Ensemble Debiasing Across Class and Sample Levels for Fairer Prompting Accuracy

Ruixi Lin   **Ziqiao Wang**   **Yang You**
Department of Computer Science
National University of Singapore
`ruixi@comp.nus.edu.sg`
`e1374492@u.nus.edu`
`youy@comp.nus.edu.sg`

Abstract

Language models are strong few-shot learners and achieve good overall accuracy in text classification tasks, masking the fact that their results suffer from great class accuracy imbalance. We believe that the pursuit of overall accuracy should not come from enriching the strong classes, but from raising up the weak ones. To address the imbalance, we propose a Heaviside step function based ensemble debiasing method, which enables flexible rectifications of in-context learned class probabilities at both class and sample levels. Evaluations with Llama-2-13B on seven text classification benchmarks show that our approach achieves state-of-the-art overall accuracy gains with balanced class accuracies. More importantly, we perform analyses on the resulted probability correction scheme, showing that sample-level corrections are necessary to elevate weak classes. Due to effectively correcting weak classes, our method also brings significant performance gains to a larger model variant, Llama-2-70B, especially on a biomedical domain task, further demonstrating the necessity of ensemble debiasing at both levels. Our source code is available at `https://github.com/NUS-HPC-AI-Lab/DCS`.

1 Introduction

Language models are good few-shot learners, but prompting can result in inevitable class accuracy imbalance. A language model can perform significantly better on certain classes compared to others, leading to biased performances across different classes (illustrated in Figure 1). This issue is particularly relevant in tasks like classification, or any scenario where the model must distinguish between multiple categories. For example, on DBpedia ontology classification (Auer et al., 2007), the overall accuracy for 14 classes can reach 88% when prompting Llama-2-13B (Touvron et al., 2023), but class *Nature*'s accuracy is 28%. Broadly, for open-ended generation tasks, when treating each token in the vocabulary as a distinct class, generating the most likely next token can be alternatively viewed as classifying over the entire vocabulary or soft labels (Thrampoulidis, 2024; Zhao et al., 2024), also susceptible to class accuracy imbalance. Therefore, accuracy imbalance is prevalent and affecting users of language models, where classification is the most basic case manifesting such imbalances.

In text classifications, the prevalence of class accuracy imbalance is sometimes hidden by a single or a dominant evaluation metric, such as overall accuracy. Especially for relatively large models, the fact that they excel on benchmarks which emphasize overall performances may hide a poor performance in some classes. To enable a focused evaluation of whether language model performances are balanced across classes, the COBias metric was proposed (Lin & You, 2024a) to dedicate evaluating pairwise class accuracy differences.

The class accuracy imbalance issue is not particularly easy to solve from the root. Its causes come from within the model architecture. In details, language models rely on priors learned during training to make predictions. However, built on self-attention, the models are prone to the copying mechanism of the induction heads (Elhage et al., 2021; Olsson et al., 2022).

These heads predict repeated sequences, which can over-capture surface-level patterns in the data, leading to biased priors, and then biased predictions. Therefore, if certain classes or concepts were more frequently emphasized in the training data, the model may disproportionately favor them (Zhao et al., 2021). Although a systemic change at model level or selecting high-quality data may help, it can be inefficient.

Besides what is rooted in the model and data, the way prompts are phrased can further bias the model towards certain classes (Lin & Ng, 2023; Ceballos-Arroyo et al., 2024; Sclar et al., 2024). In few-shot prompting, imbalanced class accuracy can be exacerbated because of under-represented classes in the demonstrations of a prompt. Nevertheless, it only helps to a limited extent by making all classes well-represented in the demonstrations (Lin & You, 2024a). In addition, Chain-of-Thought style outputs may facilitate self-correcting responses without relying on prompt engineering (Wang & Zhou, 2024), but the outcome is guided and restricted by reward

Figure 1: The accuracy imbalance and what we can achieve with debiasing

signals. Lastly, although certain types of biases, such as gender bias or truthfulness, can be traced back to Transformer layers and intervened at inference time (Vig et al., 2020; Geiger et al., 2021; Li et al., 2023), interventions can be inefficient to mitigate the accuracy bias, especially when it could be better treated without altering of the model's internal structure.

Therefore, to effectively and efficiently reduce class accuracy imbalances, post-hoc optimization methods are introduced. As such, post-hoc probability correction complements training and prompting strategies to prevent the model from overly relying on learned patterns, and to incorporate necessary information in downstream tasks. This type of correction involves correcting In-context learned (ICL) output probabilities in inference time by combinatorially optimizing a relatively small mathematical model. These direct rectifications are useful, because they can directly correct outputs, no matter what biases are encoded in the model prior, or what prompts are used.

Related Work. Prior post-hoc correction work has investigated re-weighting ICL output probabilities through combinatorial optimization addressing COBias(Lin & You, 2024a;b) or calibration addressing biased predictions (Zhou et al., 2024; Zhao et al., 2021). The former uses labeled instances for optimization while the latter does not. Both lines of work advance overall classification accuracy with low costs and the flexibility of no language model parameter updates. The performance difference is that the combinatorial optimization works achieve better balanced class performances while improving overall accuracy. We follow the combinatorial optimization perspective for our ensemble debiasing.

For a background on combinatorial optimization, it provides a mathematical framework for maximizing or minimizing an objective function of decision variables, subject to constraints on some or all of the variables, involving inequalities, equalities, and integrality restrictions (Nemhauser & Wolsey, 1988). This framework essentially transforms a rich variety of tasks into discrete optimization models, which is robust and remarkably versatile. It solves real-world problems such as planning, scheduling, etc. in operations research, and has diverse applications in NLP, from structured prediction such as named entity recognition and dependency parsing, to reading comprehension, then to grammatical error correction, and to align LLM outputs with human preferences (Martins et al., 2009; Wang et al., 2010; Rush et al., 2010; Koo et al., 2010; Goldwasser et al., 2012; Wu & Ng, 2013; Berant et al., 2014; Roth & Srikumar, 2017; Lin & Ng, 2021; Dakle et al., 2023; Zhang et al., 2024; Srikumar & Roth, 2023; I. Garmendia et al., 2024). The core use of the framework is to translate a problem into a mathematical statement of one of the forms of linear mixed-integer programming problem (MIP), integer programming problem (IP), nonlinear integer programming problem (NIP), or other types (Roth & Yih, 2004; 2005; Hemmecke et al., 2010; Srikumar & Roth, 2023). Solving combinatorial optimization models is not an easy task. Among which, NIP is highly challenging, because the NIP model can contain discontinuous functions and is
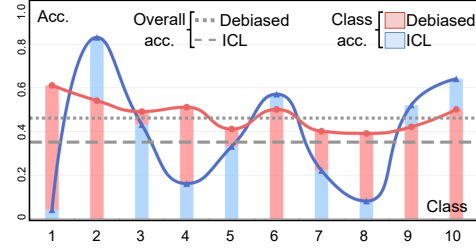
inherently non-differentiable. Lin & You (2024a;b) proposed to address the search difficulty with metaheuristics, which we follow (more discussions in Appendix A).

Back to the combinatorial optimization based debiasing methods, their drawbacks lie in the single consideration at either class level (Lin & You, 2024a) or sample level (Lin & You, 2024b). In (Lin & You, 2024a), any sample gets re-weighted by the same set of per-class correction weights, making it hard to capture sample-level variances, which is particularly needed for weak classes. In (Lin & You, 2024b), corrections are made at sample level by assigning per-sample per-class correction weights that are computed by membership functions, however, capturing too much nuance may over-corrects strong classes.

To overcome the insufficiency in previous works, we propose a meticulous post-hoc correction framework **DCS** (**D**ebiasing at **C**lass and **S**ample levels). DCS is an organic integration of broad and fine-grained corrections across class and sample levels, pushing forward fairer prompting accuracy for LLMs. DCS automatically detects which classes need corrections that account for sample-level differences and which do not. In the following, we first introduce the Heaviside step function based ensembling strategy for DCS (Section 2) and the overall framework (Section 3), followed by quantitative evaluations and in-depth analyses (Section 4). In Section 5, we conclude with prospective applications and open problems.

Our contributions are summarized as below:

- *Post-hoc debiasing unifies class and sample level corrections.* We ensemble weight correction and membership correction to enable rectifications of class probabilities at both class and sample levels, to enhance the performance of LLMs directly from their outputs.
- *SoTA scores on overall accuracy while maintaining low COBias* DCS can achieve leading overall accuracy improvements while good COBias reduction across seven benchmark text classification tasks compared to the previous SoTA.
- *Optimal solutions need weight correction to elevate weak classes.* Quantitative analyses show that the average ratio of classes using membership correction to weight correction is between 0 and 1, where most classes apply weight corrections but lowest-accuracy (weakest) classes more often apply membership corrections, suggesting the necessity of keeping membership correction as an option and letting the optimization process determine if it is needed for any particular class.

2 Using Heaviside Step Function to Ensemble Class and Sample Level Debiasing

We introduce a robust ensemble debiasing framework across class and sample levels, which is targeted specifically for balancing class accuracy while improving overall accuracy over individual methods. To achieve ensembling, a Heaviside step function seamlessly unifies two unique ICL output correction methods, bringing together the advantages of weight coefficient and fuzzy rule based corrections. This unified framework captures the intricacies of ICL output class probabilities with greater precision, reaching solutions that elevate weak classes while maintaining the strong ones. List of notations are presented in Appendix B.

2.1 The Mapping Function for ICL Probability Correction

As mentioned, a common method for correcting probabilities involves multiplying each class probability by a coefficient (Lin & You, 2024a). This is a class-level correction. For an N-class task, we use a set of weights to adjust each dimension of the ICL output N-dimensional probabilities without considering the variations in the initial probabilities across different instances. In contrast, a more refined correction method employs membership functions from fuzzy logic (Lin & You, 2024b), where a mapping function is applied to adjust each dimension. Different classes use different mapping functions. As a result, class probabilities of different instances undergo varying degrees of transformation - some are adjusted significantly, others minimally, and some remain unchanged, reflecting a finer, sample-level correction approach.

We propose enabling a model-driven selection between coarse (class-level) and finer (sample-level) corrections in the optimization process, promoting efficient and precise post-hoc corrections for any class. Let's denote the probability of the m-th instance's i-th class as p_{mi}. We correct the per-sample per-class probability by a mapping function $f_i : p'_{mi} \leftarrow f_i(p_{mi}), p_{mi} \in [0,1]$. To achieve our goal, the set of mapping functions Ψ combine weight correction with membership function correction, i.e., $\Psi = \mathbb{R}^{\text{Weight}} \cup \mathbb{R}^{\text{Membership}}$, where $\mathbb{R}^{\text{Weight}} = \{\omega_1, \ldots, \omega_{D_W}\}$ and $\mathbb{R}^{\text{Membership}} = \{\mu_1, \ldots, \mu_{D_F}\}$. To dynamically choose the correction, we first introduce the Heaviside step function:

$$H(x) = \begin{cases} 1, & x \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

Let ξ_i denote the selection variable for class i, each ξ_i will be optimized to choose a most suitable correction from Ψ. We conventionally note when ξ_i is optimized to be one of the indices from $\{1, ..., D_F\}$, the i-th class probability of each instance gets a personalized adjustment by a membership function determined by ξ_i; when ξ_i is one of the indices from $\{D_F + 1, ..., D_F + D_W\}$, the i-th class probability of any input instance is corrected by a weight coefficient determined by ξ_i. Consequently, when $\xi_i = k$ and $k \leq D_F$, the triangular membership function (following (Lin & You, 2024b)) based correction function μ_k is selected, the updated probability is computed by:

$$\mu_k(p_{mi}) = \begin{cases} 0, & p_{mi} \leq a_k \\ \dfrac{p_{mi} - a_k}{b_k - a_k}, & a_k < p_{mi} \leq b_k \\ \dfrac{c_k - p_{mi}}{c_k - b_k}, & b_k < p_{mi} \leq c_k \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

Special cases: when $a_k = b_k = 0, \mu_k(p_{mi}) = \frac{c_k - p_{mi}}{c_k}$ for $0 \leq p_{mi} \leq c_k$ and 0 for $p_{mi} \geq c_k$; when $b_k = c_k = 1, \mu_k(p_{mi}) = \frac{p_{mi} - a_k}{1 - a_k}$ for $a_k \leq p_{mi} \leq 1$ and 0 for $p_{mi} \leq a_k$.

Otherwise, the weight correction function ω_k is selected, and the updated probability is obtained by:

$$\omega_k(p_{mi}) = \frac{k - D_F}{D_W} p_{mi} \tag{3}$$

Therefore, the per-sample per-class probability p_{mi} is precisely updated as follows.

$$p'_{mi} = f_i(p_{mi}) = \mu_{\xi_i}(p_{mi}) \cdot H(D_F - \xi_i) + \omega_{\xi_i}(p_{mi}) \cdot H(\xi_i - D_F - 1) \tag{4}$$

This results in a more accurate correction for the ICL output probability, due to applying flexible fixes at either the broad level or the fine-grained level.

2.2 Mathematical Model

The mathematical model used to drive the selection is based on the nonlinear integer programming model in (Lin & You, 2024a) with accuracy-maximizing and COBias-minimizing objectives. The combination of weight and membership corrections that optimizes the model's objective function will be selected. In short, the model is:

$$\min Z = Z^{\text{Err}}(\xi) + \beta Z^{\text{COBias}}(\xi) + \tau Z^{\text{PMI}}(\xi)$$
$$\text{s.t. } \boldsymbol{\xi} = (\xi_1, \ldots, \xi_N), \xi_i \in \{1, \ldots, D_F + D_W\} \tag{5}$$

where $Z^{\text{Err}}, Z^{\text{COBias}}, Z^{\text{PMI}}$ respectively corresponds to the overall error rate, COBias of all class pairs, and point-wise mutual information (PMI) between predicted and ground-truth classes (Appendix C). During optimization, at the beginning of each simulated annealing iteration, candidate selection ξ_i's are generated and the correction functions f_i's correct each sample in the optimization set, yielding corrected predictions $\{\hat{y}_m\}_{m=1}^M$. Each of the individual term in Z is essentially an evaluation score computed using the corrected
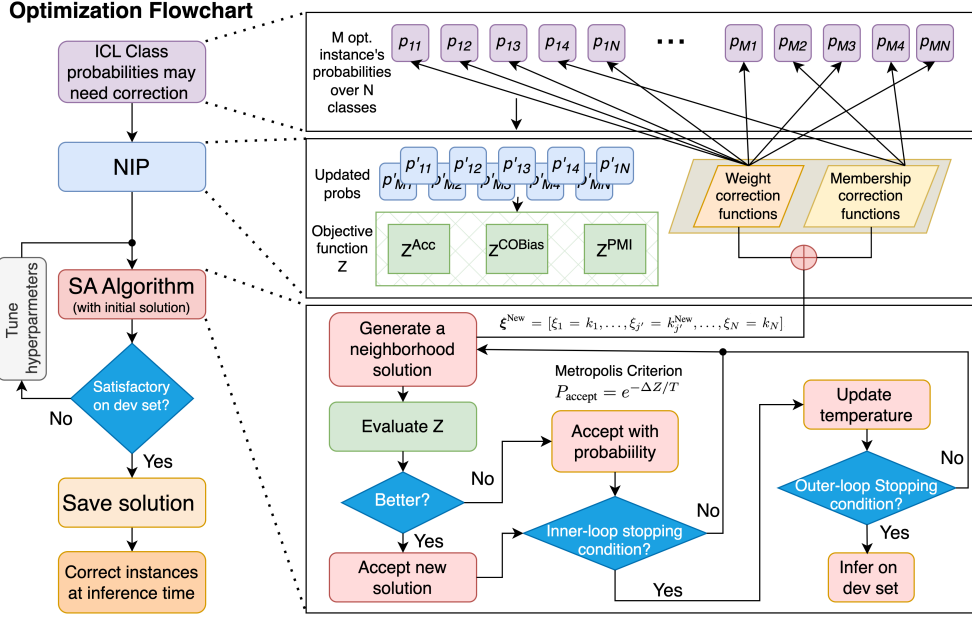
Figure 2: Overall framework of DCS

predictions and ground-truth labels $\{y_m\}_{m=1}^{M}$. If the new Z is smaller than the current Z, the candidate selection is accepted and the best selection is updated; otherwise, the algorithm decides whether to accept the new selection based on the acceptance probability. Iterations repeat until a stopping criterion is met. The optimization process will be elaborated in Section 3.

Model parameters. This is a budget-friendly model with a total number of parameters (here we refer to search space size) of $N(D_F + D_W)$.

3 Simulated Annealing for Solving DCS

Due to the non-differentiable nature of the math model, we turn to heuristics and employ the simulated annealing (SA) algorithm to solve the DCS model. The SA algorithm was proposed by (Kirkpatrick et al., 1983; Cerny, 1985). The forerunners successfully introduced the concept of annealing into combinatorial optimization, inspired by the annealing process of solid materials. It starts at an initial temperature and cools down at a specified rate. As the temperature gradually decreases, the algorithm leverages probabilistic jumps to explore the solution space randomly, searching for the global optimum of the objective function, so it can probabilistically escape local optima and eventually converge to the global optimum (Steinbrunn et al., 1997). The optimization of DCS model is a search problem in multidimensional space, where each solution makes a potential ***probability correction scheme***. Next, we uncover the overall framework that apply SA to solve our model.

3.1 Overall Framework

The overall solution framework for DCS is depicted in Fig. 2. The combinatorial optimization relies on a labeled set of these initial class probabilities and their ground-truth labels. The labeled set can be simply taken from the full or a subset of the classification task's training set. After ICL output class probabilities are obtained, the optimization process starts with an initial solution. The iteration repeats until either a minimum temperature is reached or a maximum number of iterations is reached, and we obtain the optimal correction functions. In actual inference, the selected functions directly rectify a test instance's ICL output class probabilities and result in corrected predictions.

3.2 Initial Solution

We can keep the initial probabilities unchanged as the initial solution for the DCS math model. The fuzzy rule set $\mathbb{R}^{\text{Membership}} = \{\mu_k\}_{k=1}^{D_F}$ designed in this paper must include *Don't Change*, the triangular membership function with $a_k = 0, b_k = c_k = 1$, which will keep the original probability unchanged. If the corresponding selection index is k_0, the initial solution can be chosen as: $\xi_j = k_0$, for class $j = 1, \ldots, N$. For SA, the initial solution does not affect the quality of the final solution, but it influences the annealing process, and we empirically start with unchanged probabilities (Kirkpatrick et al., 1983).

3.3 Neighborhood Solution

It is necessary to generate a new solution in the neighborhood of the current solution. If the current solution is $\xi_j = k_j$, for class $j = 1, \ldots, N$, we can randomly select a class, for example, j' whose current solution is $\xi_{j'} = k_{j'}$, and perturb it by sampling $k_{j'}^{\text{New}}$ from the set $\mathbb{D} \setminus k_{j'}$ as the new value for the variable $\xi_{j'}$. That is, $\xi_{j'} = k_{j'}^{\text{New}}$. In this way, we obtain a neighboring solution: $\boldsymbol{\xi}^{\text{New}} = (\xi_1 = k_1, \ldots, \xi_{j'} = k_{j'}^{\text{New}}, \ldots, \xi_N = k_N)$. The transition from the current solution at a given temperature to the new solution is a Markov chain (Kirkpatrick et al., 1983; Cerny, 1985), and the transition probability is determined by the Metropolis criterion $P_{\text{accept}} = e^{-\Delta Z/T}$. If the new solution is better ($\Delta Z < 0$), accept the new solution; if the new solution is worse ($\Delta Z \geq 0$) and a uniform random number $r \sim U(0,1)$ is smaller than P_{accept}, namely $r < P_{\text{accept}}$, the new solution is also accepted.

3.4 Annealing Schedule

An annealing schedule is characterized by the initial value of the temperature, the cooling rate, the inner-loop criterion, and the outer-loop criterion.

Initial Temperature: The initial temperature is chosen so that the acceptance probability is relatively close to one. The higher the initial temperature, the more possible to obtain a high-quality solution, but the longer the computation time. In this work, the initial temperature is set to 200,000.

Inner-Loop Criterion: The inner loop simulates the process of the system reaching equilibrium at a given temperature. Theoretically, there should be a sufficient number of iterations. The more decision variables there are, the more iterations the inner loop should have. Therefore, in this paper, the inner loop stops when either the number of accepted solutions reaches λ_1 times of the number of variables, or the number of solutions generated exceeds λ_2 times of the number of variables. The value of λ_1, λ_2 are tuned alongside other hyperparameters.

Outer-Loop Criterion and Cooling Rate: Generally, the termination temperature should be set to a sufficiently small positive number. The temperature update function follows a geometric rate: $T_{\text{curr}+1} = \alpha \cdot T_{\text{curr}}$, where α is set as 0.95 in this paper.

4 Experiments

We assess the capabilities of DCS in reducing class accuracy differences and enhancing overall accuracy with three ICL settings: 1-shot, 5-shot, and N-shot. Demonstrations in the prompt are randomly selected from the optimization set for 1-shot and 5-shot cases, while the N-shot demonstration cascades examples from each class, to minimize under-representations of any class. DCS is compared against SoTA methods, DNIP (Lin & You, 2024a) and FuRud (Lin & You, 2024b). We prompt LLMs with an A100 GPU to obtain initial ICL outputs, following DNIP's hyperparameters, then we run simulated annealing on CPUs. Evaluations are performed on seven multi-class text classification tasks across general and biomedical domains, including AGNews (AGN, 4-class;(Zhang et al., 2015)), DBpedia (DBP, 14-class;(Auer et al., 2007)), SST-5 (SST, 5-class;(Socher et al., 2013)), TREC (6-class;(Voorhees & Tice, 2000; Li & Roth, 2002)), RTE (binary;(Dagan et al., 2006)), DDI

(5-class;(Segura-Bedmar et al., 2013)), and PubMedQA (PMQA, 3-class;(Jin et al., 2019)). Appendix D presents more details.

4.1 Main Results

Setup We perform quantitative evaluations on Llama-2-13B, for its representativeness as a widely applied open-source LLM. Indeed, Llama-2-13B is not particularly big in parameter size, but its architecture can be seen as basic building blocks for many advanced LLMs (Chiang et al., 2023; Rozière et al., 2024; Liu et al., 2023), and we expect similar findings in larger model variants or models similar in architecture, which will be discussed in Section 4.3. Table 1 reports our main findings.

Method	Acc										COBias									
	All	Avg (General Domain)	AGN	DBP	SST	TREC	RTE	Avg (Biomed.)	DDI	PMQA	All	Avg (General Domain)	AGN	DBP	SST	TREC	RTE	Avg (Biomed.)	DDI	PMQA
1-shot ICL + post-hoc debiasing																				
ICL	59.4	70.7	$79.9_{7.0}$	$88.2_{1.0}$	$44.9_{4.3}$	$68.5_{10.8}$	$71.5_{2.2}$	31.2	$7.2_{0.9}$	$55.1_{2.9}$	40.5	35.4	$28.3_{16.1}$	$16.2_{3.7}$	$53.1_{5.0}$	$35.9_{6.5}$	$43.4_{7.0}$	53.4	$45.6_{5.9}$	$61.2_{1.9}$
+DNIP	69.2	76.2	$87.9_{0.7}$	$93.4_{0.6}$	$48.3_{1.9}$	$77.1_{2.0}$	$74.3_{0.8}$	51.8	$40.4_{6.0}$	$63.1_{14.0}$	14.3	10.2	$6.3_{0.6}$	$7.7_{0.6}$	$18.7_{10.1}$	$14.2_{1.3}$	$4.3_{3.3}$	24.3	$7.5_{3.2}$	$41.1_{29.6}$
+FuRud	67.2	76.5	$86.8_{2.0}$	$92.7_{0.2}$	$50.0_{3.0}$	$78.4_{2.4}$	$74.5_{1.8}$	44.0	$29.4_{17.0}$	$58.6_{4.1}$	15.1	13.3	$10.9_{4.7}$	$8.8_{0.3}$	$26.2_{11.6}$	$13.5_{1.4}$	$7.1_{1.7}$	19.5	$11.8_{6.7}$	$27.2_{8.1}$
+DCS (Ours)	72.2	77.8	$88.2_{1.0}$	$94.7_{0.7}$	$50.2_{1.4}$	$81.5_{2.5}$	$74.4_{0.4}$	58.1	$52.9_{18.3}$	$63.3_{12.2}$	15.0	10.7	$6.2_{1.5}$	$6.0_{0.8}$	$23.7_{11.6}$	$12.8_{3.3}$	$5.0_{3.6}$	25.8	$16.4_{13.5}$	$35.2_{22.3}$
5-shot ICL + post-hoc debiasing																				
ICL	63.9	68.5	$82.5_{2.0}$	$93.6_{1.3}$	$45.8_{6.0}$	$58.7_{23.3}$	$61.9_{16.9}$	52.3	$34.1_{42.1}$	$70.4_{7.1}$	39.2	35.6	$24.4_{4.2}$	$9.0_{2.0}$	$48.0_{15.4}$	$35.4_{13.7}$	$61.3_{40.4}$	48.3	$44.4_{5.4}$	$52.2_{19.3}$
+DNIP	71.9	77.9	$88.5_{0.6}$	$95.8_{0.7}$	$52.9_{2.5}$	$76.6_{8.7}$	$75.8_{4.5}$	57.0	$54.1_{5.1}$	$59.9_{1.4}$	10.3	9.2	$7.0_{0.9}$	$5.7_{1.1}$	$15.8_{12.6}$	$14.4_{7.5}$	$3.1_{1.9}$	13.1	$16.5_{7.6}$	$9.6_{4.6}$
+FuRud	70.6	77.9	$87.8_{0.9}$	$95.5_{0.7}$	$53.3_{4.9}$	$76.9_{3.9}$	$75.8_{4.2}$	52.3	$40.3_{7.3}$	$64.3_{5.6}$	14.7	11.8	$10.6_{0.7}$	$5.7_{0.9}$	$22.1_{6.0}$	$19.6_{10.7}$	$1.0_{0.3}$	22.1	$19.1_{3.9}$	$25.1_{19.5}$
+DCS (Ours)	74.8	79.0	$88.8_{0.8}$	$96.3_{0.6}$	$54.2_{2.9}$	$79.7_{4.9}$	$76.1_{4.3}$	64.3	$57.1_{4.2}$	$71.4_{8.5}$	14.7	7.7	$6.7_{0.7}$	$4.7_{1.1}$	$11.7_{8.3}$	$14.0_{4.1}$	$1.6_{0.6}$	32.2	$21.0_{4.4}$	$43.4_{23.4}$
N-shot ICL + post-hoc debiasing																				
ICL	61.9	74.2	$83.5_{1.5}$	$95.2_{1.2}$	$50.3_{2.3}$	$67.0_{12.7}$	$75.0_{0.8}$	31.0	$9.7_{1.0}$	$52.3_{5.3}$	25.6	23.8	$14.9_{5.1}$	$7.0_{2.2}$	$36.3_{7.2}$	$38.2_{5.1}$	$22.5_{13.2}$	30.3	$39.7_{3.5}$	$20.9_{4.2}$
+DNIP	70.7	79.2	$88.7_{0.6}$	$96.0_{0.5}$	$51.3_{1.0}$	$82.7_{1.4}$	$76.7_{3.7}$	49.5	$43.6_{6.1}$	$55.3_{2.6}$	7.5	6.3	$7.3_{0.5}$	$4.3_{0.7}$	$2.8_{1.5}$	$12.1_{5.5}$	$5.0_{3.3}$	10.6	$12.5_{4.3}$	$8.7_{1.2}$
+FuRud	73.6	78.7	$87.9_{0.8}$	$96.5_{0.5}$	$53.7_{2.5}$	$78.3_{5.0}$	$77.1_{3.2}$	60.7	$61.7_{6.8}$	$59.7_{7.2}$	16.2	13.1	$8.5_{3.2}$	$4.9_{0.7}$	$29.0_{11.6}$	$15.8_{3.1}$	$7.3_{4.9}$	23.9	$28.9_{3.8}$	$18.8_{14.9}$
+DCS (Ours)	74.7	80.0	$88.9_{0.5}$	$96.8_{0.4}$	$54.2_{0.6}$	$83.4_{3.1}$	$76.7_{3.7}$	61.4	$62.4_{16.5}$	$60.3_{5.3}$	13.0	8.8	$7.1_{1.1}$	$4.1_{0.6}$	$16.9_{5.3}$	$11.1_{5.3}$	$5.0_{3.3}$	23.3	$28.6_{9.9}$	$17.9_{6.6}$

Table 1: Main results

The proposed method achieves **SoTA overall accuracy improvements** on all tasks except for RTE across all ICL settings. As for RTE, due to its simplicity as a binary classification task, all three methods obtain similar accuracy and COBias improvements. Meanwhile, it achieves **SoTA COBias performance on general domain tasks**. In general, its COBias is on par with the current best methods, with average COBias over all seven tasks falling between that of DNIP and FuRud. On AGNews, DBpeida, and TREC, the proposed method achieves **both more overall accuracy improvements and more COBias reduction** than DNIP and FuRud across all ICL settings, demonstrating the superior performance gains achieved by ensembling class-level and sample-level debiasing. **Weak classes are elevated.** We show in Section 4.2 that DCS powers up the low-accuracy classes. We observe that weaker classes are often corrected by rules, while stronger classes are most likely corrected by weights, further highlighting the need of an ensemble method.

Annealing Time

Figure 3 plots the time used to optimize correction functions (left axis) and the search space size (right axis) for each task. For fair comparisons, we fix the weight scale for weight correction to 30 alongside 19 triangular membership functions, making a total of 49 possible correction functions and $49 \cdot N$ possible assignments across all variables. DCS is computationally efficient where all tasks finish optimization within 150 outer loops, ranging from a maximum of 96.6 seconds to a minimum of 5.4 seconds. Notably, SST-5, DDI, and AGNews have a relatively longer time due to having a larger optimization set (Appendix D). In general, although larger optimization sets and search spaces need more time to converge,

Figure 3: Task annealing time

the annealing steps needed for a task can be further tuned to achieve cost-effective gains on downstream applications.

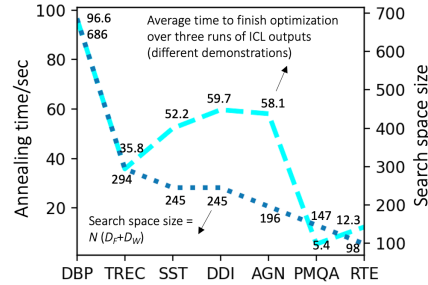## 4.2 Overall Probability Correction Scheme

The overall probability correction scheme demonstrates the ratio of class-level corrections to sample-level corrections among all classes. Which type of correction should be used for a class is automatically determined by our method. Recall that: *Class-level correction*: Probabilities of the same output class are corrected uniformly using a same correction weight across all samples. It operates at the class level, not accounting for sample-based differences. *Sample-level correction*: This is a more refined correction using membership correction functions. Instead of a single weight, a unique mapping function is applied for each class. This allows varying degrees of correction across different samples - some probabilities may be adjusted significantly, others minimally, and some may remain unchanged - capturing the nuances of individual samples, leading to more precise adjustments. By combining them, our method ensures a balanced and adaptive correction, enhancing performances.

Dataset	# Classes	# Classes that use weight corrections	# Classes that use membership corrections	Ratio of memberhsip corrections to weight corrections	#Classes that use membership corrections in seed0/1/2 run	Remarks on any weakest class that use membership corrections
AGN	4	3.7	0.3	0.08	0/1/0	The weakest class *Sports* from seed1 run, Acc: 0.75 (ICL) → 0.98 (corrected)
DBP	14	12.3	1.7	0.14	2/2/1	-
SST	5	4.3	0.7	0.16	1/0/1	The weakest class *Terrible* from seed0 run, Acc: 0.21 (ICL) → 0.54 (corrected) The weakest class *Terrible* from seed2 run, Acc: 0.09 (ICL) → 0.64 (corrected)
TREC	6	5	1	0.20	0/2/1	The weakest class *Entity* from seed2 run, Acc: 0.04 (ICL) → 0.76 (corrected)
RTE	2	0.7	1.3	1.86	1/2/1	The weakest class *False* from seed1 run, Acc: 0.54 (ICL) → 0.78 (corrected)
DDI	5	2.7	2.3	0.85	3/2/2	The weakest class *Negative* from seed0 run, Acc: 0 (ICL) → 0.73 (corrected) The weakest class *Negative* from seed1 run, Acc: 0 (ICL) → 0.46 (corrected) The weakest class *Negative* from seed2 run, Acc: 0.02 (ICL) → 0.93 (corrected)
PMQA	3	2.7	0.3	0.11	0/1/0	The weakest class *Yes* from seed1 run, Acc: 0.38 (ICL) → 0.62 (corrected)

Table 2: Overall probability correction scheme (*N*-shot)

We exemplify the overall probability correction scheme with the *N*-shot case in Table 2. Numbers of classes are averaged over three runs of different demonstrations. As expected, most classes select class-level weight corrections. Membership corrections are less selected, but weakest classes benefit greatly from them, where 6 out of 7 tasks have selected membership corrections to correct their weakest classes (at least once over three runs). This demonstrates the necessity of enabling sample-level corrections to boost the performance of weak classes, and letting the optimization decide if sample-level corrections are needed.

4.3 Larger Models on DDI (Biomedical-domain)

Similar to its smaller variant Llama-2-13B, 1-shot ICL on Llama-2-70B also manifests class accuracy imbalances. Especially on the biomedical domain DDI classification, using 1-shot prompting, Llama-2-13B's average ICL accuracy on DDI is 7.2% with CO-Bias 45.6% as shown in Table 1, while Llama-2-70B obtains 8.8% accuracy with COBias 36.0% as shown in Table 3, showing that the larger variant is still highly prone to biased predictions on biomedical tasks. This suggests that the class accuracy imbalance may not be effectively alleviated by increasing model sizes, and the proposed post-hoc corrections can exactly help with the situation. Due to effectively correcting probabilities for

Model	Acc.	CB	Prompt Demon.	Class Acc.				
				Neg.	Eff.	Mech.	Adv.	Int.
Llama-2-13B								
ICL	7.2	45.6	seed0	0.0	0.0	100.0	0.0	2.3
			seed1	0.0	91.4	0.3	17.7	39.6
			seed2	0.0	0.8	84.1	54.8	88.5
+DCS	52.9 (+45.7)	16.4 (-29.2)	seed0	32.5	38.1	32.1	29.0	32.3
			seed1	62.1	30.6	32.8	42.1	31.3
			seed2	80.7	12.8	8.6	6.3	11.5
Llama-2-70B								
ICL	8.8	36.0	seed0	0.1	86.9	3.0	4.1	19.8
			seed1	8.7	64.2	3.0	18.1	56.3
			seed2	0.5	64.4	6.6	37.6	58.3
+DCS	63.7 (+54.9)	24.3 (-11.7)	seed0	69.9	34.4	19.9	18.6	47.9
			seed1	63.6	23.6	39.4	29.0	35.4
			seed2	80.5	16.1	26.2	18.6	33.3

Table 3: Significant DDI Performance gains on both Llama-2-13B and Llama-2-70B.

low-accuracy classes as shown by Table 3, our method demonstrates significant accuracy enhancements on DDI, across both small and large model variants. For **more Llama-2-70B performance gains**, detailed scores can be found in Appendix E.

4.4 Ablations

Some may contend that more overall accuracy enhancements can be compensated by less COBias reduction. While tradeoffs between overall accuracy and COBias generally exist, relaxing COBias reduction (resulting in larger COBias score) can not guarantee higher overall accuracy (Table 4), showing the need of a dedicated Z^{COBias} in the objective function.

Prompt	Obj.	AGN		DBP		SST		TREC		RTE		DDI		PMQA	
		Acc.	CB.	Acc.	CB.	Acc.	CB.	Acc.	CB.	Acc.	CB.	Acc.	CB.	Acc.	CB.
1-shot	Full	88.2	6.2	94.7$_{(\uparrow6.1)}$	6.0$_{(\downarrow10.2)}$	50.2	23.7	81.5$_{(\uparrow12.9)}$	12.8$_{(\downarrow23.1)}$	74.4	5.0	52.9	16.4	63.3	35.2
	Z^{Err}	88.4	7.8	92.8$_{(\uparrow4.2)}$	8.6$_{(\downarrow7.6)}$	51.4	50.9	80.6$_{(\uparrow12.0)}$	16.7$_{(\downarrow19.2)}$	75.7	12.8	82.2	39.8	71.3	58.8
	$Z^{Err}+Z^{PMI}$	88.5	7.5	94.1$_{(\uparrow5.5)}$	7.5$_{(\downarrow8.7)}$	51.7	53.0	81.1$_{(\uparrow12.5)}$	15.1$_{(\downarrow20.8)}$	75.8	13.0	82.9	40.0	71.1	59.1
5-shot	Full	88.8	6.7	96.3$_{(\uparrow2.7)}$	4.7$_{(\downarrow4.3)}$	54.2	11.7	79.7$_{(\uparrow21.0)}$	14.0$_{(\downarrow21.4)}$	76.1	1.6	57.1	21.0	71.4	43.4
	Z^{Err}	88.9	7.8	95.0$_{(\uparrow1.4)}$	7.1$_{(\downarrow1.9)}$	54.2	41.6	79.5$_{(\uparrow20.8)}$	14.6$_{(\downarrow20.8)}$	76.2	13.7	82.8	40.0	76.7	58.8
	$Z^{Err}+Z^{PMI}$	88.9	7.9	95.9$_{(\uparrow2.3)}$	5.5$_{(\downarrow3.5)}$	54.9	41.4	78.9$_{(\uparrow20.2)}$	21.0$_{(\downarrow14.4)}$	76.9	10.2	82.9	40.2	76.8	58.8
N-shot	Full	88.9	7.1	96.8$_{(\uparrow1.7)}$	4.1$_{(\downarrow2.8)}$	54.2	16.9	83.4$_{(\uparrow16.4)}$	11.1$_{(\downarrow27.1)}$	76.7	5.0	62.4	28.6	60.3	17.9
	Z^{Err}	88.8	7.5	96.3$_{(\uparrow1.2)}$	5.1$_{(\downarrow1.8)}$	54.4	37.4	81.9$_{(\uparrow14.9)}$	18.4$_{(\downarrow19.8)}$	76.9	17.0	82.8	40.1	76.0	60.5
	$Z^{Err}+Z^{PMI}$	89.0	7.5	96.5$_{(\uparrow1.4)}$	4.8$_{(\downarrow2.1)}$	54.8	40.8	83.3$_{(\uparrow16.3)}$	15.3$_{(\downarrow22.9)}$	77.5	15.7	82.9	40.1	76.1	60.7

Table 4: Z^{Err} objective ablations; average scores over three random seeds are reported. Improvements over ICL scores are shown by ($\uparrow$) and ($\downarrow$).

To see this, we ablate the overall error term Z^{Err} from the objective function and optimize solely for Z^{Err}. On DBpedia and TREC, DCS with only Z^{Err} did not obtain a higher accuracy than the DCS with the full objective function. This is because the optimization without an explicit Z^{COBias} term may favor higher-accuracy classes which are easier to improve, and becomes "lazy" to improve classes that have low accuracies initially. Applying the learned correction functions will result in a good overall test accuracy with fair COBias. On the other hand, explicitly setting Z^{COBias} in the objective function can help enforce fair improvements for all classes while optimizing overall accuracy. It could bring more accuracy gains for lower-accuracy classes, and thus obtaining even higher overall test accuracy with lower COBias in some cases, especially when there are only a few low-accuracy classes among many higher-accuracy classes initially, like DBpedia and TREC. This suggests that balancing class accuracies can help with overall accuracy gains. Additionally, we ablate both accuracy objectives Z^{Err} and Z^{PMI}, and optimize for these two terms without Z^{COBias}. On DBpedia and TREC, its performance generally falls between optimizing for Z^{Err} and the full objective, further suggesting the importance of Z^{COBias} in reaching optimal solutions.

4.5 Discussions

We visualize the **ablation of sample-level and class-level rebalancing** in Appendix F, demonstrating that sample-level rebalancing significantly elevates weak classes, while class-level rebalancing steadily balances the stronger ones. Moreover, **regarding more model families**, we provide experimental results with Gemma-2-2B in Appendix G, which obtain consistent improvements similar to the Llama-2 cases, demonstrating that our method is applicable to LLMs of varied sizes and families. Lastly, with respect to **the time and cost of applying the DCS method in real-world scenarios**, we elaborate below.

Accessibility: DCS is model-agnostic, requiring only output probabilities, making it particularly suitable for large or closed LLMs where only logits/output probabilities are accessible.

Cost of post-hoc optimization: The post-hoc, offline optimization of DCS requires no model architecture modification and prompt engineering. The annealing time averages 46 seconds (computed from Figure 3). Hyperparameter tuning can be done efficiently because most hyperparameters are pre-configured in practice, with only a few scalar values (e.g., β, τ) requiring tuning. Moreover, since the class-level correction method DNIP demonstrates low-

data optimization capability, we'd infer that DCS is amenable to low-data scenarios. These make DCS particularly valuable in specialized applications where fairness and accuracy are paramount.

Negligible inference overhead: The learned correction functions are reused on-the-fly with mere milliseconds in prediction, introducing virtually no computation overhead or latency.

Scalability: Our method exhibits linear scaling with the number of classes (N). At each temperature in simulated annealing, the number of searches remains within several multiples of the search space size $N(D_F + D_W)$, ensuring practical feasibility across small and large classification tasks.

5 Conclusion and Limitations

This paper proposes DCS, a post-hoc ensemble debiasing framework to correct ICL probabilities across class and sample level. It boosts low-accuracy classes, while keeping the strong ones. It achieves SoTA accuracy on seven benchmark text classification tasks, while effectively mitigating class performance imbalances. The resulting probability correction scheme clearly shows that combining membership correction and weight correction is essential to elevate weak classes. For limitations and future work, the post-hoc correction framework can be extended to other useful scenarios, such as text generation. Combining DCS into the decoding process can help rectify the probabilities of top candidates for the first token (or first several tokens) to lead a better, less biased response. More future directions are discussed in Appendix H.

Acknowledgments

References

Tobias Achterberg. SCIP: Solving constraint integer programs. *Mathematical Programming Computation*, 1(1):1–41, 2009. doi: 10.1007/s12532-008-0001-1. URL https://link.springer.com/article/10.1007/s12532-008-0001-1.

Akshay Agrawal, Robin Verschueren, Steven Diamond, and Stephen Boyd. A rewriting system for convex optimization problems. *Journal of Control and Decision*, 5(1):42–60, 2018.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. DBpedia: A Nucleus for A Web of Open Data. In *Proceedings of the 6th International The Semantic Web and 2nd Asian Conference on Asian Semantic Web Conference*, pp. 722–735, 2007.

Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang, Peter Clark, and Christopher D. Manning. Modeling biological processes for reading comprehension. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1499–1510, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1159. URL https://aclanthology.org/D14-1159/.

Alberto Mario Ceballos-Arroyo, Monica Munnangi, Jiuding Sun, Karen Zhang, Jered McInerney, Byron C. Wallace, and Silvio Amir. Open (clinical) LLMs are sensitive to instruction phrasings. In Dina Demner-Fushman, Sophia Ananiadou, Makoto Miwa, Kirk Roberts, and Junichi Tsujii (eds.), *Proceedings of the 23rd Workshop on Biomedical Natural Language Processing*, pp. 50–71, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.bionlp-1.5. URL https://aclanthology.org/2024.bionlp-1.5/.

V. Cerny. Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications*, 45:41–51, 1985. URL https://doi.org/10.1007/BF00940812.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality, March 2023. URL https://lmsys.org/blog/2023-03-30-vicuna/.

Ido Dagan, Oren Glickman, and Bernardo Magnini. The PASCAL Recognising Textual Entailment Challenge. In *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pp. 177–190, 2006.

Parag Pravin Dakle, Serdar Kadioglu, Karthik Uppuluri, Regina Politi, Preethi Raghavan, SaiKrishna Rallabandi, and Ravisutha Srinivasamurthy. Ner4Opt: Named entity recognition for optimization modelling from natural language. In *CPAIOR*, pp. 299–319, 2023. URL https://doi.org/10.1007/978-3-031-33271-5_20.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423/.

Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann†, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. URL https://transformer-circuits.pub/2021/framework/index.html.

Atticus Geiger, Hanson Lu, Thomas F Icard, and Christopher Potts. Causal abstractions of neural networks. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=RmuXDtjDhG.

Dan Goldwasser, Vivek Srikumar, and Dan Roth. Predicting structures in NLP: Constrained conditional models and integer linear programming in NLP. In *Tutorial Abstracts at the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Montréal, Canada, June 2012. Association for Computational Linguistics. URL https://aclanthology.org/N12-4008/.

Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2024. URL https://www.gurobi.com.

Raymond Hemmecke, Matthias Köppe, Jon Lee, and Robert Weismantel. *Nonlinear Integer Programming*, pp. 561–618. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010. ISBN 978-3-540-68279-0. doi: 10.1007/978-3-540-68279-0_15. URL https://doi.org/10.1007/978-3-540-68279-0_15.

Kashif Hussain, Mohd Najib Mohd Salleh, Shi Cheng, and Yuhui Shi. Metaheuristic research: A comprehensive survey. *Artificial Intelligence Review*, 52(4):2191–2233, 2019. doi: 10.1007/s10462-017-9605-z. URL https://link.springer.com/article/10.1007/s10462-017-9605-z.

Andoni I. Garmendia, Josu Ceberio, and Alexander Mendiburu. Applicability of neural combinatorial optimization: A critical view. *ACM Trans. Evol. Learn. Optim.*, 4(3), July 2024. doi: 10.1145/3647644. URL https://doi.org/10.1145/3647644.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. Certified robustness to adversarial word substitutions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4129–4142, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1423. URL https://aclanthology.org/D19-1423/.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. PubMedQA: A Dataset for Biomedical Research Question Answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 2567–2577, 2019. URL https://aclanthology.org/D19-1259.

Michael Jünger, Thomas M. Liebling, Denis Naddef, George L. Nemhauser, William R. Pulleyblank, Gerhard Reinelt, Giovanni Rinaldi, and Laurence A. Wolsey (eds.). *50 Years of Integer Programming 1958-2008: From the Early Years to the State-of-the-Art*. Springer, 2010. ISBN 978-3-540-68274-5. doi: 10.1007/978-3-540-68279-0. URL https://link.springer.com/book/10.1007/978-3-540-68279-0.

S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *Science*, pp. 671–680, 1983. URL https://www.science.org/doi/abs/10.1126/science.220.4598.671.

Terry Koo, Alexander M. Rush, Michael Collins, Tommi Jaakkola, and David Sontag. Dual decomposition for parsing with non-projective head automata. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 1288–1298, Cambridge, MA, October 2010. Association for Computational Linguistics. URL https://aclanthology.org/D10-1125/.

A. H. Land and A. G. Doig. An automatic method of solving discrete programming problems. *Econometrica*, 28(3):497–520, 1960. ISSN 00129682, 14680262. URL http://www.jstor.org/stable/1910129.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=aLLuYpn83y.

Xin Li and Dan Roth. Learning Question Classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*, 2002. URL https://aclanthology.org/C02-1150.

Ruixi Lin and Hwee Tou Ng. System combination for grammatical error correction based on integer programming. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pp. 824–829, Held Online, September 2021. INCOMA Ltd. URL https://aclanthology.org/2021.ranlp-1.94/.

Ruixi Lin and Hwee Tou Ng. Mind the biases: Quantifying cognitive biases in language model prompting. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 5269–5281, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.324. URL https://aclanthology.org/2023.findings-acl.324/.

Ruixi Lin and Yang You. COBias and Debias: Minimizing Language Model Pairwise Accuracy Bias via Nonlinear Integer Programming, 2024a. URL https://arxiv.org/abs/2405.07623.

Ruixi Lin and Yang You. Let the Fuzzy Rule Speak: Enhancing In-context Learning Debiasing with Interpretability, 2024b. URL https://arxiv.org/abs/2412.19018.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=w0H2xGHlkw.

David G. Luenberger and Yinyu Ye. *Linear and Nonlinear Programming*, volume 116 of *International Series in Operations Research & Management Science*. Springer, 3rd edition, 2008. ISBN 978-0-387-74503-9. doi: 10.1007/978-0-387-74503-9. URL https://link.springer.com/book/10.1007/978-0-387-74503-9.

André Martins, Noah Smith, and Eric Xing. Concise integer linear programming formulations for dependency parsing. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pp. 342–350, Suntec, Singapore, August 2009. Association for Computational Linguistics. URL https://aclanthology.org/P09-1039/.

George Nemhauser and Laurence Wolsey. *The Scope of Integer and Combinatorial Optimization*, chapter I.1, pp. 1–26. John Wiley & Sons, Ltd, 1988. ISBN 9781118627372. doi: https://doi.org/10.1002/9781118627372.ch1. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118627372.ch1.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads, 2022. URL https://arxiv.org/abs/2209.11895.

Dan Roth and Vivek Srikumar. Integer linear programming formulations in natural language processing. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Tutorial Abstracts*, Valencia, Spain, April 2017. Association for Computational Linguistics. URL https://aclanthology.org/E17-5005/.

Dan Roth and Wen-tau Yih. A linear programming formulation for global inference in natural language tasks. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pp. 1–8, Boston, Massachusetts, USA, May 6 - May 7 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-2401/.

Dan Roth and Wen-tau Yih. Integer linear programming inference for conditional random fields. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML '05, pp. 736–743, New York, NY, USA, 2005. Association for Computing Machinery. ISBN 1595931805. doi: 10.1145/1102351.1102444. URL https://doi.org/10.1145/1102351.1102444.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. Code llama: Open foundation models for code, 2024. URL https://arxiv.org/abs/2308.12950.

Alexander M. Rush, David Sontag, Michael Collins, and Tommi Jaakkola. On dual decomposition and linear programming relaxations for natural language processing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 1–11, Cambridge, MA, October 2010. Association for Computational Linguistics. URL https://aclanthology.org/D10-1001/.

Linus Schrage. *Optimization Modeling with LINGO*. LINDO Systems Inc., Chicago, IL, 5th edition, 1999.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying

about prompt formatting. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=RIu5lyNXjT.

Isabel Segura-Bedmar, Paloma Martínez, and María Herrero-Zazo. SemEval-2013 Task 9 : Extraction of Drug-Drug Interactions from Biomedical Texts (DDIExtraction 2013). In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pp. 341–350, 2013. URL https://aclanthology.org/S13-2056.pdf.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, 2013. URL https://aclanthology.org/D13-1170.

Vivek Srikumar and Dan Roth. The integer linear programming inference cookbook, 2023. URL https://arxiv.org/abs/2307.00171.

Michael Steinbrunn, Guido Moerkotte, and Alfons Kemper. Heuristic and randomized optimization for the join ordering problem. *The VLDB Journal*, 6:191–208, 1997. URL https://doi.org/10.1007/s007780050040.

Christos Thrampoulidis. Implicit optimization bias of next-token prediction in linear models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 22624–22656, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/2858e880333b3cd64f8192f13ddcca2f-Paper-Conference.pdf.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open Foundation and Fine-Tuned Chat Models. *arXiv preprint arXiv:2307.09288*, 2023.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. Investigating gender bias in language models using causal mediation analysis. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 12388–12401. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/92650b2e92217715fe312e6fa7b90d82-Paper.pdf.

Ellen M. Voorhees and Dawn M. Tice. Building a question answering test collection. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 200–207, 2000. URL https://doi.org/10.1145/345508.345577.

Ming-Wei Wang, Nicholas Rizzolo, and Dan Roth. Integer linear programming in NLP - constrained conditional models. In *NAACL HLT 2010 Tutorial Abstracts*, pp. 9–14, Los Angeles, California, June 2010. Association for Computational Linguistics. URL https://aclanthology.org/N10-4005/.

Xuezhi Wang and Denny Zhou. Chain-of-thought reasoning without prompting. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=4Zt7S0B0Jp.

Yuanbin Wu and Hwee Tou Ng. Grammatical error correction using integer linear programming. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1456–1465, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL https://aclanthology.org/P13-1143/.

Haoting Zhang, Jinghai He, Jingxu Xu, Jingshen Wang, and Zeyu Zheng. Enhancing language model with both human and artificial intelligence feedback data. In *2024 Winter Simulation Conference (WSC)*, pp. 299–310, 2024. doi: 10.1109/WSC63780.2024.10838929.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level Convolutional Networks for Text Classification. In *Advances in Neural Information Processing Systems*, 2015. URL https://proceedings.neurips.cc/paper_files/paper/2015/file/250cf8b51c773f3f8dc8b4be867a9a02-Paper.pdf.

Yize Zhao, Tina Behnia, Vala Vakilian, and Christos Thrampoulidis. Implicit geometry of next-token prediction: From language sparsity patterns to model representations. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=qyilOnIRHI.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate Before Use: Improving Few-shot Performance of Language Models. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 12697–12706, 2021. URL https://proceedings.mlr.press/v139/zhao21c.html.

Han Zhou, Xingchen Wan, Lev Proleev, Diana Mincu, Jilin Chen, Katherine A Heller, and Subhrajit Roy. Batch Calibration: Rethinking Calibration for In-Context Learning and Prompt Engineering. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=L3FHMoKZcS.

A Discussion on Solution Methods for NIP

Typical solutions for combinatorial optimization problems include the branch-and-bound (B&B) algorithm, which is useful for solving mixed-integer linear programs (Land & Doig, 1960). Solvers are made into both open-source software such as CVXPY (Diamond & Boyd, 2016; Agrawal et al., 2018) or SCIP (Achterberg, 2009), and commercial solvers like Gurobi (Gurobi Optimization, LLC, 2024) or LINGO (Schrage, 1999).

We need to point out that B&B and other related algorithms are not very suitable to search solutions for the proposed nonlinear integer programming based optimization model, which contain discontinuous functions and is inherently non-differentiable. Relaxation to subproblems is needed if B&B has to be used. Relaxation can be done by creating Lagrange multipliers and Lagrangian functions for each constraint, which is often easier to solve than the original nonlinear integer programming problem, because only continuous variables are involved (Luenberger & Ye, 2008; Jünger et al., 2010). The relaxed problem finds the optimal value of the Lagrangian function, which is a lower bound for the original problem, and it is used as a bound in the B&B search. However, solving the subproblems efficiently poses challenges, as B&B is generally not a polynomial-time algorithm.

Instead, we can opt for metaheuristics (Hussain et al., 2019), such as simulated annealing (Kirkpatrick et al., 1983; Cerny, 1985), to find the optimal solution. Although heuristic search can get slower with increasing search space size (Jia et al., 2019), its computational complexity is within polynomial time (Lin & You, 2024a). To enable application of our proposed method to tasks with even more classes, we adopt simulated annealing (SA) to solve our model.

In SA, we define an initial temperature T and an annealing schedule including the cooling rate and the stopping criteria. In each iteration, a neighborhood solution is generated by perturbing the current solution. The algorithm calculates the objective function value for the new solution (Step *Evaluate Z* in Fig. 2). If the new solution is better, accept it; if the new solution is worse, accept it with a probability determined by the Metropolis criterion. Then the temperature is reduced according to the cooling rate, until convergence. In addition, the

initial solution can also be randomly generated, where N numbers are randomly selected from $\mathbb{D} = \{1, \ldots, D_F + D_W\}$ as the initial values for the selection variables ξ_j's.

B List of notations

The parameters, sets and decision variables used throughout this paper are listed in Table 5.

Symbol	Description
Parameters	
p_m	The ICL output token's class probabilities of the m-th instance in a dataset
ω_k	The k-th weight coefficient where $k \in \{1, ..., D_F\}$
μ_k	The k-th membership function where $k \in \{D_F + 1, ..., D_F + D_W\}$
Sets	
$\mathbb{R}^{\text{Weight}}$	The set of D_W weight coefficients, equally spaced discrete values from 0 to 1
$\mathbb{R}^{\text{Membership}}$	The set of D_F triangular membership functions
$\mathbb{D}$	The set of selection indices
Variables	
$y_m, \hat{y}_m$	The label and prediction of the m-th instance
ξ_j	The integer selection variable for class j

Table 5: List of notations

C Details on the Objective Function

$Z^{\text{Err}}, Z^{\text{COBias}}, Z^{\text{PMI}}$ respectively follows from the three terms of Eq. 5 in (Lin & You, 2024a), which include:

The overall error rate for M optimization instances:

$$Z^{\text{Err}} = \frac{1}{M} \sum_{m=1}^{M} \mathbb{1}\{\hat{y}_m \neq y_m\}, \text{ where } \hat{y}_m = \underset{n \in \{1, ..., N\}}{\arg\max} \, p'_{mn}, m = 1, \ldots, M \quad (6)$$

COBias of all class pairs:

$$Z^{\text{COBias}} = \binom{N}{2}^{-1} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \left| A_{c_i} - A_{c_j} \right|, \text{ where } A_{c*} \text{ is class } c* \text{'s accuracy} \quad (7)$$

PMI between ground-truth instances of class j and predictions of class j:

$$Z^{\text{PMI}} = - \sum_{j=1}^{N} \text{PMI}_j \quad (8)$$

where

$$\text{PMI}_j = \text{PMI}(\hat{S}^j, S^j) = \log \frac{f(\hat{S}^j, S^j)}{f(\hat{S}^j) f(S^j)}, \text{ for } j = 1, \ldots, N \quad (9)$$

Here, $\hat{S}^j$ and S^j denote the instances of prediction j and true label j respectively, $f(\hat{S}^j)$ is the ratio between the number of instances with prediction j and the total number of instances, similarly, $f(S^j)$ is the ratio between the number of instances labeled j and the total number of instances, $f(\hat{S}^j, S^j)$ is the ratio between the number of correct predictions of class j and the total number of instances.

D   Experimental Setups

For each evaluation task, we tune the weight scale for weight correction, β and τ used in the objective function, and the simulated annealing algorithm's number of outer loops, inner loops, the initial temperature, and stopping criteria (stopping temperature, the maximal number of solutions generated, and the maximal number of accepted solutions) on a development set. The optimization/development set split is 0.95/0.05. Dataset preprocessing follows DNIP (Lin & You, 2024a), and their statistics are shown in Table 6.

Dataset	Optimization set size (incl. dev)	Evaluation set size	Prompt
AGN	10,000	5,000	Please classify the news articles into the categories of World, Sports, Business, and Technology.\n\n Article: [Input] Answer:
DBP	10,000	5,000	Please classify the documents based on whether they are about a Company, School, Artist, Athlete, Politician, Transportation, Building, Nature, Village, Animal, Plant, Album, Film, or Book.\n\n Article: [Input] Answer:
SST	8,544	2,210	Review: [Input] Sentiment:
TREC	5,452	500	Please classify the questions based on whether their answer type is a Number, Location, Person, Description, Entity, or Abbreviation.\n\n Question: [Input] Answer Type:
RTE	2,490	277	[Input premise] question: [Input hypothesis] True or False? answer:
DDI	10,000	5,716	Please choose a most suitable answer from Negative, Effect, Mechanism, Advice, or Interaction, for the drug-drug interaction relation between the @drug$ pair in the following description.\n\n Description: [Input] Answer:
PMQA	1,000	500	Please choose a most suitable answer from yes, no, or maybe, for the following question given a context.\n\n Context: [Input context] Question: [Input question] yes, no, or maybe? Answer:

Table 6: Dataset statistics

E   Full Performance Comparisons on Llama-2-70B

Our method effectively applies to larger LLM variants, showcased by Llama-2-70B. Debiasing results using DCS on 1-shot ICL outputs in Table 7 show consistent improvements in average overall accuracy across three seeds of demonstrations, outperforming DNIP and FuRud. Our method achieves better COBias reduction than the other two methods on general domain tasks, and better accuracy with comparable COBias reduction on biomedical domain tasks. The takeaway is, increasing LLM sizes without post-hoc corrections may not always be an effective way to resolve output class accuracy imbalances, which further validates the necessity of the proposed post-hoc correction method.

Method		Acc									COBias									
	All	General Domain						Biomedical Domain			All	General Domain						Biomedical Domain		
		Avg	AGN	DBP	SST	TREC	RTE	Avg	DDI	PMQA		Avg	AGN	DBP	SST	TREC	RTE	Avg	DDI	PMQA
1-shot ICL + post-hoc debiasing																				
ICL	66.4	76.2	87.4_{22}	94.1_{10}	41.8_{86}	79.3_{25}	78.6_{26}	41.7	8.8_{37}	74.5_{23}	31.8	25.1	14.5_{56}	8.3_{12}	54.7_{14}	22.4_{20}	25.5_{76}	48.8	36.0_{21}	61.5_{33}
+DNIP	73.6	79.0	89.8_{02}	96.5_{05}	49.4_{20}	79.3_{40}	80.0_{11}	60.0	52.5_{144}	67.5_{24}	17.3	15.9	6.5_{04}	3.8_{03}	47.1_{140}	17.3_{48}	4.6_{18}	21.1	16.4_{90}	25.7_{20}
+FuRud	75.9	79.0	89.4_{06}	96.2_{08}	50.0_{22}	79.5_{38}	79.9_{91}	68.1	60.0_{118}	76.1_{22}	21.4	15.0	7.1_{10}	3.8_{06}	40.1_{83}	19.0_{52}	4.8_{15}	37.5	25.2_{75}	49.7_{94}
+DCS	76.8	79.6	89.9_{02}	96.7_{05}	50.1_{17}	81.0_{36}	80.1_{11}	70.0	63.7_{62}	76.2_{24}	20.7	14.3	6.6_{02}	3.6_{02}	40.3_{82}	15.6_{56}	5.3_{16}	36.8	24.3_{55}	49.2_{91}

Table 7: Quantitative evaluations on Llama-2-70B

F   Ablation of Sample-level and Class-level Rebalancing

We visualize the ablation of sample-level and class-level rebalancing using TREC (general domain) and PubMedQA (biomedical domain) in Figure 4a and 4b. Both datasets show that sample-level rebalancing significantly elevates weak classes, while class-level rebalancing steadily balances the stronger ones. Ensembling yields overall gains in both metrics.
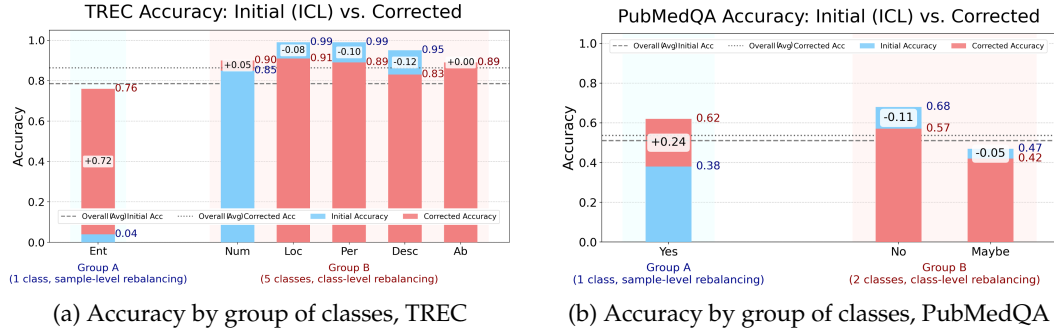

(a) Accuracy by group of classes, TREC   (b) Accuracy by group of classes, PubMedQA

Figure 4: Ablation of sample-/class-level rebalancing

G   Additional Performance Comparisons on Gemma-2-2B

Method		Acc									COBias									
	All	General Domain						Biomedical Domain			All	General Domain						Biomedical Domain		
		Avg	AGN	DBP	SST	TREC	RTE	Avg	DDI	PMQA		Avg	AGN	DBP	SST	TREC	RTE	Avg	DDI	PMQA
1-shot ICL + post-hoc debiasing																				
ICL	53.4	59.0	79.9_{34}	72.6_{27}	27.6_{43}	44.6_{47}	70.3_{66}	39.4	15.4_{106}	63.4_{10}	38.5	35.9	25.7_{69}	36.6_{18}	42.0_{15}	50.2_{07}	25.2_{131}	44.8	35.1_{130}	54.4_{32}
+DNIP	65.1	70.1	86.9_{11}	88.4_{09}	33.2_{79}	69.6_{08}	72.6_{23}	52.5	41.5_{182}	63.4_{113}	19.9	15.9	7.1_{05}	14.0_{10}	34.3_{99}	21.4_{28}	2.6_{18}	30.0	18.3_{72}	41.8_{273}
+FuRud	66.9	71.6	85.3_{18}	88.4_{17}	44.1_{85}	68.5_{06}	71.7_{47}	55.3	43.0_{189}	67.5_{12}	23.1	18.3	12.3_{23}	12.7_{36}	40.1_{170}	21.6_{12}	4.7_{30}	35.1	18.2_{109}	51.9_{06}
+DCS (Ours)	69.2	74.5	87.0_{11}	92.5_{09}	49.4_{17}	70.9_{25}	72.6_{23}	56.0	44.1_{03}	67.9_{23}	23.3	17.8	7.7_{10}	6.8_{03}	50.7_{09}	21.3_{28}	2.6_{18}	37.0	22.4_{42}	51.5_{48}

Table 8: Quantitative evaluations on Gemma-2-2B

We additionally experimented with Gemma-2-2B, and obtained consistent improvements (Table 8) similar to the Llama cases, showing that our method is applicable to LLMs of varied sizes and families.

H   More Future Directions

This work deals with prediction biases when directly prompting LLMs. When LLMs or smaller LMs like BERT (Devlin et al., 2019) are finetuned on tasks, which may result in further imbalanced class accuracies, post-hoc corrections can make class accuracies fairer. It would be valuable to understand how these output class probabilities differ from those by direct prompting, and enable custom extensions to the current framework to provide more targeted corrections. It would also be interesting to see how it may help with data-filtered or data-augmented finetuning. In terms of interpretability, those classes that apply class-level correction to any instance is interpretable only at the broader level, so further investigations on how to enforce more interpretability to weight corrections could be conducted.