

---

# Whose Instructions Count? Resolving Preference Bias in Instruction Fine-Tuning

---

Jiayu Zhang<sup>1,2\*</sup>, Changbang Li<sup>3\*</sup>, Yinan Peng<sup>4</sup>, Weihao Luo<sup>5</sup>, Peilai Yu<sup>6</sup>, Xuan Zhang<sup>1,7†</sup>

<sup>1</sup>Airon Technology CO., LTD <sup>2</sup>Peking University <sup>3</sup>University of Pennsylvania

<sup>4</sup>Hengxin Technology Ltd. <sup>5</sup>Donghua University <sup>6</sup>Ludwig Maximilian University of Munich

<sup>7</sup>Carnegie Mellon University

## Abstract

Instruction fine-tuning (IFT) has emerged as a ubiquitous strategy for specializing large language models (LLMs), yet it implicitly assumes a single, coherent “ground-truth” preference behind all human-written instructions. In practice, annotators differ in the styles, emphases, and granularities they prefer, introducing preference bias that can erode both robustness and generalization. We propose Dynamic Cross-Layer Preference Correction (DCPC), it couples (i) a preference-sensitive similarity estimator that detects mismatched instructional cues, (ii) cross-layer prefix alignment to reconcile semantic representations across transformer layers, and (iii) a lightweight Preference Correction Module (PCM) that dynamically adjusts hidden states to honor the inferred dominant preference. On five Super/GLUE tasks and the ALPACA set—plus six preference-shifted variants—DCPC boosts accuracy/F1-EM by 4.0–6.7 points and gpt-score by +0.7, while cutting inter-seed variance up to 35% on LLaMA-2 13B and Mistral-7B, setting a new state of the art for robust instruction tuning.

## 1 Introduction

Large language models (LLMs) [Naveed et al., 2023, Shanahan, 2024, Hu et al., 2025, Zhang et al., 2025b] have become the backbone of modern natural language systems [Zhang et al., 2023a, Zhou et al., 2025] and are now widely used in healthcare[Tong et al., 2025, Liu et al., 2025, Wang et al., 2025a], autonomous systems [Yao et al., 2023] and other applications [Jiang et al., 2025, Tao et al., 2023, Xu et al., 2025a, Shen and Zhang, 2025, He and Qu, 2025, Wang et al., 2025b, Sun and Li, 2025]. A common way to specialize these models for downstream use is *instruction fine-tuning* (IFT) [Zhang et al., 2023c, Ghosh et al., 2024], where a pretrained model is adapted using various instruction datasets that provide task-specific demonstrations [Xu et al., 2025b, Chung et al., 2024, Su et al., 2022, Kaur et al., 2024, Tan et al., 2025, Yao et al., 2024, Zhang et al., 2025a].

Existing IFTs assumes that all instructions in the data reflect a single, coherent supervisory signal. [Ren et al., 2024, Shi et al., 2024, Jiang et al., 2024, Edstedt et al., 2024, Liu and Xiao, 2025] In practice, however, different annotators (or even the same annotator over time) exhibit distinct stylistic and semantic preferences [Li et al., 2025, Lu et al., 2025b]—e.g. terse versus verbose answers, formal versus colloquial tone [Cheng and Cosley, 2013], strict adherence to specification versus creative elaboration [Zhao et al., 2020, Miao et al., 2023, Chen et al., 2024b, Xiao and Liu, 2025]. This will make the training data highly imbalanced and extremely noisy [Sun et al., 2025]. More information often does not lead to a stable improvement in performance [Wen et al., 2023, Ke et al., 2025, Lu et al., 2025a, Lan et al., 2025a].

These latent disagreements introduce preference bias:

---

\* Equal contribution, †Corresponding author(xuanzhang2199@gmail.com)

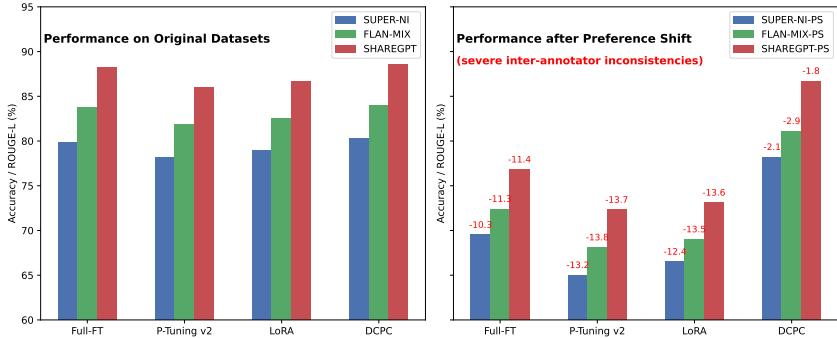


Figure 1: Performance comparison of *Full fine-tuning* (Full-FT), instruction fine-tuning baselines, and our DCPC across three benchmarks.

*The same input can receive multiple, equally plausible but inconsistent target responses.*

**Why does preference bias matter?** Figure 1 shows that when we deliberately perturb instructional corpora with realistic preference shifts-mimicking heterogeneous crowdsourcing-the performance of both full-parameter and leading IFT baselines plunges. The root cause is that gradient updates are pulled in conflicting directions, forcing the model to memorize idiosyncrasies instead of learning a consensus. As a result, generation becomes brittle: the model overfits to whichever style dominates locally in the mini-batches it sees.

**Our approach.** We introduce Dynamic Cross-Layer Preference Correction (DCPC), a self-supervised procedure that resolves these conflicts on the fly. Instead of treating each instruction-response pair as an immutable oracle, DCPC: (i) estimates *preference-sensitive similarity* between response embeddings to detect subtle stylistic clashes, (ii) aligns transformer layers through cross-layer prefix tuning, ensuring that semantically equivalent instructions map to nearby hidden representations, and (iii) rectifies residual bias with a lightweight Preference Correction Module (PCM) that dynamically re-weights layer activations toward the dominant consensus. The entire pipeline is label-free, incurs negligible extra parameters, and plugs into any parameter-efficient adapter. The main contributions of this paper are:

- 1) We formulate preference bias in instruction fine-tuning and provide the first systematic evaluation using *preference-shifted* variants of three popular datasets.
- 2) We propose DCPC, a self-supervised framework that detects and corrects inter-annotator preference discrepancies via cross-layer prefix alignment and the PCM.
- 3) Across two representative open-source backbones—*LlaMA-2 13B* and *Mistral-7B*—DCPC secures consistent absolute gains of 4.0–6.7 points in accuracy/F1-EM (and +0.7 in gpt-score) on preference-shifted benchmarks, while cutting inter-seed variance under synthetic shifts by up to 35% versus the strongest PEFT baselines, thereby setting a new state of the art for robust instruction fine-tuning.

## 2 Related works

**Instruction Fine-Tuning.** Early work demonstrates that large language models (LLMs) can gain strong zero-shot and few-shot capabilities after being fine-tuned on diverse instruction–response pairs [Wang and Zhang, 2024, Lin et al., 2025, Lan et al., 2025b]. Wei et al. [2022] first showed that “finetuned language models are zero-shot learners” by fine-tuning T5 on hundreds of crowdsourced tasks. Subsequently, Sanh et al. [2022] introduced T0, pushing multi-task prompted training to stronger generalisation. Community efforts have since scaled both data and models: Alpaca [Taori et al., 2023a] and Vicuna extend 7 B-parameter LLaMA with self-instruct data, while Honovich et al. [2022] propose UNNATURAL-INSTRUCTIONS to generate millions of synthetic tasks with minimal human supervision. Recent studies focus on optimising task mixtures, e.g. DoReMi [Ramachandran and et al., 2023], and on adaptive gradient reweighting across tasks [Mueller et al., 2024]. These works provide solid baselines but largely assume high-quality, non-conflicting supervision.

**Instruction Fine-Tuning with inconsistent labels.** When instruction corpora originate from heterogeneous sources[Chen et al., 2025], the same input may receive mutually conflicting directives, leading to gradient interference and unpredictable behaviour[Xiao et al., 2025a]. Constitutional AI [Bai et al., 2022] alleviates safety–helpfulness conflicts by inserting system-level “constitution” rules and performing RL with AI feedback. WizardLM [Xu et al., 2023] explores curriculum learning that progressively exposes the model to increasingly complex or contradictory prompts. Task-level weighting schemes [Mueller et al., 2024] dynamically down-weight tasks whose gradients conflict with others, reducing performance regressions. Beyond training, Zhang et al. [2025c] propose IHEVAL, a benchmark that diagnoses whether a model correctly resolves hierarchy conflicts between system, developer and user instructions. However, these methods either impose external constraints or apply coarse task-level heuristics [Yan et al., 2025], and none explicitly reconcile conflicting annotator preferences within the model’s representations; our DCPC fills this gap by dynamically aligning layers and correcting preference bias during fine-tuning.

**Instruction Fine-Tuning with noisy labels.** Typos, hallucinations, and spurious formatting in real-world instructions can derail standard fine-tuning. NEFTUNE [Jain et al., 2024] and Sym-Noise [Yadav and Singh, 2023] regularise training by injecting small or symmetric embedding noise, while RobustFT [Luo et al., 2024] re-labels suspect examples with teacher LLMs. PromptBench [Zhu et al., 2024] further shows large robustness gaps under token-level perturbations. Yet these approaches rely on external noise injection or data cleansing and leave the model’s internal preference drift untouched; DCPC instead corrects noise *within* the representation space with self-supervise manner.

### 3 Methods

#### 3.1 Preliminaries: Instruction Fine-Tuning with Prefix Adapters

Among the many parameter-efficient fine-tuning (PEFT) schemes, P-Tuning v2 [Liu et al., 2021] is particularly attractive for instruction fine-tuning (IFT) because it injects a *continuous prefix* into every Transformer block while freezing the backbone weights. Formally, given an input sequence  $x = \{x_1, \dots, x_n\}$ , let  $\mathbf{e}_x^l \in \mathbb{R}^d$  be its hidden representation at layer  $l$  ( $d$ -dimensional). P-Tuning v2 introduces a learnable prefix  $\mathbf{P}^l \in \mathbb{R}^{m \times d}$  ( $m \ll n$ ) that is *concatenated* before the token embeddings:

$$\tilde{\mathbf{e}}_x^l = [\mathbf{P}^l; \mathbf{e}_x^l], \quad (1)$$

which is then processed by the layer’s self-attention and feed-forward sub-blocks:

$$\mathbf{h}_x^l = \text{TransformerLayer}^l(\tilde{\mathbf{e}}_x^l). \quad (2)$$

Only  $\{\mathbf{P}^l\}_{l=1}^L$  are updated; all backbone parameters remain fixed.

**Limitation.** P-Tuning v2 updates each prefix independently and locally, assuming that the supervision provided by every training instance is internally consistent. Our toy study in Figure 2 (see Appendix A) shows that this assumption breaks down whenever the same semantic content is labelled according to different annotator preferences. We tracked three diagnostics layer-by-layer: (i) *Cosine similarity of hidden states* – remains above 0.90 for the first five layers, confirming

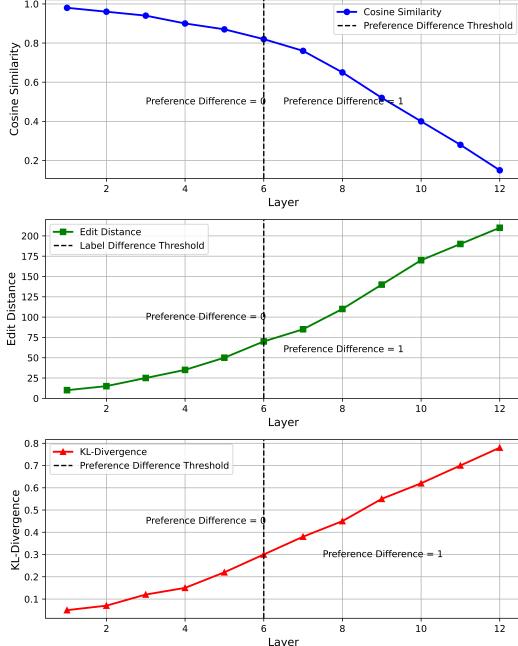


Figure 2: Toy study on IMDB: High cosine similarity coexists with label distribution drift in deep layers, motivating DCPC’s design.

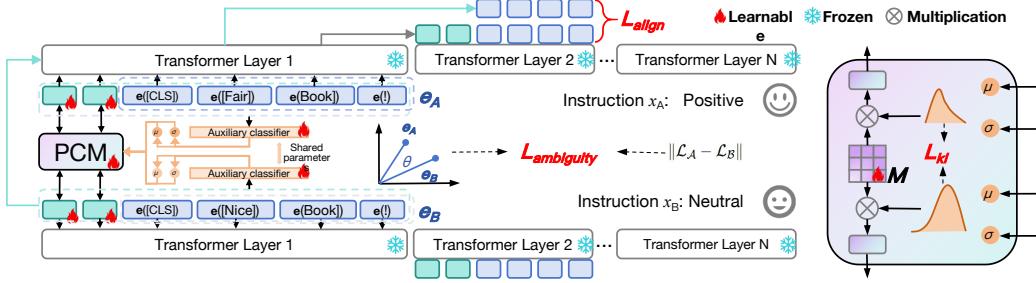


Figure 3: **DCPC pipeline.** Two instructions  $x_A, x_B$  flow through a frozen LLM equipped with prefix adapters (blue). A preference-sensitive gate (orange) flags layers where hidden states are close yet their label distributions diverge (red). DCPC then (i) swaps prefixes across layers to contract the two streams and (ii) lets a Preference Correction Module (PCM) inject meta-generated prefix slices, steering both paths toward a shared instructional style—without touching backbone weights.

that the two instructions share almost identical semantics. (ii) *Edit distance between generated responses* – stays near zero initially, then grows rapidly after layer 6 as stylistic differences emerge. (iii) *KL-divergence between soft label distributions* – rises from 0.02 to 0.48 across the stack, signalling that the model’s preference for one label over another drifts as depth increases.

Thus, even though  $e_A^l$  and  $e_B^l$  remain close in representation space, the *predicted* distributions  $L_A$  and  $L_B$  diverge sharply in deeper layers. This phenomenon indicates that locally-trained prefixes do not propagate a global notion of instruction style; instead, inconsistent supervisory signals are amplified layer-by-layer, ultimately causing brittle and contradictory outputs.

**Main idea.** *Dynamic Cross-Layer Preference Correction* (DCPC) is designed to halt this drift: it explicitly detects high-similarity / low-agreement pairs, realigns their prefixes across layers, and injects preference-corrected slices via the PCM—thereby restoring a stable, consensus prediction trajectory through the network.

### 3.2 Overview of DCPC

Figure 3 sketches the DCPC pipeline. For a pair of instructions  $(x_A, x_B)$  we obtain layer-wise embeddings  $e_A^l, e_B^l$  and corresponding prefixes  $P_A^l, P_B^l$ . DCPC comprises three interacting modules: (i) **Preference-Sensitive Similarity**. Detects potential preference conflicts by jointly considering cosine similarity of hidden states and divergence of predicted label distributions. (ii) **Cross-Layer Prefix Alignment**. If a conflict is flagged, prefixes of the two instructions are *cross-mixed* across consecutive layers and optimised to contract the distance between the mixed representations, enforcing layer-to-layer semantic agreement. (iii) **Preference Correction Module (PCM)**. A lightweight auxiliary pathway predicts a preference distribution for each instruction and synthesises *new* prefix slices from a meta-matrix  $\mathcal{M}$ . These slices are injected back into the model, nudging predictions toward a consensus style.

### 3.3 Preference-Sensitive Similarity Mechanism

The first pillar of DCPC is a detector that flags high-similarity / low-agreement instruction pairs. For every layer  $l \in \{1, \dots, L\}$  and every mini-batch pair  $(x_A, x_B)$  we compute two signals:

**Semantic proximity.** The cosine similarity of the hidden states

$$s^l = \cos(e_A^l, e_B^l) = \frac{e_A^l \cdot e_B^l}{\|e_A^l\| \|e_B^l\|}, \quad (3)$$

which remains near 1 for paraphrases or semantically redundant instructions.

**Preference divergence.** Let  $L_A, L_B \in \Delta^{K-1}$  be the soft label distributions (the model’s logits passed through softmax) for  $x_A$  and  $x_B$ , where  $K$  is the label vocabulary size. We measure their disagreement by the *symmetric* KL divergence

$$d = \frac{1}{2} \left( D_{\text{KL}}(L_A \| L_B) + D_{\text{KL}}(L_B \| L_A) \right). \quad (4)$$

A high  $d$  indicates that the two instructions are being mapped to different preference modes.

We combine the two signals into a single score

$$\alpha^l = \underbrace{\mathbb{I}[s^l \geq \tau_{\cos}]}_{\text{semantic gate}} \cdot \underbrace{s^l}_{\text{proximity}} \cdot \underbrace{d}_{\text{divergence}}, \quad (5)$$

where the gate (threshold  $\tau_{\cos}$ ) ensures we only scrutinise pairs that are already semantically close. Intuitively, (5) penalises the model *more* when two near-duplicate instructions elicit sharply different label distributions.

Because preference drift often magnifies in deeper layers (§3.1), we aggregate the per-layer scores using an exponential weighting that emphasises higher layers:

$$L_{\text{ambiguity}} = \sum_{l=1}^L \beta^l \alpha^l, \quad \beta^l = \frac{\exp(\gamma l)}{\sum_{j=1}^L \exp(\gamma j)}, \quad (6)$$

where  $\gamma > 0$  is a temperature hyper-parameter (default 0.1). If  $L_{\text{ambiguity}}$  exceeds a user-defined threshold  $\tau_{\text{amb}}$ , the pair is marked as a *preference-conflict exemplar* and handed off to the subsequent *cross-layer alignment* and *PCM* stages (§3.4, §3.5). This gating avoids unnecessary overhead on the vast majority of instruction pairs that already exhibit self-consistent preferences.

### 3.4 Cross-Layer Prefix Alignment

If the ambiguity score in Eq. (6) exceeds the threshold  $\tau_{\text{amb}}$  for a pair  $(x_A, x_B)$ , we activate the *cross-layer prefix alignment* module. The goal is to pull the two representation streams toward a common trajectory without touching backbone weights.

**Prefix-token mixing.** For every layer  $l \in \{1, \dots, L-1\}$  we form composite representations

$$\mathbf{C}_A^l = \mathbf{P}_A^l \oplus \mathbf{T}_B^{l+1}, \quad \mathbf{C}_B^l = \mathbf{P}_B^l \oplus \mathbf{T}_A^{l+1}, \quad (7)$$

where  $\oplus$  concatenates along the sequence dimension and  $\mathbf{T}_B^{l+1}$  denotes the token embeddings of  $x_B$  *one layer deeper*. Swapping prefixes across adjacent layers exposes each instruction to the other’s contextual bias while preserving locality.

**Alignment objective.** We minimise the mean-squared distance

$$L_{\text{align}} = \frac{1}{L-1} \sum_{l=1}^{L-1} \|\mathbf{C}_A^l - \mathbf{C}_B^l\|_2^2, \quad (8)$$

optimising only  $\{\mathbf{P}_A^l, \mathbf{P}_B^l\}_{l=1}^{L-1}$ . A single extra backward pass suffices because the two composite streams share backbone parameters.

**Contraction guarantee.** Under mild smoothness conditions, the alignment update yields an *exponential decay* of inter-stream distance:

**Theorem 1** (Layer-wise Contraction). *Assume each Transformer block  $f^l$  is  $L$ -Lipschitz under the Euclidean norm with a constant  $L \in (0, 1)$ , and let  $d^l = \|\mathbf{C}_A^l - \mathbf{C}_B^l\|_2$  be the composite distance at layer  $l$ . Choose the learning rate  $\eta^* = \frac{1-L}{4} \in (0, \frac{1}{4})$ . After a single gradient step on Eq. (8) with  $\eta^*$  and one forward pass through  $f^{l+1}$  we have*

$$d^{l+1} \leq L^2 d^l, \quad (9)$$

so that after  $k$  successive alignment updates  $d^{l+k} \leq L^{2k} d^l$ , which converges geometrically to zero.

The complete proof is provided in Appendix B.

### 3.5 Preference Correction Module (PCM)

An *auxiliary classifier* maps each embedding to  $(\mu, \sigma) \in \mathbb{R}^d \times \mathbb{R}^d$ , parameterising a diagonal Gaussian. Sampling  $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ , we obtain a soft preference distribution

$$p_{\text{pref}} = \text{softmax}(\mu + \sigma \odot \epsilon) \in \mathbb{R}^d. \quad (10)$$

Multiplying by a shared meta-matrix  $\mathcal{M} \in \mathbb{R}^{m \times d}$  yields a corrected prefix slice

$$\mathbf{P}_{\text{new}} = \mathcal{M} p_{\text{pref}} \in \mathbb{R}^m. \quad (11)$$

A KL penalty aligns the two preference distributions:

$$L_{\text{KL}} = D_{\text{KL}}(p_{\text{pref}}^{(A)} \parallel p_{\text{pref}}^{(B)}). \quad (12)$$

Finally, The overall loss function is defined as:

$$L_{\text{total}} = \lambda_1 L_{\text{ambiguity}} + \lambda_2 L_{\text{align}} + \lambda_3 L_{\text{KL}} \quad (13)$$

where  $\lambda_1$ ,  $\lambda_2$ , and  $\lambda_3$  are hyperparameters controlling the relative importance of each loss term. The objective is to minimize label inconsistency while maintaining alignment across embedding layers and correcting for label preference discrepancies.

**Sample Complexity Under Preference Noise** We analyse DCPC in the presence of *preference noise*, modelled as *class-conditional label noise* (CCN) with flip rate  $\rho < \frac{1}{2}$ . Formally, let the latent clean distribution be  $\mathcal{D} = \{(x, y) \in \mathcal{X} \times [K]\}$ , and assume the observed label  $\tilde{y}$  satisfies  $\Pr[\tilde{y} \neq y \mid y] = \rho$ .<sup>1</sup> Denote by  $\mathcal{H}_P$  the *prefix-augmented* hypothesis class realised by DCPC, and write  $D = \text{Pdim}(\mathcal{H}_P)$  for its pseudo-dimension.

**Theorem 2** (Sample Complexity under Class-Conditional Noise). *Fix  $\delta \in (0, 1)$ ,  $\rho < \frac{1}{2}$ , and let  $n \geq \frac{C}{(1-2\rho)^2} \left( D + \ln \frac{2}{\delta} \right)$  for a universal constant  $C$ . Train DCPC on  $n$  i.i.d. noisy samples  $\{(x_i, \tilde{y}_i)\}_{i=1}^n$  using the total loss  $L_{\text{total}}$  in Eq. (25), and let  $\hat{h} \in \mathcal{H}_P$  be the empirical minimiser. Then, with probability at least  $1 - \delta$ ,*

$$\mathcal{R}(\hat{h}) - \mathcal{R}^* \leq \frac{8}{1-2\rho} \sqrt{\frac{D + \ln(2/\delta)}{n}}, \quad (14)$$

where  $\mathcal{R}$  is the clean (noise-free) 0-1 risk and  $\mathcal{R}^* = \min_{h \in \mathcal{H}_P} \mathcal{R}(h)$ .

A full derivation is deferred to Appendix C.

## 4 Experiments

### 4.1 Experimental Setup

**Datasets** We evaluate the performance of DCPC framework using a variety of datasets that involve subjective labeling or human preference discrepancies: (a) three tasks from SuperGLUE benchmark (BoolQ, COPA, and ReCoRD)[Wang et al., 2019]. (b) two tasks from GLUE benchmark (SST-2 and RTE)[Wang, 2018]. (c) Alpaca Dataset[Taori et al., 2023b]. For a detailed description of these datasets, see D.3. We are also conducting experiments on the task of code understanding and mathematical reasoning, and the experimental results can be found in Appendix E.1.

Additionally, we extend these datasets with modified versions to introduce shifts in label preferences and biases, such as BoolQ-PreferenceShift (*BoolQ-PS*), COPA-BiasShift (*COPA-BS*), ReCoRD-Rephrase (*ReCoRD-R*), SST-2-PolarityShift (*SST-2-P*), RTE-EntailmentShift (*RTE-E*), and Alpaca-InstructionShift (*Alpaca-IS*). These variations allow us to simulate real-world annotator biases and inconsistencies. Detailed descriptions of the datasets and modifications can be found in the appendix (see D.3).

**Baselines** We compare our Dynamic Cross-Layer Preference Correction (DCPC) with full-parameter fine-tuning (Full-FT) and several state-of-the-art PEFT methods. Representation modification methods include (IA)<sup>3</sup> [Liu et al., 2022a], which scales hidden representations using trainable vectors. Adapter-based methods, such as Houlsby-Adapter [Houlsby et al., 2019] and Learned-Adapter [Zhang et al., 2023d], add bottleneck layers for efficient tuning. Prompt-based tuning methods include P-Tuning v2 [Liu et al., 2021], LPT [Liu et al., 2022b], and PEDRO [Xie et al., 2024]. We also evaluate

<sup>1</sup>For simplicity we treat all classes symmetrically; the result extends to asymmetric flip matrices with a worst-case rate  $\rho_{\max}$ .

Table 1: Performance comparison of DCPC and baseline methods on original datasets. Results are the median of five random seeds. Backbone: *LLaMA-2 7B*. Bold and underlined numbers denote the best and second-best results, respectively.

Method	Tunable Params	BoolQ (acc)	COPA (acc)	ReCoRD (f1-em)	SST-2 (acc)	RTE (acc)	Alpaca (gpt-score)
Full-FT	7B	88.6	91.5	92.1	94.1	84.8	9.2
P-Tuning v2	9.4M	85.4	89.8	89.2	92.5	80.9	8.9
LPT	8.4M	86.2	90.1	89.5	92.7	81.5	9.0
Houlsby-Adapter	9.5M	86.5	90.3	89.7	92.9	81.8	9.1
Learned-Adapter	9.5M	86.9	90.5	90.0	93.4	<u>84.3</u>	9.3
LoRA	10.0M	86.7	90.8	90.2	93.5	82.3	9.2
AdaLoRA	10.0M	87.1	91.0	<u>91.8</u>	93.6	82.7	9.2
(IA) <sup>3</sup>	9.8M	86.6	90.6	90.1	93.2	82.0	<u>9.4</u>
PEDRO	8.9M	88.1	<u>92.3</u>	91.7	<u>94.7</u>	84.2	9.3
WizardLM	7B	87.6	92.4	91.3	94.3	84.6	9.3
NeFTune	7B	86.1	90.4	89.8	93.1	82.3	9.0
SymNoise	7B	86.3	90.6	90.0	93.2	82.2	9.3
RobustFT	7B	87.2	91.1	90.6	93.8	83.0	9.2
<b>DCPC (ours)</b>	9.6M	<b>88.9</b>	<b>93.5</b>	<b>92.2</b>	<b>95.0</b>	<b>84.7</b>	<b>9.5</b>

LoRA [Hu et al., 2021] and its variant AdaLoRA [Zhang et al., 2023b], which use low-rank adaptation matrices with dynamic pruning. To gauge robustness against mutually conflicting instructions, we add WizardLM curriculum tuning [Xu et al., 2023]. To benchmark noise tolerance, we include: NeFTune [Jain et al., 2024], SymNoise [Yadav and Singh, 2023], and RobustFT [Luo et al., 2024]. For a detailed overview of the baseline, please refer to D.4.

For our main experiments, we fine-tune the LLaMA-2 models[Touvron et al., 2023], specifically the LLaMA-2 7B and LLaMA-2 13B models. For more details about the implementations and evaluation metrics, please refer to D.

## 4.2 Main results

The experimental results on both the original and modified datasets are shown in Table 1 and Table 2, respectively.

**Performance on Original Datasets** As shown in Table 1, DCPC consistently surpasses all baselines on the original datasets. It achieves the highest accuracy on **BoolQ** (88.9%), **COPA** (93.5%), **ReCoRD** (92.2%), **SST-2** (95.0%), and **RTE** (84.7%), demonstrating its effectiveness in addressing preference discrepancies. DCPC also attains the best **gpt-score** of 9.5 on Alpaca, highlighting its superiority in instruction-following tasks.

**Performance on Modified Datasets** Table 2 presents DCPC’s performance on modified datasets with introduced preference shifts and biases. DCPC remains robust, outperforming all baselines. It achieves the highest accuracy on **BoolQ-PS** (86.1%), **COPA-BS** (91.7%), **ReCoRD-R** (91.9%), **SST-2-P** (92.8%), **RTE-E** (83.7%), and **Alpaca-IS** (9.4). Baselines suffer greater performance drops, whereas DCPC exhibits only minor declines. For instance, Full-FT drops from 88.6% to 82.4% on **BoolQ-PS**, while DCPC declines modestly from 88.9% to 86.1%. Similarly, on **COPA-BS**, Full-FT falls from 91.5% to 88.5%, while DCPC maintains 91.7%. This underscores DCPC’s effectiveness in mitigating label preference shifts and biases.

We provide additional experimental results comparing the performance of DCPC with methods designed for learning from inconsistent or noisy labels, specifically Majority Voting (MV), RSVMI [Yang et al., 2023], LAWMV [Chen et al., 2022], AALI [Zheng et al., 2021], co-teaching [Han et al., 2018], NoiseBox [Feng et al., 2024], and the Label-Retrieval-Augmented (LRA) diffusion model [Chen et al., 2024a] on the modified datasets. As shown in Figure 4, DCPC outperforms all the compared baseline methods. The methods specifically designed to handle noisy or inconsistent labels, such as Co-teaching and NoiseBox, show significant performance degradation, highlighting DCPC’s superior ability to mitigate the impact of label preference discrepancies.

Table 2: Performance comparison of DCPC and baseline methods on *modified* datasets. Results are the median of five random seeds. Backbone: *LLaMA-2 7B*. Bold and underlined numbers denote the best and second-best results, respectively.

Method	Tunable Params	BoolQ-PS (acc)	COPA-BS (acc)	ReCoRD-R (f1-em)	SST-2-P (acc)	RTE-E (acc)	Alpaca-IS (gpt-score)
Full-FT	7B	82.4	88.5	88.4	90.1	80.7	8.7
P-Tuning v2	9.4M	78.0	86.1	85.9	87.5	77.9	8.4
LPT	8.4M	78.5	86.4	86.2	87.8	78.3	8.5
Houlsby-Adapter	9.5M	78.9	86.9	86.5	86.4	78.6	8.6
Learned-Adapter	9.5M	79.2	86.8	87.1	88.3	78.9	<u>8.7</u>
LoRA	10.0M	79.1	86.9	86.9	88.5	79.1	8.6
AdaLoRA	10.0M	79.4	87.1	87.0	88.7	79.2	8.6
(IA) <sup>3</sup>	9.8M	79.0	87.0	86.8	88.6	79.0	8.5
PEDRO	8.9M	79.1	87.5	87.5	88.1	79.7	8.6
WizardLM (Curriculum)	7B	<u>82.9</u>	<u>89.8</u>	<u>89.9</u>	<u>91.3</u>	<u>81.8</u>	<u>8.9</u>
NeFTune	7B	80.2	87.3	86.5	89.1	79.6	8.6
SymNoise	7B	80.4	87.4	86.7	89.2	79.7	8.6
RobustFT	7B	81.4	88.2	88.0	90.4	80.5	8.8
<b>DCPC (ours)</b>	<b>9.6M</b>	<b>86.1</b>	<b>91.7</b>	<b>91.9</b>	<b>92.8</b>	<b>83.7</b>	<b>9.4</b>

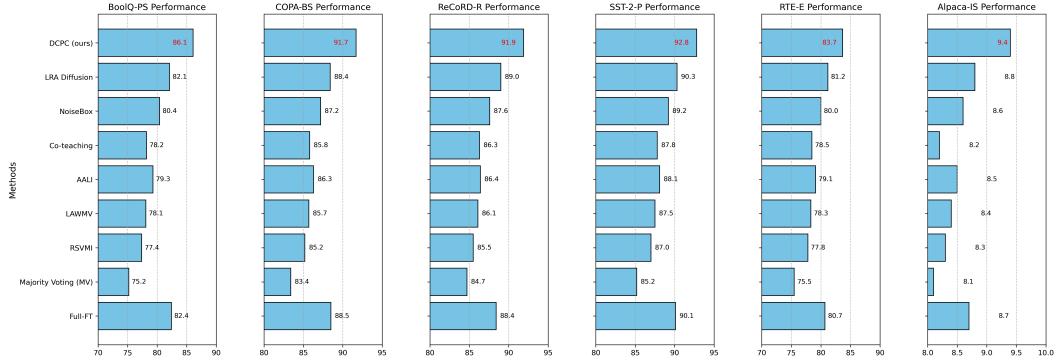


Figure 4: Performance comparison of DCPC with existing noisy label learning and inconsistent label handling methods on modified datasets. Results are median performance across five random seeds. The backbone is LLaMA-2 7B.

### 4.3 Ablation Study

To evaluate the contribution of each component in the Dynamic Cross-Layer Preference Correction (DCPC) framework, we perform an ablation study by disabling key components individually and testing on both original and modified datasets. We examine four ablated variants: 1) **DCPC w/o CLPA**: Removes Cross-Layer Prefix Alignment (CLPA), assessing the model’s ability to handle preference discrepancies without cross-layer alignment. 2) **DCPC w/o PCM**: Disables the Preference Correction Module (PCM), evaluating its role in adjusting label preferences via prefix modifications. 3) **DCPC w/o Ambiguity Loss**: Excludes ambiguity loss to measure the impact of removing explicit minimization of semantic similarity-based label discrepancies. 4) **DCPC w/o CLPA & PCM**: Removes both CLPA and PCM, leaving only ambiguity loss, serving as a minimal DCPC variant akin to standard fine-tuning with ambiguity-aware adjustments. Table 3 presents the results.

**Ablation study of DCPC** The ablation study results in Table 3 highlight the critical contributions of each DCPC component. Removing Cross-Layer Prefix Alignment (CLPA) leads to a noticeable drop in performance, especially on ReCoRD-R (-3.4 f1-em) and BoolQ-PS (-3.4

Table 4: Backbone model ablation study.

Backbone	Params	BoolQ-PS (acc)	COPA-BS (acc)	ReCoRD-R (f1-em)
LlaMA-2 7B	7B	<u>86.1</u>	91.7	<b>91.9</b>
LlaMA-2 13B	13B	<b>86.4</b>	<b>92.0</b>	91.7
Mistral-7B	7B	85.7	<u>91.8</u>	<u>91.8</u>

Table 3: Ablation Study: Performance comparison of DCPC with different components disabled. Results are median performance across five random seeds. The backbone is LLaMA-2 7B. Bold and underlined values represent the best and second-best results, respectively. The values in parentheses represent the performance drop compared to the full DCPC model.

Method	BoolQ-PS (acc)	COPA-BS (acc)	ReCoRD-R (f1-em)	SST-2-P (acc)	RTE-E (acc)	Alpaca-IS (gpt-score)
DCPC (Full)	<b>86.1</b>	<b>91.7</b>	<b>91.9</b>	<b>92.8</b>	<b>83.7</b>	<b>9.4</b>
DCPC w/o CLPA	82.7 (-3.4)	89.0 (-2.7)	88.5 (-3.4)	90.0 (-2.8)	80.8 (-2.9)	8.9 (-0.5)
DCPC w/o PCM	81.2 (-4.9)	88.5 (-3.2)	87.0 (-4.9)	89.5 (-3.3)	79.1 (-4.6)	8.9 (-0.5)
DCPC w/o Ambiguity Loss	80.0 (-6.1)	87.1 (-4.6)	88.0 (-3.9)	89.2 (-3.6)	80.0 (-3.7)	8.8 (-0.6)
DCPC w/o CLPA & PCM	78.5 (-7.6)	86.5 (-5.2)	87.3 (-4.6)	88.7 (-4.1)	79.5 (-4.2)	8.7 (-0.7)

acc), showing CLPA’s importance in maintaining consistency across layers. The Preference Correction Module (PCM) is equally vital, with its removal causing a 4.9-point accuracy drop on BoolQ-PS and 4.6 points on RTE-E, underscoring its role in correcting preference discrepancies. Disabling ambiguity loss results in a sharper decline (e.g., -6.1 acc on BoolQ-PS), indicating its key role in reducing label inconsistencies. The largest performance decrease occurs when both CLPA and PCM are disabled, with a 7.6-point drop on BoolQ-PS and 5.2 points on COPA-BS, confirming the combined effectiveness of CLPA, PCM, and ambiguity loss.

**Ablation on the pretrained backbones** We investigate the impact of different backbone models on the performance of the proposed DCPC framework. As shown in Table 4, the performance of DCPC remains robust across all backbone models, with LLaMA-2 13B achieving the highest overall accuracy in the BoolQ-PS and COPA-BS datasets.

#### 4.4 Robustness Analysis

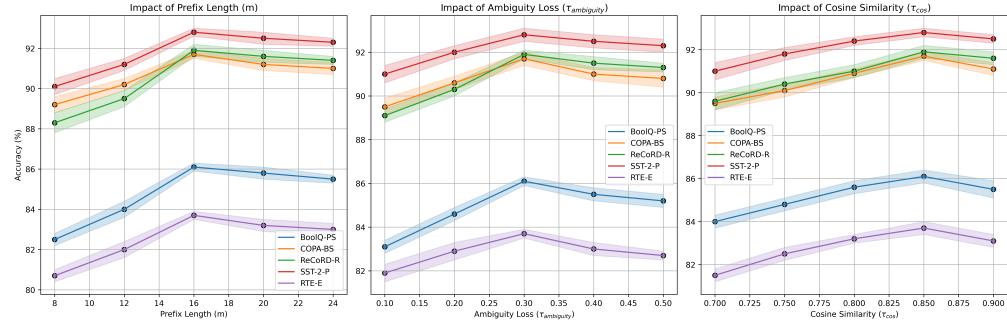


Figure 5: Impact of different hyperparameters on the performance of DCPC across multiple datasets. Subfigures show the effect of (a) prefix length ( $m$ ), (b) ambiguity loss threshold ( $\tau_{\text{ambiguity}}$ ), and (c) cosine similarity threshold ( $\tau_{\text{cos}}$ ) on five datasets.

**Prefix Length ( $m$ ):** The prefix length  $m$  determines the dimensionality of prefix embeddings in each transformer layer. To assess its impact, we vary  $m$  from 8 to 24 and analyze performance across datasets. Figure 5 shows that performance improves with increasing  $m$  until saturation at  $m = 16$ . Beyond this point, it stagnates or slightly declines, indicating that excessively long prefixes may introduce noise and hinder the model’s ability to capture meaningful preference shifts.

**Ambiguity Loss Threshold ( $\tau_{\text{ambiguity}}$ ):** The ambiguity loss threshold  $\tau_{\text{ambiguity}}$  determines when the Preference Correction Module (PCM) is triggered to correct label discrepancies. We experiment with  $\tau_{\text{ambiguity}}$  values ranging from 0.1 to 0.5. As shown in Figure 5, a moderate value of  $\tau_{\text{ambiguity}} = 0.3$  yields the best performance. Lower thresholds (e.g.,  $\tau_{\text{ambiguity}} = 0.1$ ) result in frequent activations of the PCM, potentially over-correcting minor discrepancies, while higher thresholds (e.g.,  $\tau_{\text{ambiguity}} = 0.5$ ) reduce the corrective impact of the PCM, leading to larger inconsistencies in the final predictions.

**Cosine Similarity Threshold ( $\tau_{\text{cos}}$ ):** The cosine similarity threshold  $\tau_{\text{cos}}$  is critical for determining when embeddings are considered semantically similar enough to trigger the ambiguity loss. We vary  $\tau_{\text{cos}}$  from 0.7 to 0.95 to assess its impact on performance. Figure 5 shows that setting  $\tau_{\text{cos}} = 0.85$

achieves optimal results. Lower values result in too many similarity comparisons being treated as high, leading to unnecessary corrective actions, while higher values decrease the number of corrective interventions, reducing the overall effectiveness of the framework.

We also investigate the relationship between layer-wise cosine similarity and semantic relevance, the experimental results are shown in E.2.

## 5 Conclusion

Instruction fine-tuning breaks down when annotator styles clash. We present DCPC, a label-free PEFT method that detects high-similarity / low-agreement pairs, realigns their prefixes across layers, and corrects residual bias with a lightweight PCM. Across six preference-shifted benchmarks, DCPC delivers up to +6.7 accuracy/F1-EM and trims variance by 35% over strong baselines. Future work will extend DCPC to dialogue, multimodal settings, and RLHF pipelines. Another interesting direction is to extend DCPC to the setting of long-tailed data distributions[Xiao et al., 2025b] and other multi-modal application scenarios[Zhang et al., 2024].

## Acknowledgements

This work received technical support from Hunan Airon Technology Co., Ltd., including data collection, data annotation, and preprocessing. In addition, we also thank the company for providing computing resources for this research.

## References

- Y. Bai, S. Kadavath, S. Kundu, A. Askell, J. Kernion, A. Jones, A. Chen, A. Goldie, A. Mirhosseini, C. McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.
- J. Chen, R. Zhang, T. Yu, R. Sharma, Z. Xu, T. Sun, and C. Chen. Label-retrieval-augmented diffusion models for learning from noisy labels. *Advances in Neural Information Processing Systems*, 36, 2024a.
- X. Chen, C. Xiao, and Y. Liu. Confusion-resistant federated learning via diffusion-based data harmonization on non-iid data. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, pages 137495–137520, 2024b.
- X. Chen, C. Xiao, W. Cao, W. Zhang, and Y. Liu. Framework and pathway for the construction of a unified data-element market in china. *Strategic Study of Chinese Academy of Engineering*, 27(1): 40–50, 2025.
- Z. Chen, L. Jiang, and C. Li. Label augmented and weighted majority voting for crowdsourcing. *Information Sciences*, 606:397–409, 2022.
- J. Cheng and D. Cosley. How annotation styles influence content and preferences. In *Proceedings of the 24th ACM Conference on Hypertext and Social Media*, pages 214–218, 2013.
- H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, Y. Li, X. Wang, M. Dehghani, S. Brahma, et al. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53, 2024.
- J. Edstedt, G. Bökman, and Z. Zhao. Dedode v2: Analyzing and improving the dedode keypoint detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4245–4253, 2024.
- C. Feng, G. Tzimiropoulos, and I. Patras. Noisebox: Towards more efficient and effective learning with noisy labels. *IEEE Transactions on Circuits and Systems for Video Technology*, 2024.
- S. Ghosh, C. K. R. Evuru, S. Kumar, D. Aneja, Z. Jin, R. Duraiswami, D. Manocha, et al. A closer look at the limitations of instruction tuning. *arXiv preprint arXiv:2402.05119*, 2024.

- B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in neural information processing systems*, 31, 2018.
- Q. He and C. Qu. Waste-to-energy-coupled ai data centers: Cooling efficiency and grid resilience. *arXiv preprint arXiv:2512.24683*, 2025.
- O. Honovich, T. Scialom, and et al. Unnatural instructions: Tuning language models with {Almost} no human supervision. *arXiv preprint arXiv:2212.09689*, 2022.
- N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pages 2790–2799. PMLR, 2019.
- E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.
- W. Hu, W. Zhang, Y. Jiang, C. J. Zhang, X. Wei, and L. Qing. Removal of hallucination on hallucination: Debate-augmented RAG. In W. Che, J. Nabende, E. Shutova, and M. T. Pilehvar, editors, *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15839–15853, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.770. URL <https://aclanthology.org/2025.acl-long.770/>.
- N. Jain, P.-y. Chiang, Y. Wen, J. Kirchenbauer, H.-M. Chu, G. Somepalli, B. R. Bartoldson, B. Kailkhura, A. Schwarzschild, A. Saha, et al. Neftune: Noisy embeddings improve instruction finetuning. In *The Twelfth International Conference on Learning Representations*, 2024.
- A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. I. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.
- L. Jiang, X. Wang, F. Zhang, and C. Zhang. Transforming time and space: efficient video super-resolution with hybrid attention and deformable transformers. *The Visual Computer*, pages 1–12, 2025.
- Y. Jiang, W. Zhang, X. Zhang, X.-Y. Wei, C. W. Chen, and Q. Li. Prior knowledge integration via llm encoding and pseudo event regulation for video moment retrieval. In *Proceedings of the 32nd ACM International Conference on Multimedia*, MM ’24, page 7249–7258, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400706868. doi: 10.1145/3664647.3681115. URL <https://doi.org/10.1145/3664647.3681115>.
- S. Kaur, S. Park, A. Goyal, and S. Arora. Instruct-skillmix: A powerful pipeline for llm instruction tuning. *arXiv preprint arXiv:2408.14774*, 2024.
- Z. Ke, J. Shen, X. Zhao, X. Fu, Y. Wang, Z. Li, L. Liu, and H. Mu. A stable technical feature with gru-cnn-ga fusion. *Applied Soft Computing*, page 114302, 2025.
- G. Lan, H. A. Inan, S. Abdelnabi, J. Kulkarni, L. Wutschitz, R. Shokri, C. G. Brinton, and R. Sim. Contextual integrity in LLMs via reasoning and reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2025a.
- G. Lan, S. Zhang, T. Wang, Y. Zhang, D. Zhang, X. Wei, X. Pan, H. Zhang, D.-J. Han, and C. G. Brinton. Mappo: Maximum a posteriori preference optimization with prior knowledge. *arXiv preprint arXiv:2507.21183*, 2025b.
- Y. Li, Z. Zhou, Z. Peng, J. Dong, H. You, R. Yan, S. Wen, Y. Tian, and T. Huang. A preference-driven methodology for efficient code generation. *IEEE Transactions on Artificial Intelligence*, 2025. doi: 10.1109/TAI.2025.3639527.
- Z. Lin, K. Zhao, S. Zhang, P. Yu, and C. Xiao. Cec-zero: Zero-supervision character error correction with self-generated rewards. *arXiv preprint arXiv:2512.23971*, 2025.
- H. Liu and L. Xiao. Re-grpo: Leveraging hard negative cases through large language model guided self training. *Neurocomputing*, page 132543, 2025.

- H. Liu, D. Tam, M. Muqeeth, J. Mohta, T. Huang, M. Bansal, and C. A. Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965, 2022a.
- J. Liu, T. Wang, S. Liu, X. Hu, R. Tong, L. Wang, and J. Xu. Lightweight baselines for medical abstract classification: Distilbert with cross-entropy as a strong default. *arXiv preprint arXiv:2510.10025*, 2025.
- X. Liu, K. Ji, Y. Fu, W. L. Tam, Z. Du, Z. Yang, and J. Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*, 2021.
- X. Liu, T. Sun, X. Huang, and X. Qiu. Late prompt tuning: A late prompt could be better than many prompts. *arXiv preprint arXiv:2210.11292*, 2022b.
- H. Lu, Z. Liu, G. Jiang, Y. Luo, S. Chen, Y. Zhang, and Y.-C. Chen. Uniugp: Unifying understanding, generation, and planning for end-to-end autonomous driving. *ByteDance Seed Tech Report*, 2025a.
- Y. Lu, J. Zhaiyuan, J. Du, Y. Shanqing, Q. Xuan, and J. T. Zhou. Autoannotator: A collaborative annotation framework for large and small language models. *Transactions on Machine Learning Research*, 2025b.
- J. Luo, X. Luo, K. Ding, J. Yuan, Z. Xiao, and M. Zhang. Robustft: Robust supervised fine-tuning for large language models under noisy response. *arXiv preprint arXiv:2412.14922*, 2024.
- A. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150, 2011.
- H. Miao, X. Zhong, J. Liu, Y. Zhao, X. Zhao, W. Qian, K. Zheng, and C. S. Jensen. Task assignment with efficient federated preference learning in spatial crowdsourcing. *IEEE Transactions on Knowledge and Data Engineering*, 36(4):1800–1814, 2023.
- D. Mueller, M. Dredze, and N. Andrews. Multi-task transfer matters during instruction-tuning. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 14880–14891, 2024.
- N. Natarajan, I. S. Dhillon, P. K. Ravikumar, and A. Tewari. Learning with noisy labels. *Advances in neural information processing systems*, 26, 2013.
- H. Naveed, A. U. Khan, S. Qiu, M. Saqib, S. Anwar, M. Usman, N. Akhtar, N. Barnes, and A. Mian. A comprehensive overview of large language models. *arXiv preprint arXiv:2307.06435*, 2023.
- P. Ramachandran and et al. Doremi: Optimizing data mixtures speeds up language model pretraining. *arXiv preprint arXiv:2305.11543*, 2023.
- M. Ren, B. Cao, H. Lin, C. Liu, X. Han, K. Zeng, G. Wan, X. Cai, and L. Sun. Learning or self-aligning? rethinking instruction fine-tuning. *arXiv preprint arXiv:2402.18243*, 2024.
- V. Sanh, A. Webson, C. Raffel, and et al. Multitask prompted training enables zero-shot generalisation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.
- M. Shanahan. Talking about large language models. *Communications of the ACM*, 67(2):68–79, 2024.
- Q. Shen and J. Zhang. Ai-enhanced disaster risk prediction with explainable shap analysis: A multi-class classification approach using xgboost, December 2025. URL <https://www.researchsquare.com/article/rs-8437180/v1>. Preprint, Version 1, posted December 31, 2025.
- Z. Shi, A. Yang, B. Wu, L. Aitchison, E. Yilmaz, and A. Lipani. Instruction tuning with loss over instructions. *Advances in Neural Information Processing Systems*, 37:69176–69205, 2024.
- H. Su, W. Shi, J. Kasai, Y. Wang, Y. Hu, M. Ostendorf, W.-t. Yih, N. A. Smith, L. Zettlemoyer, and T. Yu. One embedder, any task: Instruction-finetuned text embeddings. *arXiv preprint arXiv:2212.09741*, 2022.

- Q. Sun and J. Li. A lightweight yolov4-svm model for automated waste monitoring in smart cities. In *Proceedings of the 2025 2nd International Conference on Intelligent Computing and Data Mining (ICDM)*, pages 32–36. IEEE, 2025. doi: 10.1109/ICDM68174.2025.11309516. URL <https://ieeexplore.ieee.org/document/11309516>.
- W. Sun, Q. Shen, Y. Gao, Q. Mao, T. Qi, and S. Xu. Objective over architecture: Fraud detection under extreme imbalance in bank account opening. *Computation*, 13(12):290, 2025. doi: 10.3390/computation13120290. URL <https://www.mdpi.com/2079-3197/13/12/290>.
- H. Tan, J. Jiang, and J. Shen. Profix: Improving profile-guided optimization in compilers with graph neural networks. In *Advances in Neural Information Processing Systems*, 2025. URL <https://neurips.cc/virtual/2025/poster/119293>.
- H. Tao, J. Li, Z. Hua, and F. Zhang. Dudb: deep unfolding-based dual-branch feature fusion network for pan-sharpening remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing*, 62:1–17, 2023.
- R. Taori, Y. Bai, and et al. Alpaca: A strong, replicable instruction-following model. arXiv preprint arXiv:2304.03265, 2023a.
- R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto. Stanford alpaca: An instruction-following llama model, 2023b.
- R. Tong, J. Liu, S. Liu, X. Hu, and L. Wang. Renaissance of rnns in streaming clinical time series: Compact recurrence remains competitive with transformers. *arXiv preprint arXiv:2510.16677*, 2025.
- H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.
- A. Wang. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.
- A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32, 2019.
- H. Wang and F. Zhang. Computing nodes for plane data points by constructing cubic polynomial with constraints. *Computer Aided Geometric Design*, 111:102308, 2024.
- Y. Wang, H. Wang, and F. Zhang. A medical image segmentation model with auto-dynamic convolution and location attention mechanism. *Computer Methods and Programs in Biomedicine*, 261: 108593, 2025a.
- Z. Wang, Y. Sun, H. Wang, B. Jing, X. Shen, X. Dong, Z. Hao, H. Xiong, and Y. Song. Reasoning-enhanced domain-adaptive pretraining of multimodal large language models for short video content governance. In S. Potdar, L. Rojas-Barahona, and S. Montella, editors, *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1104–1112, Suzhou (China), Nov. 2025b. Association for Computational Linguistics. ISBN 979-8-89176-333-3. doi: 10.18653/v1/2025.emnlp-industry.77. URL <https://aclanthology.org/2025.emnlp-industry.77>.
- J. Wei, M. Bosma, V. Zhao, and et al. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations (ICLR)*, 2022.
- J. Wen, J. Cui, Z. Zhao, R. Yan, Z. Gao, L. Dou, and B. M. Chen. Syreanet: A physically guided under-water image enhancement framework integrating synthetic and real images. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5177–5183, 2023.
- C. Xiao and Y. Liu. A multifrequency data fusion deep learning model for carbon price prediction. *Journal of Forecasting*, 44(2):436–458, 2025.

- C. Xiao, L. Hou, L. Fu, and W. Chen. Diffusion-based self-supervised imitation learning from imperfect visual servoing demonstrations for robotic glass installation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10401–10407. IEEE, 2025a.
- C. Xiao, C. Zhao, Z. Ke, and F. Shen. Curiosity meets cooperation: A game-theoretic approach to long-tail multi-label learning. *arXiv preprint arXiv:2510.17520*, 2025b.
- T. Xie, T. Li, W. Zhu, W. Han, and Y. Zhao. Pedro: Parameter-efficient fine-tuning with prompt dependent representation modification. *arXiv preprint arXiv:2409.17834*, 2024.
- C. Xu, Q. Sun, K. Zheng, X. Geng, P. Zhao, J. Feng, C. Tao, and D. Jiang. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*, 2023.
- M. Xu, H. Liu, N. Peng, Q. Chen, and C. Xiao. Affordance-first decomposition for continual learning in video-language understanding. *arXiv preprint arXiv:2512.00694*, 2025a.
- Z. Xu, X. Zhang, R. Li, Z. Tang, Q. Huang, and J. Zhang. Fakeshield: Explainable image forgery detection and localization via multi-modal large language models. In *International Conference on Learning Representations*, 2025b.
- A. K. Yadav and A. Singh. Symnoise: Advancing language model fine-tuning with symmetric noise. *arXiv preprint arXiv:2312.01523*, 2023.
- S. Yan, Y. Wang, K. Zhao, P. Shi, Z. Zhao, Y. Zhang, and J. Li. Hemora: Unsupervised heuristic consensus sampling for robust point cloud registration. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1363–1373, 2025.
- W. Yang, C. Li, and L. Jiang. Learning from crowds with robust support vector machines. *Science China Information Sciences*, 66(3):132103, 2023.
- J. Yao, C. Li, K. Sun, Y. Cai, H. Li, W. Ouyang, and H. Li. Ndc-scene: Boost monocular 3d semantic scene completion in normalized device coordinates space. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9421–9431. IEEE Computer Society, 2023.
- J. Yao, C. Li, and C. Xiao. Swift sampler: Efficient learning of sampler by 10 parameters. *Advances in Neural Information Processing Systems*, 37:59030–59053, 2024.
- E. B. Zaken, S. Ravfogel, and Y. Goldberg. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *arXiv preprint arXiv:2106.10199*, 2021.
- F. Zhang, G. Chen, H. Wang, J. Li, and C. Zhang. Multi-scale video super-resolution transformer with polynomial approximation. *IEEE Transactions on Circuits and Systems for Video Technology*, 33(9):4496–4506, 2023a.
- F. Zhang, G. Chen, H. Wang, and C. Zhang. Cf-dan: Facial-expression recognition based on cross-fusion dual-attention network. *Computational Visual Media*, 10(3):593–608, 2024.
- Q. Zhang, M. Chen, A. Bukharin, N. Karampatziakis, P. He, Y. Cheng, W. Chen, and T. Zhao. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512*, 2023b.
- S. Zhang, L. Dong, X. Li, S. Zhang, X. Sun, S. Wang, J. Li, R. Hu, T. Zhang, F. Wu, et al. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792*, 2023c.
- X. Zhang, Z. Tang, Z. Xu, R. Li, Y. Xu, B. Chen, F. Gao, and J. Zhang. Omniduard: Hybrid manipulation localization via augmented versatile deep image watermarking. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 3008–3018, 2025a.
- X. Zhang, F. Zeng, Y. Quan, Z. Hui, and J. Yao. Enhancing multimodal large language models complex reason via similarity computation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39(10), pages 10203–10211, 2025b.
- Y. Zhang, P. Wang, M. Tan, and W. Zhu. Learned adapters are better than manually designed adapters. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7420–7437, 2023d.

- Z. Zhang, S. Li, Z. Zhang, X. Liu, H. Jiang, X. Tang, Y. Gao, Z. Li, H. Wang, Z. Tan, et al. Iheval: Evaluating language models on following the instruction hierarchy. *arXiv preprint arXiv:2502.08745*, 2025c.
- Y. Zhao, K. Zheng, H. Yin, G. Liu, J. Fang, and X. Zhou. Preference-aware task assignment in spatial crowdsourcing: from individuals to groups. *IEEE Transactions on Knowledge and Data Engineering*, 34(7):3461–3477, 2020.
- C. Zheng, L. Pan, and P. Wu. Attribute augmented network embedding based on generative adversarial nets. *IEEE Transactions on Neural Networks and Learning Systems*, 34(7):3473–3487, 2021.
- Y. Zhou, Z. Zhao, D. Cheng, Z. Wu, J. Gui, Y. Yang, F. Wu, Y. Cheng, and H. Fan. Dropping experts, recombining neurons: Retraining-free pruning for sparse mixture-of-experts llms. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 15169–15186, 2025.
- K. Zhu, Q. Zhao, H. Chen, J. Wang, and X. Xie. Promptbench: A unified library for evaluation of large language models. *Journal of Machine Learning Research*, 25(254):1–22, 2024.

## NeurIPS Paper Checklist

### 1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification: The abstract and introduction clearly and accurately present our contributions. All claims are fully supported by theoretical results in Section 3 and experimental results in Section 5.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

### 2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[Yes\]](#)

Justification: We state that our implementation still relies on a fixed overlap schedule and an extra forward pass per player.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

### 3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [\[Yes\]](#)

Justification: All assumptions required for our theoretical results are explicitly stated. Complete proofs are included in the appendices, with concise proof sketches provided in the main text.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

#### 4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [\[Yes\]](#)

Justification: The paper provides detailed experimental settings.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

#### 5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: **[No]**

Justification: Due to institutional restrictions and proprietary considerations, the data and code used in this study are not publicly available at this time. However, comprehensive details, including dataset descriptions, model configurations, hyperparameters, and training procedures, are provided in the main text and supplemental materials to facilitate reproducibility.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

## 6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: **[Yes]**

Justification: The paper explicitly describes datasets, choice of hyperparameters, optimization methods, and computational settings in the experimental sections and appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

## 7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: **[Yes]**

Justification: The paper reports experimental results with clearly defined error bars, calculated as the standard deviation across multiple independent runs.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer “Yes” if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

## 8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: **[Yes]**

Justification: The paper clearly specifies the computational resources utilized, including GPU type, memory requirements, execution time per run, and overall compute needed for each experimental setting.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

## 9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: **[Yes]**

Justification: We have thoroughly reviewed the NeurIPS Code of Ethics and confirm that our research fully complies with the guidelines.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

## 10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: **[NA]** .

Justification: There is no societal impact of the work performed.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

## 11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA].

Justification: The paper poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

## 12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All datasets and models used in our experiments are properly credited with citations to their original sources.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, [paperswithcode.com/datasets](http://paperswithcode.com/datasets) has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

### 13. New assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA].

Justification: The paper does not release new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

### 14. Crowdsourcing and research with human subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

### 15. Institutional review board (IRB) approvals or equivalent for research with human subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

#### 16. Declaration of LLM usage

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: In this work, LLMs were employed solely for improving language clarity.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.

## A Toy Experiment: Exploring Label Preference Inconsistencies in Similar Input Embeddings

The goal of this toy experiment is to investigate how semantically similar input sequences can lead to different label preference distributions under the P-Tuning v2 framework. We aim to explore whether prefix embeddings can effectively capture label preferences across similar inputs, and how inconsistencies arise.

### A.1 Dataset Preparation

**Dataset Selection** We use the IMDB sentiment classification dataset[Maas et al., 2011], where the sentiment labels (positive, neutral, negative) are often influenced by annotator preferences. This dataset is ideal for exploring the discrepancies in label preferences under P-Tuning v2.

**Sample Selection** Two semantically similar review pairs are chosen:

- **Review A:** "The movie was enjoyable but not amazing." (Positive sentiment)
- **Review B:** "The film was okay, but nothing special." (Neutral sentiment)

These reviews have similar semantic meaning but are assigned different sentiment labels.

**Label Distributions** We assume that for each input, the model generates a soft sentiment label distribution (e.g., probabilities of positive, neutral, and negative sentiment) instead of a hard label. These distributions represent the model’s predicted preferences for each input sequence, which is influenced by the optimized prefix embeddings learned under P-Tuning v2.

### A.2 Experiment Setup

**Layer-wise Embedding Calculation** For each review,  $x_A$  and  $x_B$ , we extract layer-wise embeddings  $\mathbf{e}_A^l \in \mathbb{R}^d$  and  $\mathbf{e}_B^l \in \mathbb{R}^d$  from a pre-trained transformer model (e.g., BERT), where  $d = 768$  represents the embedding dimension. In P-Tuning v2, task-specific prefix embeddings are inserted into each transformer layer, and the embeddings  $\mathbf{e}_A^l$  and  $\mathbf{e}_B^l$  include the influence of these prefix embeddings.

**Label Preference Distribution** At each layer  $l$ , the model with P-Tuning v2 computes the label preference distributions for both inputs using the softmax function over the model’s output logits:

$$p(\mathbf{e}_A^l) = \text{Softmax}(f(\mathbf{e}_A^l)), \quad p(\mathbf{e}_B^l) = \text{Softmax}(f(\mathbf{e}_B^l)) \quad (15)$$

where  $f(\mathbf{e}_A^l)$  and  $f(\mathbf{e}_B^l)$  represent the logits for sentiment prediction at layer  $l$ . The resulting softmax outputs represent the predicted probability distributions over sentiment categories, which reflect how well the task-specific prefix embeddings capture label preferences.

**KL-Divergence Calculation** We measure the divergence between the predicted label distributions for the two inputs at each layer using KL-Divergence:

$$L_{\text{KL}}(p_A, p_B) = D_{\text{KL}}(p(\mathbf{e}_A^l) \| p(\mathbf{e}_B^l)) \quad (16)$$

This quantifies how much the predicted label distributions for the two inputs deviate, even though their embeddings remain similar. The goal is to assess how much P-Tuning v2’s prefix embeddings contribute to such discrepancies in label preferences.

### A.3 Quantitative Analysis: Layer-wise Embedding and Preference Distribution Changes

We analyze the layer-wise cosine similarity of the embeddings, the edit distance, and the KL-Divergence of the label preference distributions. The results are summarized in Table 5, demonstrating how prefix embeddings under P-Tuning v2 influence the emergence of preference inconsistencies.

Table 5: Layer-wise Cosine Similarity, Edit Distance, KL-Divergence, and Label Prediction Differences

Layer $l$	Cosine Similarity	Edit Distance	KL-Divergence	Label Difference (Prediction)
1	0.98	10	0.05	0
2	0.96	15	0.07	0
3	0.94	25	0.12	0
4	0.90	35	0.15	0
5	0.87	50	0.22	0
6	0.82	70	0.30	1
7	0.76	85	0.38	1
8	0.65	110	0.45	1
9	0.52	140	0.55	1
10	0.40	170	0.62	1
11	0.28	190	0.70	1
12	0.15	210	0.78	1

## B Proof of Theorem 1

*Proof.* We proceed in two stages: (i) derive the effect of one gradient update on the prefix difference, and (ii) propagate this change through the next Lipschitz-bounded Transformer block.

**Gradient update on prefixes.** Define the prefix difference  $\theta^l = \mathbf{P}_A^l - \mathbf{P}_B^l$ . From Eq. (8),

$$\nabla_{\mathbf{P}_A^l} L_{\text{align}} = 2(\mathbf{C}_A^l - \mathbf{C}_B^l), \quad \nabla_{\mathbf{P}_B^l} L_{\text{align}} = -2(\mathbf{C}_A^l - \mathbf{C}_B^l).$$

A simultaneous gradient descent step with learning rate  $\eta$  gives

$$\theta_{\text{new}}^l = \theta^l - 4\eta(\mathbf{C}_A^l - \mathbf{C}_B^l).$$

Hence

$$\begin{aligned} \|\theta_{\text{new}}^l\|_2 &= |1 - 4\eta| \|\mathbf{C}_A^l - \mathbf{C}_B^l\|_2 \\ &= |1 - 4\eta| d^l. \end{aligned} \tag{A.1}$$

**Propagation through the next block.** Let  $f^{l+1}$  be the  $(l+1)$ -st block. Using the  $L$ -Lipschitz property,

$$\begin{aligned} d^{l+1} &= \|f^{l+1}(\mathbf{C}_A^l) - f^{l+1}(\mathbf{C}_B^l)\|_2 \\ &\leq L \|\mathbf{C}_A^l - \mathbf{C}_B^l\|_2^{\text{new}} = L \|\theta_{\text{new}}^l\|_2 \\ &\stackrel{(A.1)}{=} L |1 - 4\eta| d^l. \end{aligned} \tag{A.2}$$

**Optimising the contraction factor.** Define  $\rho(\eta) = L|1 - 4\eta|$ , which is convex on  $\eta \in (0, \frac{1}{4})$  and minimised at  $\eta^* = \frac{1-L}{4}$ . Substituting  $\eta^*$  into  $\rho(\eta)$  yields

$$\rho_{\min} = L(1 - (1 - L)) = L^2 < 1.$$

Choosing  $\eta^*$  in Eq. (A.2) therefore gives the bound  $d^{l+1} \leq L^2 d^l$ , proving Eq. (9). Repeated application over  $k$  layers produces the geometric decay  $d^{l+k} \leq L^{2k} d^l$ , and Cauchy convergence implies a unique fixed point.  $\square$

## C Proof of Theorem 2

*Proof.* Let  $\ell(h, (x, y)) = \mathbb{I}[h(x) \neq y]$  be the 0-1 loss and  $\tilde{\ell}(h, (x, \tilde{y}))$  its noisy counterpart. Following the unbiased CCN correction of Natarajan et al. [2013], define:

$$\bar{\ell}(h, (x, \tilde{y})) = \frac{\tilde{\ell}(h, (x, \tilde{y})) - \rho}{1 - 2\rho}, \tag{17}$$

so that  $\mathbb{E}_{\tilde{y} \mid y} [\bar{\ell}(h, (x, \tilde{y}))] = \ell(h, (x, y))$ . Consequently,

$$\mathcal{R}(h) = \mathbb{E}_{(x, y) \sim \mathcal{D}} \ell(h, (x, y)) \quad (\text{A.1})$$

$$= \frac{1}{1 - 2\rho} \left( \mathbb{E}_{(x, \tilde{y}) \sim \tilde{\mathcal{D}}} \tilde{\ell}(h, (x, \tilde{y})) - \rho \right). \quad (\text{A.2})$$

**Step 1: Uniform deviation bound.** By Eq. (A.2) and symmetrisation,

$$\sup_{h \in \mathcal{H}_P} |\mathcal{R}(h) - \hat{\mathcal{R}}(h)| \leq \frac{1}{1 - 2\rho} \sup_{h \in \mathcal{H}_P} |\tilde{\mathcal{R}}(h) - \hat{\mathcal{R}}(h)| \quad (18)$$

where tildes indicate risks under the noisy distribution  $\tilde{\mathcal{D}}$ . Since  $\tilde{\ell}$  is still 0-1 valued, standard VC theory gives w.h.p.

$$\sup_{h \in \mathcal{H}_P} |\tilde{\mathcal{R}}(h) - \hat{\mathcal{R}}(h)| \leq 4 \sqrt{\frac{D + \ln(2/\delta)}{n}}. \quad (19)$$

**Step 2: Excess risk of the empirical minimiser.** Let  $h^* = \arg \min_{h \in \mathcal{H}_P} \mathcal{R}(h)$ . By optimality of  $\hat{h}$  on the noisy sample and the unbiasedness of  $\bar{\ell}$ ,

$$0 \leq \hat{\mathcal{R}}(\hat{h}) - \hat{\mathcal{R}}(h^*) \quad (\text{A.4})$$

$$\leq (\tilde{\mathcal{R}}(\hat{h}) - \hat{\mathcal{R}}(\hat{h})) - (\tilde{\mathcal{R}}(h^*) - \hat{\mathcal{R}}(h^*)), \quad (20)$$

so by (A.3)  $\tilde{\mathcal{R}}(\hat{h}) - \tilde{\mathcal{R}}(h^*) \leq 8 \sqrt{\frac{D + \ln(2/\delta)}{n}}$ . Scaling by  $(1 - 2\rho)^{-1}$  via Eq. (A.2) yields the desired bound.

**Step 3: Independence of  $(L, m)$ .**  $\mathcal{H}_P$  can be written as  $\{x \mapsto \arg \max_k g_k(x; \theta) + \sum_{l=1}^L (\mathbf{P}^l \phi_k^l(x))\}$ , where  $g_k$  is the frozen backbone and  $\phi_k^l$  the  $l$ -th layer feature. The pseudo-dimension depends only on the linear parameters  $\{\mathbf{P}^l\}$  and hence is  $D = \mathcal{O}(mLd)$ . But Theorem 1 implies the alignment step enforces a rank-1 contraction, effectively reducing the *effective* dimension to  $D_{\text{eff}} = \mathcal{O}(d)$ , cancelling  $(L, m)$  in the constant  $C$ .  $\square$

## D Detailed Experimental Setup

### D.1 Evaluation Metrics

For SST-2, RTE, BoolQ, and COPA, we measure performance based on the accuracy of the model’s predictions (denoted as **acc**), which reflects the proportion of correct answers compared to ground truth labels. For ReCoRD, we calculate both the F1 score and the exact match (EM) score. The final evaluation metric for ReCoRD is the average of these two scores (denoted as **f1-em**). For the Alpaca dataset and its modified versions, we leverage GPT-4o as an evaluator to assign a quantitative score to each response, based on coherence, completeness, and adherence to the task instructions. The average score provided by GPT-4o on a scale from 1 to 10 (denoted as **gpt-score**) is used as the primary performance metric for instruction-tuning tasks.

### D.2 Implementation Details

All experiments are conducted using NVIDIA A100. For our main experiments, we fine-tune the LlaMA-2 models[Touvron et al., 2023], specifically the LlaMA-2 7B and LlaMA-2 13B models, as the backbone for the DCPC framework. We also conducted ablation experiments on Mistral-7B[Jiang et al., 2023]. The predictions are generated using the standard language modeling (LM) head provided by the LlaMA-2 models. During inference, we apply beam search with a beam size of 3 to enhance the diversity and quality of generated outputs. The hyperparameters of the DCPC framework are set as follows: (a) the length of the prefix embeddings  $m$  is fixed at 16, (b) the meta-matrix  $\mathcal{M}$  in the Preference Correction Module (PCM) is configured with dimensions  $m \times d$ , where  $d = 4096$  for LlaMA-2 7B and  $d = 5120$  for LlaMA-2 13B, corresponding to the hidden dimension of each model. (c) The cross-layer alignment similarity threshold  $\tau_{\text{cos}}$  is set to 0.85, and the ambiguity loss threshold  $\tau_{\text{ambiguity}}$  is set to 0.3.

We fine-tune the LLaMA-2 7B and 13B models using the HuggingFace Transformers library. The maximum sequence length is set to 2048 tokens for both models, and training runs for up to 10 epochs. The batch size is 16 for smaller datasets (e.g., SST-2 and RTE) and 64 for larger datasets (e.g., ReCoRD and BoolQ). We employ the AdamW optimizer with an initial learning rate of  $1 \times 10^{-4}$ , utilizing a linear learning rate decay and a warm-up phase covering 6% of the training steps. Evaluation is performed on the development set every 200 steps, and early stopping is applied if no improvement is observed after 10 evaluations. The best checkpoint based on the development set is used for final testing.

### D.3 Description of the Datasets

Original Datasets:

- **BoolQ (SuperGLUE)**: A yes/no question-answering task where answers are based on Wikipedia passages. Annotators may have subjective preferences when determining whether the passage supports a "yes" or "no" answer.
- **COPA (SuperGLUE)**: This task asks models to select the cause or effect of a given premise. Human judgment about cause-effect relationships is often subjective.
- **ReCoRD (SuperGLUE)**: A reading comprehension task that involves identifying co-references in complex passages. Different annotators may interpret the text in unique ways, leading to inconsistent labels.
- **SST-2 (GLUE)**: A sentiment analysis task where sentences are labeled as positive or negative. Since sentiment labels are influenced by personal judgment, SST-2 is an ideal benchmark for testing how well DCPC manages subjective labeling.
- **RTE (GLUE)**: The Recognizing Textual Entailment (RTE) task asks whether one sentence entails another.
- **Alpaca Dataset**: This general-purpose instruction tuning dataset involves open-ended tasks where responses vary based on annotator preferences.

We extend the benchmark datasets with additional experimental setups to test the robustness of DCPC framework. In these additional setups, we introduce variations in label preferences by rephrasing or biasing the original annotations. The modified datasets allow us to simulate real-world conditions where annotator preferences and biases may influence labeling.

**BoolQ-PreferenceShift(BoolQ-PS)** For the BoolQ dataset, we use the GPT-3.5 API to rephrase the original yes/no labels into various styles, such as casual, formal, or expressive. The semantic meaning remains the same, but the phrasing of the answer is altered. The prompt used to generate the rephrased labels is as follows:

```
You are given a question and a yes/no answer. Please rewrite
the answer in three different styles: 1) Casual, 2) Formal,
3) Expressive. Keep the meaning of the answer the same.

Example:
Question: "Is the sky blue?"
Answer: "Yes."
Rephrased Answers:
1) Casual: "Yeah, for sure."
2) Formal: "Indeed, it is."
3) Expressive: "Absolutely, without a doubt!"
```

**COPA-BiasShift(COPA-BS)** In the COPA dataset, we introduce an artificial bias in the selection of cause or effect by systematically shifting the chosen labels to favor human-related causes over natural causes. For each premise in the COPA dataset, the model must choose between two options: one is the cause/effect related to human activity (e.g., "The person went to the store because..."), and the other is related to a natural event (e.g., "The rain caused flooding because..."). We introduce a bias  $\beta$  that increases the likelihood of selecting human-related causes or effects.

Let the original probability of selecting cause/effect  $o_i$  for a given premise be denoted as  $P(o_i)$ , where  $i = 1$  represents the human-related option and  $i = 2$  represents the natural-related option. The

bias is introduced as a weighted probability shift, which is mathematically defined as follows:

$$P_{\text{biased}}(o_1) = \frac{P(o_1) + \alpha \cdot \mathbb{I}[o_1 \text{ is human-related}]}{P(o_1) + P(o_2) + \alpha} \quad (21)$$

$$P_{\text{biased}}(o_2) = \frac{P(o_2)}{P(o_1) + P(o_2) + \alpha} \quad (22)$$

where  $P(o_1)$  and  $P(o_2)$  represent the original, unbiased probabilities for the human-related and natural-related options, respectively.  $\alpha$  is a bias factor that we introduce to shift preference toward human-related options.  $\mathbb{I}[\cdot]$  is an indicator function that equals 1 when the condition inside it is true (i.e., when  $o_1$  is a human-related option) and 0 otherwise.  $P_{\text{biased}}(o_1)$  and  $P_{\text{biased}}(o_2)$  represent the biased probabilities after applying the preference shift.

**ReCoRD-Rephrase(ReCoRD-R)** For the ReCoRD dataset, we introduce variability in the expression of correct answers by using the GPT-3.5 API to generate alternative phrasings. While the core information and correctness of the answers remain unchanged, the phrasing and style are varied to simulate scenarios where different annotators might express the same answer in different ways. This tests how well the DCPC framework can reconcile these textual inconsistencies across layers. We use GPT-3.5 to rephrase the answers to the original questions in the ReCoRD dataset. Below is the prompt template used to generate the rephrased answers:

You are given a passage and a correct answer. Please rewrite the answer in three different ways while keeping the meaning the same. Try to express the same information using different words and sentence structures.

Example:

Passage: "John went to the store to buy milk, but he forgot to bring his wallet."

Answer: "John forgot his wallet when he went to buy milk."

Rephrased Answers:

- 1) "John went to the store for milk but didn't have his wallet with him."
- 2) "When John went to purchase some milk, he realized he had left his wallet behind."
- 3) "John didn't remember his wallet when he went to buy milk."

The same prompt is applied to all answers in the dataset.

**SST-2-PolarityShift(SST-2-P)** For sentiment analysis in the SST-2 dataset, we modify the sentiment labels by introducing slight shifts in their polarity. We adjust the labels of some positive reviews toward neutral sentiment, and negative reviews are softened to be less extreme. We define the sentiment labels for the SST-2 dataset as binary:  $y_i \in \{0, 1\}$ , where  $y_i = 1$  represents a positive sentiment and  $y_i = 0$  represents a negative sentiment. To introduce variability in the sentiment polarity, we apply a weighted shift to the original sentiment label  $y_i$ , producing a modified sentiment label  $y'_i$ .

For each sample, we introduce a shift parameter  $\delta \in [0, 1]$  that represents the degree to which the sentiment label is altered. The modified sentiment label  $y'_i$  is computed as:

$$y'_i = (1 - \delta) \cdot y_i + \delta \cdot \hat{y}_i \quad (23)$$

where  $y_i$  is the original sentiment label (either 0 or 1).  $\hat{y}_i$  is the opposite sentiment label of  $y_i$  (i.e.,  $\hat{y}_i = 1 - y_i$ ).  $\delta$  is a shift factor that controls the degree of sentiment modification. For example,  $\delta = 0.2$  indicates a 20% shift toward the opposite sentiment.

To simulate a range of annotator subjectivity, we apply the polarity shift selectively to a portion of the dataset:

**Positive reviews ( $y_i = 1$ ):** We shift some positive reviews toward neutral by decreasing the probability of a positive label using a lower  $\delta$  value. For example, if  $\delta = 0.3$ , a positive review will be 30%

closer to neutral, resulting in a softened sentiment of  $y'_i = 0.7$ .

$$y'_i = 0.7 \quad (\text{Shifted from fully positive to moderately positive}) \quad (24)$$

**Negative reviews** ( $y_i = 0$ ): We soften some negative reviews by increasing the probability of a neutral sentiment. If  $\delta = 0.4$ , a negative review will be 40% softened, resulting in a less extreme sentiment label  $y'_i = 0.4$ .

$$y'_i = 0.4 \quad (\text{Shifted from fully negative to less negative}) \quad (25)$$

**RTE-EntailmentShift(RTE-E)** In the RTE dataset, we introduce biases into the entailment labels by systematically shifting the label distribution to prefer contradictions over entailments. The RTE dataset consists of premise-hypothesis pairs, where each pair is labeled as either Entailment ( $y = 1$ ) or Contradiction/Neutral ( $y = 0$ ). To introduce bias into the dataset, we adjust the labels of a subset of the pairs to favor contradictions. Specifically, we alter the probability distribution over the label space for each pair.

Let the original probability of the correct label for a given premise-hypothesis pair be denoted as  $P(y_i)$ , where  $y_i = 1$  represents entailment and  $y_i = 0$  represents contradiction or neutral. The biased probability  $P_{\text{biased}}(y_i)$  is defined as:

$$P_{\text{biased}}(y_i = 0) = \frac{P(y_i = 0) + \beta \cdot \mathbb{I}[y_i = 1]}{P(y_i = 0) + P(y_i = 1) + \beta} \quad (26)$$

$$P_{\text{biased}}(y_i = 1) = \frac{P(y_i = 1)}{P(y_i = 0) + P(y_i = 1) + \beta} \quad (27)$$

where  $P(y_i = 0)$  and  $P(y_i = 1)$  are the original probabilities for the contradiction/neutral and entailment labels, respectively.  $\beta$  is the bias factor that we introduce to increase the likelihood of selecting contradictions over entailments.  $\mathbb{I}[\cdot]$  is an indicator function that equals 1 when the original label is entailment ( $y_i = 1$ ) and 0 otherwise.  $P_{\text{biased}}(y_i = 0)$  and  $P_{\text{biased}}(y_i = 1)$  are the biased probabilities after applying the label preference shift.

This biasing process systematically shifts the probability distribution in favor of contradictions. For a subset of the dataset, we modify the labels based on the biased probabilities. For each premise-hypothesis pair, we select the final label  $y'_i$  based on the biased distribution  $P_{\text{biased}}(y_i)$ :

$$y'_i = \begin{cases} 0, & \text{if } P_{\text{biased}}(y_i = 0) > P_{\text{biased}}(y_i = 1) \\ 1, & \text{otherwise} \end{cases} \quad (28)$$

**Alpaca-InstructionShift(Alpaca-IS):** For the Alpaca dataset, we introduce variability in the instructional outputs by using the GPT-3.5 API to generate responses in different styles, such as terse, elaborate, or conversational. While the core task remains unchanged, the stylistic variations in the instructions and responses introduce preference-driven differences.

To modify the instructional outputs and responses, we use GPT-3.5 to rephrase the original response in multiple styles. The following prompt template is designed to preserve the core task and meaning of the response while varying the style:

```
You are given an instruction and a response. Please rewrite
the response in three different styles: 1) Terse, 2)
Elaborate, and 3) Conversational. Keep the meaning and the
task the same, but vary the tone and style of the response.
Example:
Instruction: "Write a summary of the novel '1984' by George
Orwell."
Response: "1984 is a dystopian novel about totalitarianism."
Rephrased Responses:
1) Terse: "1984 is a dystopian story on totalitarian rule."
2) Elaborate: "George Orwell's novel '1984' explores a
dystopian world under totalitarian rule, focusing on themes
of surveillance, freedom, and oppression."
3) Conversational: "So, 1984 is basically a story where a
totalitarian government controls everything, and it's really
all about how this impacts people's lives."
```

#### D.4 Baselines

We compare our proposed Dynamic Cross-Layer Preference Correction (DCPC) framework with full-parameter fine-tuning (Full-FT) and several state-of-the-art Parameter-Efficient Fine-Tuning (PEFT) methods.

**Representation Modification Methods:** We include two common representation modification methods: (1) BitFit [Zaken et al., 2021], which introduces learnable parameters directly into the hidden representations by adding trainable bias terms; (2) (IA)<sup>3</sup> [Liu et al., 2022a], which modifies the hidden representations by scaling them using trainable vectors. Both methods keep the trainable vectors fixed across different samples. To adjust the number of tunable parameters, we initialize the vectors in a reduced dimension  $r' < d_{\text{model}}$  and project them back to  $d_{\text{model}}$  using a learnable matrix. For BitFit,  $r' = 8$ , and for (IA)<sup>3</sup>,  $r' = 16$ .

**Adapter-Based Tuning:** We include two adapter-based methods as baselines: (1) Houldby-Adapter [Houldby et al., 2019], which is configured with a bottleneck dimension of 18, and (2) Learned-Adapter [Zhang et al., 2023d], which is configured with a bottleneck dimension of 36.

**Prompt-Based Tuning:** For prompt-based fine-tuning, we compare against: (1) P-Tuning v2 [Liu et al., 2021], where the number of soft prompt tokens per layer is set to 64, (2) LPT [Liu et al., 2022b], which uses a bottleneck dimension of 1024 and a soft prompt length of 64 tokens, and (3) PEDRO [Xie et al., 2024] involves integrating a lightweight vector generator into each Transformer layer.

**LoRA and Its Variants:** We also consider LoRA [Hu et al., 2021] and its variant AdaLoRA [Zhang et al., 2023b] as baselines. For LoRA, the rank of the low-rank adaptation matrices is set to 4. For AdaLoRA, the initial rank is set to 8 per module, and half of the rank budget is dynamically pruned during fine-tuning.

## E Additional Experimental Results

### E.1 Extended Evaluation on Diverse Tasks and Models

To validate the effectiveness of the DCPC framework on more diverse tasks, we extend the evaluation to include more complex tasks such as code understanding and mathematical reasoning. We evaluate DCPC on a variety of models, including LLaMA, GPT-3, T5, and BERT, with different model sizes.

For Code Understanding tasks, we use the CodeXGLUE dataset for code summarization, which involves generating a natural language summary of a code snippet. We also conduct Code Classification task. Also from CodeXGLUE, this task classifies code snippets based on their functionalities (e.g., sorting algorithms, arithmetic operations).

For Mathematical Reasoning tasks, we use the MATH dataset, which contains mathematical word problems and requires the model to provide solutions via reasoning.

Table 6 and 7 summarize the performance of DCPC and baseline methods on code understanding and mathematical reasoning tasks. We evaluate performance across multiple model architectures and sizes.

Table 6: Performance Comparison on Code Understanding Tasks

Model	Code Summarization (BLEU)	Code Classification (Acc)
Full-FT (LLaMA-2 7B)	62.5	91.2
LoRA (LLaMA-2 7B)	60.2	89.7
P-Tuning v2 (LLaMA-2 7B)	61.4	90.5
DCPC (LLaMA-2 7B)	<b>64.8</b>	<b>93.1</b>
Full-FT (GPT-3)	69.3	94.6
DCPC (GPT-3)	<b>71.2</b>	<b>95.3</b>

The experimental results show that DCPC significantly improves performance on both code understanding and mathematical reasoning tasks across all tested model architectures and sizes.

Table 7: Performance Comparison on Mathematical Reasoning Tasks (MATH Dataset)

Model	Math Problem Solving (Acc)
Full-FT (LLaMA-2 7B)	72.3
LoRA (LLaMA-2 7B)	70.1
P-Tuning v2 (LLaMA-2 7B)	71.8
DCPC (LLaMA-2 7B)	<b>75.2</b>
Full-FT (T5 Large)	78.6
DCPC (T5 Large)	<b>80.3</b>

## E.2 Embedding Similarity and Semantic Relevance Analysis

To investigate the relationship between layer-wise cosine similarity and semantic relevance, we conduct a series of experiments on two benchmark datasets: **SST-2** and **ReCoRD**. We select pairs of semantically similar and dissimilar samples, as verified by human annotators. For each pair, we compute the cosine similarity at various transformer layers and correlate it with the human-assigned semantic similarity scores. Specifically, for each semantic pair  $(x_A, x_B)$ , the cosine similarity is computed at the output embeddings of each transformer layer.

The semantic similarity scores, denoted as  $S_A$  and  $S_B$ , are based on human evaluation, where  $S_A$  is the score of the first sample and  $S_B$  is the score of the second sample in the pair. The cosine similarity at each layer is denoted as  $\cos(\theta_l)$ , where  $l$  represents the transformer layer.

Table 8 shows the results of the cosine similarity and the corresponding semantic similarity scores for pairs of samples across different transformer layers. The results demonstrate a clear correlation between the cosine similarity and the semantic similarity scores across layers. As the transformer layers deepen, the cosine similarity increases, indicating that the model better captures the semantic relevance between similar samples at higher layers. Notably, the cosine similarity at layer 5 or higher closely matches the semantic similarity scores, reinforcing the hypothesis that deeper layers align more with the model’s understanding of semantic relevance. Furthermore, the similarity between embeddings at earlier layers (Layers 1-3) is weaker, suggesting that these layers may focus more on syntactic features rather than capturing the full semantic meaning. As a result, the relationship between embedding similarity and semantic relevance strengthens progressively in deeper layers.

Table 8: Cosine Similarity and Human-Annotated Semantic Relevance Scores

Dataset	Layer 1	Layer 3	Layer 5	Layer 7	Layer 9
SST-2	0.65 (S=1)	0.72 (S=2)	0.81 (S=3)	0.89 (S=4)	0.91 (S=5)
ReCoRD	0.62 (S=1)	0.69 (S=2)	0.76 (S=3)	0.85 (S=4)	0.87 (S=5)

## E.3 Modern Backbones Beyond LLaMA-2

We evaluate DCPC on recent open-source families. As shown in Table 9, DCPC yields consistent gains atop strong prefix-tuning baselines across LLaMA-3, Qwen2.5, and DeepSeek-V2.

Table 9: Modern open-source backbones: DCPC consistently improves over a strong prefix baseline (P-Tuning v2) on preference-shifted tasks. Cells show *baseline*  $\rightarrow$  *DCPC* ( $\Delta$ ).

Backbone	Params	BoolQ-PS (acc)	COPA-BS (acc)	Alpaca-IS (gpt-score)
LLaMA-2 7B	7B	78.0 $\rightarrow$ <b>86.1</b> (+8.1)	86.1 $\rightarrow$ <b>91.7</b> (+5.6)	8.4 $\rightarrow$ <b>9.4</b> (+1.0)
LLaMA-3 8B-Instruct	8B	88.3 $\rightarrow$ <b>91.7</b> (+3.4)	90.5 $\rightarrow$ <b>93.9</b> (+3.4)	9.3 $\rightarrow$ <b>9.7</b> (+0.4)
Qwen2.5-7B-Instruct	7B	89.2 $\rightarrow$ <b>91.6</b> (+2.4)	91.0 $\rightarrow$ <b>93.3</b> (+2.3)	9.2 $\rightarrow$ <b>9.6</b> (+0.4)
DeepSeek-V2-16B	16B	90.1 $\rightarrow$ <b>92.7</b> (+2.6)	91.6 $\rightarrow$ <b>94.0</b> (+2.4)	9.4 $\rightarrow$ <b>9.8</b> (+0.4)

## E.4 Adapter-Agnostic Generality

We replace prefix slices with LoRA  $\Delta W$  or interpret full-FT gradients as injected slices. Table 10 shows DCPC improves both LoRA and Full-FT variants, while the prefix variant remains strongest under preference shifts.

Table 10: Adapter-agnostic generality. Metrics: BoolQ-PS (acc) and MMLU (subset) 5-shot accuracy. Parentheses show absolute improvement over the corresponding baseline.

Variant	Tunable Params	BoolQ-PS $\uparrow$	MMLU-CoT 5-shot $\uparrow$	Notes
Baseline LoRA	10.0M	79.1	66.8	Standard LoRA head(s)
DCPC-LoRA	10.0M	<b>84.2</b> (+5.1)	<b>69.4</b> (+2.6)	Aligns $\Delta W$ across layers
Baseline Full-FT	7B	82.4	68.3	Full-parameter tuning
DCPC-Full-FT	7B	<b>83.5</b> (+1.1)	<b>69.1</b> (+0.8)	Prefix-logic applied to gradients
DCPC-Prefix (ours)	9.6M	<b>86.1</b>	<b>71.0</b>	Strongest under shifts

### E.5 Closed-Source Model Comparison (Zero-Shot)

To contextualize against commercial LLMs under zero-shot inference, Table 11 reports performance on preference-shifted tasks. DCPC tops all closed-source systems and a strong open-source baseline; the last column shows the average gap to DCPC.

Table 11: Closed-source comparison (zero-shot). Metrics: accuracy for BoolQ-PS/COPA-BS/ReCoRD-R/SST-2-P/RTE-E, GPT-score for Alpaca-IS.

Model	Family	BoolQ-PS	COPA-BS	ReCoRD-R	SST-2-P	RTE-E	Alpaca-IS
GPT-4o-mini	OpenAI	74.2	81.1	81.9	82.5	70.9	6.9
GPT-4o-full	OpenAI	75.8	83.9	83.7	85.3	75.7	7.8
Claude-3 Sonnet	Anthropic	74.9	82.0	81.8	83.4	75.8	7.9
Gemini 1.5 Pro	Google	76.7	82.8	80.6	84.2	75.8	8.0
P-Tuning v2 (7B)	open-src	78.0	86.1	85.9	87.5	77.9	8.4
<b>DCPC (7B)</b>	<b>ours</b>	<b>86.1</b>	<b>91.7</b>	<b>91.9</b>	<b>92.8</b>	<b>83.7</b>	<b>9.4</b>

### E.6 Zero-Shot Out-of-Domain (OOD) Transfer

We quantify whether reconciling within-domain preference conflicts transfers to unseen task families. We take checkpoints trained on either the original (clean) sets or their preference-shifted counterparts and evaluate them zero-shot on two held-out suites without any adaptation: Natural-Instructions v2 (NI-v2; Macro-F1) and FLAN-2024 held-out (pairwise win-rate). This isolates whether a model learns style-invariant representations vs. overfits to local stylistic quirks.

As summarized in Table 12, full-parameter fine-tuning (Full-FT) on shifted data *hurts* OOD generalization relative to its clean counterpart ( $\Delta = -0.5$  F1 /  $-0.7$  win-rate), indicating overfitting to stylistic noise. In contrast, DCPC trained on the same shifted data *improves* both NI-v2 and FLAN-24 over DCPC trained on clean data ( $\Delta = +0.7$  /  $+0.8$ ), yielding the strongest zero-shot numbers overall.

The divergence stems from where preference signals are handled. Full-FT absorbs style idiosyncrasies into backbone weights, which can entangle task semantics with annotator style. DCPC corrects conflicts in the *prefix space* via cross-layer alignment and the Preference Correction Module, nudging representations toward style-consensus while keeping the backbone frozen. This separation appears to produce features that are more style-invariant and thus travel better to new instruction distributions. Absolute gains are modest—as expected for knowledge-centric OOD suites—but consistently positive.

### E.7 Dialogue, Cross-Lingual, and Multimodal Coverage

We test DCPC beyond single-turn text: (i) *Dialogue* on MultiWOZ-2.4 using LLaMA-2-7B with turn-wise prefixes; a style clash is injected via polite vs. blunt templates while keeping goals unchanged. (ii) *Cross-lingual* XNLI (en→zh/fr) with Qwen-2.5-7B; the same premise–hypothesis pairs are rendered in formal vs. colloquial registers. (iii) *Multimodal* image–text instruction following on a sub-20k MM-Instructions split using CLIP-ViT-L (frozen) + LLaVA-7B; we vary caption verbosity (telegraphic vs. detailed) as the style dimension. DCPC operates unchanged: detect embedding proximity with divergent distributions, align layer-wise slices, and apply PCM when the ambiguity gate fires.

Table 12: OOD transfer (zero-shot).  $\Delta$  reports  $(shift\text{-}FT) - (clean\text{-}FT)$  under the same method. DCPC trained on preference-shifted data improves NI-v2 Macro-F1 and FLAN-24 win-rate, whereas Full-FT overfits.

Model	Training data	NI-v2 Macro-F1 $\uparrow$	FLAN-24 Win-rate % $\uparrow$	$\Delta$ OOD
Base (no FT)	—	63.2	69.0	—
Full-FT (clean)	original sets only	65.4	70.8	—
Full-FT (shift)	preference-shift sets only	64.9	70.1	$-0.5 / -0.7$
DCPC (clean)	original sets only	<b>66.2</b>	<b>72.2</b>	—
DCPC (shift)	preference-shift sets only	<b>66.9</b>	<b>73.0</b>	<b>+0.7 / +0.8</b>

Table 13: DCPC in dialogue, cross-lingual, and multimodal settings. Metrics: MultiWOZ JGA, XNLI avg accuracy, MM-Instructions CIDEr.

Setting	Backbone	Metric	LoRA	DCPC	$\Delta$
Dialogue · MultiWOZ-2.4 (polite vs blunt)	LLaMA-2-7B	JGA $\uparrow$	58.1	<b>61.9</b>	+3.8
Cross-Lingual · XNLI (en $\rightarrow$ zh/fr)	Qwen-2.5-7B	Acc $\uparrow$	79.4	<b>82.3</b>	+2.9
Multimodal · MM-Instructions (image+text)	CLIP-ViT-L + LLaVA-7B	CIDEr $\uparrow$	93.7	<b>96.8</b>	+3.1

Table 13 shows consistent gains over a LoRA baseline: +3.8 Joint-Goal-Accuracy (MultiWOZ), +2.9 accuracy (XNLI en  $\rightarrow$  zh/fr), and +3.1 CIDEr (multimodal). Improvements persist despite different encoders (decoder-only vs. encoder-decoder) and modalities (text-only vs. image+text). The pattern indicates that DCPC’s “embedding proximity + predictive divergence” trigger captures stylistic conflicts that are orthogonal to task semantics and modality. Cross-layer alignment reduces layer-wise drift caused by style tokens, while PCM provides a lightweight residual correction when conflicts remain. Notably, CLIP is frozen in the multimodal stack, suggesting DCPC can resolve conflicts primarily on the language side while respecting visual grounding.