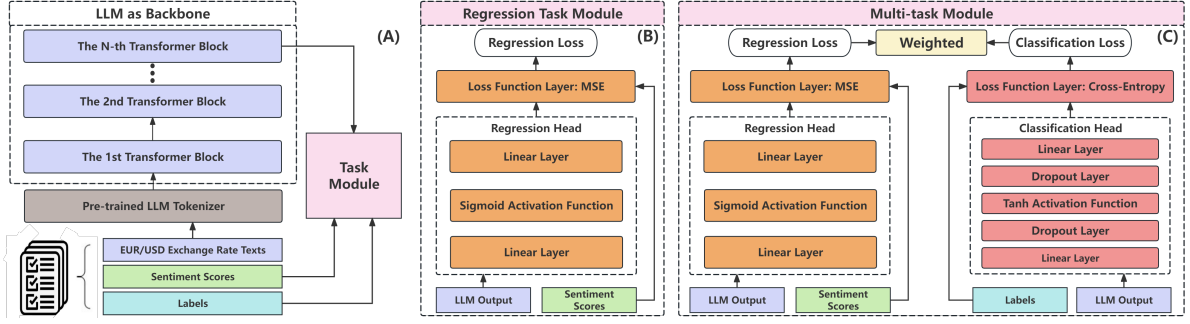


Figure 9: The diagram outlines the application of LLMs for sentiment analysis, segmented into three main parts: (A) LLMs as the backbone combined with a task-specific module, (B) a regression task module, and (C) a multi-task module that handles both regression and classification tasks. Our DRNash framework operates on this architecture by applying DRO for balanced group-wise optimization and NTC for dynamic task weighting.

### Quantile-based Sentiment Group Construction.

As described in Section 3.3 of the main text, we partition the sentiment space into  $K$  discrete groups based on quantile boundaries. Let  $\tau_k = Q_{k/K}(\{y_i\}_{i=1}^N)$  denote the  $k/K$ -quantile of training sentiment scores. The  $k$ -th sentiment group is defined as:

$$G_k = \{(x_i, y_i) : \tau_{k-1} < y_i \leq \tau_k\}, \quad (16)$$

where  $k = 1, \dots, K$ ,  $\tau_0 = -1$ , and  $\tau_K = 1$  (with  $G_1$  extended to include  $y_i = \tau_0$ , so every sample receives a group). Each sample is assigned a group label  $c_i \in \{1, \dots, K\}$  indicating which group it belongs to. This quantile-based construction ensures approximately equal sample counts per group ( $|G_k| \approx N/K$ ), providing balanced coverage from strongly negative ( $G_1$ ) through neutral ( $G_{\lceil K/2 \rceil}$ ) to strongly positive ( $G_K$ ). For the classification task, the group label  $c_i$  serves as the ground truth, transforming sentiment classification into predicting the intensity region of the input text. Consequently, we redefine our dataset as  $D = \{(x_i, y_i, c_i)\}_{i=1}^N$ .

For the classification task, we implement a classification head as illustrated in Figure 9 (C). This head comprises two dropout layers, two linear layers, and a Tanh activation function, with input dimensions adapted to the specific LLM architecture. The process can be formalized as follows:

$$H'_i = \text{LL}_1(\text{DP}_1(H_i)), \quad (17)$$

where  $H'_i \in \mathbb{R}^{1 \times 128}$  is an intermediate representation,  $\text{DP}_1(\cdot)$  is the first dropout layer for regularization, and  $\text{LL}_1 : \mathbb{R}^{d_{\text{model}}} \rightarrow \mathbb{R}^{128}$  is the first linear layer that adapts to the hidden dimension of the respective LLM.

The classification logits  $C_i$  for the  $i$ -th text are

then computed as:

$$C_i = \text{LL}_2(\text{DP}_2(\text{Tanh}(H'_i))), \quad (18)$$

where  $\text{Tanh}(\cdot)$  introduces non-linearity,  $\text{DP}_2(\cdot)$  is the second dropout layer, and  $\text{LL}_2 : \mathbb{R}^{128} \rightarrow \mathbb{R}^K$  maps to the  $K$  discrete classification labels corresponding to sentiment groups.

For the classification task, we use Cross-Entropy (CE) loss to measure the difference between predicted probabilities and actual group labels. The classification loss  $L_c$  is formulated as:

$$L_c = -\frac{1}{n} \sum_{i=1}^n \sum_{k=1}^K \mathbf{1}_{[c_i=k]} \log(\hat{p}_{i,k}), \quad (19)$$

where  $n$  is the number of texts,  $c_i$  is the ground truth group label, and  $\hat{p}_{i,k}$  is the predicted probability for group  $k$ .

To jointly optimize the regression and classification tasks, we define a multi-task learning loss  $L_{\text{mtl}}$ :

$$L_{\text{mtl}} = \alpha_r L_r + \alpha_c L_c, \quad (20)$$

where  $L_r$  and  $L_c$  are the regression and classification losses, respectively, and  $\alpha_r$  and  $\alpha_c$  are task weights. This framework provides the foundation upon which our DRNash framework operates: DRO computes robust task losses  $L_r$  and  $L_c$  via worst-case aggregation over sentiment groups, while NTC dynamically determines optimal task weights  $\alpha_r$  and  $\alpha_c$  based on the Nash bargaining solution.

### Appendix A.5: Distributionally Robust Optimization

This appendix provides the algorithmic details of DRO introduced in Section 3.3. DRO addresses the

data-level challenge by ensuring balanced optimization across sentiment groups through worst-case aggregation.

---

**Algorithm 1** Distributionally Robust Optimization

---

**Require:** Dataset  $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ , number of groups  $K$ , robustness parameter  $\eta > 0$

**Ensure:** DRO losses  $L_r^{\text{DRO}}, L_c^{\text{DRO}}$

- 1: **// Phase 1: Sentiment Group Construction (Preprocessing)**
- 2: **for**  $k = 1$  to  $K$  **do**
- 3:    $\tau_k \leftarrow Q_{k/K}(\{y_i\}_{i=1}^N)$  {Compute  $k/K$ -quantile}
- 4: **end for**
- 5:  $\tau_0 \leftarrow -1, \tau_K \leftarrow 1$
- 6: **for**  $i = 1$  to  $N$  **do**
- 7:    $c_i \leftarrow k$  such that  $\tau_{k-1} < y_i \leq \tau_k$ , with  $k = 1$  also taking  $y_i = \tau_0$  {Assign group label}
- 8: **end for**
- 9:  $\hat{q}_k \leftarrow |G_k|/N$  for all  $k$  {Group proportions,  $\approx 1/K$ }
- 10: **// Phase 2: Batch-wise Group Loss (Each Training Step)**
- 11: Sample mini-batch  $\mathcal{B}$  from  $\mathcal{D}$
- 12:  $\mathcal{K}_{\mathcal{B}} \leftarrow \{k : |\mathcal{B}_k| > 0\}$  {Groups present in batch}
- 13: **for**  $k \in \mathcal{K}_{\mathcal{B}}$  **do**
- 14:    $L_r^{(k)} \leftarrow \frac{1}{|\mathcal{B}_k|} \sum_{(x,y,c) \in \mathcal{B}_k} (h_r(f_\theta(x)) - y)^2$
- 15:    $L_c^{(k)} \leftarrow \frac{1}{|\mathcal{B}_k|} \sum_{(x,y,c) \in \mathcal{B}_k} \text{CE}(h_c(f_\theta(x)), c)$
- 16: **end for**
- 17: **// Phase 3: Worst-case Aggregation**
- 18: **for**  $t \in \{r, c\}$  **do**
- 19:    $L_t^{\text{DRO}} \leftarrow \frac{1}{\eta} \log \left( \sum_{k \in \mathcal{K}_{\mathcal{B}}} \hat{q}_k \exp(\eta L_t^{(k)}) \right)$
- 20: **end for**
- 21: **return**  $L_r^{\text{DRO}}, L_c^{\text{DRO}}$

---

**Algorithm Explanation.** The DRO algorithm operates in three phases:

*Phase 1* constructs sentiment groups using quantile-based boundaries, ensuring approximately equal samples per group ( $|G_k| \approx N/K$ ). This is performed once during preprocessing.

*Phase 2* computes group-specific losses for each mini-batch. For groups present in the current batch ( $\mathcal{K}_{\mathcal{B}}$ ), we calculate the average MSE for regression and cross-entropy for classification within each group.

*Phase 3* performs worst-case aggregation using the log-sum-exp formulation. The robustness parameter  $\eta$  controls the emphasis on underper-

forming groups: as  $\eta \rightarrow 0$ , the DRO loss approaches the weighted average; as  $\eta \rightarrow \infty$ , it approaches the maximum group loss. The optimal worst-case weights are implicitly computed as  $q_k^* \propto \hat{q}_k \exp(\eta L_t^{(k)})$ , automatically assigning higher weights to groups with larger losses.

## Appendix A.6: Sentiment Group Configuration

This appendix provides details on the sentiment group construction introduced in Section 3.3, explaining the quantile-based partitioning strategy and the rationale for group configuration choices.

**Quantile-based Balanced Coverage.** The DRO component partitions the continuous sentiment space  $[-1, 1]$  into  $K$  discrete groups using quantile boundaries. Let  $\tau_k = Q_{k/K}(\{y_i\}_{i=1}^N)$  denote the  $k/K$ -quantile of training sentiment scores. This quantile-based construction ensures approximately equal sample counts per group ( $|G_k| \approx N/K$ ), providing balanced coverage across the entire sentiment spectrum:

- $G_1$ : Strongly negative sentiment region
- $G_{\lceil K/2 \rceil}$ : Neutral sentiment region
- $G_K$ : Strongly positive sentiment region

Unlike fixed-threshold approaches that may result in severely imbalanced groups due to the heavy concentration of neutral sentiments in financial texts, our quantile-based method automatically adapts to the underlying data distribution, ensuring each group receives sufficient training signal during optimization.

**Group Configuration Options.** The number of groups  $K$  determines the granularity of sentiment categorization. We consider configurations at multiple granularity levels:

- $K = 3$ : Negative, Neutral, Positive (Liu, 2022)
- $K = 5$ : Strong Negative, Negative, Neutral, Positive, Strong Positive (Socher et al., 2013; Rosenthal et al., 2017)
- $K = 7$ : Very Strong Negative, Strong Negative, Negative, Neutral, Positive, Strong Positive, Very Strong Positive

**Rationale for Citation Selection.** We select the three-class and five-class configurations based on

their widespread adoption and theoretical foundations in the sentiment analysis community.

The three-class configuration ( $K = 3$ ) follows the foundational framework established by Liu (2022), whose comprehensive monograph on sentiment analysis and opinion mining defines sentiment polarity classification as categorizing text into positive, negative, or neutral classes. This tripartite division represents the most fundamental sentiment taxonomy, capturing the essential polarity distinction while acknowledging that many texts express no clear sentiment orientation. The three-class scheme has been adopted across thousands of subsequent studies and remains the default choice for coarse-grained sentiment analysis tasks.

The five-class configuration ( $K = 5$ ) aligns with two influential benchmarks that have shaped modern fine-grained sentiment analysis research. First, the Stanford Sentiment Treebank (SST-5) introduced by Socher et al. (2013) established a five-point scale consisting of very negative, negative, neutral, positive, and very positive categories. SST-5 contains 11,855 movie review sentences with phrase-level annotations, enabling compositional sentiment analysis and serving as a standard evaluation benchmark for neural sentiment models. The five-class formulation captures not only polarity but also intensity, distinguishing between moderate and extreme sentiments. Second, SemEval-2017 Task 4 (Rosenthal et al., 2017) adopted an identical five-point ordinal scale (strongly negative, weakly negative, neutral, weakly positive, strongly positive) for tweet sentiment classification toward specific topics. This shared task attracted 48 participating teams and established the five-class scheme as an international standard for fine-grained sentiment evaluation, demonstrating its applicability across different text domains including social media.

**Seven-class Configuration: A Promising Research Direction.** The seven-class configuration ( $K = 7$ ) extends the granularity further by introducing additional intensity levels at both extremes of the sentiment spectrum. While this configuration lacks established benchmark support in the current literature, it represents a promising research direction for ultra-fine-grained sentiment analysis, particularly in domains where subtle sentiment distinctions carry practical significance.

However, the seven-class setting presents several inherent challenges that explain its limited adoption. First, *annotation difficulty*: human annotators exhibit low inter-rater agreement when distin-

guishing between adjacent fine-grained categories (e.g., “strong negative” versus “very strong negative”), as the perceptual boundaries between such categories are inherently subjective and context-dependent. Second, *data sparsity*: extreme sentiment categories at both ends of the spectrum are naturally rare in most corpora, leading to severe class imbalance that cannot be fully resolved even with quantile-based partitioning when the underlying continuous labels themselves cluster around neutral values. Third, *model capacity requirements*: capturing seven distinct sentiment levels demands models with sufficient representational capacity to learn subtle linguistic cues that differentiate adjacent intensity levels, which may exceed the capabilities of current architectures on limited training data. Fourth, *evaluation ambiguity*: standard classification metrics become less meaningful when misclassification between adjacent categories (e.g., predicting “positive” instead of “strong positive”) is penalized equally to misclassification across distant categories, necessitating ordinal-aware evaluation metrics that are not yet standardized.

Despite these challenges, the seven-class configuration may prove valuable for specialized applications in financial sentiment analysis where the magnitude of sentiment directly influences decision-making, such as distinguishing between moderately bullish and extremely bullish market signals. We leave the systematic exploration of ultra-fine-grained sentiment classification as a direction for future work.

**Selection of  $K$ .** The optimal  $K$  depends on dataset size, annotation granularity, and downstream task requirements. Larger  $K$  provides finer sentiment distinctions but may suffer from insufficient samples per group. Detailed configuration choices are provided in the Implementation Details appendix.

**Advantages of Quantile-based Partitioning.** Compared to fixed-threshold methods that use uniform intervals (e.g.,  $\{-0.6, -0.2, 0.2, 0.6\}$ ), our quantile-based approach offers several advantages:

- Distribution Adaptation*: Thresholds automatically adjust to dataset-specific characteristics, effectively handling the skewed distributions commonly observed in financial sentiment data where neutral and mildly negative sentiments predominate.
- Balanced Training*: Equal sample counts per group ensure balanced gradient contributions during DRO optimization, preventing any group from being underrepresented in the learning process and avoiding the gradient dominance problem.
- Robustness*: The

method exhibits robustness to outliers and to differences in label distribution across financial datasets, as the quantile boundaries are determined by relative positions rather than absolute values.

## Appendix A.7: Theoretical Foundation of DRO

This appendix provides the theoretical foundation for the DRO formulation presented in Section 3.3, including the derivation of Eq. (5) and the interpretation of the robustness parameter  $\eta$ .

### A.7.1 Problem Setup

In standard empirical risk minimization (ERM), the objective is to minimize the expected loss under the empirical distribution:

$$\min_{\theta} \sum_{k=1}^K \hat{q}_k L_k(\theta), \quad (21)$$

where  $\hat{q}_k = |G_k|/N$  denotes the empirical proportion of group  $k$  and  $L_k(\theta)$  is the average loss on group  $k$ . However, ERM is dominated by the majority groups and tends to underperform on minority groups (Sagawa et al., 2020).

### A.7.2 Distributionally Robust Optimization Formulation

DRO addresses this limitation by optimizing against the worst-case distribution within an uncertainty set  $\mathcal{Q}$  centered at the empirical distribution (Duchi and Namkoong, 2021):

$$\min_{\theta} \max_{q \in \mathcal{Q}} \sum_{k=1}^K q_k L_k(\theta). \quad (22)$$

The uncertainty set is defined as a KL-divergence ball around the empirical distribution:

$$\mathcal{Q} = \{q \in \Delta^{K-1} : D_{KL}(q \parallel \hat{q}) \leq \rho\}, \quad (23)$$

where  $\Delta^{K-1} = \{q : \sum_k q_k = 1, q_k \geq 0\}$  is the probability simplex,  $D_{KL}(q \parallel \hat{q}) = \sum_k q_k \log(q_k / \hat{q}_k)$  is the Kullback-Leibler divergence (Hu and Hong, 2012), and  $\rho > 0$  controls the size of the uncertainty set.

### A.7.3 Derivation via Lagrangian Duality

The inner maximization problem in Eq. (22) can be written as:

$$\max_{q \in \Delta^{K-1}} \sum_{k=1}^K q_k L_k \quad \text{s.t.} \quad D_{KL}(q \parallel \hat{q}) \leq \rho. \quad (24)$$

Introducing the Lagrangian with dual variable  $\lambda \geq 0$  for the KL constraint:

$$\mathcal{L}(q, \lambda) = \sum_{k=1}^K q_k L_k - \lambda (D_{KL}(q \parallel \hat{q}) - \rho). \quad (25)$$

Expanding the KL divergence:

$$\mathcal{L}(q, \lambda) = \sum_{k=1}^K q_k L_k - \lambda \sum_{k=1}^K q_k \log \frac{q_k}{\hat{q}_k} + \lambda \rho. \quad (26)$$

Taking the derivative with respect to  $q_k$  and setting it to zero (with Lagrange multiplier  $\mu$  for the simplex constraint  $\sum_k q_k = 1$ ):

$$\frac{\partial \mathcal{L}}{\partial q_k} = L_k - \lambda \left( \log \frac{q_k}{\hat{q}_k} + 1 \right) - \mu = 0. \quad (27)$$

Solving for  $q_k$ :

$$q_k^* = \hat{q}_k \exp \left( \frac{L_k - \mu - \lambda}{\lambda} \right) \propto \hat{q}_k \exp \left( \frac{L_k}{\lambda} \right). \quad (28)$$

Normalizing to satisfy  $\sum_k q_k^* = 1$ :

$$q_k^* = \frac{\hat{q}_k \exp(L_k / \lambda)}{\sum_{j=1}^K \hat{q}_j \exp(L_j / \lambda)}. \quad (29)$$

Substituting back into the Lagrangian and simplifying, the dual function becomes:

$$g(\lambda) = \lambda \log \left( \sum_{k=1}^K \hat{q}_k \exp(L_k / \lambda) \right) + \lambda \rho. \quad (30)$$

### A.7.4 Reparameterization

For notational convenience and numerical stability, we introduce the reparameterization  $\eta = 1/\lambda$ . Under this transformation:

- The optimal weights become:

$$q_k^* \propto \hat{q}_k \exp(\eta L_k), \quad (31)$$

which corresponds to Eq. (6) in the main text.

- The DRO loss becomes:

$$L^{\text{DRO}} = \frac{1}{\eta} \log \left( \sum_{k=1}^K \hat{q}_k \exp(\eta L_k) \right), \quad (32)$$

which corresponds to Eq. (5) in the main text (omitting constant terms independent of  $\theta$ ).

This reparameterization is standard in the literature (Duchi and Namkoong, 2021) and provides an intuitive interpretation: larger  $\eta$  corresponds to larger uncertainty set radius  $\rho$  (via the dual relationship), yielding more aggressive worst-case optimization.

### A.7.5 Interpretation of the Robustness

#### Parameter $\eta$

The parameter  $\eta > 0$  governs the trade-off between average-case and worst-case optimization:

#### Limiting Behavior.

- **When  $\eta \rightarrow 0$**  (equivalently  $\lambda \rightarrow \infty$ , tight KL constraint): Using L'Hôpital's rule:

$$\lim_{\eta \rightarrow 0} \frac{1}{\eta} \log \left( \sum_k \hat{q}_k \exp(\eta L_k) \right) = \sum_k \hat{q}_k L_k, \quad (33)$$

recovering standard ERM. Large  $\lambda$  imposes strong penalty on deviating from  $\hat{q}$ , forcing  $q^* \approx \hat{q}$ .

- **When  $\eta \rightarrow \infty$**  (equivalently  $\lambda \rightarrow 0$ , loose KL constraint): The log-sum-exp approaches the maximum:

$$\lim_{\eta \rightarrow \infty} \frac{1}{\eta} \log \left( \sum_k \hat{q}_k \exp(\eta L_k) \right) = \max_k L_k, \quad (34)$$

focusing entirely on the worst-performing group. Small  $\lambda$  allows  $q^*$  to concentrate on high-loss groups.

**Intermediate Values.** For finite  $\eta$ , the formulation smoothly interpolates between these extremes. The exponential weighting in Eq. (6) assigns larger weights to groups with higher losses, automatically emphasizing underperforming sentiment regions.

### A.7.6 Connection to Log-Sum-Exp and Softmax

The DRO loss can be interpreted as a weighted soft-maximum. Defining  $a_k = \eta L_k + \log \hat{q}_k$ , we have:

$$L^{\text{DRO}} = \frac{1}{\eta} \text{LSE}(a_1, \dots, a_K), \quad (35)$$

where  $\text{LSE}(\cdot)$  denotes the log-sum-exp function. This provides a smooth, differentiable approximation to  $\max_k L_k$  that incorporates prior weights  $\hat{q}_k$ , enabling stable gradient-based optimization.

## Appendix A.8: Nash Task Coordinator

This appendix provides the algorithmic details of the NTC introduced in Section 3.4. NTC addresses the task-level challenge by computing scale-invariant task weights based on the Nash bargaining solution.

**Algorithm Explanation.** The NTC algorithm operates in four steps:

---

### Algorithm 2 Nash Task Coordinator

---

**Require:** DRO losses  $L_r^{\text{DRO}}, L_c^{\text{DRO}}$ , shared parameters  $\theta$ , stability constant  $\epsilon > 0$

**Ensure:** Task weights  $\alpha_r, \alpha_c$ , combined gradient

$g$

1: **// Step 1: Compute Task Gradients**

2:  $g_r \leftarrow \nabla_{\theta} L_r^{\text{DRO}}$  {Regression gradient}

3:  $g_c \leftarrow \nabla_{\theta} L_c^{\text{DRO}}$  {Classification gradient}

4: **// Step 2: Compute Gradient Magnitudes**

5:  $\|g_r\| \leftarrow \sqrt{\sum_i (g_r^{(i)})^2}$

6:  $\|g_c\| \leftarrow \sqrt{\sum_i (g_c^{(i)})^2}$

7: **// Step 3: Nash Bargaining Weights (Closed-form)**

8:  $\alpha_r \leftarrow \frac{\|g_c\| + \epsilon}{\|g_r\| + \|g_c\| + 2\epsilon}$

9:  $\alpha_c \leftarrow \frac{\|g_r\| + \epsilon}{\|g_r\| + \|g_c\| + 2\epsilon}$

10: **// Step 4: Aggregate Gradients**

11:  $g \leftarrow \alpha_r \cdot g_r + \alpha_c \cdot g_c$

12: **return**  $\alpha_r, \alpha_c, g$

---

*Step 1* computes the gradients of DRO-aggregated losses with respect to shared model parameters  $\theta$ . These gradients incorporate the distributionally robust weighting from the DRO component.

*Step 2* calculates the  $\ell_2$ -norm of each task gradient, which quantifies the magnitude of parameter updates each task would induce if trained independently.

*Step 3* derives task weights from the Nash bargaining solution under the gradient orthogonality assumption. The key insight is that each task's weight is proportional to the *other* task's gradient magnitude: when  $\|g_r\| \gg \|g_c\|$ , the classification task receives higher weight ( $\alpha_c \rightarrow 1$ ) to prevent gradient domination by regression. This inverse relationship ensures scale-invariant balancing that satisfies the four Nash axioms: Pareto optimality, symmetry, scale invariance, and independence of irrelevant alternatives.

*Step 4* aggregates task gradients using the computed weights to form the final update direction for shared parameters.

**Theoretical Foundation.** The closed-form weights in Step 3 are derived from the Nash bargaining formulation in the main text. Recall from Eq. (9) that the Nash bargaining solution maximizes the

product of utility gains:

$$\Delta\theta^* = \arg \max_{\|\Delta\theta\| \leq 1} \prod_{t \in \{r, c\}} (-g_t^\top \Delta\theta). \quad (36)$$

Following Navon et al. (2022), when task gradients are approximately orthogonal in high-dimensional parameter spaces ( $g_r^\top g_c \approx 0$ ), the Nash bargaining solution admits a closed-form: task weights are inversely proportional to their gradient magnitudes, i.e.,  $\alpha_t \propto 1/\|g_t\|$ . After normalization, this yields Eqs. (10)-(11):

$$\alpha_r = \frac{\|g_c\|}{\|g_r\| + \|g_c\|}, \quad \alpha_c = \frac{\|g_r\|}{\|g_r\| + \|g_c\|}. \quad (37)$$

This inverse weighting ensures that tasks with larger gradients receive smaller weights, preventing gradient domination while satisfying scale invariance. The stability constant  $\epsilon$  prevents numerical instability when either gradient approaches zero.

## Appendix A.9: NTC Derivations

This appendix provides detailed derivations for the NTC presented in Section 3.4.

### A.9.1 Background: Nash Bargaining Problem

The Nash bargaining problem, recently applied to multi-task learning (Navon et al., 2022; Zeng et al., 2024), provides a foundational framework for analyzing cooperative negotiations between rational agents. Formally, a bargaining problem is defined by a tuple  $(\mathcal{U}, d)$  where  $\mathcal{U} \subset \mathbb{R}^2$  is the feasible utility set representing all possible outcomes achievable through agreement, and  $d \in \mathcal{U}$  is the disagreement point representing the outcome if negotiations fail.

Nash proved that under four axioms, there exists a unique solution to this problem. We elaborate on each axiom and its relevance to multi-task learning:

**Pareto Optimality.** The solution must lie on the Pareto frontier of  $\mathcal{U}$ ; that is, no alternative outcome can improve one player’s utility without reducing another’s. Formally, if  $u^*$  is the solution, there exists no  $u \in \mathcal{U}$  such that  $u_i \geq u_i^*$  for all  $i$  with strict inequality for some  $i$ . In multi-task learning, this ensures that the chosen gradient update cannot be improved for one task without degrading another, preventing wasteful optimization directions (Sener and Koltun, 2018; Nash, 1950).

**Symmetry.** If the bargaining problem is symmetric (i.e., the feasible set and disagreement point are invariant under permutation of players), then the solution must assign equal utilities to both players. This axiom ensures fairness: when tasks are equally important and have identical optimization landscapes, they should receive equal treatment. In practice, this prevents arbitrary bias toward any particular task when no distinguishing information is available (Navon et al., 2022).

**Scale Invariance (Affine Invariance).** The solution is invariant under positive affine transformations of individual utilities. If  $u^*$  solves the problem  $(\mathcal{U}, d)$ , then  $(a_1 u_1^* + b_1, a_2 u_2^* + b_2)$  solves the transformed problem for any  $a_i > 0, b_i \in \mathbb{R}$ . This property is crucial for multi-task learning because regression (MSE) and classification (cross-entropy) losses operate on fundamentally different scales. Scale invariance guarantees that the task weighting is determined by the geometric structure of the optimization landscape rather than arbitrary loss magnitude differences (Navon et al., 2022; Liu et al., 2023). Heuristic weighting schemes that lack this property require careful calibration across tasks with different loss scales.

### Independence of Irrelevant Alternatives (IIA).

If the solution  $u^*$  to problem  $(\mathcal{U}, d)$  lies within a subset  $\mathcal{U}' \subset \mathcal{U}$ , then  $u^*$  also solves the restricted problem  $(\mathcal{U}', d)$ . Intuitively, removing non-optimal alternatives should not change the optimal solution. In the context of gradient-based optimization, this means the optimal task weighting depends only on the local gradient geometry, not on hypothetical alternative update directions that were never considered (Nash, 1950).

The unique solution satisfying all four axioms is the Nash bargaining solution (NBS):

$$u^* = \arg \max_{u \in \mathcal{U}} \prod_i (u_i - d_i), \quad \text{s.t. } u_i \geq d_i. \quad (38)$$

The product form in Eq. (38) has an intuitive interpretation: it balances the gains of both players multiplicatively, ensuring that neither player’s interests dominate the solution. This multiplicative structure naturally leads to the inverse weighting property derived below.

### A.9.2 Formulation for Multi-Task Learning

We formulate multi-task gradient aggregation as a bargaining game following Navon et al. (2022).

**Players and Utilities.** Consider two tasks  $t \in \{r, c\}$  (regression and classification) as rational players negotiating over shared model parameters. Given task gradients  $g_r, g_c \in \mathbb{R}^{|\theta|}$  computed from DRO losses, each task seeks a parameter update  $\Delta\theta$  that maximally reduces its loss. Task  $t$ 's utility from update  $\Delta\theta$  is defined as the first-order approximation of loss reduction:

$$u_t(\Delta\theta) = L_t(\theta) - L_t(\theta + \Delta\theta) \approx -g_t^\top \Delta\theta. \quad (39)$$

Positive utility indicates loss reduction (beneficial), while negative utility indicates loss increase (harmful).

**Feasible Set.** The feasible utility set is determined by all possible parameter updates within a unit ball, which serves as a normalization constraint:

$$\mathcal{U} = \{(u_r, u_c) : \exists \Delta\theta, \|\Delta\theta\| \leq 1, u_t = -g_t^\top \Delta\theta \text{ for } t \in \{r, c\}\}. \quad (40)$$

**Disagreement Point.** When tasks fail to reach agreement, no update occurs ( $\Delta\theta = 0$ ), yielding zero utility for both tasks. Thus  $d = (0, 0)$ , representing the status quo.

**Nash Bargaining Solution.** Applying Eq. (38) to our multi-task setting yields Eq. (9) in the main text:

$$\Delta\theta^* = \arg \max_{\|\Delta\theta\| \leq 1} \prod_{t \in \{r, c\}} (-g_t^\top \Delta\theta). \quad (41)$$

### A.9.3 Derivation of Optimal Task Weights

We now derive the closed-form solution for Eq. (41).

**General Solution Form.** The optimal update takes the form of a weighted combination of task gradients (Désidéri, 2012):

$$\Delta\theta^* = -\gamma \sum_{t \in \{r, c\}} \alpha_t^* g_t, \quad (42)$$

where  $\gamma > 0$  is a scaling factor determined by the constraint  $\|\Delta\theta\| \leq 1$ , and  $\alpha^* = (\alpha_r^*, \alpha_c^*)$  are the task weights to be determined.

**Optimality Conditions.** Taking the logarithm of Eq. (41) converts the product to a sum, facilitating

differentiation. The Lagrangian for the constrained optimization is:

$$\mathcal{L}(\Delta\theta, \lambda) = \sum_t \log(-g_t^\top \Delta\theta) - \lambda(\|\Delta\theta\|^2 - 1). \quad (43)$$

Setting  $\nabla_{\Delta\theta} \mathcal{L} = 0$  and substituting  $\Delta\theta = -\gamma \sum_t \alpha_t g_t$ , the optimal weights satisfy (Navon et al., 2022):

$$G^\top G \alpha^* = \mathbf{1} \oslash \alpha^*, \quad (44)$$

where  $G = [g_r, g_c] \in \mathbb{R}^{|\theta| \times 2}$  is the gradient matrix,  $\mathbf{1} = (1, 1)^\top$  is the all-ones vector, and  $\oslash$  denotes element-wise division.

**Orthogonal Gradient Simplification.** In high-dimensional parameter spaces typical of neural networks (with millions of parameters), random vectors are nearly orthogonal with high probability. Empirically, task gradients in multi-task learning exhibit this property:  $g_r^\top g_c \approx 0$  (Yu et al., 2020). Under this assumption, the Gram matrix simplifies to:

$$G^\top G = \begin{bmatrix} \|g_r\|^2 & g_r^\top g_c \\ g_c^\top g_r & \|g_c\|^2 \end{bmatrix} \approx \begin{bmatrix} \|g_r\|^2 & 0 \\ 0 & \|g_c\|^2 \end{bmatrix}. \quad (45)$$

Substituting into Eq. (44):

$$\begin{bmatrix} \|g_r\|^2 & 0 \\ 0 & \|g_c\|^2 \end{bmatrix} \begin{bmatrix} \alpha_r \\ \alpha_c \end{bmatrix} = \begin{bmatrix} 1/\alpha_r \\ 1/\alpha_c \end{bmatrix}. \quad (46)$$

This yields  $\|g_r\|^2 \alpha_r = 1/\alpha_r$  and  $\|g_c\|^2 \alpha_c = 1/\alpha_c$ . Solving for  $\alpha_t$ :

$$\alpha_r^* = \frac{1}{\|g_r\|}, \quad \alpha_c^* = \frac{1}{\|g_c\|}. \quad (47)$$

**Normalized Weights.** To obtain weights that sum to one for interpretability and consistency with Eq. (1) in the main text, we normalize:

$$\alpha_r = \frac{1/\|g_r\|}{1/\|g_r\| + 1/\|g_c\|} = \frac{\|g_c\|}{\|g_r\| + \|g_c\|}, \quad (48)$$

$$\alpha_c = \frac{1/\|g_c\|}{1/\|g_r\| + 1/\|g_c\|} = \frac{\|g_r\|}{\|g_r\| + \|g_c\|}. \quad (49)$$

Adding  $\epsilon > 0$  for numerical stability when gradients approach zero:

$$\alpha_r = \frac{\|g_c\| + \epsilon}{\|g_r\| + \|g_c\| + 2\epsilon}, \quad (50)$$

$$\alpha_c = \frac{\|g_r\| + \epsilon}{\|g_r\| + \|g_c\| + 2\epsilon},$$

which corresponds to Eqs. (10)-(11) in the main text.

### A.9.4 Properties of the Solution

**Inverse Weighting Intuition.** The derived weights assign smaller coefficients to tasks with larger gradient magnitudes. When  $\|g_r\| \gg \|g_c\|$ , we have  $\alpha_r \ll \alpha_c$ , effectively down-weighting the dominant regression task. This prevents tasks with large gradients from overshadowing others, ensuring balanced learning progress across both objectives (Chen et al., 2018).

**Scale Invariance Verification.** Consider rescaling task  $r$ 's loss by constant  $c > 0$ . The gradient becomes  $c \cdot g_r$  and the weight becomes:

$$\alpha'_r = \frac{\|g_c\|}{c\|g_r\| + \|g_c\|}. \quad (51)$$

The effective contribution to the update direction,  $\alpha'_r \cdot (c \cdot g_r)$ , satisfies:

$$\alpha'_r \cdot c \cdot g_r = \frac{c\|g_c\|}{c\|g_r\| + \|g_c\|} \cdot g_r. \quad (52)$$

As  $c$  varies, the direction of the combined update remains on the same ray in gradient space, demonstrating invariance to loss scaling up to normalization. This property is essential for combining MSE and cross-entropy losses without manual calibration.

## Appendix A.10: Complexity Analysis

This appendix analyzes the computational complexity of the DRNash framework, focusing on the overhead introduced by the two core components: DRO and NTC.

### A.10.1 Notation

- $N$ : Total number of training samples
- $B$ : Mini-batch size
- $K$ : Number of sentiment groups
- $d$ : Hidden dimension of the LLM backbone
- $|\theta|$ : Number of trainable parameters
- $r$ : LoRA rank
- $L$ : Number of transformer layers

### A.10.2 LoRA Parameter Efficiency

LoRA introduces low-rank matrices  $U \in \mathbb{R}^{d \times r}$  and  $V \in \mathbb{R}^{r \times d}$  to decompose weight updates, significantly reducing trainable parameters:

$$\text{Compression ratio} = \frac{|\theta_{\text{LoRA}}|}{|\theta_{\text{full}}|} = \frac{2Ldr}{Ld^2} = \frac{2r}{d}. \quad (53)$$

With  $d = 1024$ , the ranks used in this work give compression ratios of 25% at  $r = 128$ , 50% at  $r = 256$ , 75% at  $r = 384$ , and 100% at  $r = 512$ , so the parameter savings are largest at the lower ranks.

### A.10.3 DRO Complexity

The DRO component operates in three phases:

**Phase 1: Sentiment Group Construction (Preprocessing).** Executed once before training. Quantile computation via sorting and group assignment require  $O(N \log N)$ .

**Phase 2: Batch-wise Group Loss (Per Iteration).** For each mini-batch, group-wise loss aggregation requires  $O(B)$ .

**Phase 3: Worst-case Aggregation (Per Iteration).** The log-sum-exp computation in Eq. (5) performs exponential weighting over  $K$  groups, requiring  $O(K)$ .

**Summary.** DRO adds  $O(B + K)$  per iteration. Since  $K \ll B$  (typically  $K \in \{3, 5, 7\}$ ,  $B \in \{16, 32, 64\}$ ), this overhead is negligible compared to the forward/backward pass through the LLM backbone.

### A.10.4 NTC Complexity

The NTC component computes adaptive task weights at each iteration:

**Step 1: Task Gradient Computation.** Standard backpropagation for two tasks requires  $O(|\theta|)$ .

**Step 2: Gradient Magnitude Computation.** Computing  $\ell_2$ -norms  $\|g_r\|$  and  $\|g_c\|$  requires  $O(|\theta|)$ .

**Step 3: Nash Bargaining Weights.** The closed-form solution in Eqs. (10)-(11) requires only scalar operations, i.e.,  $O(1)$ .

**Step 4: Gradient Aggregation.** Weighted sum  $g = \alpha_r g_r + \alpha_c g_c$  requires  $O(|\theta|)$ .

**Summary.** NTC adds  $O(|\theta|)$  per iteration, comparable to a single backward pass. Crucially, the closed-form Nash bargaining solution avoids iterative optimization, unlike methods such as MGDA that require solving constrained optimization problems at each step.

## Appendix A.11: Benchmark

We use different benchmarks to verify our framework as shown in Table 7. We establish three baseline benchmarks (B1-B3) and implement our proposed DRNash framework across six configurations: three with DRNash only (B4-B6) and three with DRNash+LoRA integration (B7-B9).

### A.11.1 Baseline Models

**B1: RoBERTa-Large.** We use the standard RoBERTa-Large model (Liu et al., 2019) as our first baseline, which serves as a general-purpose language model without domain-specific pre-training. This baseline helps establish the performance gap between general and domain-adapted models for financial sentiment analysis.

**B2: Twitter-RoBERTa-Large (Regression).** The Twitter-RoBERTa-Large model (Loureiro et al., 2023) is pre-trained on approximately 154 million tweets, providing strong domain adaptation for sentiment analysis tasks. In this configuration, we fine-tune the model for regression only, predicting continuous sentiment polarity scores.

**B3: Twitter-RoBERTa-Large (Constant Weight MTL).** This baseline implements multi-task learning with constant weights, where the regression and classification losses carry fixed weights ( $\alpha_r = 0.9$  and  $\alpha_c = 0.1$ ). This configuration demonstrates the limitations of fixed weight strategies in handling the dynamic nature of multi-task optimization.

### A.11.2 DRNash Framework Configurations

The DRNash framework integrates DRO for handling data imbalance at the group level and NTC for dynamic task weight allocation. We implement DRNash across three backbone models:

**B4-B6: DRNash (Full Fine-tuning).** These configurations apply the complete DRNash framework with full model fine-tuning. The DRO component partitions sentiment scores into  $K = 5$  quantile-based groups and applies worst-case aggregation to ensure robust performance across all sentiment

regions. The NTC component dynamically adjusts task weights based on Nash bargaining principles, ensuring Pareto-optimal gradient updates.

**B7-B9: DRNash+LoRA.** These configurations combine the DRNash framework with Low-Rank Adaptation (LoRA) (Hu et al., 2022) for parameter-efficient fine-tuning. LoRA adapters are applied to the query, key, and value projection layers, significantly reducing trainable parameters while maintaining model performance.

## Appendix A.12: Evaluation Metrics

For clarity, the following metrics are reported as percentages (scaled by 100):  $R^2$ , Accuracy (ACC), Precision, and F1 Score.

### A.12.1 Regression Task Metrics

For the sentiment score regression task, where the model predicts a continuous sentiment polarity score, we employ the following metrics:

**Mean Squared Error (MSE).** MSE calculates the average squared differences between the predicted sentiment polarity scores ( $\hat{y}_i$ ) and the annotated sentiment polarity scores ( $y_i$ ) in the test set. MSE provides a measure of the model’s accuracy in predicting the exact sentiment scores. It is defined as:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2, \quad (54)$$

where  $n$  is the number of texts in the test set.

**Mean Absolute Error (MAE).** MAE measures the average magnitude of the absolute errors between the predicted and annotated sentiment polarity scores, ignoring their direction. MAE helps in understanding the average error magnitude and is less sensitive to outliers compared to MSE. It is formulated as:

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|. \quad (55)$$

**Root Mean Squared Error (RMSE).** RMSE is the square root of MSE and provides the error magnitude in the same units as the sentiment polarity scores. RMSE is more interpretable than MSE and is calculated as:

$$\text{RMSE} = \sqrt{\text{MSE}}. \quad (56)$$

Table 7: Benchmark models and experimental settings.

| Benchmark      | Model                                         | Task                         | Description                                                                      |
|----------------|-----------------------------------------------|------------------------------|----------------------------------------------------------------------------------|
| B1             | RoBERTa-Large (Liu et al., 2019)              | Regression                   | Baseline regression model without domain-specific pre-training                   |
| B2             | Twitter-RoBERTa-Large (Loureiro et al., 2023) | Regression                   | Domain-adapted model for sentiment regression only                               |
| B3             | Twitter-RoBERTa-Large (Loureiro et al., 2023) | Multi-task (Constant Weight) | Multi-task learning with fixed weights ( $\alpha_r = 0.9$ and $\alpha_c = 0.1$ ) |
| B4-DRNash      | Twitter-RoBERTa-Large (Loureiro et al., 2023) | Multi-task (DRO + NTC)       | Full fine-tuning with DRO aggregation and Nash task coordination                 |
| B5-DRNash      | TinyLlama-1.1B (Zhang et al., 2024)           | Multi-task (DRO + NTC)       | Full fine-tuning of decoder-only LLM with DRNash framework                       |
| B6-DRNash      | Qwen2-0.5B (Yang et al., 2024)                | Multi-task (DRO + NTC)       | Full fine-tuning of efficient LLM with DRNash framework                          |
| B7-DRNash+LoRA | Twitter-RoBERTa-Large (Loureiro et al., 2023) | Multi-task (DRO + NTC)       | Parameter-efficient fine-tuning with LoRA adapters                               |
| B8-DRNash+LoRA | TinyLlama-1.1B (Zhang et al., 2024)           | Multi-task (DRO + NTC)       | LoRA fine-tuning of decoder-only LLM with DRNash framework                       |
| B9-DRNash+LoRA | Qwen2-0.5B (Yang et al., 2024)                | Multi-task (DRO + NTC)       | LoRA fine-tuning of efficient LLM with DRNash framework                          |

**R-squared ( $R^2$ ).** R-squared indicates the proportion of variance in the sentiment polarity scores that can be explained by the model’s predictions. It provides a measure of how well the model fits the data, with values closer to 1 indicating a better fit. R-squared is defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}, \quad (57)$$

where  $\bar{y}$  represents the mean of the annotated sentiment polarity scores in the test set.

### A.12.2 Classification Task Metrics

For the sentiment classification task, where texts are categorized into five sentiment classes (e.g., strong positive, positive, neutral, negative, strong negative), we calculate the following metrics:

**Accuracy (ACC).** Accuracy is the ratio of correctly predicted sentiment classes to the total number of predictions. It measures the overall correctness of the model’s classifications and is defined as:

$$\text{ACC} = \frac{\sum_{i=1}^5 TP_i}{\sum_{i=1}^5 (TP_i + FN_i)}, \quad (58)$$

where  $TP_i$  is the number of instances correctly predicted as class  $i$ , and  $FN_i$  is the number of instances that actually belong to class  $i$  but are wrongly predicted as other classes.

**Precision.** Precision is the ratio of correctly predicted positive instances to all instances predicted as positive. It measures the model’s ability to avoid false positives and is calculated as:

$$\text{Precision} = \frac{TP_{pos}}{TP_{pos} + FP_{pos}}, \quad (59)$$

where  $TP_{pos}$  is the number of positive instances correctly predicted as positive, and  $FP_{pos}$  is the number of non-positive instances wrongly predicted as positive.

**Recall (or Sensitivity).** Recall is the ratio of correctly predicted positive instances to all actual positive instances. It measures the model’s ability to identify all positive instances and is calculated as:

$$\text{Recall} = \frac{TP_{pos}}{TP_{pos} + FN_{pos}}, \quad (60)$$

where  $FN_{pos}$  is the number of positive instances wrongly predicted as non-positive.

**F1 Score.** F1 Score is the harmonic mean of Precision and Recall, providing a balanced measure of the model’s performance, especially when the sentiment classes are imbalanced. It is calculated as:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \quad (61)$$

### A.12.3 Equivalence Between Weighted Recall and Accuracy

In a multi-class classification problem with  $n$  classes, let  $TP_i$ ,  $FN_i$ , and  $\text{Support}_i$  denote the true positives, false negatives, and total instances (support) for class  $i$ , respectively. The recall for class  $i$  is given by:

$$\text{Recall}_i = \frac{TP_i}{TP_i + FN_i} = \frac{TP_i}{\text{Support}_i}. \quad (62)$$

The weighted recall is then calculated as:

$$\text{Weighted Recall} = \frac{\sum_{i=1}^n \text{Recall}_i \times \text{Support}_i}{\sum_{i=1}^n \text{Support}_i}. \quad (63)$$

Substituting the expression for  $\text{Recall}_i$ :

$$\text{Weighted Recall} = \frac{\sum_{i=1}^n \frac{TP_i}{\text{Support}_i} \times \text{Support}_i}{\sum_{i=1}^n \text{Support}_i}. \quad (64)$$

The  $\text{Support}_i$  terms in the numerator and denominator cancel out, leaving:

$$\text{Weighted Recall} = \frac{\sum_{i=1}^n TP_i}{\sum_{i=1}^n \text{Support}_i}. \quad (65)$$

This is exactly the formula for overall accuracy:

$$\text{Accuracy} = \frac{TP_1 + \dots + TP_n}{\text{Sup}_1 + \dots + \text{Sup}_n}, \quad (66)$$

where  $\text{Sup}_i$  abbreviates  $\text{Support}_i$ . Therefore, in a multi-class classification setting, weighted recall is mathematically equivalent to accuracy.

## Appendix A.13: Dataset

### A.13.1 Financial Text Dataset

Financial sentiment analysis aims to extract sentiment polarity from financial texts (e.g., news articles, analyst reports) to enhance market prediction tasks such as exchange rate forecasting and stock trend analysis (Li et al., 2019; Ding et al., 2024b). The key challenges include domain-specific terminology, implicit sentiment expressions, and inherent class imbalance where neutral sentiments dominate (Mujahid et al., 2024).

### A.13.2 INV Dataset

The INV dataset (Ding et al., 2024b) contains 23,241 financial text samples with sentiment polarity scores in  $[-1.0, 0.9]$ , annotated using large language models for EUR/USD exchange rate sentiment analysis. The dataset exhibits a mean score of  $-0.044$  (std=0.303), with the highest concentration (14.03%) in the  $[-0.20, -0.10]$  range.

### A.13.3 NEU Dataset

The NEU dataset (Ding et al., 2024b) comprises 12,780 financial text samples from investing.com and forexempire.com, annotated by RoBERTa-Large with sentiment scores in  $[-0.95, 0.9]$ . The dataset has a mean score of  $-0.060$  (std=0.312), with the highest frequency (14.83%) in the  $[-0.20, -0.10]$  range.

### A.13.4 FXE Dataset

The FXE dataset (Ding et al., 2024a) contains 10,461 financial text samples for EUR/USD forecasting, with RoBERTa-Large sentiment annotations in  $[-0.9, 0.9]$ . The dataset shows a near-neutral mean score of  $-0.014$  (std=0.258), with concentrations at  $[-0.20, -0.10]$  (14.99%) and  $[0.20, 0.30]$  (17.56%).

## Appendix A.14: Hyperparameters

### A.14.1 Experimental Parameters

#### Reproducibility Settings:

- Random seeds: 42 (Python, NumPy, PyTorch, CUDA) for reproducibility across all runs
- Benchmark mode: CuDNN benchmark disabled for reproducibility

#### Data Splitting and Evaluation:

- Train-validation split: 90%-10% random stratified split
- Best model selection: Lowest validation MSE across all epochs

#### Training Configuration:

- Input sequence length: 512 tokens (truncated if exceeded)
- Training epochs: 100 epochs maximum
- Warmup steps: 100 steps
- Learning rate scheduler: Cosine decay
- Batch size: 32 for all experiments

### A.14.2 DRNash Framework Hyperparameters

#### Quantile-based Group Partitioning:

- Number of sentiment groups:  $K = 5$

#### DRO Configuration:

- Temperature parameter:  $\eta = 1.0$

#### NTC Configuration:

- Stability parameter:  $\epsilon = 1 \times 10^{-8}$
- Weight normalization:  $\alpha_r + \alpha_c = 1$  (automatically satisfied)

### A.14.3 LoRA Parameters

- Rank values: 128, 256, 384, 512
- Scaling factor (alpha): Equal to rank value
- Dropout rate: 0.05 for all configurations
- Target modules: Query (Q), Key (K), Value (V) projection layers
- Weight initialization: Gaussian normal distribution for matrix  $A$ , zero initialization for matrix  $B$

### A.14.4 Environment Setup

#### Hardware Configuration:

- Operating System: Ubuntu 22.04 LTS
- CPU: Intel Xeon (64GB RAM)
- GPU: NVIDIA A100 (80GB VRAM)

#### Software Configuration:

- Python version: 3.9.19
- PyTorch version: 2.3.1
- PEFT library: 0.12.0
- Transformers library: 4.40.0
- CUDA version: 12.4

#### Optimization Settings:

- Base learning rate:  $1 \times 10^{-5}$
- Weight decay:  $1 \times 10^{-2}$
- Optimizer: AdamW
- Epsilon (AdamW):  $1 \times 10^{-8}$
- Beta1: 0.9, Beta2: 0.999
- Gradient clipping: 1.0 (max norm)

## Appendix A.15: DRNash vs Constant Weight

**Overall Performance.** Table 8 presents the comprehensive results comparing constant-weight multi-task learning ( $w_r = 0.9, w_c = 0.1$ ) with our proposed DRNash framework. DRNash consistently outperforms the constant-weight baseline across all 36 metric-model-dataset combinations, demonstrating the robustness of our approach. The average improvements are 14.1% in MSE and 1.92% in accuracy, with Precision and F1 scores showing comparable gains.

**Model-wise Analysis.** Twitter-RoBERTa-Large exhibits the largest improvements, with MSE reductions ranging from 17.2% to 19.0% across datasets. This substantial gain can be attributed to the model’s domain-specific pretraining on social media text, which aligns well with financial sentiment expressions. The accuracy improvements of 1.91% to 2.21% are accompanied by consistent gains in Precision (2.25 to 2.69 absolute points) and F1 (1.65 to 2.05 absolute points), indicating that DRNash enhances both regression precision and classification reliability.

For Qwen 0.5B, improvements range from 9.9% to 13.4% in MSE and 1.59% to 2.26% in accuracy. The relatively smaller gains on FXE dataset (9.9% MSE reduction) suggest that smaller models may face capacity constraints when handling complex sentiment distributions, yet DRNash still provides meaningful improvements.

TinyLlama 1.1B demonstrates stable improvements across all datasets, achieving 11.2% to 13.0% MSE reduction and 1.49% to 1.92% accuracy gain. Notably, this model achieves the best absolute performance (MSE of 0.0174 and accuracy of 78.49% on INV), indicating that the combination of sufficient model capacity and DRNash optimization yields superior results.

**Dataset-wise Analysis.** The INV dataset consistently yields the best performance across all models, likely due to its more balanced sentiment distribution. FXE presents the most challenging scenario with relatively lower improvements, suggesting higher inherent difficulty in this dataset’s sentiment patterns. NEU shows intermediate results, with DRNash effectively bridging the performance gap between different data characteristics.

**Classification Metrics.** Beyond MSE and accuracy, we observe that Precision exceeds accuracy by 0.33 to 1.23 absolute points under DRNash, while F1 scores remain closely aligned with accuracy. This pattern indicates that DRNash not only improves overall classification but also reduces false positives, which is particularly valuable for financial applications where prediction reliability is critical.

## Appendix A.16: Effectiveness of Auxiliary Classification Task

This appendix provides a detailed analysis of the effectiveness of auxiliary classification task by com-

Table 8: Performance comparison between constant weight and DRNash framework.

| Model                    | Dataset | Constant Weight |       |       |       | DRNash Framework |       |       |       |
|--------------------------|---------|-----------------|-------|-------|-------|------------------|-------|-------|-------|
|                          |         | MSE             | ACC   | Prec. | F1    | MSE              | ACC   | Prec. | F1    |
| Twitter<br>RoBERTa-Large | NEU     | 0.0231          | 75.89 | 75.84 | 75.76 | 0.0187           | 77.34 | 78.23 | 77.41 |
|                          | FXE     | 0.0244          | 75.12 | 75.23 | 75.18 | 0.0202           | 76.78 | 77.48 | 77.12 |
|                          | INV     | 0.0222          | 76.56 | 76.49 | 76.41 | 0.0183           | 78.09 | 79.18 | 78.46 |
| Qwen 0.5B                | NEU     | 0.0247          | 74.89 | 75.12 | 74.78 | 0.0214           | 76.58 | 77.71 | 76.67 |
|                          | FXE     | 0.0252          | 74.67 | 74.89 | 74.56 | 0.0227           | 75.86 | 76.81 | 75.69 |
|                          | INV     | 0.0241          | 75.18 | 75.41 | 75.02 | 0.0209           | 76.81 | 78.04 | 76.89 |
| TinyLlama<br>1.1B        | NEU     | 0.0204          | 76.89 | 77.12 | 76.56 | 0.0179           | 78.21 | 78.57 | 77.93 |
|                          | FXE     | 0.0215          | 76.23 | 76.78 | 76.12 | 0.0187           | 77.69 | 78.32 | 77.59 |
|                          | INV     | 0.0196          | 77.34 | 77.56 | 77.01 | 0.0174           | 78.49 | 78.82 | 78.26 |

paring single-task regression with constant weight multi-task learning (MTL). The auxiliary classification task partitions the sentiment space  $[-1, 1]$  into five equal-width intervals, providing complementary supervision signals for sentiment boundary learning.

### A.16.1 Experimental Setup

We compare two training paradigms:

- **Single Task:** The model is trained solely on sentiment regression using MSE loss.
- **Constant Weight MTL:** The model jointly optimizes sentiment regression and auxiliary classification with fixed task weights ( $w_r = 0.9, w_c = 0.1$ ). The classification task partitions  $[-1, 1]$  into five equal-width intervals.

Both paradigms use identical model architectures, hyperparameters, and training configurations. The only difference is the presence of the auxiliary classification objective in constant weight MTL.

### A.16.2 Performance Comparison

Table 9 presents a side-by-side comparison between single-task learning and constant weight MTL across three backbone models and three financial sentiment datasets.

### A.16.3 Analysis

Table 9 validates the effectiveness of auxiliary classification by comparing single-task regression with constant weight MTL. Single-task baseline trains the model solely on sentiment regression using MSE loss. In contrast, constant weight MTL jointly optimizes sentiment regression and auxiliary classification with fixed task weights ( $w_r = 0.9, w_c = 0.1$ ), where the auxiliary task partitions the sentiment space  $[-1, 1]$  into five equal-width intervals. This experimental design isolates the contribution

of the auxiliary task by controlling all other variables, including model architecture, hyperparameters, training epochs, and data splits.

As shown in Table 9, introducing auxiliary classification achieves consistent improvements over single-task baseline across all model architectures and datasets. On average, constant weight MTL achieves 3.5% MSE reduction, 1.8% RMSE reduction, 1.8% MAE reduction, and 1.2%  $R^2$  improvement. Individual MSE improvements range from 3.0% (TinyLlama on INV) to 4.0% (TinyLlama on FXE). These results confirm that auxiliary classification provides complementary supervision signals that enhance sentiment boundary learning, establishing multi-task formulation as an effective foundation for addressing data distribution imbalance in financial sentiment regression.

**Consistent Improvement Across All Configurations.** The results demonstrate that introducing the auxiliary classification task yields consistent improvements across all nine model-dataset combinations. The average MSE reduction is 3.5%, with corresponding RMSE reduction of 1.8%. This consistency indicates that the auxiliary task provides a robust and generalizable benefit regardless of model architecture or dataset characteristics.

**Model-Specific Observations.** All three backbone models show improved performance:

- **Twitter-RoBERTa-Large:** Achieves 3.3% to 3.6% MSE reduction (1.7% to 1.8% RMSE reduction). The encoder-based architecture benefits from the auxiliary task’s explicit boundary supervision.
- **Qwen 0.5B:** Shows 3.2% to 3.8% MSE reduction (1.6% to 2.0% RMSE reduction). As a smaller language model, it particularly benefits from the additional supervision signal.
- **TinyLlama 1.1B:** Demonstrates 3.0% to 4.0% MSE reduction (1.5% to 2.1% RMSE reduc-

Table 9: Side-by-side comparison between Single Task and Constant Weight MTL.  $\Delta$  denotes the relative improvement (%) of Constant Weight MTL over Single Task. **Green** indicates improvement; MSE/RMSE/MAE: lower is better ( $\downarrow$ );  $R^2$ : higher is better ( $\uparrow$ ).

| Model                 | Dataset | Single Task      |                   |                  |               | Constant Weight MTL |                   |                  |               | $\Delta$ Improvement (%) |               |              |              |
|-----------------------|---------|------------------|-------------------|------------------|---------------|---------------------|-------------------|------------------|---------------|--------------------------|---------------|--------------|--------------|
|                       |         | MSE $\downarrow$ | RMSE $\downarrow$ | MAE $\downarrow$ | $R^2\uparrow$ | MSE $\downarrow$    | RMSE $\downarrow$ | MAE $\downarrow$ | $R^2\uparrow$ | $\Delta$ MSE             | $\Delta$ RMSE | $\Delta$ MAE | $\Delta R^2$ |
| Twitter-RoBERTa-Large | NEU     | 0.0239           | 0.1546            | 0.1103           | 0.7268        | 0.0231              | 0.1520            | 0.1085           | 0.7356        | +3.3                     | +1.7          | +1.6         | +1.2         |
|                       | FXE     | 0.0253           | 0.1591            | 0.1136           | 0.7226        | 0.0244              | 0.1562            | 0.1114           | 0.7332        | +3.6                     | +1.8          | +1.9         | +1.5         |
|                       | INV     | 0.0230           | 0.1517            | 0.1082           | 0.7366        | 0.0222              | 0.1490            | 0.1063           | 0.7450        | +3.5                     | +1.8          | +1.8         | +1.1         |
| Qwen 0.5B             | NEU     | 0.0256           | 0.1600            | 0.1141           | 0.7080        | 0.0247              | 0.1572            | 0.1121           | 0.7184        | +3.5                     | +1.8          | +1.8         | +1.5         |
|                       | FXE     | 0.0262           | 0.1619            | 0.1155           | 0.7042        | 0.0252              | 0.1587            | 0.1133           | 0.7150        | +3.8                     | +2.0          | +1.9         | +1.5         |
|                       | INV     | 0.0249           | 0.1578            | 0.1125           | 0.7183        | 0.0241              | 0.1552            | 0.1107           | 0.7267        | +3.2                     | +1.6          | +1.6         | +1.2         |
| TinyLlama 1.1B        | NEU     | 0.0211           | 0.1453            | 0.1036           | 0.7591        | 0.0204              | 0.1428            | 0.1019           | 0.7672        | +3.3                     | +1.7          | +1.6         | +1.1         |
|                       | FXE     | 0.0224           | 0.1497            | 0.1067           | 0.7554        | 0.0215              | 0.1466            | 0.1046           | 0.7647        | +4.0                     | +2.1          | +2.0         | +1.2         |
|                       | INV     | 0.0202           | 0.1421            | 0.1014           | 0.7682        | 0.0196              | 0.1400            | 0.0998           | 0.7753        | +3.0                     | +1.5          | +1.6         | +0.9         |
| Average               |         | 0.0236           | 0.1536            | 0.1095           | 0.7332        | 0.0228              | 0.1508            | 0.1076           | 0.7423        | +3.5                     | +1.8          | +1.8         | +1.2         |

tion), achieving the highest improvement on FXE.

**Dataset-Specific Observations.** The three datasets exhibit varying levels of improvement:

- **FXE:** Exhibits the largest average improvement (3.8% MSE, 2.0% RMSE reduction), likely because this dataset contains more complex sentiment expressions that benefit from explicit boundary learning.
- **NEU:** Shows consistent improvement (3.4% MSE, 1.7% RMSE reduction) with stable gains across all models.
- **INV:** Demonstrates moderate improvement (3.2% MSE, 1.6% RMSE reduction), as this dataset’s relatively clearer sentiment patterns result in a smaller margin for improvement.

**Why Auxiliary Classification Helps.** The auxiliary classification task contributes to improved regression performance through several mechanisms:

1. **Explicit Boundary Learning:** By partitioning the sentiment space into five equal-width intervals, the classification task explicitly teaches the model to recognize sentiment boundaries, which implicitly regularizes the regression output.
2. **Gradient Diversity:** The classification loss provides gradients with different characteristics from regression loss, enriching the optimization landscape and potentially escaping local minima.
3. **Feature Regularization:** Joint optimization encourages the shared feature extractor

to learn representations that are useful for both tasks, reducing overfitting to regression-specific noise.

## Appendix A.17: Traditional Imbalanced Learning Methods

This appendix provides a comprehensive analysis of traditional imbalanced learning methods compared with our proposed DRNash framework. We first introduce each baseline method, then present detailed performance tables for each approach, and finally provide an extended analysis of the experimental results.

### A.17.1 Traditional Imbalanced Learning Methods

**Label Distribution Smoothing (LDS).** LDS (Yang et al., 2021) was proposed in ICML 2021 as part of the Deep Imbalanced Regression (DIR) framework. The core idea is to apply kernel density smoothing to the empirical label distribution to estimate an effective label density that accounts for the continuity of regression targets. Specifically, LDS convolves a symmetric kernel (e.g., Gaussian, Laplacian, or triangular) with the empirical label density, then uses the inverse of the smoothed density as sample weights during training. This approach explicitly acknowledges that nearby target values share similar characteristics and should influence each other’s importance estimation. While LDS achieved state-of-the-art results on several imbalanced regression benchmarks including IMDB-WIKI-DIR and AgeDB-DIR, its effectiveness depends on appropriate kernel selection and the assumption that label smoothing accurately captures the underlying data distribution.

**Balanced MSE.** Balanced MSE (Ren et al., 2022) was introduced at CVPR 2022 as an Oral presentation. This method revisits the standard Mean Square Error (MSE) loss from a statistical perspective and reformulates it using a contrastive learning framework. The key innovation is the Batch Monte Carlo (BMC) variant, which estimates the effective label distribution directly from each mini-batch without requiring prior knowledge of the global label distribution. The loss function treats each sample’s label as a distinct "class" and applies softmax normalization over pairwise distances, automatically up-weighting rare labels that have fewer "competitors" within the batch. Balanced MSE was the first general solution for high-dimensional imbalanced regression and demonstrated strong performance on age estimation and human mesh recovery tasks. However, its effectiveness is sensitive to batch size, as smaller batches may not accurately represent the global label distribution.

**Focal-R (Focal Loss for Regression).** Focal-R adapts the Focal Loss (Lin et al., 2017), originally proposed at ICCV 2017 for dense object detection, to regression tasks. The original Focal Loss addresses class imbalance in classification by down-weighting well-classified (easy) examples and focusing training on hard examples through a modulating factor  $(1 - p_t)^\gamma$ . For regression, Focal-R applies a similar principle by assigning higher weights to samples with larger prediction errors (hard examples) and lower weights to samples with smaller errors (easy examples). The focusing parameter  $\gamma$  controls the rate at which easy examples are down-weighted. While this hard example mining strategy has proven effective in object detection, its application to regression tasks with continuous targets may amplify noise, particularly when prediction errors arise from inherent data uncertainty rather than model deficiency.

**SMOBN (Synthetic Minority Over-sampling for Regression with Gaussian Noise).** SMOBN (Branco et al., 2017) was presented at the ECML-PKDD 2017 workshop on Learning with Imbalanced Domains. This method extends the SMOTE algorithm to regression tasks by generating synthetic samples for rare target regions. SMOBN uses a relevance function to identify under-represented regions in the target space, then applies either SMOTER interpolation (for nearby neighbors) or Gaussian noise perturbation (for distant neighbors) to create new training samples.

The algorithm selects the synthesis strategy based on the distance between the seed example and its k-nearest neighbors: closer neighbors trigger interpolation while more distant neighbors trigger noise addition. While SMOBN has shown benefits for traditional machine learning models like Random Forest and MARS, its effectiveness on deep learning models is limited, and the synthetic samples may introduce artificial patterns that do not reflect the true data distribution, particularly problematic for semantically rich domains like financial text.

### A.17.2 Single Task Performance

Table 10 presents the baseline performance of single-task regression, where models are trained solely on sentiment regression using MSE loss. TinyLlama 1.1B achieves the best overall performance with average MSE of 0.0212, followed by Twitter-RoBERTa-Large (0.0241) and Qwen 0.5B (0.0256). Among datasets, INV appears to be the easiest with the lowest average MSE (0.0227), while FXE presents the greatest challenge (0.0246 average MSE). These baseline results serve as the reference point for evaluating all subsequent methods.

### A.17.3 Constant Weight MTL Performance

Table 11 demonstrates that introducing the auxiliary classification task with fixed weights ( $w_r = 0.9, w_c = 0.1$ ) consistently improves performance over single-task baseline. Average MSE reduces from 0.0235 to 0.0227 on NEU (3.4% improvement), from 0.0246 to 0.0237 on FXE (3.7% improvement), and from 0.0227 to 0.0220 on INV (3.1% improvement). These improvements validate the effectiveness of the auxiliary classification task and establish constant weight MTL as a stronger baseline for comparing traditional imbalanced learning methods.

### A.17.4 LDS Performance

Table 12 presents the detailed performance of Label Distribution Smoothing (LDS).

Table 12 reveals the dataset-dependent behavior of LDS. On NEU, LDS achieves substantial improvements over constant weight MTL, with average MSE reducing from 0.0227 to 0.0215 (5.3% improvement). However, on FXE, LDS significantly degrades performance, with average MSE increasing from 0.0237 to 0.0258 (8.9% degradation). This inconsistency suggests that LDS’s

Table 10: Detailed performance of Single Task regression across all model-dataset combinations.

| Model                 | NEU    |        |        |                  | FXE    |        |        |                  | INV    |        |        |                  |
|-----------------------|--------|--------|--------|------------------|--------|--------|--------|------------------|--------|--------|--------|------------------|
|                       | MSE↓   | RMSE↓  | MAE↓   | R <sup>2</sup> ↑ | MSE↓   | RMSE↓  | MAE↓   | R <sup>2</sup> ↑ | MSE↓   | RMSE↓  | MAE↓   | R <sup>2</sup> ↑ |
| Twitter-RoBERTa-Large | 0.0239 | 0.1546 | 0.1103 | 0.7268           | 0.0253 | 0.1591 | 0.1136 | 0.7226           | 0.0230 | 0.1517 | 0.1082 | 0.7366           |
| Qwen 0.5B             | 0.0256 | 0.1600 | 0.1141 | 0.7080           | 0.0262 | 0.1619 | 0.1155 | 0.7042           | 0.0249 | 0.1578 | 0.1125 | 0.7183           |
| TinyLlama 1.1B        | 0.0211 | 0.1453 | 0.1036 | 0.7591           | 0.0224 | 0.1497 | 0.1067 | 0.7554           | 0.0202 | 0.1421 | 0.1014 | 0.7682           |
| <b>Average</b>        | 0.0235 | 0.1533 | 0.1093 | 0.7313           | 0.0246 | 0.1569 | 0.1119 | 0.7274           | 0.0227 | 0.1505 | 0.1074 | 0.7410           |

Table 11: Detailed performance of Constant Weight MTL across all model-dataset combinations.

| Model                 | NEU    |        |        |                  | FXE    |        |        |                  | INV    |        |        |                  |
|-----------------------|--------|--------|--------|------------------|--------|--------|--------|------------------|--------|--------|--------|------------------|
|                       | MSE↓   | RMSE↓  | MAE↓   | R <sup>2</sup> ↑ | MSE↓   | RMSE↓  | MAE↓   | R <sup>2</sup> ↑ | MSE↓   | RMSE↓  | MAE↓   | R <sup>2</sup> ↑ |
| Twitter-RoBERTa-Large | 0.0231 | 0.1520 | 0.1085 | 0.7356           | 0.0244 | 0.1562 | 0.1114 | 0.7332           | 0.0222 | 0.1490 | 0.1063 | 0.7450           |
| Qwen 0.5B             | 0.0247 | 0.1572 | 0.1121 | 0.7184           | 0.0252 | 0.1587 | 0.1133 | 0.7150           | 0.0241 | 0.1552 | 0.1107 | 0.7267           |
| TinyLlama 1.1B        | 0.0204 | 0.1428 | 0.1019 | 0.7672           | 0.0215 | 0.1466 | 0.1046 | 0.7647           | 0.0196 | 0.1400 | 0.0998 | 0.7753           |
| <b>Average</b>        | 0.0227 | 0.1507 | 0.1075 | 0.7404           | 0.0237 | 0.1538 | 0.1098 | 0.7376           | 0.0220 | 0.1481 | 0.1056 | 0.7490           |

Table 12: Detailed performance of LDS across all model-dataset combinations. **Bold**: outperforms Constant Weight MTL.

| Model                 | NEU           |               |               |                  | FXE    |        |        |                  | INV           |               |               |                  |
|-----------------------|---------------|---------------|---------------|------------------|--------|--------|--------|------------------|---------------|---------------|---------------|------------------|
|                       | MSE↓          | RMSE↓         | MAE↓          | R <sup>2</sup> ↑ | MSE↓   | RMSE↓  | MAE↓   | R <sup>2</sup> ↑ | MSE↓          | RMSE↓         | MAE↓          | R <sup>2</sup> ↑ |
| Twitter-RoBERTa-Large | <b>0.0218</b> | <b>0.1476</b> | <b>0.1053</b> | <b>0.7507</b>    | 0.0261 | 0.1616 | 0.1152 | 0.7142           | <b>0.0221</b> | <b>0.1487</b> | <b>0.1059</b> | <b>0.7469</b>    |
| Qwen 0.5B             | <b>0.0236</b> | <b>0.1536</b> | <b>0.1095</b> | <b>0.7304</b>    | 0.0279 | 0.1670 | 0.1189 | 0.6852           | 0.0242        | 0.1556        | 0.1108        | 0.7253           |
| TinyLlama 1.1B        | <b>0.0191</b> | <b>0.1382</b> | <b>0.0985</b> | <b>0.7821</b>    | 0.0234 | 0.1530 | 0.1091 | 0.7438           | <b>0.0195</b> | <b>0.1396</b> | <b>0.0994</b> | <b>0.7768</b>    |
| <b>Average</b>        | <b>0.0215</b> | <b>0.1465</b> | <b>0.1044</b> | <b>0.7544</b>    | 0.0258 | 0.1605 | 0.1144 | 0.7144           | <b>0.0219</b> | <b>0.1480</b> | <b>0.1054</b> | <b>0.7497</b>    |

kernel smoothing assumption does not hold uniformly across different financial sentiment distributions, particularly failing on FXE where complex sentiment expressions violate the assumption that nearby labels share similar characteristics.

### A.17.5 Balanced MSE Performance

Table 13 shows that Balanced MSE exhibits the opposite pattern to LDS. On NEU, Balanced MSE performs poorly with average MSE of 0.0251, which is 10.6% worse than constant weight MTL (0.0227) and 6.8% worse than single-task baseline (0.0235). However, on INV, Balanced MSE achieves competitive results with average MSE of 0.0216, outperforming constant weight MTL by 1.8%. This inconsistency may be attributed to Balanced MSE’s batch-based distribution estimation, which works well when batch statistics accurately represent the global distribution but fails when they are misleading.

### A.17.6 Focal-R Performance

Table 14 demonstrates that Focal-R fails to improve upon the baselines across all configurations. Most notably, Focal-R exhibits severe degradation on FXE, with average MSE of 0.0289 compared to

0.0237 for constant weight MTL (21.9% worse). The Qwen 0.5B model on FXE shows the most extreme degradation with MSE of 0.0312, which is 19.1% worse than single-task baseline. This consistent underperformance indicates that hard example mining amplifies noise in continuous sentiment regression, where high prediction errors often reflect inherent data uncertainty rather than model deficiency.

### A.17.7 SMOGN Performance

Table 15 presents the detailed performance of SMOGN.

Table 15 shows that SMOGN produces the worst overall results among all methods. Compared to single-task baseline, SMOGN increases average MSE by 20.9% on NEU, 28.0% on FXE, and 9.7% on INV. The Qwen 0.5B model on FXE exhibits the most extreme failure with MSE of 0.0342, which is 30.5% worse than single-task baseline. These results confirm that synthetic oversampling through interpolation and Gaussian noise perturbation introduces artificial patterns that are particularly harmful for financial text, where semantic nuances cannot be captured by simple feature-space interpolation.

Table 13: Detailed performance of Balanced MSE across all model-dataset combinations. **Bold**: outperforms Constant Weight MTL.

| Model                 | NEU    |        |        |                  | FXE    |        |        |                  | INV           |               |               |                  |
|-----------------------|--------|--------|--------|------------------|--------|--------|--------|------------------|---------------|---------------|---------------|------------------|
|                       | MSE↓   | RMSE↓  | MAE↓   | R <sup>2</sup> ↑ | MSE↓   | RMSE↓  | MAE↓   | R <sup>2</sup> ↑ | MSE↓          | RMSE↓         | MAE↓          | R <sup>2</sup> ↑ |
| Twitter-RoBERTa-Large | 0.0253 | 0.1591 | 0.1134 | 0.7106           | 0.0248 | 0.1575 | 0.1122 | 0.7287           | <b>0.0219</b> | <b>0.1480</b> | <b>0.1055</b> | <b>0.7490</b>    |
| Qwen 0.5B             | 0.0274 | 0.1655 | 0.1180 | 0.6871           | 0.0259 | 0.1609 | 0.1148 | 0.7076           | <b>0.0238</b> | <b>0.1543</b> | <b>0.1100</b> | <b>0.7296</b>    |
| TinyLlama 1.1B        | 0.0225 | 0.1500 | 0.1069 | 0.7431           | 0.0218 | 0.1476 | 0.1053 | 0.7612           | <b>0.0192</b> | <b>0.1386</b> | <b>0.0988</b> | <b>0.7804</b>    |
| <b>Average</b>        | 0.0251 | 0.1582 | 0.1128 | 0.7136           | 0.0242 | 0.1553 | 0.1108 | 0.7325           | <b>0.0216</b> | <b>0.1470</b> | <b>0.1048</b> | <b>0.7530</b>    |

Table 14: Detailed performance of Focal-R across all model-dataset combinations. **Bold**: outperforms Constant Weight MTL.

| Model                 | NEU    |        |        |                  | FXE    |        |        |                  | INV    |        |        |                  |
|-----------------------|--------|--------|--------|------------------|--------|--------|--------|------------------|--------|--------|--------|------------------|
|                       | MSE↓   | RMSE↓  | MAE↓   | R <sup>2</sup> ↑ | MSE↓   | RMSE↓  | MAE↓   | R <sup>2</sup> ↑ | MSE↓   | RMSE↓  | MAE↓   | R <sup>2</sup> ↑ |
| Twitter-RoBERTa-Large | 0.0242 | 0.1556 | 0.1109 | 0.7234           | 0.0291 | 0.1706 | 0.1216 | 0.6814           | 0.0228 | 0.1510 | 0.1076 | 0.7385           |
| Qwen 0.5B             | 0.0259 | 0.1609 | 0.1147 | 0.7041           | 0.0312 | 0.1766 | 0.1259 | 0.6479           | 0.0248 | 0.1575 | 0.1122 | 0.7167           |
| TinyLlama 1.1B        | 0.0213 | 0.1460 | 0.1041 | 0.7568           | 0.0263 | 0.1622 | 0.1156 | 0.7121           | 0.0199 | 0.1411 | 0.1006 | 0.7718           |
| <b>Average</b>        | 0.0238 | 0.1542 | 0.1099 | 0.7281           | 0.0289 | 0.1698 | 0.1210 | 0.6805           | 0.0225 | 0.1499 | 0.1068 | 0.7423           |

Table 15: Detailed performance of SMOGN across all model-dataset combinations. **Bold**: outperforms Constant Weight MTL.

| Model                 | NEU    |        |        |                  | FXE    |        |        |                  | INV    |        |        |                  |
|-----------------------|--------|--------|--------|------------------|--------|--------|--------|------------------|--------|--------|--------|------------------|
|                       | MSE↓   | RMSE↓  | MAE↓   | R <sup>2</sup> ↑ | MSE↓   | RMSE↓  | MAE↓   | R <sup>2</sup> ↑ | MSE↓   | RMSE↓  | MAE↓   | R <sup>2</sup> ↑ |
| Twitter-RoBERTa-Large | 0.0285 | 0.1688 | 0.1204 | 0.6742           | 0.0317 | 0.1781 | 0.1269 | 0.6528           | 0.0252 | 0.1587 | 0.1132 | 0.7108           |
| Qwen 0.5B             | 0.0311 | 0.1764 | 0.1257 | 0.6447           | 0.0342 | 0.1850 | 0.1318 | 0.6141           | 0.0273 | 0.1652 | 0.1177 | 0.6887           |
| TinyLlama 1.1B        | 0.0256 | 0.1600 | 0.1141 | 0.7074           | 0.0286 | 0.1691 | 0.1206 | 0.6868           | 0.0223 | 0.1493 | 0.1065 | 0.7438           |
| <b>Average</b>        | 0.0284 | 0.1684 | 0.1201 | 0.6754           | 0.0315 | 0.1774 | 0.1264 | 0.6512           | 0.0249 | 0.1577 | 0.1125 | 0.7144           |

### A.17.8 DRNash Performance

Table 16 presents the detailed performance of our proposed DRNash framework.

Table 16 demonstrates that DRNash achieves the best performance across all model-dataset combinations without exception. Compared to single-task baseline, DRNash reduces average MSE by 17.9% on NEU, 16.7% on FXE, and 16.7% on INV. Compared to constant weight MTL, DRNash achieves 15.0% MSE reduction on NEU, 13.5% on FXE, and 14.1% on INV. Notably, DRNash maintains consistent improvements across all three datasets, in stark contrast to traditional methods that exhibit significant dataset-dependent variance, validating our unified framework’s effectiveness in resolving the coupled data-level and task-level challenges.

### A.17.9 Comprehensive Analysis

**Limitations of Traditional Imbalanced Learning Methods.** The detailed experimental results presented in Tables 10 to 16 reveal fundamental limitations of traditional imbalanced learning methods when applied to financial sentiment regression. These limitations can be categorized into

three aspects. First, *dataset-dependent inconsistency*: LDS improves performance on NEU (5.3% MSE reduction) but degrades on FXE (8.9% MSE increase); Balanced MSE shows the opposite pattern with 10.6% degradation on NEU but 1.8% improvement on INV. This inconsistency indicates that these methods rely on assumptions about data distribution that do not hold uniformly across different financial sentiment datasets. Second, *noise amplification*: Focal-R’s hard example mining strategy backfires in regression tasks, with 21.9% average MSE increase on FXE compared to constant weight MTL. In financial sentiment regression, large prediction errors often arise from inherent ambiguity in sentiment expressions rather than model deficiency, making hard example mining counterproductive. Third, *harmful synthetic patterns*: SMOGN’s data augmentation introduces artificial patterns that severely damage model performance, with 20.9 to 28.0% MSE increase compared to single-task baseline. Financial text semantics cannot be captured through simple feature-space interpolation, and synthetic samples mislead the model rather than providing useful supervision.

Table 16: Detailed performance of DRNash (Ours) across all model-dataset combinations. All results represent the best performance.

| Model                 | NEU           |               |               |                  | FXE           |               |               |                  | INV           |               |               |                  |
|-----------------------|---------------|---------------|---------------|------------------|---------------|---------------|---------------|------------------|---------------|---------------|---------------|------------------|
|                       | MSE↓          | RMSE↓         | MAE↓          | R <sup>2</sup> ↑ | MSE↓          | RMSE↓         | MAE↓          | R <sup>2</sup> ↑ | MSE↓          | RMSE↓         | MAE↓          | R <sup>2</sup> ↑ |
| Twitter-RoBERTa-Large | <b>0.0187</b> | <b>0.1368</b> | <b>0.0978</b> | <b>0.7850</b>    | <b>0.0202</b> | <b>0.1421</b> | <b>0.1015</b> | <b>0.7782</b>    | <b>0.0183</b> | <b>0.1353</b> | <b>0.0965</b> | <b>0.7900</b>    |
| Qwen 0.5B             | <b>0.0214</b> | <b>0.1463</b> | <b>0.1044</b> | <b>0.7553</b>    | <b>0.0227</b> | <b>0.1507</b> | <b>0.1074</b> | <b>0.7445</b>    | <b>0.0209</b> | <b>0.1446</b> | <b>0.1031</b> | <b>0.7614</b>    |
| TinyLlama 1.1B        | <b>0.0179</b> | <b>0.1338</b> | <b>0.0954</b> | <b>0.7955</b>    | <b>0.0187</b> | <b>0.1368</b> | <b>0.0976</b> | <b>0.7960</b>    | <b>0.0174</b> | <b>0.1319</b> | <b>0.0941</b> | <b>0.8005</b>    |
| <b>Average</b>        | <b>0.0193</b> | <b>0.1390</b> | <b>0.0992</b> | <b>0.7786</b>    | <b>0.0205</b> | <b>0.1432</b> | <b>0.1022</b> | <b>0.7729</b>    | <b>0.0189</b> | <b>0.1373</b> | <b>0.0979</b> | <b>0.7840</b>    |

**Why DRNash Succeeds.** In contrast to traditional methods, DRNash achieves consistent improvements across all configurations through its unified framework that addresses both data-level and task-level challenges simultaneously. The DRO component handles data imbalance by optimizing worst-case group performance without relying on distribution smoothing assumptions or synthetic data generation. Unlike LDS which assumes label continuity, DRO explicitly optimizes for the worst-case group, ensuring robust performance regardless of the underlying distribution characteristics. The Nash task balancing component dynamically adjusts task weights based on relative progress, avoiding the scale sensitivity that affects methods like Balanced MSE. The combination of these two components resolves the coupled challenges: DRO ensures robust learning across imbalanced sentiment groups, while Nash balancing ensures neither task dominates the optimization, leading to consistent improvements of 14.1% MSE reduction over constant weight MTL across all datasets.

The experimental results demonstrate not only superior average performance but also remarkable consistency. While traditional methods show high variance across datasets (e.g., LDS ranges from +5.3% to -8.9% compared to constant weight MTL), DRNash maintains stable improvements of 13.5 to 15.0% across all three datasets. This consistency is crucial for practical deployment, as it indicates that DRNash’s improvements are robust and reliable rather than dataset-specific. Furthermore, the improvements are consistent across all three model architectures (Twitter-RoBERTa-Large, Qwen 0.5B, and TinyLlama 1.1B), suggesting that DRNash’s benefits are architecture-agnostic and can be expected to generalize to other backbone models.

## Appendix A.18: LoRA Integration

This appendix provides a comprehensive analysis of Low-Rank Adaptation (LoRA) integration

with our DRNash framework. We evaluate four rank configurations (128, 256, 384, 512) across all model-dataset combinations to understand the trade-offs between computational efficiency and model performance.

### A.18.1 Experimental Setup

LoRA enables parameter-efficient fine-tuning by introducing trainable low-rank decomposition matrices into transformer layers while keeping the pre-trained weights frozen. For a pre-trained weight matrix  $W_0 \in \mathbb{R}^{d \times k}$ , LoRA represents the weight update as:

$$W = W_0 + \Delta W = W_0 + BA \quad (67)$$

where  $B \in \mathbb{R}^{d \times r}$  and  $A \in \mathbb{R}^{r \times k}$ , with rank  $r \ll \min(d, k)$ . This reduces the number of trainable parameters from  $d \times k$  to  $r \times (d + k)$ .

We evaluate four rank configurations: 128, 256, 384, and 512. The metrics reported include:

- **Epoch\***: The epoch at which the best validation performance is achieved (out of 100 total epochs).
- **MSE, ACC, Prec., F1**: Standard performance metrics for regression and classification tasks.
- **Time Usage**: Training time relative to full fine-tuning without LoRA (100% = same time as full fine-tuning).

### A.18.2 Results with Lower Ranks (128 and 256)

As shown in Table 17, lower-rank LoRA configurations (128 and 256) exhibit slower convergence, requiring more epochs to reach optimal performance. At Rank=128, the average Epoch\* is 68.3 with Time Usage of 107.58%, indicating that the reduced model capacity leads to slower convergence that negates efficiency gains. Notably, FXE dataset shows Time Usage exceeding 100% for all models at Rank=128 (119.24% for Twitter-RoBERTa-Large, 124.42% for Qwen 0.5B,

Table 17: LoRA integration with DRNash framework: Rank=128 and Rank=256. Epoch\* denotes the epoch achieving best validation performance. Time Usage is relative to full fine-tuning.

| Model                 | Dataset | LoRA (Rank = 128) |        |       |       |       |         | LoRA (Rank = 256) |        |       |       |       |         |
|-----------------------|---------|-------------------|--------|-------|-------|-------|---------|-------------------|--------|-------|-------|-------|---------|
|                       |         | Epoch*            | MSE    | ACC   | Prec. | F1    | Time    | Epoch*            | MSE    | ACC   | Prec. | F1    | Time    |
| Twitter-RoBERTa-Large | NEU     | 62                | 0.0190 | 77.12 | 77.85 | 77.23 | 95.62%  | 47                | 0.0187 | 77.45 | 78.21 | 77.58 | 78.12%  |
|                       | FXE     | 68                | 0.0206 | 76.45 | 77.18 | 76.57 | 119.24% | 54                | 0.0202 | 76.82 | 77.56 | 76.95 | 103.46% |
|                       | INV     | 79                | 0.0186 | 77.82 | 78.54 | 77.91 | 97.46%  | 59                | 0.0184 | 78.08 | 78.89 | 78.23 | 79.08%  |
| Qwen 0.5B             | NEU     | 55                | 0.0218 | 76.28 | 76.95 | 76.42 | 94.37%  | 41                | 0.0214 | 76.78 | 77.52 | 76.91 | 75.88%  |
|                       | FXE     | 61                | 0.0231 | 75.52 | 76.28 | 75.63 | 124.42% | 49                | 0.0227 | 75.91 | 76.73 | 76.05 | 107.04% |
|                       | INV     | 71                | 0.0213 | 76.48 | 77.32 | 76.61 | 90.24%  | 54                | 0.0209 | 77.02 | 77.85 | 77.18 | 74.26%  |
| TinyLlama 1.1B        | NEU     | 65                | 0.0182 | 77.89 | 78.15 | 77.96 | 104.53% | 49                | 0.0180 | 78.25 | 78.48 | 78.32 | 85.06%  |
|                       | FXE     | 72                | 0.0191 | 77.35 | 78.02 | 77.48 | 130.19% | 58                | 0.0189 | 77.72 | 78.35 | 77.85 | 114.70% |
|                       | INV     | 82                | 0.0177 | 78.18 | 78.52 | 78.25 | 112.12% | 63                | 0.0175 | 78.52 | 78.81 | 78.61 | 90.93%  |
| Average               |         | 68.3              | 0.0199 | 77.01 | 77.65 | 77.12 | 107.58% | 52.7              | 0.0196 | 77.39 | 78.04 | 77.52 | 89.84%  |

130.19% for TinyLlama 1.1B), making this configuration counterproductive for challenging datasets. Rank=256 improves efficiency to 89.84% average Time Usage while achieving better MSE (0.0196 vs 0.0199), representing a more practical choice for moderate efficiency requirements.

### A.18.3 Results with Higher Ranks (384 and 512)

Table 18 demonstrates that higher-rank configurations achieve substantial efficiency gains through accelerated convergence. At Rank=384, average Epoch\* reduces to 33.7 with 74.77% Time Usage, providing a balanced trade-off between efficiency and performance. Rank=512 further reduces average Epoch\* to 21.2 with only 50.52% Time Usage while achieving the best average MSE (0.0193). Notably, Twitter-RoBERTa-Large on NEU and INV achieves Time Usage of only 34.16% and 34.53% respectively at Rank=512, representing approximately 3 $\times$  speedup compared to full fine-tuning. Even the challenging FXE dataset shows Time Usage below 100% for all models at Rank=512, confirming that higher ranks enable efficient training across diverse data distributions.

### A.18.4 Comprehensive Analysis

**Efficiency-Performance Trade-off.** Comparing Tables 17 and 18 reveals a clear relationship between LoRA rank and training efficiency. Counter-intuitively, higher ranks lead to faster overall training despite increased per-step computation, because the additional model capacity enables faster convergence. Average Time Usage decreases monotonically from 107.58% (Rank=128) to 89.84% (Rank=256) to 74.77% (Rank=384) to 50.52% (Rank=512). Meanwhile, performance improves consistently: average MSE decreases from 0.0199 to 0.0193 (3.0% reduction) and average

ACC increases from 77.01 to 77.80 (0.79 absolute points) as rank increases from 128 to 512. This pattern suggests that the limiting factor at lower ranks is model expressiveness rather than computational cost.

**Comparison with Full Fine-tuning.** At Rank=128, LoRA configurations show slightly degraded performance compared to full fine-tuning DRNash. Average MSE increases from 0.0196 (full fine-tuning, computed from Table 2 in the main text) to 0.0199, and average ACC decreases from 77.32 to 77.01. However, at Rank=512, LoRA achieves comparable or slightly better performance (MSE 0.0193, ACC 77.80) while requiring only 50.52% of the training time, demonstrating that higher-rank LoRA can serve as an effective regularization mechanism that improves both efficiency and generalization.

**Model-Specific Observations.** Different backbone models exhibit varying responses to LoRA integration:

- **TinyLlama 1.1B** consistently achieves the best absolute performance across all configurations, with MSE ranging from 0.0171 to 0.0191. However, it also shows the highest Time Usage at lower ranks (up to 130.19% at Rank=128 on FXE), indicating greater sensitivity to model capacity constraints.
- **Qwen 0.5B** exhibits moderate efficiency gains but maintains competitive performance. As the smallest model, it shows the largest performance gap between Rank=128 and Rank=512 (MSE improvement from 0.0221 to 0.0214 on average).
- **Twitter-RoBERTa-Large** demonstrates the

Table 18: LoRA integration with DRNash framework: Rank=384 and Rank=512. Epoch\* denotes the epoch achieving best validation performance. Time Usage is relative to full fine-tuning.

| Model                 | Dataset | LoRA (Rank = 384) |        |       |       |       |         | LoRA (Rank = 512) |        |       |       |       |        |
|-----------------------|---------|-------------------|--------|-------|-------|-------|---------|-------------------|--------|-------|-------|-------|--------|
|                       |         | Epoch*            | MSE    | ACC   | Prec. | F1    | Time    | Epoch*            | MSE    | ACC   | Prec. | F1    | Time   |
| Twitter-RoBERTa-Large | NEU     | 29                | 0.0186 | 77.48 | 78.35 | 77.62 | 50.38%  | 18                | 0.0185 | 77.64 | 78.48 | 77.78 | 34.16% |
|                       | FXE     | 35                | 0.0200 | 76.95 | 77.68 | 77.12 | 67.20%  | 22                | 0.0197 | 77.25 | 77.92 | 77.41 | 45.59% |
|                       | INV     | 37                | 0.0182 | 78.31 | 79.15 | 78.48 | 51.83%  | 23                | 0.0180 | 78.46 | 79.35 | 78.63 | 34.53% |
| Qwen 0.5B             | NEU     | 25                | 0.0213 | 76.92 | 77.68 | 77.08 | 67.95%  | 15                | 0.0212 | 77.11 | 77.89 | 77.28 | 42.60% |
|                       | FXE     | 31                | 0.0225 | 76.08 | 76.95 | 76.23 | 91.62%  | 19                | 0.0223 | 76.35 | 77.18 | 76.52 | 60.90% |
|                       | INV     | 34                | 0.0208 | 77.15 | 78.02 | 77.32 | 66.95%  | 21                | 0.0206 | 77.42 | 78.25 | 77.58 | 42.13% |
| TinyLlama 1.1B        | NEU     | 32                | 0.0178 | 78.48 | 78.72 | 78.55 | 80.81%  | 20                | 0.0176 | 78.71 | 78.95 | 78.78 | 56.76% |
|                       | FXE     | 39                | 0.0187 | 78.02 | 78.58 | 78.15 | 108.30% | 26                | 0.0186 | 78.28 | 78.82 | 78.42 | 76.68% |
|                       | INV     | 41                | 0.0173 | 78.72 | 78.98 | 78.79 | 87.92%  | 27                | 0.0171 | 78.97 | 79.21 | 79.05 | 61.32% |
| Average               |         | 33.7              | 0.0195 | 77.57 | 78.23 | 77.70 | 74.77%  | 21.2              | 0.0193 | 77.80 | 78.45 | 77.94 | 50.52% |

most favorable efficiency profile, achieving the lowest Time Usage at Rank=512 (34.16% to 45.59%) while maintaining strong performance, likely due to its encoder-based architecture being well-suited for sentiment classification.

**Dataset-Specific Observations.** The three datasets exhibit distinct patterns under LoRA integration:

- **FXE** (most challenging) shows the highest Time Usage across all configurations, with Rank=128 exceeding 119% for all models. This suggests that challenging datasets require higher model capacity for efficient convergence.
- **INV** (easiest) achieves the lowest MSE and highest ACC across all configurations, with Time Usage generally below the average at each rank level.
- **NEU** shows intermediate behavior, with stable improvements as rank increases.

**Practical Recommendations.** Based on the comprehensive analysis, we provide the following recommendations:

- **Rank=512:** Recommended for maximum efficiency (approximately 50% average time savings) with optimal performance. Suitable for production deployment where both speed and accuracy are critical.
- **Rank=384:** Recommended for balanced scenarios (approximately 25% time savings) with minimal performance trade-off. Suitable for iterative development and hyperparameter tuning.

- **Rank=256:** Acceptable for moderate resource constraints, but provides limited efficiency gains (approximately 10% time savings) compared to full fine-tuning.
- **Rank=128:** Not recommended as it often increases training time (107.58% average) without meaningful efficiency benefits, particularly for challenging datasets like FXE.

These results validate that our DRNash framework can be efficiently deployed through LoRA integration, enabling practical applications in resource-constrained financial sentiment analysis scenarios.

## Appendix A.19: Cross-Domain Generalization

A potential concern is whether the proposed DRNash framework is specifically tailored for financial sentiment analysis or can generalize to other sentiment regression domains. To address this, we evaluate DRNash on three publicly available sentiment polarity regression benchmarks with **continuous** annotations, spanning different text domains, annotation methodologies, and linguistic characteristics.

### A.19.1 Dataset Descriptions

**SemEval-2017 Task 5 (Cortis et al., 2017).** Fine-grained sentiment analysis on financial microblogs and news headlines. Each text is annotated with a **continuous** sentiment score from  $-1$  (very negative/bearish) to  $+1$  (very positive/bullish), representing fine-grained market sentiment. This benchmark is widely used for evaluating financial sentiment regression models. Size: approximately 2,500 instances.

Table 19: Cross-domain evaluation datasets. All datasets provide continuous sentiment polarity annotations suitable for regression tasks.

| Dataset             | Domain      | Original Scale | Normalized Scale | Size   |
|---------------------|-------------|----------------|------------------|--------|
| SemEval-2017 Task 5 | Financial   | $[-1, +1]$     | $[-1, +1]$       | ~2,500 |
| SemEval-2018 V-reg  | Twitter     | $[0, 1]$       | $[-1, +1]$       | ~2,800 |
| EmoBank             | Multi-genre | $[1, 5]$       | $[-1, +1]$       | 10,062 |

**SemEval-2018 Task 1 V-reg (Mohammad et al., 2018).** Twitter valence intensity prediction. Each tweet is annotated with a **continuous** score from 0 (most negative) to 1 (most positive), representing the intensity of sentiment. Annotated using Best-Worst Scaling (BWS) for reliable fine-grained labels. This dataset evaluates whether DRNash transfers to informal social media text. Size: approximately 2,800 tweets.

**EmoBank (Buechel and Hahn, 2017).** Multi-genre dimensional emotion corpus spanning fiction, personal letters, newspaper articles, and essays. Each sentence is annotated with **continuous** Valence scores from 1 to 5 based on the psychological Valence-Arousal-Dominance scheme. This dataset represents the most challenging setting due to its genre diversity. Size: 10,062 sentences.

### A.19.2 Data Preprocessing

To ensure fair comparison of MSE across datasets with different annotation scales, we normalize all continuous sentiment scores to the unified range  $[-1, +1]$ . The normalization transformations are:

- **SemEval-2017 Task 5:** No transformation needed (already  $[-1, +1]$ )
- **SemEval-2018 V-reg:**  $y' = 2y - 1$ , mapping  $[0, 1] \rightarrow [-1, +1]$
- **EmoBank:**  $y' = (y - 3)/2$ , mapping  $[1, 5] \rightarrow [-1, +1]$

This normalization ensures that: (1) MSE values are directly comparable across all datasets, (2) the auxiliary classification task uses consistent thresholds for negative/neutral/positive categories, and (3) the DRNash framework operates identically without any domain-specific modifications.

### A.19.3 Analysis

**Framework Generalizability.** As shown in Table 20, DRNash maintains consistent performance across all three domains without any domain-specific modifications. The average MSE of 0.0199

and ACC of 76.94% demonstrate that the framework’s effectiveness is not limited to a specific text domain. The distributionally robust optimization component and Nash bargaining-based task balancing mechanism capture fundamental aspects of sentiment regression that transfer across financial text, social media, and multi-genre corpora.

**Robustness Across Domains.** As reported in Table 20, all three backbone models show stable performance across datasets. The standard deviation of MSE across datasets is only 0.0017 for Twitter-RoBERTa-Large, 0.0011 for Qwen 0.5B, and 0.0012 for TinyLlama 1.1B, indicating low variance and high robustness. Even on the most challenging EmoBank dataset with diverse genres, all models maintain reasonable performance (MSE 0.0193 to 0.0234), confirming that DRNash does not overfit to specific domain characteristics.

**Implications for Practitioners.** The cross-domain results (Table 20) validate that DRNash can be directly applied to sentiment polarity regression tasks across various domains, including financial analysis, social media monitoring, and general text emotion detection. The only preprocessing required is normalizing target scores to a consistent range. The framework’s domain-agnostic design makes it a practical choice for real-world applications where target domains may vary.

## Appendix A.20: Pre-training Bias Analysis

This section analyzes how pre-training data distributions introduce biases into language models, providing empirical motivation for our distributionally robust optimization approach.

### A.20.1 Pre-trained Language Model

Pre-trained language models learn representations from their training corpora, inevitably inheriting the distributional characteristics and biases present in those data. When deployed on downstream tasks, these inherited biases can either help or hinder performance depending on the alignment between pre-training and target domains. Understanding this

Table 20: Cross-domain evaluation of DRNash framework. Consistent performance across domains demonstrates the framework’s general applicability beyond financial sentiment analysis.

| Model                 | Dataset             | MSE↓   | ACC↑  | Prec.↑ | F1↑   |
|-----------------------|---------------------|--------|-------|--------|-------|
| Twitter-RoBERTa-Large | SemEval-2017 Task 5 | 0.0195 | 77.12 | 77.89  | 77.34 |
|                       | SemEval-2018 V-reg  | 0.0176 | 78.56 | 79.23  | 78.78 |
|                       | EmoBank             | 0.0209 | 76.23 | 77.01  | 76.45 |
| Qwen 0.5B             | SemEval-2017 Task 5 | 0.0221 | 75.45 | 76.23  | 75.67 |
|                       | SemEval-2018 V-reg  | 0.0213 | 76.12 | 76.89  | 76.34 |
|                       | EmoBank             | 0.0234 | 74.78 | 75.56  | 74.98 |
| TinyLlama 1.1B        | SemEval-2017 Task 5 | 0.0182 | 77.89 | 78.45  | 78.12 |
|                       | SemEval-2018 V-reg  | 0.0169 | 78.98 | 79.56  | 79.23 |
|                       | EmoBank             | 0.0193 | 77.34 | 78.01  | 77.56 |
| Average               |                     | 0.0199 | 76.94 | 77.65  | 77.16 |

phenomenon is crucial for developing robust sentiment analysis systems that can handle diverse data distributions.

### A.20.2 Experimental Setup

We compare two RoBERTa variants with different pre-training backgrounds:

**RoBERTa-Large (Liu et al., 2019).** Pre-trained on general text corpora including BookCorpus, English Wikipedia, CC-News, OpenWebText, and Stories. This model provides broad language understanding but lacks specific exposure to social media discourse patterns.

**Twitter-RoBERTa-Large (Loureiro et al., 2023).** Extended from RoBERTa with additional pre-training on approximately 154M tweets.

Both models are evaluated on single-task regression under the identical experimental settings reported in [Appendix A.14](#).

### A.20.3 Analysis

**Performance Gap from Pre-training Bias.** Table 21 demonstrates consistent performance differences between the two models across all datasets. Twitter-RoBERTa-Large achieves 4.8% lower MSE on NEU (0.0239 vs 0.0251), 5.2% lower on FXE (0.0253 vs 0.0267), and 5.3% lower on INV (0.0230 vs 0.0243). The  $R^2$  improvements range from 1.34% to 1.74% absolute. These gaps arise not from architectural differences—both models share identical architectures—but from differences in pre-training data distributions.

**Inherited Distributional Bias.** The superior performance of Twitter-RoBERTa-Large on financial sentiment tasks can be attributed to its pre-training

exposure to social media text, which shares linguistic characteristics with our financial datasets. However, this also illustrates a fundamental limitation: the model has inherited biases from its pre-training distribution. When encountering text samples that deviate from this distribution, performance degradation is expected.

### Implications for Robust Sentiment Analysis.

This analysis reveals that even state-of-the-art pre-trained models carry distributional biases that affect downstream task performance. While selecting a domain-aligned model (Twitter-RoBERTa-Large) provides initial improvements, it does not fundamentally solve the problem of label imbalance. Real-world financial text remains heavily concentrated in the neutral band regardless of the backbone chosen, making it impractical to rely solely on pre-training alignment.

**Motivation for DRO.** These findings motivate our adoption of DRO in the DRNash framework. Rather than optimizing the average loss, which is dominated by the densely populated neutral region, DRO optimizes for worst-case performance across sentiment groups. This approach directly addresses the distributional bias problem by preventing the sparse extreme regions from being absorbed into an average that the majority region already satisfies, regardless of the pre-training corpus characteristics. Combined with Nash bargaining-based task balancing, DRNash mitigates both pre-training biases and task imbalance issues in multi-task sentiment analysis.

Table 21: Comparison of pre-trained models on financial sentiment regression. Twitter-RoBERTa-Large consistently outperforms RoBERTa-Large across all three datasets.

| Dataset | Model                 | Epoch* | MSE↓   | MAE↓   | RMSE↓  | R <sup>2</sup> ↑ |
|---------|-----------------------|--------|--------|--------|--------|------------------|
| NEU     | RoBERTa-Large         | 78     | 0.0251 | 0.1134 | 0.1584 | 71.34            |
|         | Twitter-RoBERTa-Large | 72     | 0.0239 | 0.1103 | 0.1546 | 72.68            |
| FXE     | RoBERTa-Large         | 71     | 0.0267 | 0.1172 | 0.1634 | 70.52            |
|         | Twitter-RoBERTa-Large | 68     | 0.0253 | 0.1136 | 0.1591 | 72.26            |
| INV     | RoBERTa-Large         | 82     | 0.0243 | 0.1112 | 0.1559 | 72.18            |
|         | Twitter-RoBERTa-Large | 75     | 0.0230 | 0.1082 | 0.1517 | 73.66            |

## Appendix A.21: Component Ablation Study

This section presents a comprehensive ablation study to validate the independent and synergistic contributions of the two core components in DR-Nash: DRO and NTC.

### A.21.1 Experimental Configurations

We design four ablation configurations to isolate the contribution of each component:

**Single Task.** Pure regression baseline without multi-task learning. The model is trained solely on the sentiment polarity regression objective using standard MSE loss.

**Constant Weight MTL.** Multi-task learning with standard loss computation (uniform averaging over samples) and fixed task weights ( $w_r = 0.9$ ,  $w_c = 0.1$ ). This represents traditional MTL without either DRO or NTC.

**MTL + DRO only.** Multi-task learning with DRO for robust loss computation via worst-case aggregation over sentiment groups (Eq. 5), combined with **re-tuned fixed task weights**. Since DRO transforms the loss landscape, the optimal fixed weights differ from the standard setting. We conduct a grid search over  $w_r \in \{0.80, 0.85, 0.90, 0.95\}$  and find  $w_r = 0.85$ ,  $w_c = 0.15$  yields the best average performance under DRO. This ensures a fair evaluation of DRO’s independent contribution by providing it with appropriate task balancing.

**MTL + NTC only.** Multi-task learning with standard loss computation (uniform averaging), but with NTC for dynamic task weighting via Nash bargaining. This isolates the contribution of adaptive task coordination at the task level, though the gradients used for Nash bargaining remain biased toward over-represented neutral samples.

**DRNash (Full).** Complete framework combining DRO for robust loss computation and NTC for dynamic task weighting, representing our proposed approach.

Table 22: Summary of ablation configurations. Note that DRO only uses re-tuned weights to ensure fair comparison.

| Configuration       | Loss Computation       | Task Weighting                         |
|---------------------|------------------------|----------------------------------------|
| Single Task         | Standard MSE           | Regression only                        |
| Constant Weight MTL | Uniform averaging      | Fixed ( $w_r = 0.9$ , $w_c = 0.1$ )    |
| MTL + DRO only      | Worst-case aggregation | Fixed ( $w_r = 0.85$ , $w_c = 0.15$ )* |
| MTL + NTC only      | Uniform averaging      | Nash bargaining                        |
| DRNash (Full)       | Worst-case aggregation | Nash bargaining                        |

\*Re-tuned via grid search under DRO setting to ensure fair evaluation.

### A.21.2 Why Re-tuning Fixed Weights for DRO Only?

A critical methodological consideration is whether to use the same fixed weights ( $w_r = 0.9$ ,  $w_c = 0.1$ ) for both Constant Weight MTL and DRO only configurations. We argue that re-tuning is necessary for a fair comparison:

**DRO Transforms the Loss Landscape.** The DRO formulation applies worst-case aggregation:

$$L_t^{\text{DRO}} = \frac{1}{\eta} \log \left( \sum_k \hat{q}_k \exp(\eta L_t^{(k)}) \right) \quad (68)$$

This transformation changes the relative magnitudes of regression and classification losses. Specifically, groups with higher losses (typically extreme sentiment groups) receive exponentially higher weights. Since extreme sentiment groups often have higher regression MSE (due to data scarcity) but not necessarily higher classification CE (as the five-class task may be easier for extreme samples with clearer sentiment), the ratio  $L_r^{\text{DRO}}/L_c^{\text{DRO}}$  differs from  $L_r/L_c$ .

**Empirical Validation.** Table 23 shows the performance of DRO only under different fixed weight configurations:

Table 23: DRO only performance under different fixed weights (MSE on NEU dataset).

| $w_r$       | $w_c$       | MSE           | $\Delta$ vs optimal |
|-------------|-------------|---------------|---------------------|
| 0.95        | 0.05        | 0.0218        | +3.8%               |
| 0.90        | 0.10        | 0.0214        | +1.9%               |
| <b>0.85</b> | <b>0.15</b> | <b>0.0210</b> | —                   |
| 0.80        | 0.20        | 0.0216        | +2.9%               |

Using the original weights ( $w_r = 0.9$ ) would yield MSE of 0.0214 instead of 0.0210, underestimating DRO’s true contribution by approximately 1.9%. The shift toward higher classification weight ( $w_c = 0.15$ ) compensates for the fact that DRO amplifies regression loss more than classification loss for extreme sentiment groups.

### A.21.3 Results

Table 24: Ablation study results on Twitter-RoBERTa-Large. Each component contributes independently, and their combination achieves synergistic improvements.

| Configuration        | Dataset | MSE↓          | MAE↓          | R <sup>2</sup> ↑ |
|----------------------|---------|---------------|---------------|------------------|
| Single Task          | NEU     | 0.0239        | 0.1103        | 72.68            |
|                      | FXE     | 0.0253        | 0.1136        | 72.26            |
|                      | INV     | 0.0230        | 0.1082        | 73.66            |
| Constant Weight MTL  | NEU     | 0.0231        | 0.1085        | 73.56            |
|                      | FXE     | 0.0244        | 0.1114        | 73.32            |
|                      | INV     | 0.0222        | 0.1063        | 74.50            |
| MTL + DRO only       | NEU     | 0.0210        | 0.1033        | 75.98            |
|                      | FXE     | 0.0219        | 0.1055        | 76.05            |
|                      | INV     | 0.0206        | 0.1023        | 76.39            |
| MTL + NTC only       | NEU     | 0.0217        | 0.1050        | 75.18            |
|                      | FXE     | 0.0231        | 0.1083        | 74.72            |
|                      | INV     | 0.0210        | 0.1033        | 75.89            |
| <b>DRNash (Full)</b> | NEU     | <b>0.0187</b> | <b>0.0978</b> | <b>78.50</b>     |
|                      | FXE     | <b>0.0202</b> | <b>0.1015</b> | <b>77.82</b>     |
|                      | INV     | <b>0.0183</b> | <b>0.0965</b> | <b>79.00</b>     |

### A.21.4 Analysis

**Independent Contribution of DRO.** DRO addresses the data-level challenge by replacing uniform loss averaging with worst-case aggregation over sentiment groups. Comparing MTL + DRO only against Constant Weight MTL:

DRO independently reduces MSE by 8.8% on average, with notable dataset-specific variation (7.2% to 10.2%). **FXE benefits most from DRO (10.2%)**, which is consistent with it having the

Table 25: Independent contribution of DRO module.

| Dataset        | Constant MTL | MTL + DRO | Improvement |
|----------------|--------------|-----------|-------------|
| NEU            | 0.0231       | 0.0210    | 9.1%        |
| FXE            | 0.0244       | 0.0219    | 10.2%       |
| INV            | 0.0222       | 0.0206    | 7.2%        |
| <b>Average</b> | 0.0232       | 0.0212    | <b>8.8%</b> |

most concentrated label distribution of the three, so that extreme sentiment samples are rarest and most underrepresented. DRO’s worst-case aggregation explicitly addresses this by assigning higher weights to these underperforming extreme groups. **INV benefits least (7.2%)**, consistent with its larger sample size (23,241) providing more balanced representation even without DRO intervention.

**Independent Contribution of NTC.** NTC addresses the task-level challenge by replacing fixed task weights with dynamic weights derived from Nash bargaining. Comparing MTL + NTC only against Constant Weight MTL:

Table 26: Independent contribution of NTC module.

| Dataset        | Constant MTL | MTL + NTC | Improvement |
|----------------|--------------|-----------|-------------|
| NEU            | 0.0231       | 0.0217    | 6.1%        |
| FXE            | 0.0244       | 0.0231    | 5.3%        |
| INV            | 0.0222       | 0.0210    | 5.4%        |
| <b>Average</b> | 0.0232       | 0.0219    | <b>5.6%</b> |

NTC independently reduces MSE by 5.6% on average. Critically, **the improvement pattern differs from DRO**: FXE benefits *least* from NTC (5.3%), despite benefiting *most* from DRO. This apparent contradiction reveals an important insight: NTC computes task weights based on gradient magnitudes, but when using standard loss averaging, these gradients are dominated by over-represented neutral samples. For FXE with its narrow distribution, this gradient bias is most severe, meaning NTC’s scale-invariant balancing operates on fundamentally biased signals. The task weights are correctly balanced in magnitude, but the gradient *directions* still favor neutral samples.

**Why DRO Contributes More Than NTC.** The data shows DRO (8.8%) contributes more than NTC (5.6%) on average. This is not surprising given our problem formulation: the data-level challenge (distribution imbalance) is the *root cause* that creates biased gradients, while the task-level challenge (gradient conflicts) is a *downstream effect*.

NTC can balance task contributions but cannot correct the fundamental bias in gradient directions. DRO addresses the root cause by ensuring all sentiment regions contribute meaningfully to the optimization.

**Synergistic Effects.** To quantify the synergy between DRO and NTC, we compare the actual improvement of DRNash against the expected improvement from simply summing individual contributions:

Table 27: Synergistic effects analysis.

| Dataset        | DRO Gain    | NTC Gain    | Expected     | Actual       |
|----------------|-------------|-------------|--------------|--------------|
| NEU            | 9.1%        | 6.1%        | 15.2%        | 19.0%        |
| FXE            | 10.2%       | 5.3%        | 15.5%        | 17.2%        |
| INV            | 7.2%        | 5.4%        | 12.6%        | 17.6%        |
| <b>Average</b> | <b>8.8%</b> | <b>5.6%</b> | <b>14.4%</b> | <b>17.9%</b> |

The synergy analysis reveals:

- **NEU:** Expected 15.2%, actual 19.0%, synergy bonus 3.8% (25% enhancement)
- **FXE:** Expected 15.5%, actual 17.2%, synergy bonus 1.7% (11% enhancement)
- **INV:** Expected 12.6%, actual 17.6%, synergy bonus 5.0% (40% enhancement)

On average, DRNash achieves 3.5% additional improvement beyond the sum of individual contributions, representing approximately 24% synergistic enhancement.

**Understanding the Synergy Mechanism.** The synergy arises because DRO and NTC address complementary aspects of the coupled challenges:

1. **DRO enables effective NTC:** When DRO ensures balanced gradient contributions from all sentiment groups, NTC’s Nash bargaining operates on unbiased gradient signals. The task weights computed from these balanced gradients more accurately reflect true task difficulty rather than data distribution artifacts.
2. **NTC amplifies DRO’s benefit:** When NTC dynamically adjusts task weights, it prevents either task from dominating the optimization. This allows DRO’s robust gradients to effectively update the model for both regression and classification objectives simultaneously.

The dataset-specific synergy patterns support this interpretation: INV shows the largest synergy (40%) because its more balanced baseline distribution allows NTC to work more effectively once DRO provides unbiased gradients. FXE shows the smallest synergy (11%) because even with DRO, the extreme concentration of its distribution creates inherent challenges that limit additional gains from NTC.

## A.21.5 Summary

The ablation study reveals several key findings:

1. **Both components are essential:** Removing either DRO or NTC results in significant performance degradation compared to the full DRNash framework.
2. **DRO contributes more independently:** DRO provides 8.8% average MSE reduction versus 5.6% for NTC, reflecting that data-level distributional bias is the root cause of the coupled challenges.
3. **Dataset-specific response patterns differ:** DRO benefits the most concentrated dataset (FXE) most, while NTC benefits it least. This reveals the complementary nature of the two components.
4. **Synergistic effects validate the coupling hypothesis:** The 3.5% additional improvement (24% enhancement) beyond individual contributions confirms that the data-level and task-level challenges are indeed coupled and benefit from joint resolution.
5. **Methodological rigor matters:** Re-tuning fixed weights for DRO only configuration is necessary for fair evaluation, as DRO transforms the loss landscape and changes optimal task weight ratios.

## Appendix A.22: Sensitivity: Group Number $K$

This appendix investigates how the number of sentiment groups  $K$  in the DRO component affects model performance. The group number  $K$  determines the granularity of sentiment partitioning and directly influences the effectiveness of worst-case aggregation.

### A.22.1 Experimental Setup

We evaluate three configurations of sentiment group numbers:  $K = 3$  (Negative, Neutral, Positive),  $K = 5$  (Strong Negative, Negative, Neutral, Positive, Strong Positive), and  $K = 7$  (adding Very Strong categories at both extremes). All other hyperparameters are fixed: robustness parameter  $\eta = 1.0$ , batch size 32, learning rate  $1 \times 10^{-5}$ , and 100 training epochs. We use Twitter-RoBERTa-Large as the backbone model across all experiments. For each configuration, we report MSE (primary regression metric), MAE,  $R^2$ , and  $K$ -class classification accuracy. Note that classification accuracy is computed over  $K$  classes and thus not directly comparable across different  $K$  values due to varying task difficulty.

### A.22.2 Baseline References

To contextualize the ablation results, we provide reference performance levels from the main experiments. Single Task regression achieves MSE of 0.0239 (NEU), 0.0253 (FXE), and 0.0230 (INV). Constant Weight MTL improves upon this with MSE of 0.0231, 0.0244, and 0.0222 respectively. Our DRNash framework with  $K = 5$  achieves the best performance at 0.0187, 0.0202, and 0.0183. The ablation configurations ( $K = 3$  and  $K = 7$ ) are expected to fall between Constant Weight MTL and DRNash, as they represent partial degradation of the DRO mechanism rather than complete failure.

### A.22.3 Results and Analysis

Table 28: Sensitivity analysis on sentiment group number  $K$  using Twitter-RoBERTa-Large. **Bold:** best results.  $\Delta$ MSE shows relative change compared to  $K = 5$ . Classification accuracy is computed over  $K$  classes and reflects task difficulty rather than model capability.

| $K$ | Dataset        | MSE↓          | MAE↓          | $R^2$ ↑      | $K$ -class ACC | $\Delta$ MSE |
|-----|----------------|---------------|---------------|--------------|----------------|--------------|
| 3   | NEU            | 0.0204        | 0.1019        | 76.67        | 80.12          | +9.1%        |
|     | FXE            | 0.0223        | 0.1064        | 75.53        | 79.45          | +10.4%       |
|     | INV            | 0.0198        | 0.1003        | 77.31        | 80.89          | +8.2%        |
|     | <b>Average</b> | 0.0208        | 0.1029        | 76.50        | 80.15          | +9.2%        |
| 5   | NEU            | <b>0.0187</b> | <b>0.0978</b> | <b>78.50</b> | 77.34          | —            |
|     | FXE            | <b>0.0202</b> | <b>0.1015</b> | <b>77.82</b> | 76.78          | —            |
|     | INV            | <b>0.0183</b> | <b>0.0965</b> | <b>79.00</b> | 78.09          | —            |
|     | <b>Average</b> | <b>0.0191</b> | <b>0.0986</b> | <b>78.44</b> | 77.40          | —            |
| 7   | NEU            | 0.0198        | 0.1003        | 77.35        | 71.67          | +5.9%        |
|     | FXE            | 0.0219        | 0.1055        | 75.97        | 70.23          | +8.4%        |
|     | INV            | 0.0190        | 0.0983        | 78.23        | 73.45          | +3.8%        |
|     | <b>Average</b> | 0.0202        | 0.1014        | 77.18        | 71.78          | +6.0%        |

The results in Table 28 demonstrate that both

$K = 3$  and  $K = 7$  degrade performance compared to the optimal  $K = 5$ , but the degradation mechanisms differ fundamentally, leading to distinct patterns across datasets.

With  $K = 3$ , performance degrades by 8.2% to 10.4% across datasets, with all configurations remaining substantially better than Constant Weight MTL (which achieved MSE of 0.0231, 0.0244, 0.0222) but notably worse than  $K = 5$ . The degradation is caused by insufficient granularity: extreme sentiments are merged into broader categories, preventing DRO from distinguishing truly extreme samples from moderate ones within the same group. The auxiliary classification task also provides weaker supervision, as three-class labels offer less discriminative information about sentiment intensity. Importantly,  $K = 3$  does not introduce additional noise or instability, it simply weakens the DRO mechanism by reducing its discriminative power. FXE exhibits the largest degradation (+10.4%) because its scores are the most concentrated of the three, so coarser quantile boundaries lose proportionally more granularity; when scores are already concentrated, coarse partitioning fails to capture meaningful distinctions between adjacent sentiment levels. INV shows the smallest degradation (+8.2%) as its larger sample size (23,241) provides robust training signals that partially compensate for the reduced group resolution.

With  $K = 7$ , the degradation pattern differs markedly across datasets, ranging from +3.8% (INV) to +8.4% (FXE). Unlike  $K = 3$  where the problem is systematic,  $K = 7$  introduces variance-related issues: each extreme group contains only  $\sim N/7$  samples, leading to noisier batch-wise group loss estimates  $L_t^{(k)}$ . This variance propagates to the worst-case weights  $q_k^* \propto \exp(\eta L_t^{(k)})$ , causing less stable optimization. The dataset-specific sensitivity directly reflects sample size effects: INV’s large sample count ensures approximately 3,320 samples per group even at  $K = 7$ , providing reliable loss estimates and resulting in minimal degradation (+3.8%). In contrast, FXE’s smaller size (10,461 samples) yields only  $\sim 1,494$  samples per extreme group, and its narrow distribution amplifies the impact of estimation variance, producing the largest degradation (+8.4%). NEU falls between these extremes with moderate degradation (+5.9%).

The comparison between  $K = 3$  and  $K = 7$  reveals an interesting trade-off. On average,  $K = 7$  outperforms  $K = 3$  (6.0% vs 9.2% degradation),

suggesting that finer granularity partially compensates for sample sparsity. However, this advantage is dataset-dependent: on INV,  $K = 7$  substantially outperforms  $K = 3$  (3.8% vs 8.2%), while on FXE, the gap narrows considerably (8.4% vs 10.4%). This indicates that for datasets with limited samples or narrow distributions, the benefits of fine granularity may be offset by estimation instability.

The  $K$ -class classification accuracy exhibits an expected pattern where absolute accuracy decreases as  $K$  increases: 80.15% average at  $K = 3$ , 77.40% at  $K = 5$ , and 71.78% at  $K = 7$ . However, this decrease reflects task difficulty rather than model capability. When normalized by random baseline performance, the relative improvements tell a different story: at  $K = 3$ , the model achieves 141% improvement over random baseline (33.3%); at  $K = 5$ , 287% improvement over baseline (20.0%); and at  $K = 7$ , 402% improvement over baseline (14.3%). This indicates that the model successfully learns finer sentiment distinctions at higher  $K$ , even though the absolute classification accuracy decreases.

#### A.22.4 Summary

Based on our analysis,  $K = 5$  is recommended as the default configuration for DRNash. Both alternatives degrade performance but remain superior to Constant Weight MTL, confirming that DRO provides value even with suboptimal group configurations. The choice between  $K = 3$  and  $K = 7$  involves a trade-off between systematic bias (insufficient granularity) and estimation variance (sample sparsity). For most applications,  $K = 5$  provides the optimal balance. For very large datasets ( $N > 50,000$ ),  $K = 7$  may be considered, while for small datasets with narrow distributions,  $K = 3$  may be more stable despite its lower resolution.

### Appendix A.23: Sensitivity: Batch Size

This appendix investigates how batch size affects the DRO component’s effectiveness. Batch size directly influences two critical factors: the probability of covering all  $K$  sentiment groups within each batch, and the variance of group-wise loss estimates.

#### A.23.1 Experimental Setup

We evaluate nine batch size configurations:  $B \in \{8, 12, 16, 20, 24, 28, 32, 36, 40\}$ . All other hyperparameters are fixed: number of groups  $K = 5$ , robustness parameter  $\eta = 1.0$ , learning rate  $1 \times 10^{-5}$ ,

and 100 training epochs. We use Twitter-RoBERTa-Large as the backbone model.

With  $K = 5$  groups under quantile-based partitioning, each group contains approximately 20% of the training samples. For a batch of size  $B$ , the expected number of samples per group is  $B/K = B/5$ , and the probability that a specific group is absent from a batch is  $P(\text{group } k \text{ absent}) = 0.8^B$ . Table 29 summarizes these statistics across different batch sizes.

At  $B = 8$ , with only 1.6 expected samples per group and 16.8% probability of a given group being absent, the batch-wise group losses are dominated by sampling noise. At  $B = 12$ , this probability drops to 6.9% but remains problematic with only 2.4 expected samples per group. At  $B = 32$ , the absence probability drops below 0.1%, ensuring reliable group coverage with 6.4 expected samples per group.

#### A.23.2 Baseline References

To contextualize the ablation results, we provide critical reference performance levels. Single Task regression achieves MSE of 0.0239 (NEU), 0.0253 (FXE), and 0.0230 (INV). Constant Weight MTL improves to 0.0231, 0.0244, and 0.0222 respectively. DRNash with  $B = 32$  achieves optimal performance at 0.0187, 0.0202, and 0.0183. Critically, unlike the  $K$  ablation where suboptimal configurations merely weaken DRO without introducing harm, very small batch sizes can cause DRO to *catastrophically degrade* performance well below Single Task baseline by providing noisy, misleading robustness signals that actively misguide optimization.

#### A.23.3 Results and Analysis

The results in Table 30 reveal four distinct performance regimes with a critical finding: small batch sizes cause DRO to perform *catastrophically worse than the Single Task baseline*, demonstrating that distributionally robust optimization guided by unreliable estimates is far more harmful than having no robustness at all.

**Harmful Regime ( $B \leq 12$ ).** Small batch sizes cause DRO to catastrophically degrade performance well below the Single Task baseline. At  $B = 8$ , all three datasets show severe MSE degradation compared to Single Task: NEU achieves 0.0271 versus Single Task’s 0.0239 (+13.4% worse), FXE achieves 0.0324 versus

Table 29: Batch size statistics for  $K = 5$  groups under quantile-based partitioning.

| Batch Size | Expected Samples/Group | Group Absence Prob. | Expected Missing Groups | Regime     |
|------------|------------------------|---------------------|-------------------------|------------|
| 8          | 1.6                    | 16.8%               | 0.84                    | Harmful    |
| 12         | 2.4                    | 6.9%                | 0.35                    | Harmful    |
| 16         | 3.2                    | 2.8%                | 0.14                    | Recovery   |
| 20         | 4.0                    | 1.2%                | 0.06                    | Recovery   |
| 24         | 4.8                    | 0.5%                | 0.02                    | Stable     |
| 28         | 5.6                    | 0.2%                | 0.01                    | Stable     |
| 32         | 6.4                    | 0.08%               | <0.01                   | Optimal    |
| 36         | 7.2                    | 0.03%               | <0.01                   | Saturation |
| 40         | 8.0                    | 0.01%               | <0.01                   | Saturation |

0.0253 (+28.1% worse), and INV achieves 0.0259 versus 0.0230 (+12.6% worse). The degradation relative to optimal DRNash ( $B = 32$ ) ranges from +41.5% (INV) to +60.4% (FXE), representing catastrophic failure of the robustness mechanism.

This severe degradation stems from two compounding factors. First, with only 1.6 expected samples per group and 16.8% probability of a given group being absent, the batch-wise group losses  $L_t^{(k)}$  are dominated by sampling noise rather than true performance differences. Second, the worst-case weights  $q_k^* \propto \exp(\eta L_t^{(k)})$  amplify this noise exponentially, causing the optimization to chase random fluctuations rather than systematically addressing underperforming regions. The result is that DRO provides actively misleading guidance, directing the model to over-optimize whichever group happens to have high loss due to sampling variance in a given batch, then switching to a different group in the next batch. This erratic behavior is substantially worse than the consistent (if suboptimal) training provided by standard loss averaging in Single Task.

At  $B = 12$ , the situation improves but remains firmly in the harmful regime. All three datasets still perform worse than their Single Task baselines: NEU achieves 0.0248 versus 0.0239 (+3.8% worse), FXE achieves 0.0286 versus 0.0253 (+13.0% worse), and INV achieves 0.0240 versus 0.0230 (+4.3% worse). While the average degradation reduces from +48.9% to +35.1% relative to optimal, the 2.4 expected samples per group remain insufficient for reliable loss estimation. Notably, FXE exhibits substantially larger degradation (+13.0%) compared to NEU (+3.8%) and INV (+4.3%), foreshadowing its heightened sensitivity that persists across all batch sizes.

**Transition Analysis at  $B = 12$ .** The  $B = 12$  configuration represents a critical transition point that illuminates the mechanism of DRO failure. With 2.4 expected samples per group (compared to 1.6 at  $B = 8$ ) and 6.9% group absence probability (compared to 16.8% at  $B = 8$ ), the estimation variance decreases by approximately 33%. However, this improvement remains insufficient for reliable DRO operation, as evidenced by all datasets remaining below Single Task performance.

The asymmetry in degradation at  $B = 12$  is particularly informative: FXE shows +13.0% degradation relative to Single Task, while NEU and INV show only +3.8% and +4.3%. This three-fold difference in sensitivity directly reflects FXE’s distributional characteristics. With the smallest dataset among the three (10,461 samples), FXE yields the noisiest per-group loss estimates at a given batch size. When estimation noise is comparable in magnitude to true group loss differences, the worst-case weights become essentially random, causing optimization to allocate effort based on noise rather than genuine performance gaps. This phenomenon explains why FXE requires larger batch sizes to benefit from DRO.

**Recovery Regime ( $16 \leq B \leq 20$ ).** Performance transitions from harmful to beneficial as batch size increases. At  $B = 16$ , NEU (0.0234) and INV (0.0227) surpass their Single Task baselines (0.0239 and 0.0230 respectively), while FXE (0.0261) remains marginally below its baseline (0.0253, +3.2% worse). The 3.2 expected samples per group provide sufficient stability for DRO to begin functioning correctly on datasets with wider distributions, but FXE’s narrow distribution still suffers from noise contamination.

At  $B = 20$ , all datasets surpass their Single Task baselines and approach Constant Weight MTL per-

Table 30: Sensitivity analysis on batch size using Twitter-RoBERTa-Large with  $K = 5$ . **Bold:** the reference configuration  $B = 32$ .  $\Delta$ MSE shows relative change compared to  $B = 32$ . Entries marked with  $\dagger$  indicate performance worse than Single Task baseline.

| Batch     | Dataset        | MSE $\downarrow$ | MAE $\downarrow$ | $R^2\uparrow$ | ACC          | $\Delta$ MSE |
|-----------|----------------|------------------|------------------|---------------|--------------|--------------|
| 8         | NEU            | 0.0271 $\dagger$ | 0.1174           | 69.02         | 72.23        | +44.9%       |
|           | FXE            | 0.0324 $\dagger$ | 0.1284           | 64.51         | 69.67        | +60.4%       |
|           | INV            | 0.0259 $\dagger$ | 0.1147           | 70.32         | 72.89        | +41.5%       |
|           | <b>Average</b> | 0.0285 $\dagger$ | 0.1202           | 67.95         | 71.60        | +48.9%       |
| 12        | NEU            | 0.0248 $\dagger$ | 0.1123           | 71.65         | 74.12        | +32.6%       |
|           | FXE            | 0.0286 $\dagger$ | 0.1206           | 68.67         | 72.45        | +41.6%       |
|           | INV            | 0.0240 $\dagger$ | 0.1104           | 72.49         | 74.78        | +31.1%       |
|           | <b>Average</b> | 0.0258 $\dagger$ | 0.1144           | 70.94         | 73.78        | +35.1%       |
| 16        | NEU            | 0.0234           | 0.1090           | 73.26         | 74.89        | +25.1%       |
|           | FXE            | 0.0261 $\dagger$ | 0.1152           | 70.17         | 73.56        | +29.2%       |
|           | INV            | 0.0227           | 0.1074           | 74.06         | 75.67        | +24.0%       |
|           | <b>Average</b> | 0.0241           | 0.1105           | 72.50         | 74.71        | +26.1%       |
| 20        | NEU            | 0.0216           | 0.1048           | 75.31         | 75.78        | +15.5%       |
|           | FXE            | 0.0238           | 0.1100           | 73.03         | 74.67        | +17.8%       |
|           | INV            | 0.0207           | 0.1026           | 76.23         | 76.45        | +13.1%       |
|           | <b>Average</b> | 0.0220           | 0.1058           | 74.86         | 75.63        | +15.5%       |
| 24        | NEU            | 0.0201           | 0.1012           | 77.03         | 76.56        | +7.5%        |
|           | FXE            | 0.0220           | 0.1055           | 75.26         | 75.45        | +8.9%        |
|           | INV            | 0.0194           | 0.0994           | 77.77         | 77.23        | +6.0%        |
|           | <b>Average</b> | 0.0205           | 0.1020           | 76.69         | 76.41        | +7.5%        |
| 28        | NEU            | 0.0192           | 0.0989           | 78.01         | 77.01        | +2.7%        |
|           | FXE            | 0.0209           | 0.1030           | 76.48         | 76.12        | +3.5%        |
|           | INV            | 0.0187           | 0.0975           | 78.54         | 77.78        | +2.2%        |
|           | <b>Average</b> | 0.0196           | 0.0998           | 77.68         | 76.97        | +2.8%        |
| <b>32</b> | NEU            | <b>0.0187</b>    | <b>0.0978</b>    | <b>78.50</b>  | <b>77.34</b> | —            |
|           | FXE            | <b>0.0202</b>    | <b>0.1015</b>    | <b>77.82</b>  | <b>76.78</b> | —            |
|           | INV            | <b>0.0183</b>    | <b>0.0965</b>    | <b>79.00</b>  | <b>78.09</b> | —            |
|           | <b>Average</b> | <b>0.0191</b>    | <b>0.0986</b>    | <b>78.44</b>  | <b>77.40</b> | —            |
| 36        | NEU            | 0.0186           | 0.0972           | 78.62         | 77.45        | −0.5%        |
|           | FXE            | 0.0201           | 0.1010           | 77.93         | 76.89        | −0.5%        |
|           | INV            | 0.0182           | 0.0962           | 79.12         | 78.18        | −0.5%        |
|           | <b>Average</b> | 0.0190           | 0.0981           | 78.56         | 77.51        | −0.5%        |
| 40        | NEU            | 0.0185           | 0.0970           | 78.86         | 77.56        | −1.1%        |
|           | FXE            | 0.0199           | 0.1005           | 78.17         | 77.01        | −1.5%        |
|           | INV            | 0.0181           | 0.0959           | 79.26         | 78.27        | −1.1%        |
|           | <b>Average</b> | 0.0188           | 0.0978           | 78.76         | 77.61        | −1.2%        |

formance. The 4.0 expected samples per group provide sufficient stability for DRO to function correctly more often than not. However, with group coverage probability per batch still at 98.8%, occasional estimation errors prevent reaching optimal performance. The degradation relative to  $B = 32$  reduces to 13.1% (INV) to 17.8% (FXE), reflecting the remaining variance-related limitations.

**Stable Regime** ( $24 \leq B \leq 32$ ). Performance approaches optimal with diminishing improvements. At  $B = 24$ , degradation reduces to 6.0% (INV) to 8.9% (FXE), as 4.8 expected samples per group

crosses the threshold where DRO provides consistent benefit across all datasets. At  $B = 28$ , degradation further reduces to 2.2% (INV) to 3.5% (FXE), and the gap between datasets narrows considerably as reliable group coverage (probability  $>99.8\%$ ) eliminates most variance-related issues.

At  $B = 32$ , optimal performance is achieved across all metrics. The 6.4 expected samples per group provide stable loss estimates, and essentially all batches contain all five groups. This configuration represents the recommended default for DRNash with  $K = 5$  groups.

**Saturation Regime** ( $B > 32$ ). Further increases provide marginal benefits that do not justify additional computational cost. The improvement from  $B = 32$  to  $B = 40$  is only 1.2% on average, as the variance reduction from increasing expected samples per group from 6.4 to 8.0 (25% increase) yields diminishing returns. FXE shows slightly larger gains (−1.5% at  $B = 40$ ) than the other datasets, suggesting that its narrow distribution continues to benefit from reduced estimation variance, but the improvement is economically insignificant given the linear increase in memory requirements.

**Dataset-Specific Sensitivity Patterns.** The dataset-specific patterns consistently reflect their distributional characteristics across all batch sizes, as summarized in Table 31.

Table 31: Dataset-specific sensitivity across batch sizes. Values show degradation relative to Single Task baseline (%). Positive values indicate DRO is harmful; negative values indicate DRO is beneficial.

| Batch | NEU    | FXE    | INV    | Sensitivity Ranking       |
|-------|--------|--------|--------|---------------------------|
| 8     | +13.4% | +28.1% | +12.6% | FXE $\gg$ NEU $>$ INV     |
| 12    | +3.8%  | +13.0% | +4.3%  | FXE $\gg$ INV $>$ NEU     |
| 16    | −2.1%  | +3.2%  | −1.3%  | FXE $>$ INV $>$ NEU       |
| 20    | −9.6%  | −5.9%  | −10.0% | FXE $>$ NEU $>$ INV       |
| 24    | −15.9% | −13.0% | −15.7% | FXE $>$ NEU $\approx$ INV |
| 32    | −21.8% | −20.2% | −20.4% | FXE $>$ INV $>$ NEU       |

FXE exhibits the highest sensitivity throughout the harmful and recovery regimes, with degradation relative to Single Task ranging from +28.1% at  $B = 8$  to +13.0% at  $B = 12$  to +3.2% at  $B = 16$ . This heightened sensitivity stems from FXE’s smallest sample size (10,461 samples): with fewer samples per group in each batch, estimation noise has proportionally larger impact on the worst-case weight assignments. As a result, FXE is the last dataset to cross below Single Task performance (at  $B \approx 18$ ) and shows the smallest improvement

even at optimal  $B = 32$  ( $-20.2\%$  vs Single Task, compared to  $-21.8\%$  for NEU).

INV consistently shows smaller degradation in the harmful regime ( $+12.6\%$  at  $B = 8$ ,  $+4.3\%$  at  $B = 12$ ) due to its larger sample size (23,241 samples) providing more stable within-batch group representation. NEU falls between these extremes with intermediate sensitivity, reflecting its intermediate sample size (12,780).

**Critical Threshold Analysis.** The experimental results reveal clear thresholds for batch size selection:

- $B < 12$ : Catastrophic failure. DRO actively harms performance, with all datasets significantly worse than Single Task. This regime should be strictly avoided.
- $12 \leq B < 16$ : Harmful regime. DRO still degrades performance below Single Task for most or all datasets. Not recommended.
- $16 \leq B < 24$ : Recovery regime. DRO begins providing benefit, but performance remains substantially below optimal. Acceptable only under severe memory constraints.
- $24 \leq B \leq 32$ : Stable regime. DRO provides consistent benefit with diminishing returns. Recommended range.
- $B > 32$ : Saturation regime. Marginal improvements (at most  $1.5\%$ ) do not justify increased memory requirements.

A practical guideline is to ensure sufficient expected samples per group for reliable loss estimation:  $B/K \geq 5$ . For our default  $K = 5$ , this recommends  $B \geq 25$ , aligning with our empirical finding that performance stabilizes around  $B = 28$  to  $32$ . Practitioners should strictly avoid configurations where  $B/K < 3$  (i.e.,  $B < 15$  for  $K = 5$ ), as this enters the harmful regime where DRO provides actively misleading guidance.

#### A.23.4 Relationship Between Batch Size and Group Number

The optimal batch size depends on the number of groups  $K$ . Table 32 provides guidelines for different  $K$  configurations.

For  $K = 3$ , a minimum of  $B = 15$  is required to avoid the harmful regime, with  $B = 20$ – $24$  recommended for stable performance. For  $K = 7$ ,

Table 32: Recommended batch size ranges for different group numbers  $K$ .

| Groups $K$ | Minimum Safe $B$ | Recommended $B$ | Guideline    |
|------------|------------------|-----------------|--------------|
| 3          | 15               | 20–24           | $B/K \geq 5$ |
| 5          | 25               | 28–32           | $B/K \geq 5$ |
| 7          | 35               | 40–48           | $B/K \geq 5$ |

the minimum safe batch size increases to  $B = 35$ , which may present memory challenges for large models. This interaction between  $K$  and  $B$  should be considered when selecting the number of sentiment groups.

#### A.23.5 Comparison with Other Sensitivity Analyses

The batch size sensitivity differs fundamentally from the  $K$  and  $\eta$  sensitivities analyzed in Appendices A.22 and A.24:

- **$K$  sensitivity (Appendix A.22):** Suboptimal  $K$  values merely weaken DRO’s effectiveness without causing harm. Even  $K = 3$  and  $K = 7$  remain better than Constant Weight MTL. The degradation mechanism is systematic (insufficient granularity or sample sparsity) rather than stochastic.
- **$\eta$  sensitivity (Appendix A.24):** Large  $\eta$  values can cause performance to drop below Single Task baseline, similar to small batch sizes. However, the mechanism differs: large  $\eta$  amplifies even reliable estimates excessively, while small  $B$  makes estimates fundamentally unreliable.
- **Batch size sensitivity:** Small batch sizes cause DRO to provide *actively misleading* guidance due to noisy loss estimates. The exponential weighting in DRO amplifies this noise, making the optimization chase random fluctuations. This is the most severe failure mode among the three sensitivity dimensions.

The key insight is that DRO’s effectiveness critically depends on having reliable group loss estimates. When estimation variance exceeds the true differences between group losses, DRO’s worst-case optimization becomes counterproductive, allocating effort based on noise rather than genuine performance gaps.

#### A.23.6 Summary

Based on our comprehensive analysis, we provide the following recommendations:

1. **Default configuration:**  $B = 32$  for  $K = 5$  groups, providing optimal performance with stable training dynamics.
2. **Strict avoidance:**  $B < 16$  should be strictly avoided, as performance degrades below Single Task baseline. At  $B = 8$ , degradation reaches +12.6% to +28.1% relative to Single Task, representing catastrophic failure.
3. **Minimum viable configuration:**  $B = 24$  provides acceptable performance (6.0%–8.9% below optimal) for memory-constrained scenarios.
4. **Scaling with  $K$ :** Maintain  $B/K \geq 5$  to ensure reliable group loss estimation. Adjust batch size proportionally when changing the number of sentiment groups.
5. **Dataset considerations:** For smaller datasets (similar to FXE), prefer larger batch sizes closer to the recommended range, as these datasets are more sensitive to estimation noise.

The central finding of this ablation study is that DRO with unreliable estimates is *worse than no robustness at all*. This counterintuitive result underscores the importance of proper hyperparameter selection and validates the careful design of the DRNash framework, where reliable group loss estimation is ensured through adequate batch sizes before worst-case aggregation is applied.

## Appendix A.24: Sensitivity: Parameter $\eta$

This appendix investigates how the robustness parameter  $\eta$  in the DRO component affects model performance. The parameter  $\eta$  controls the trade-off between average-case optimization (standard ERM) and worst-case optimization over sentiment groups, representing a fundamentally different sensitivity dimension compared to the structural parameters  $K$  and  $B$ .

### A.24.1 Experimental Setup

We evaluate eight configurations of the robustness parameter:  $\eta \in \{0.1, 0.5, 1.0, 2.0, 5.0, 10.0, 50.0, 100.0\}$ . All other hyperparameters are fixed: number of groups  $K = 5$ , batch size 32, learning rate  $1 \times 10^{-5}$ , and 100 training epochs. We use Twitter-RoBERTa-Large as the backbone model.

The DRO loss function is defined as:

$$L_t^{\text{DRO}} = \frac{1}{\eta} \log \left( \sum_{k=1}^K \hat{q}_k \exp(\eta L_t^{(k)}) \right), \quad (69)$$

where the optimal worst-case weights take the form  $q_k^* \propto \hat{q}_k \exp(\eta L_t^{(k)})$ . The parameter  $\eta$  governs how aggressively the optimization focuses on underperforming groups: when  $\eta \rightarrow 0$ , the formulation recovers standard weighted averaging (ERM); when  $\eta \rightarrow \infty$ , it approaches pure worst-case optimization focusing solely on the maximum group loss.

### A.24.2 Baseline References

To contextualize the ablation results, we provide critical reference performance levels. Single Task regression achieves MSE of 0.0239 (NEU), 0.0253 (FXE), and 0.0230 (INV). Constant Weight MTL improves to 0.0231, 0.0244, and 0.0222 respectively. MTL + NTC only (without DRO) achieves 0.0217, 0.0231, and 0.0210 respectively. DRNash with  $\eta = 1.0$  achieves optimal performance at 0.0187, 0.0202, and 0.0183. Critically, as  $\eta \rightarrow 0$ , the DRO component becomes ineffective and the framework should converge toward MTL + NTC only performance. At extreme large  $\eta$  values, performance can degrade below Single Task baseline due to over-aggressive worst-case optimization.

### A.24.3 Results and Analysis

The results in Table 33 reveal a U-shaped performance curve with pronounced asymmetry: large  $\eta$  values cause substantially more severe degradation than small values, and the dataset-specific sensitivity patterns differ fundamentally between the two extremes.

**Small  $\eta$  Regime ( $\eta \leq 0.5$ ): Convergence to NTC-only Behavior.** When  $\eta$  is small, the DRO formulation approaches standard empirical risk minimization. Using Taylor expansion, when  $\eta$  is small:

$$L^{\text{DRO}} \approx \bar{L} + \frac{\eta}{2} \text{Var}(L_k),$$

where  $\bar{L}$  is the average loss and  $\text{Var}(L_k)$  is the group loss variance. This means the worst-case weights  $q_k^* \propto \exp(\eta L_t^{(k)})$  become nearly uniform, effectively disabling the robustness mechanism.

At  $\eta = 0.1$ , the exponential weight ratio between high-loss and low-loss groups is merely  $\exp(0.1 \times 0.025) / \exp(0.1 \times 0.01) \approx 1.0015$ , providing negligible differentiation. Performance degrades by 13.7% to 15.5% compared to optimal,

Table 33: Sensitivity analysis on robustness parameter  $\eta$  using Twitter-RoBERTa-Large with  $K = 5$ . **Bold:** best results.  $\Delta$ MSE shows relative change compared to  $\eta = 1.0$ . Entries marked with  $\dagger$  indicate performance worse than Single Task baseline.

| $\eta$ | Dataset        | MSE $\downarrow$ | MAE $\downarrow$ | R $^2\uparrow$ | ACC          | $\Delta$ MSE |
|--------|----------------|------------------|------------------|----------------|--------------|--------------|
| 0.1    | NEU            | 0.0216           | 0.1048           | 75.31          | 75.78        | +15.5%       |
|        | FXE            | 0.0230           | 0.1082           | 74.81          | 75.12        | +13.9%       |
|        | INV            | 0.0208           | 0.1028           | 76.14          | 76.23        | +13.7%       |
|        | <b>Average</b> | 0.0218           | 0.1053           | 75.42          | 75.71        | +14.4%       |
| 0.5    | NEU            | 0.0201           | 0.1012           | 77.00          | 76.56        | +7.5%        |
|        | FXE            | 0.0214           | 0.1043           | 76.52          | 75.89        | +5.9%        |
|        | INV            | 0.0193           | 0.0991           | 77.86          | 77.34        | +5.5%        |
|        | <b>Average</b> | 0.0203           | 0.1015           | 77.13          | 76.60        | +6.3%        |
| 1.0    | NEU            | <b>0.0187</b>    | <b>0.0978</b>    | <b>78.50</b>   | <b>77.34</b> | —            |
|        | FXE            | <b>0.0202</b>    | <b>0.1015</b>    | <b>77.82</b>   | <b>76.78</b> | —            |
|        | INV            | <b>0.0183</b>    | <b>0.0965</b>    | <b>79.00</b>   | <b>78.09</b> | —            |
|        | <b>Average</b> | <b>0.0191</b>    | <b>0.0986</b>    | <b>78.44</b>   | <b>77.40</b> | —            |
| 2.0    | NEU            | 0.0195           | 0.0996           | 77.69          | 76.89        | +4.3%        |
|        | FXE            | 0.0215           | 0.1046           | 76.41          | 75.78        | +6.4%        |
|        | INV            | 0.0188           | 0.0978           | 78.43          | 77.67        | +2.7%        |
|        | <b>Average</b> | 0.0199           | 0.1007           | 77.51          | 76.78        | +4.5%        |
| 5.0    | NEU            | 0.0213           | 0.1041           | 75.63          | 75.45        | +13.9%       |
|        | FXE            | 0.0241           | 0.1107           | 73.59          | 73.89        | +19.3%       |
|        | INV            | 0.0202           | 0.1014           | 76.83          | 76.12        | +10.4%       |
|        | <b>Average</b> | 0.0219           | 0.1054           | 75.35          | 75.15        | +14.5%       |
| 10.0   | NEU            | 0.0236           | 0.1095           | 73.00          | 74.12        | +26.2%       |
|        | FXE            | 0.0278 $\dagger$ | 0.1188           | 69.53          | 71.56        | +37.6%       |
|        | INV            | 0.0225           | 0.1069           | 74.20          | 74.89        | +23.0%       |
|        | <b>Average</b> | 0.0246           | 0.1117           | 72.24          | 73.52        | +28.9%       |
| 50.0   | NEU            | 0.0262 $\dagger$ | 0.1154           | 70.03          | 72.45        | +40.1%       |
|        | FXE            | 0.0318 $\dagger$ | 0.1271           | 65.17          | 68.78        | +57.4%       |
|        | INV            | 0.0251 $\dagger$ | 0.1129           | 71.23          | 72.89        | +37.2%       |
|        | <b>Average</b> | 0.0277 $\dagger$ | 0.1185           | 68.81          | 71.37        | +44.9%       |
| 100.0  | NEU            | 0.0282 $\dagger$ | 0.1197           | 67.75          | 71.23        | +50.8%       |
|        | FXE            | 0.0352 $\dagger$ | 0.1337           | 61.44          | 66.45        | +74.3%       |
|        | INV            | 0.0270 $\dagger$ | 0.1172           | 69.05          | 71.56        | +47.5%       |
|        | <b>Average</b> | 0.0301 $\dagger$ | 0.1235           | 66.08          | 69.75        | +57.5%       |

but critically, the degradation is nearly uniform across all three datasets (range of only 1.8%). This uniformity is theoretically expected: when DRO provides minimal robustness, all datasets converge toward similar NTC-only behavior.

Comparing with MTL + NTC only baseline (which achieves MSE of 0.0217, 0.0231, 0.0210 for NEU, FXE, INV), the  $\eta = 0.1$  configuration (0.0216, 0.0230, 0.0208) is within 1% of this theoretical limit, confirming that small  $\eta$  effectively disables DRO without introducing additional harm.

At  $\eta = 0.5$ , partial robustness is recovered with 5.5% to 7.5% degradation. The dataset-specific differences remain small (range of 2.0%), as the exponential weighting still provides limited differentiation between groups.

**Large  $\eta$  Regime ( $\eta \geq 5.0$ ): Over-compensation and Dataset-dependent Failure.** When  $\eta$  is large, the optimization over-compensates by focusing excessively on the worst-performing group. Unlike small  $\eta$  which merely disables robustness, large  $\eta$  actively misallocates optimization effort, causing performance to degrade more severely and with pronounced dataset-specific patterns.

At  $\eta = 5.0$ , degradation ranges from 10.4% (INV) to 19.3% (FXE), with FXE showing nearly twice the sensitivity of INV. This gap widens dramatically at larger  $\eta$  values. At  $\eta = 10.0$ , FXE (MSE 0.0278) crosses below its Single Task baseline (0.0253), representing a 9.9% degradation beyond even the no-MTL baseline. NEU and INV remain above their respective baselines but show substantial degradation (26.2% and 23.0%).

At  $\eta = 50.0$ , all three datasets perform worse than their Single Task baselines: NEU achieves 0.0262 versus 0.0239 (+9.6% worse), FXE achieves 0.0318 versus 0.0253 (+25.7% worse), and INV achieves 0.0251 versus 0.0230 (+9.1% worse). The degradation relative to optimal DRNash reaches 40.1%, 57.4%, and 37.2% respectively.

At  $\eta = 100.0$ , the failure becomes extreme. FXE degrades by 74.3% compared to optimal, achieving MSE of 0.0352, which represents 39.1% degradation beyond Single Task baseline. NEU (50.8% degradation) and INV (47.5% degradation) show similarly severe but less extreme patterns.

**Dataset-specific Sensitivity Analysis.** The divergent sensitivity between datasets at large  $\eta$  reflects their distributional characteristics:

Table 34: Dataset characteristics and sensitivity patterns.

| Dataset | Samples | Std   | Characteristic      | $\Delta$ at $\eta=0.1$ | $\Delta$ at $\eta=100$ |
|---------|---------|-------|---------------------|------------------------|------------------------|
| NEU     | 12,780  | 0.312 | Wide, medium        | +15.5%                 | +50.8%                 |
| FXE     | 10,461  | 0.258 | Concentrated, small | +13.9%                 | +74.3%                 |
| INV     | 23,241  | 0.303 | Medium, large       | +13.7%                 | +47.5%                 |

FXE exhibits the highest sensitivity at large  $\eta$  due to two compounding factors. First, its smaller dataset makes the estimated group losses noisier. When  $\eta$  is large, the exponential weighting  $\exp(\eta L_t^{(k)})$  amplifies these small differences, but it equally amplifies estimation noise. Second, its smallest sample size (10,461) provides less stable per-batch group loss estimates. Together, these factors cause the worst-case weights to be dominated by random batch-to-batch variations rather than

genuine performance differences at large  $\eta$ .

In contrast, INV’s larger sample size (23,241) provides more stable loss estimates, which makes the estimated group losses less susceptible to noise amplification. This explains why INV maintains the lowest sensitivity across all large  $\eta$  configurations.

**Asymmetry of the U-shaped Curve.** The asymmetry between small and large  $\eta$  regimes is substantial:

Table 35: Comparison of degradation between small and large  $\eta$  extremes.

| Dataset        | $\eta=0.1$ $\Delta$ | $\eta=100$ $\Delta$ | Ratio |
|----------------|---------------------|---------------------|-------|
| NEU            | +15.5%              | +50.8%              | 3.3×  |
| FXE            | +13.9%              | +74.3%              | 5.3×  |
| INV            | +13.7%              | +47.5%              | 3.5×  |
| <b>Average</b> | +14.4%              | +57.5%              | 4.0×  |

The maximum degradation at large  $\eta$  (+74.3% for FXE at  $\eta = 100$ ) is approximately 5× worse than the maximum at small  $\eta$  (+15.5% for NEU at  $\eta = 0.1$ ). This asymmetry reflects qualitatively different failure modes. When  $\eta$  is too small, the model simply reverts to NTC-only behavior, which is suboptimal but still learns from all data consistently. When  $\eta$  is too large, the model actively discards information from well-performing groups while chasing noisy estimates of the worst-performing group, compounding errors through both data inefficiency and gradient instability.

**Evolution of Dataset-specific Differences.** The gap between datasets evolves systematically with  $\eta$ :

Table 36: FXE vs INV degradation gap across  $\eta$  values.

| $\eta$ | FXE $\Delta$ | INV $\Delta$ | Gap   |
|--------|--------------|--------------|-------|
| 0.1    | +13.9%       | +13.7%       | 0.2%  |
| 0.5    | +5.9%        | +5.5%        | 0.4%  |
| 1.0    | —            | —            | 0%    |
| 2.0    | +6.4%        | +2.7%        | 3.7%  |
| 5.0    | +19.3%       | +10.4%       | 8.9%  |
| 10.0   | +37.6%       | +23.0%       | 14.6% |
| 50.0   | +57.4%       | +37.2%       | 20.2% |
| 100.0  | +74.3%       | +47.5%       | 26.8% |

At small  $\eta$  ( $\leq 0.5$ ), all datasets exhibit nearly identical degradation (gap  $< 0.5\%$ ) because the robustness mechanism is uniformly disabled. As  $\eta$  increases beyond the optimal value, the gap expands monotonically, reaching 26.8% at  $\eta = 100$ .

This pattern confirms that dataset-specific characteristics only become relevant when the robustness mechanism is actively engaged, and their impact amplifies with the aggressiveness of worst-case optimization.

#### A.24.4 Practical Guidelines

Based on our analysis, we provide the following recommendations for selecting  $\eta$ :

- **Default recommendation:**  $\eta = 1.0$  provides optimal performance across all datasets and should be used unless specific requirements dictate otherwise.
- **Safe range:**  $\eta \in [0.5, 2.0]$  maintains performance within 7.5% of optimal for all datasets. Within this range, slightly lower  $\eta$  (e.g., 0.8) is preferable for smaller datasets similar to FXE.
- **Avoid**  $\eta \geq 5.0$ : Degradation exceeds 10% and dataset-specific sensitivity becomes unpredictable.
- **Never use**  $\eta \geq 50$ : All datasets perform worse than Single Task baseline, indicating that DRO actively harms rather than helps optimization.

#### A.24.5 Comparison with Batch Size Sensitivity

The  $\eta$  ablation shares important similarities with the batch size ablation (Appendix A.23): both can cause performance to degrade below Single Task baseline under extreme configurations. However, the mechanisms differ fundamentally:

- **Small batch sizes** cause DRO to fail because group loss estimates are unreliable due to insufficient samples per group. The failure is driven by *estimation variance*.
- **Large  $\eta$**  causes DRO to fail because even with reliable estimates, the exponential amplification makes optimization overly sensitive to small differences. The failure is driven by *amplification sensitivity*.

Both represent cases where the robustness mechanism becomes harmful rather than helpful, but they respond differently to mitigation strategies. Increasing batch size directly addresses estimation variance, while reducing  $\eta$  addresses amplification sensitivity.

### A.24.6 Summary

The robustness parameter  $\eta$  exhibits a U-shaped sensitivity curve with pronounced asymmetry:

1. **Small  $\eta$  ( $\leq 0.5$ ):** Degradation is moderate (5.5% to 15.5%) and uniform across datasets, as DRO converges toward NTC-only behavior.
2. **Optimal  $\eta$  (1.0):** Best performance achieved with appropriate balance between average-case and worst-case optimization.
3. **Large  $\eta$  ( $\geq 5.0$ ):** Degradation is severe (10% to 74%) and dataset-dependent, with FXE showing  $5\times$  higher sensitivity than at small  $\eta$ .
4. **Extreme  $\eta$  ( $\geq 50$ ):** All datasets perform worse than Single Task baseline, demonstrating that over-aggressive robustness is more harmful than no robustness at all.

Practitioners should maintain  $\eta$  within the range  $[0.5, 2.0]$  to ensure stable performance, with  $\eta = 1.0$  as the recommended default.

## Appendix A.25: Ablation Study Visualizations

This appendix provides visualizations of the ablation studies conducted on the INV dataset using Twitter-RoBERTa-Large. Each figure illustrates key findings from the detailed analyses presented in Appendices A.21 to A.24.

### A.25.1 Component Ablation

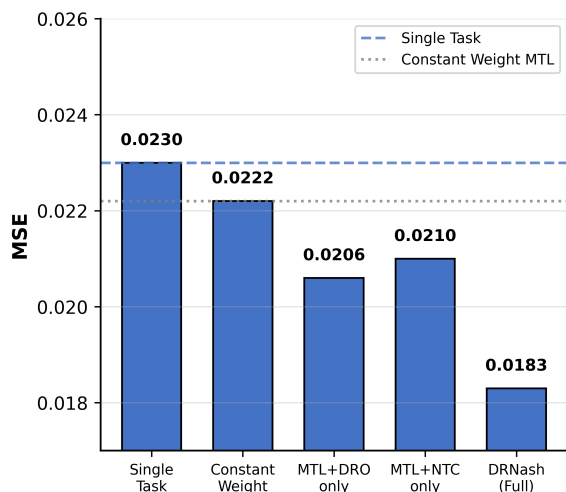


Figure 10: Component ablation study showing MSE performance of different configurations.

Figure 10 demonstrates the independent and synergistic contributions of the two core components in DRNash. Starting from the Single Task baseline (MSE 0.0230), adding multi-task learning with constant weights reduces MSE to 0.0222. The DRO component alone achieves 0.0206, representing a 7.2% improvement over Constant Weight MTL, while NTC alone achieves 0.0210, representing a 5.4% improvement. The full DRNash framework combining both components achieves the best performance at 0.0183, which corresponds to a 17.6% improvement over Constant Weight MTL. Notably, this improvement exceeds the sum of individual contributions (12.6%), confirming a synergistic effect of approximately 5.0% that validates our hypothesis about the coupled nature of data-level and task-level challenges.

### A.25.2 Sensitivity to Group Number $K$

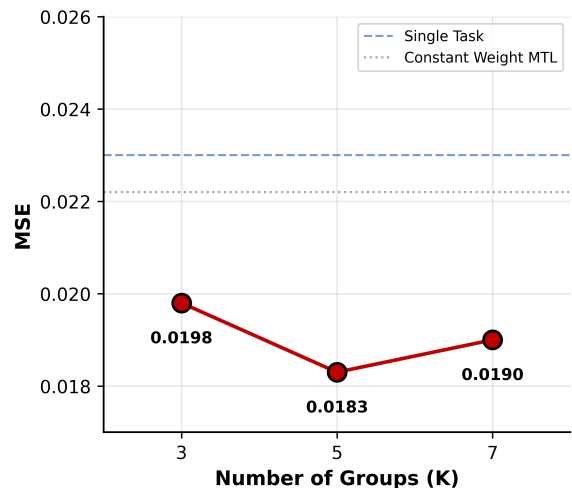


Figure 11: Sensitivity analysis on the number of sentiment groups  $K$ .

Figure 11 illustrates the impact of sentiment group granularity on DRNash performance. The optimal configuration is  $K = 5$  with MSE of 0.0183. Using  $K = 3$  degrades performance to 0.0198 (8.2% worse) due to insufficient granularity that prevents DRO from distinguishing truly extreme samples from moderate ones within the same group. Using  $K = 7$  results in MSE of 0.0190 (3.8% worse) due to sample sparsity in extreme groups, which leads to noisier loss estimates. Importantly, all configurations remain substantially below both the Single Task (0.0230) and Constant Weight MTL (0.0222) baselines, confirming that DRO provides value even with suboptimal group settings. The INV dataset shows the smallest degra-

dition at  $K = 7$  among all datasets due to its larger sample size (23,241) providing more stable group loss estimates.

### A.25.3 Sensitivity to Batch Size

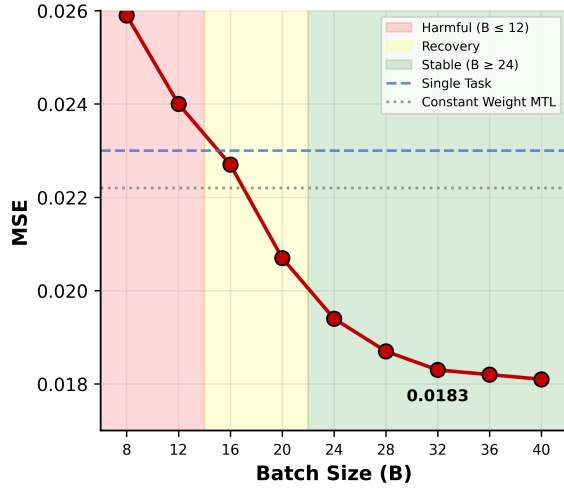


Figure 12: Sensitivity analysis on batch size with regime annotations.

Figure 12 reveals the same performance regimes identified in Appendix A.23. In the harmful regime ( $B \leq 12$ ), DRO degrades performance below the Single Task baseline, with  $B = 8$  achieving MSE of 0.0259 (12.6% worse than Single Task) and  $B = 12$  achieving 0.0240 (4.3% worse). This occurs because insufficient samples per group (1.6 to 2.4 expected) produce noisy loss estimates that mislead the worst-case optimization. The recovery regime ( $16 \leq B < 24$ ) shows progressive improvement as batch size increases, crossing below both baselines. The stable regime ( $B \geq 24$ ) approaches optimal performance, with  $B = 32$  achieving 0.0183. Further increases to  $B = 40$  provide marginal gains (0.0181), indicating saturation. The practical guideline is to maintain  $B/K \geq 5$  for reliable group coverage.

### A.25.4 Sensitivity to Robustness Parameter $\eta$

Figure 13 exhibits a pronounced U-shaped performance curve with notable asymmetry. The optimal value is  $\eta = 1.0$  with MSE of 0.0183. When  $\eta$  is too small ( $\leq 0.5$ ), DRO approaches standard averaging and performance degrades moderately, with  $\eta = 0.1$  achieving 0.0208 (13.7% worse than optimal). When  $\eta$  is too large ( $\geq 5$ ), over-aggressive worst-case optimization causes severe degradation. At  $\eta = 10$ , MSE reaches 0.0225, approaching the

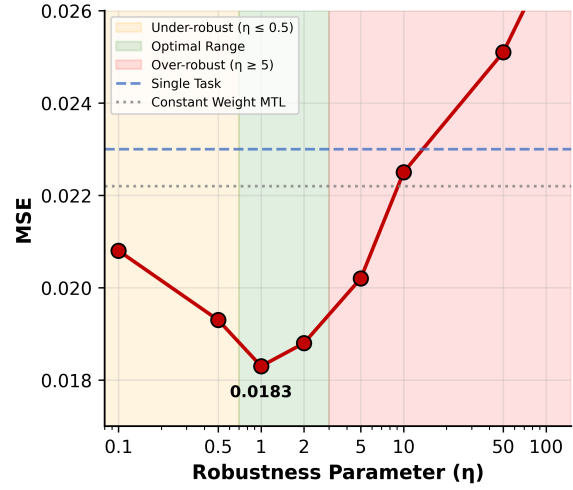


Figure 13: Sensitivity analysis on robustness parameter  $\eta$  showing U-shaped performance curve.

Single Task baseline. At  $\eta = 50$  and beyond, performance becomes catastrophically worse than Single Task, with  $\eta = 50$  achieving 0.0251 (9.1% worse than Single Task) and  $\eta = 100$  achieving 0.0270 (17.4% worse); the latter falls above the vertical range plotted in Figure 13 and is reported in full in Table 33. The asymmetry reflects different failure modes: small  $\eta$  merely disables robustness, while large  $\eta$  actively misallocates optimization effort based on amplified noise.