Flip-Flop Consistency: Unsupervised Training for Robustness to Prompt Perturbations in LLMs

Parsa Hejabi **Elnaz Rahmati** **Alireza S. Ziabari** **Morteza Dehghani**

University of Southern California
{hejabi, erahmati, salkhord, mdehghan}@usc.edu

Abstract

Large Language Models (LLMs) often produce inconsistent answers when faced with different phrasings of the same prompt. In this paper, we propose *Flip-Flop Consistency* ($\mathrm{F^2C}$), an unsupervised training method that improves robustness to such perturbations. $\mathrm{F^2C}$ is composed of two key components. The first, *Consensus Cross-Entropy* (CCE), uses a majority vote across prompt variations to create a hard pseudo-label. The second is a representation alignment loss that pulls lower-confidence and non-majority predictors toward the consensus established by high-confidence, majority-voting variations. We evaluate our method on 11 datasets spanning four NLP tasks, with 4–15 prompt variations per dataset. On average, $\mathrm{F^2C}$ raises observed agreement by 11.62%, improves mean F_1 by 8.94%, and reduces performance variance across formats by 3.29%. In out-of-domain evaluations, $\mathrm{F^2C}$ generalizes effectively, increasing $\overline{F_1}$ and agreement while decreasing variance across most source-target pairs. Finally, when trained on only a subset of prompt perturbations and evaluated on held-out formats, $\mathrm{F^2C}$ consistently improves both performance and agreement while reducing variance. These findings highlight $\mathrm{F^2C}$ as an effective unsupervised method for enhancing LLM consistency, performance, and generalization under prompt perturbations.[1]

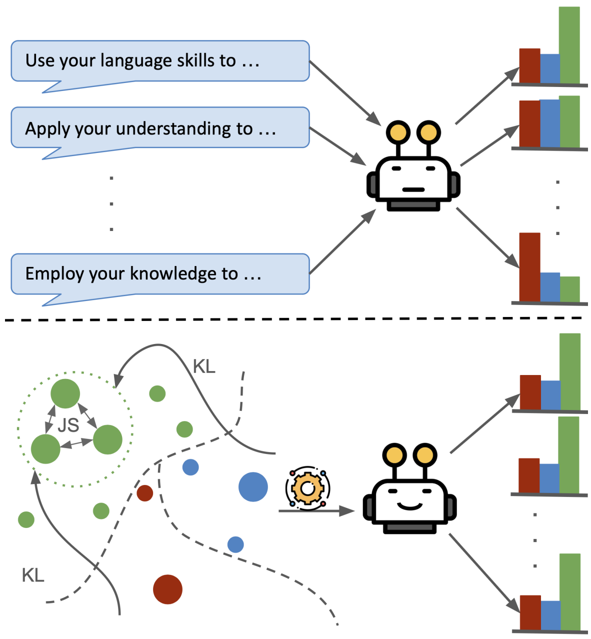

Figure 1: Our method aligns representations of input variations to promote consistency. To this end, we minimize the JS divergence among variations within the high-confidence consensus group, and the KL divergence between all other variations and that group.

1 Introduction

Large Language Models (LLMs) are increasingly deployed across diverse domains, including high-stakes settings such as law and medicine (OpenAI, 2025; Singhal et al., 2025; Guha et al., 2023), which raises the bar for reliability and trustworthiness. A core requirement for a trustworthy model is *semantic consistency*: when the phrasing of a question varies but its meaning remains the same, the model's answer should remain consistent. Recent studies show that LLM predictions can vary sharply under prompt perturbations such as formatting, casing, separators, paraphrasing, item ordering in few-shot settings, and other surface changes, often shifting reported accuracy by large margins (Sclar et al., 2024; Qiang et al., 2024; Sun et al., 2024; Lu et al., 2022; Cummins, 2025). Accordingly, several works advocate reporting performance ranges (or variance) across prompt variants rather than a single point estimate (Mizrahi et al., 2024; Polo et al., 2024; Alzahrani et al., 2024).

While numerous studies evaluate the consistency in existing models and propose new metrics to quantify it (Chatterjee et al., 2024; Cao et al., 2024; Nalbandyan et al., 2025; Datta et al., 2023), fewer works aim to improve consistency within the mod-

[1] Code is available at our GitHub repository.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1571–1587

els themselves. One line of work addresses this issue using prompt engineering techniques to search for the highest-performing prompt (Fu et al., 2024; Sclar et al., 2024; Ngweta et al., 2025; Cao et al., 2024; Raj et al., 2025; Voronov et al., 2024; Salinas and Morstatter, 2024). However, while effective, these techniques add computational overhead for prompt optimization and do not resolve the internal inconsistency of the models. Other approaches try to solve this issue via supervised fine-tuning (SFT) (Qiang et al., 2024; Yan et al., 2024; Sun et al., 2024; Fu and Barez, 2025), although these methods are limited by the availability of labeled data. Lastly, inference-time intervention approaches try to address this issue through model editing (Yang et al., 2024) and activation steering (Yang et al., 2025). According to Yang et al. (2024), despite being transparent, these methods fall behind SFT methods for improving performance. In the unsupervised setting, Zhou et al. (2022) propose "swarm distillation," a pairwise consistency loss that aligns the representations of input variations. They use a diverse set of variations provided by the Public Pool of Prompts (P3; Bach et al., 2022) for input-output pairs of 11 datasets. However, Cao et al. (2024) show that while this method improves consistency, it decreases overall performance.

We target the gap of improving consistency in the absence of supervision while preserving task performance, and propose *Flip-Flop Consistency* ($\mathrm{F}^2\mathrm{C}$), an unsupervised training algorithm that improves robustness to prompt perturbations by aligning their representations without sacrificing the performance. Prior work (Chen et al., 2025, 2024) has demonstrated that when a model consistently outputs the same label across variations, that label is more likely to be correct, whereas incorrect labels tend to be scattered, reflecting low confidence. Building on this insight, $\mathrm{F}^2\mathrm{C}$ takes the majority answer across variations as a *pseudo-label* for each data point. It then combines two components: (1) a cross-entropy loss that treats the pseudo-label as a hard label for all variations and (2) a divergence loss that aligns the distributions of less-confident and non-majority variations with those that confidently predict the majority. Together, these terms both increase the pseudo-label's probability and enforce consistency across variations.

We evaluate $\mathrm{F}^2\mathrm{C}$ on 11 datasets through three studies: (1) a comprehensive analysis of its performance against the base model and swarm distillation; (2) testing out-of-domain (OOD) generalization, where a model trained on one dataset is evaluated on the others; and (3) examining generalization to unseen prompt perturbations, where the model is trained on the first K prompt formats and evaluated on held-out formats. In Section 5.1, we demonstrate that $\mathrm{F}^2\mathrm{C}$ significantly raises observed agreement (P_o) on average by 11.62%, whereas swarm distillation slightly decreases it (-0.38%), showing a +12.00% agreement margin over swarm. As a beneficial byproduct, $\mathrm{F}^2\mathrm{C}$ also improves $\overline{F_1}$ on 9 out of 11 datasets with an average gain of +8.94% (vs. CCE: +8.36%, swarm: +1.40%) and reduces across-format σ_{F_1} by 3.29% on average (vs. CCE: 3.05%, swarm: 0.47%). For OOD generalization (Section 5.2), $\mathrm{F}^2\mathrm{C}$ successfully generalizes to the OOD data, yielding higher $\overline{F_1}$ on 74/80 train→test pairs, increases P_o on 64/80, and lowers σ_{F_1} on 66/80 compared to the base model. Finally, under limited format diversity (Section 5.3), we demonstrate that increasing the number of training formats in $\mathrm{F}^2\mathrm{C}$ consistently lifts $\overline{F_1}$ and P_o while shrinking σ_{F_1}, demonstrating robustness to unseen formats, despite only being trained on 5 or 10 prompt variations.

2 Related Work

2.1 Evaluating Consistency in LLMs

Despite strong zero-shot performance across many tasks (Brown et al., 2020), LLMs can be inconsistent-even contradictory-when responding to prompts that are semantically equivalent but phrased differently. Therefore, recent works advocate reporting performance as a *range* across prompt variants, rather than a single score that may reflect only a best-case (Mizrahi et al., 2024; Polo et al., 2024; Alzahrani et al., 2024; Wang et al., 2024). Empirical studies show large accuracy variations from simple format changes such as paraphrasing, casing, separators, spacing, and option ordering (Sclar et al., 2024; Cao et al., 2024; Qiang et al., 2024; Sun et al., 2024; Lu et al., 2022; Cummins, 2025; Alzahrani et al., 2024; Wang et al., 2024). Beyond raw accuracy spread, several frameworks and metrics target consistency more directly. Nalbandyan et al. (2025) propose various non-adversarial perturbations, such as paraphrasing, option reordering, and temperature sampling with multiple independent samples, to yield more realistic estimates. In contrast, Chatterjee et al. (2024) argue that accuracy variance across templates overlooks response distribution and therefore cannot distinguish a model that is consistently wrong from one that produces different wrong answers depending on the template. The authors then introduce a sensitivity index, `POSIX`, capturing response overlap, entropy, semantic coherence, and confidence variation. For each meaning-preserving input format and its model-generated answer, it averages the difference of the probabilities of generating the same response across all prompt variants.

2.2 Improving Consistency in LLMs

Work on improving consistency spans several directions. A line of work addresses inconsistency via prompt engineering without changing model weights. Fu et al. (2024) train a small seq2seq "paraphrase generator" to rewrite queries into expressions the target LLM prefers. Their method improves accuracy across QA, commonsense, and math tasks. Ngweta et al. (2025) propose *Mixture of Formats* (MOF), in which each few-shot example in the prompt uses a distinct format. Raj et al. (2025) introduce *Ask-to-Choose* (A2C), which samples multiple candidate answers and then prompts an LLM to select the best answer from those candidates. These approaches are effective at the prompt level but do not resolve the model's internal inconsistency while incurring inference-time overhead to obtain a strong prompt.

Supervised training has also been used to improve consistency. Yan et al. (2024) take a contrastive-learning approach and make hidden states for paraphrased instructions with the same input-output pair closer and push apart hard negatives (same instruction, different input-output). They use paraphrasing to create perturbations for the training data; however, more diverse perturbations, e.g., typos, word substitutions, appending random sequences at the end of instructions, and multilingual paraphrases, are used for evaluation data. Their method improves robustness to unseen perturbed instructions, with an average accuracy gain of 2.5% over continual instruction tuning. Zhao et al. (2024) introduce a two-stage alignment framework with two metrics, *Consistency Rate* (pairwise agreement across paraphrases with an LLM-as-judge) and *Maximum Consistency Rate* (the fraction of responses in the largest mutually consistent group). Stage 1 performs SFT on paraphrased instructions that share the same input–output, and Stage 2 generates multiple responses per input, scores them on format validity and correctness (using the gold label), forms preference pairs, and optimizes a DPO-style (Rafailov et al., 2023) ranking loss. A Vicuna-13B (Chiang et al., 2023) model trained with this pipeline surpasses GPT-4 on CR. Similarly, Qiang et al. (2024) propose Prompt Perturbation Consistency Learning (PPCL): during fine-tuning, they feed both a clean utterance and its perturbed version (oronyms, synonyms, or paraphrases) and optimize the cross-entropy on each. They also add a Jensen-Shannon (JS) divergence term between their token-level output distributions, which recovers much of the performance lost under prompt noise. Sun et al. (2024) add small trainable soft-prompt embeddings and optimize them to make representations of semantically equivalent instructions more similar, which consistently improves zero-shot robustness to new phrasing. Finally, Fu and Barez (2025) propose Latent Adversarial Paraphrasing (LAP): a bi-level scheme where an inner loop learns a constrained latent perturbation that acts as a continuous paraphrase while preserving semantics, and an outer loop fine-tunes the model on these perturbed inputs, improving worst-case win-rate by about 0.5–4% without adding inference-time latency.

Model editing (Meng et al., 2022) and activation steering (Turner et al., 2024) have also been adapted to this problem. Yang et al. (2024) use a *locate-then-edit* pipeline. They build paraphrase pairs, label each pair by whether the model's predictions agree (consistency), concatenate hidden states from both prompts, and train linear classifiers on last-token activations from each attention/MLP layer to predict the consistency label. They select the top-K components with the highest classifier accuracy as key components for semantic consistency. For each component, they compute the difference between the mean hidden output of the consistent pairs and the mean over all pairs, and add this as a bias to that component's hidden state. This increases accuracy and reduces the across-variant standard deviation on NLU tasks, and increases mean pairwise cosine similarity across variants for NLG tasks. Yang et al. (2025) use the same idea to identify the most influential transformer layer for consistency, then train a Top-K Sparse Autoencoder (SAE, Gao et al., 2025) to decompose its representation into a higher dimension. Using contrastive prompt pairs (correct vs. incorrect outputs), they select key SAE features with average activation differences exceeding a threshold and, at inference, add the learned feature offsets when the corresponding features activate, steering the model toward consistency. Yang et al. (2024) reported that, although these inference-time intervention methods are transparent, they generally fall behind SFT in performance.

Outside prompt-consistency settings, Shen et al. (2024) improve robustness to textual adversarial attacks with a model-level dynamic-attention mechanism tailored to transformer architectures. Their method modifies attention through attention rectification and dynamic modeling, showing that model-level interventions can improve robustness even without relying solely on prompt search or standard supervised fine-tuning.

Zhou et al. (2022) propose an unsupervised *swarm distillation* loss. For each instance, they sample a pair of prompt formats with the same semantical meaning and apply pairwise distillation (Hinton et al., 2015) so that one prompt's output distribution teaches the other (each prompt format can be both teacher and student). Using Fleiss' *kappa* (Fleiss, 1971) to measure agreement across prompt variations, they report a relative 14.6% increase over the T0-3B baseline on 8 out of 11 NLP datasets.

Huang et al. (2023) leverage Self-Consistency decoding (Wang et al., 2023), and sample multiple chain-of-thought solutions for each unlabeled question and take the majority answer as the "high-confidence." They then retain all reasoning paths that yield the majority answer, convert each path into four mixed input-output formats, and then perform supervised fine-tuning on the resulting set and gain up to 7.7% on GSM8K (Cobbe et al., 2021).

3 Method

To improve prompt perturbation robustness, we drew inspiration from two prior works. First, majority voting across prompt variations (Salinas and Morstatter, 2024) builds on Self-Consistency (Wang et al., 2023) and achieves the highest overall accuracy across 11 classification tasks. Second, swarm distillation (Zhou et al., 2022) encourages consistency across prompt variations by minimizing KL divergence between all pairs. This is implemented via sequence-level distillation (Kim and Rush, 2016), where each prompt simultaneously acts as both teacher and student.

Building on these two ideas, we focus on an unsupervised setting where no gold labels are available during training and pseudo-labels must instead be inferred from the model's own responses across variations. Because the most frequent label produced across perturbations tends to represent the model's most confident and often correct prediction (Chen et al., 2024, 2025), it is natural to treat the majority answer as a training signal. However, plain cross-entropy alone does not guarantee consistency across semantically equivalent formats. It only increases the probability of the pseudo-labeled answer within each format without aligning distributions between formats.

Swarm distillation addresses this issue by enforcing agreement, pulling all variations' distributions toward their average (uniform mixture; see Theorem A.1) regardless of each variation's prediction. Yet, Cao et al. (2024) show that this averaging can harm overall performance, likely because the model overfits to noisy or lower quality mixtures.

To address both limitations, we introduce "Flip-Flop Consistency" by combining two complementary components: (1) supervising with majority-vote pseudo-labels using Consensus Cross-Entropy (CCE; Section 3.2) and (2) aligning distributions to a stronger target, defined as the average distribution computed only from variations that confidently select the majority label (Section 3.3).

3.1 Problem Formulation

Let $\mathcal{T}$ be a classification task with L labels $\mathcal{Y} = \{\ell_1, \ldots, \ell_L\}$. The dataset consists of N instances $\{(x_i, y_i)\}_{i=1}^{N}$, where $y_i \in \mathcal{Y}$ denotes the gold label. In our unsupervised setting, the gold labels are not used for training. We assume a set of V prompt templates that preserve semantic meaning, $\mathcal{R} = \{r_1, \ldots, r_V\}$. Each template r_v renders inputs and label options:

$$x_i^{(v)} = r_v(x_i), \tag{1}$$

$$y_c^{(v)} = r_v(\ell_c) \quad \text{for } c = 1, \ldots, L. \tag{2}$$

Per-variation scoring. Using the model's length-normalized token-level log-likelihood for choosing label ℓ_c under template v, denoted $\mathrm{LL}_i[v, c]$ (see Appendix A.3 for the exact computation), we define the per-variation label distribution as $\pi_{i,v,c} = \mathrm{softmax}(\mathrm{LL}_i[v, c])$ and the per-variation prediction

$$\hat{y}_{i,v} = \arg\max_{c \in \{1,\ldots,L\}} \pi_{i,v,c}. \tag{3}$$

3.2 Consensus Cross-Entropy

We construct a pseudo-label via majority vote across variations and then fit the model to that label.

Consensus label. Define vote counts

$$n_{i,c} \;=\; \sum_{v=1}^{V} \mathbf{1}[\hat{y}_{i,v} = c]\,, \tag{4}$$

and set the "consensus" (strict majority) label

$$c_i^\star \;=\; \arg\max_c n_{i,c} \quad \text{with} \quad n_{i,c_i^\star} > \tfrac{V}{2}; \tag{5}$$

otherwise, no consensus is formed for instance i.

Loss. When a consensus exists for example i with label $c_i^\star$, let $\ell_{i,v}$ denote the negative log-likelihood of the consensus answer $y^{(v)}_{c_i^\star}$ under variation v given $x_i^{(v)}$ (scoring only the answer tokens). The instance-level CCE is

$$\mathcal{L}_{\text{CCE}}(i) \;=\; \mathbf{1}\big[n_{i,c_i^\star} > \tfrac{V}{2}\big] \;\lambda_{\text{CCE}}\, \frac{1}{V}\sum_{v=1}^{V} \ell_{i,v}, \tag{6}$$

and the training objective averages over examples:

$$\mathcal{L}_{\text{CCE}} \;=\; \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}_{\text{CCE}}(i). \tag{7}$$

Only examples with a strict majority contribute to the loss; if no consensus exists, $\mathcal{L}_{\text{CCE}}(i) = 0$. The coefficient λ_{CCE} controls this term's strength.

3.3 Flip-Flop Consistency

We combine CCE with a representation alignment objective. Among the variations that vote for the consensus label $c_i^\star$, we identify confident prompts (the consensus-confident, or *CC*, set) and align the remaining prompts (the non-confident or non-consensus, *NC*, set) toward the CC set, while also encouraging agreement within the CC set. If no strict consensus exists, the example is skipped (no loss). We describe the details on forming the CC and NC sets in algorithm 1.

Given the strict majority consensus $c_i^\star$ and its consensus set $G = \{v : \hat{y}_{i,v} = c_i^\star\}$, the algorithm checks if a majority exists (line 1-2), computes per-variation confidence margins m_v by calculating the difference of log-likelihood between the consensus label and the most probable non-consensus label. Then, it takes the median m_{med} as a representative for the consensus set (lines 3-5). When $|G|{=}V$ and $m_{\text{med}} \geq \tau_{\text{unanimous}}$, all variations predict the same label confidently, so the algorithm returns $T_i{=}|G|$ (CC set) and $S_i{=}\emptyset$ (NC set) (line 7). If variations are not confident in producing the majority label or they produce different labels, the algorithm forms a CC/NC split to pull the NC set's representations toward the mean distribution of the CC set. This is done by picking the top-k majority voter variations as the CC set and assigning the rest to the NC set (lines 10, 13-14). Lastly, w_{flip} weight is calculated to control the intensity of alignment between the CC and NC sets by applying a sigmoid function to the difference of average log-likelihoods for producing the consensus label between the CC and NC sets (Δ). Finally, this weight is capped between $f_{\min}$ and $f_{\max}$ hyperparameters (lines 15-18). Degenerate branches (fewer than two variations in the CC set) return empty sets and zero weight.

All loss components operate *only* on the consensus answer tokens. For each variation v, let $\log \mathbf{q}^\star_{i,v}$ denote the model's token-level log-softmax over the full vocabulary when outputting the consensus answer $y^{(v)}_{c_i^\star}$ under $x_i^{(v)}$, aggregated over answer positions. We use $\mathbf{q}^\star_{i,v} = \exp(\log \mathbf{q}^\star_{i,v})$ inside divergence losses. For the CC set T_i, define the CC mixture $\bar{\mathbf{q}}_i^{T\star}$ as the (probability-space) average of $\{\mathbf{q}^\star_{i,t}\}_{t\in T_i}$.

Let T_i (CC set) and S_i (NC set) be the sets returned by Algorithm 1. There are three possible cases for each instance i:

Case 1: No strict majority ($|G| \leq V/2$, line 2). No pseudo-label is trusted; we skip the example and apply *no loss*.

Case 2: Unanimous & confident ($|G|{=}V$ and $m_{\text{med}} \geq \tau_{\text{unanimous}}$, line 7). Here $T_i = G$, $S_i = \emptyset$, and $w_{\text{flip}} = 0$. All variations are confident in outputting the majority answer, and in order to make them even more consistent, we apply a JSD loss with β_{jsd} hyperparameter to make them even closer to their average point.

$$\mathcal{L}_{\text{jsd}}(i) = \beta_{\text{jsd}}\ \text{JSD}\big(\{\mathbf{q}^\star_{i,t}\}_{t\in T_i}\big). \tag{8}$$

Case 3: Consensus with split ($|T_i| \geq 2$, lines 5-19). In this case, we want to only pick the top K most confident majority voters as the CC set and align the NC set toward their average distribution:

$$\mathcal{L}_{\text{flip}}(i) = w_{\text{flip}}\, \frac{1}{|S_i|}\sum_{s\in S_i} \text{KL}\big(\mathbf{q}^\star_{i,s} \,\|\, \bar{\mathbf{q}}_i^{T\star}\big), \tag{9}$$

and also encourage agreement within the CC set using the same $\mathcal{L}_{\text{jsd}}(i)$ as in Case 2 (Equation (8)). We control the strength w_{flip} of $\mathcal{L}_{\text{flip}}(i)$ using the hyperparameters $f_{\min}$, $f_{\max}$ and temperature t.

Algorithm 1: F^{2}C for instance i

Data: $\mathrm{LL}_i \in \mathbb{R}^{V\times L}$,
Consensus label $c_i^\star$,
Consensus set $G = \{\, v : \hat{y}_{i,v} = c_i^\star \,\}$
Input: Unanimous margin $\tau_{\text{unanimous}}$,
CC set size cap $k_{\max} \ge 2$,
weight bounds $f_{\min} \le f_{\max}$,
temperature $t > 0$
Result: CC set T_i,
NC set S_i,
flip weight w_{flip}

if $|G| \le V/2$ **then**
  return $(\emptyset, \emptyset, 0)$; `// no strict majority`
foreach $v \in G$ **do**
  $m_v \leftarrow \mathrm{LL}_i[v, c_i^\star] - \max_{c \neq c_i^\star} \mathrm{LL}_i[v, c]$;
$m_{\text{med}} \leftarrow \mathrm{median}\{m_v : v \in G\}$;
if $|G| = V$ ***and*** $m_{med} \ge \tau_{unanimous}$ **then**
  return $(G, \emptyset, 0)$; `// unanimous & confident`
if $|G| < 2$ **then**
  return $(\emptyset, \emptyset, 0)$
$k \leftarrow \min\bigl(k_{\max},\, V-1\bigr)$; `// leave at least one variation in NC set`
if $k < 2$ **then**
  return $(\emptyset, \emptyset, 0)$; `// need ≥ 2 variations in CC`
$T_i \leftarrow$ top-k members of G by m_v (descending);
$S_i \leftarrow \{1, \ldots, V\} \setminus T_i$;
$\bar{\ell}_T \leftarrow \frac{1}{|T_i|} \sum_{t \in T_i} \mathrm{LL}_i[t, c_i^\star]$;
$\bar{\ell}_S \leftarrow \frac{1}{|S_i|} \sum_{s \in S_i} \mathrm{LL}_i[s, c_i^\star]$;
$\Delta \leftarrow \bar{\ell}_T - \bar{\ell}_S$; `// gap on consensus label`
$w_{\text{flip}} \leftarrow f_{\min} + (f_{\max} - f_{\min}) \cdot \sigma(\Delta/t)$;
return (T_i, S_i, w_{flip})

Total loss and hyperparameters. To summarize, the total loss per example is:

$$\mathcal{L}_{\text{FF}}(i) = \begin{cases} 0, & \text{Case 1}, \\ \mathcal{L}_{\text{CCE}}(i) + \mathcal{L}_{\text{jsd}}(i), & \text{Case 2}, \\ \mathcal{L}_{\text{CCE}}(i) + \mathcal{L}_{\text{jsd}}(i) + \mathcal{L}_{\text{flip}}(i), & \text{Case 3}. \end{cases} \tag{10}$$

With hyperparameters λ_{CCE} (CCE weight), $\tau_{\text{unanimous}}$ (confidence threshold), $k_{\max}$ (max CC size), $f_{\min}, f_{\max}$ and t (flip loss caps, temperature), β_{jsd} (agreement weight within the CC set).

3.4 Metrics

Following Zhao et al. (2024) we use the raw observed agreement (P_o) to measure consistency across prompt variations. We omit Fleiss' κ used in (Zhou et al., 2022) due to the prevalence/bias paradox noted by Hoehler (2000). Using the vote counts $n_{i,c}$ from Eq. 4, the per-item agreement is calculated as in Eq. 11, and P_o is the average of P_i over items. Intuitively, P_i represents the probability that two uniformly sampled prompt variations for the same input predict the same label.

$$P_i = \frac{1}{V(V-1)} \sum_{c=1}^{L} n_{i,c}\,\bigl(n_{i,c} - 1\bigr), \tag{11}$$

High agreement alone may result from a collapsed model that predicts a single label for all prompts. To ensure consistency does not come at the expense of task performance, we also report $\overline{F_1}$. Prior work has further used performance spread- best- vs. worst-case performance- (Sclar et al., 2024), but this metric is highly sensitive to outliers. Instead, we report the standard deviation σ_{F_1} across prompt variations to capture performance stability.

4 Experimental Setup

4.1 Datasets

Following Zhou et al. (2022), we evaluate our proposed method on eleven classification datasets spanning four tasks, and use templates from the Public Pool of Prompts (P3; Bach et al., 2022) to create prompt variations. The NLP tasks include natural language inference ((ANLI R1/R2/R3, Nie et al., 2020), (CB, de Marneffe et al., 2019), (RTE, Wang et al., 2019)), sentence completion (COPA (Roemmele et al., 2011), HellaSwag (Zellers et al., 2019), StoryCloze 2016 (Mostafazadeh et al., 2016)), coreference-style commonsense (WSC (Wang et al., 2019), WinoGrande-XL (Sakaguchi et al., 2019)), and word sense disambiguation (WiC (Pilehvar and Camacho-Collados, 2019)). Most official test splits of these datasets are unlabeled. Therefore, we evaluate on the official *validation* split. We create a stratified hold-out set from the official training split for validation. Details and statistics of all datasets are reported in the Appendix A.2.

4.2 Implementation Details

We compare our approach against two baselines, the unmodified base model and the base model fine-tuned with swarm distillation. For all experiments, we fine-tune `Qwen2.5-3B-Instruct` (Qwen et al., 2025) using LoRA (Hu et al., 2022). The first experiment is also run on `Llama-3.2-3B-Instruct` (Grattafiori et al., 2024), where we compare the base model with its F^{2}C fine-tuned variant. Configuration details are provided in Appendix A.3.

5 Results

We conduct three experiments to analyze whether our method: (1) improves robustness to prompt per-

		ANLI R1	ANLI R2	ANLI R3	CB	RTE	COPA	HellaSwag	StoryCloze	WSC	WinoGrande	WiC
Base	$\overline{F_1}$	39.23	32.31	31.02	31.70	77.90	81.73	35.43	85.35	42.39	54.43	11.45
	σ_{F_1}	13.97	9.76	9.35	19.46	4.12	10.65	2.20	7.78	**11.84**	3.69	17.53
	P_o	52.29	52.80	52.84	46.73	85.08	81.54	50.89	85.24	74.29	68.87	85.00
Swarm	$\overline{F_1}$	38.77	32.06	30.95	**34.32**	78.34	81.35	34.92	84.69	**46.06**	62.30	**14.57**
	σ_{F_1}	14.27	9.92	9.36	21.01	2.31	10.95	2.26	7.64	14.40	2.66	**10.44**
	P_o	50.91	50.63	50.49	40.78	86.94	80.93	50.97	85.23	68.58	75.22	**90.73**
CCE	$\overline{F_1}$	**58.68**	39.42	34.08	32.14	81.26	90.44	67.64	96.22	41.43	63.88	9.72
	σ_{F_1}	**2.40**	**5.97**	7.07	19.27	1.54	5.52	2.80	**0.38**	12.16	2.99	16.73
	P_o	**78.96**	**70.58**	**72.35**	**50.39**	91.71	90.32	72.42	97.05	**75.91**	77.16	87.16
F^{2}C	$\overline{F_1}$	58.43	**39.58**	**35.77**	31.81	**81.55**	**90.84**	**70.28**	**96.32**	41.68	**64.98**	9.99
	σ_{F_1}	2.61	6.17	**6.24**	**19.19**	**0.82**	**5.39**	**2.12**	0.46	12.12	**2.43**	16.66
	P_o	78.85	67.64	68.68	50.26	**93.40**	**90.71**	**76.97**	**97.32**	75.23	**77.30**	87.01

Table 1: Comparison across datasets for `Qwen2.5-3B-Instruct`: the base model and variants trained with Swarm (swarm distillation), CCE, and F^{2}C. Bold blue values mark the best metric per dataset column, orange values denote the worst.

turbations without reducing task performance, (2) maintains semantic consistency in out-of-domain settings, and (3) is not bound to the prompt formats used in training and generalizes to unseen formats.

Before evaluating our method, we assess the inherent consistency of the `Qwen2.5-3B-Instruct` base model (see Appendix Figure 3). The mean interquartile range (Q3−Q1) of F_1 across prompt variations, averaged over datasets, is 13.53%. Nearly half of the datasets (5/11) exceed a 15% spread, indicating substantial inconsistency within the base model.

F^{2}C is motivated by prior evidence that majority pseudo-labels across prompt variations are more likely to be correct. Therefore, we evaluate the strict-majority labels produced by the base `Qwen2.5-3B-Instruct` model in our setting (see Appendix A.3). A strict-majority label exists for 93.9% of validation examples across their prompt formats, and the F_1 of those majority labels is 59.5 when per-dataset scores are averaged by the number of instances with strict-majority label. The signal is strongest on RTE, COPA, and StoryCloze, moderate on ANLI, HellaSwag, and WinoGrande, and weakest on CB, WSC, and especially WiC. These results help interpret the dataset-level gains and failure cases in the first experiment.

5.1 Flip-Flop Consistency Against Baselines

To examine whether aligning representations across prompt formats further improves consistency when combined with the CCE loss, we train `Qwen2.5-3B-Instruct` using two loss functions, CCE, and F^{2}C, and compare them against the baselines introduced in Section 4.2. Since training-induced gains can depend on the model family (Shao et al., 2026), we also compare the base `Llama-3.2-3B-Instruct` with its F^{2}C fine-tuned variant. Implementation details are provided in Appendix A.3. For each method and dataset, Table 1 and Table 2 report $\overline{F_1}$, σ_{F_1}, and P_o for `Qwen2.5-3B-Instruct` and `Llama-3.2-3B-Instruct`, respectively.

On `Qwen2.5-3B-Instruct`, F^{2}C achieves the largest average improvements over the base model. It improves agreement by 11.62%, increases the mean F_1 by 8.94%, and reduces F_1 variance by 3.29%. In comparison, CCE raises the average agreement by 11.68%, with slightly smaller gains in mean F_1 and σ_{F_1} reduction (8.36% and 3.05%, respectively). Both F^{2}C and CCE achieve higher $\overline{F_1}$ and P_o and lower σ_{F_1} than the baselines on most datasets, including ANLI R1/R2/R3, RTE, COPA, HellaSwag, StoryCloze, and WinoGrande. In contrast, swarm distillation is the weakest baseline that reduces agreement on average by 0.38% and produces only a small mean F_1 gain of 1.40%. These results indicate that F^{2}C not only enhances consistency but also improves task performance.

The same qualitative pattern holds on `Llama-3.2-3B-Instruct`. Relative to the base model, F^{2}C improves $\overline{F_1}$ on 9/11 datasets, lowers σ_{F_1} on 10/11, and increases P_o on 10/11, with average changes of +7.98%, -2.98%, and +14.21%, respectively. Only RTE and WiC do not see mean F_1 gains, while the remaining datasets follow the same overall trend as `Qwen2.5-3B-Instruct`. This indicates that the gains are not specific to the

		ANLI R1	ANLI R2	ANLI R3	CB	RTE	COPA	HellaSwag	StoryCloze	WSC	WinoGrande	WiC
Base	$\overline{F_1}$	32.28	29.43	28.95	25.00	**69.41**	68.21	24.16	65.02	44.74	52.23	**35.36**
	σ_{F_1}	8.56	6.69	7.32	12.00	**3.48**	5.77	7.40	7.64	8.76	3.60	22.98
	P_o	48.08	46.78	45.09	40.03	**77.46**	77.64	52.28	80.72	52.26	80.90	59.48
F²C	$\overline{F_1}$	**36.36**	**31.48**	**31.36**	**25.22**	68.80	**84.35**	**39.64**	**94.20**	**54.58**	**66.75**	29.80
	σ_{F_1}	**3.87**	**4.83**	**4.66**	**9.63**	3.80	**5.59**	**3.40**	**0.56**	**1.96**	**0.15**	**22.97**
	P_o	**73.51**	**54.40**	**65.71**	**54.42**	75.82	**85.11**	**59.17**	**96.12**	**88.80**	**98.91**	**65.07**

Table 2: Comparison across datasets for `Llama-3.2-3B-Instruct`: the base model and the variant trained with F^{2}C. Bold blue values mark the best metric per dataset column, orange values denote the worst.

Qwen family.

On `Qwen2.5-3B-Instruct`, the consensus is unreliable in cases where the base model has weak performance and high σ_{F_1} (CB, WSC, and WiC). This is consistent with the pseudo-label analysis above: these are also the datasets where strict-majority labels are least reliable. Therefore, pushing toward consensus does not improve performance; nevertheless, F^{2}C and CCE still raise P_o in these datasets.

Overall, F^{2}C increases agreement, improves task performance, and reduces across-format variance, while swarm distillation can even harm agreement.

5.2 Generalization in Out-of-Domain Settings

We assess out-of-domain (OOD) generalization by evaluating a model trained on source dataset with F^{2}C on all other target datasets. CB, WSC, and WiC are excluded as sources due to weak performance in Section 5.1 but are retained as targets. For each source→target pair, we compute the change in mean $\overline{F_1}$, σ_{F_1}, and agreement P_o on the target dataset relative to the base model (see Table 3). Appendix Figs. 6, 5, and 7 visualize the corresponding $\Delta\overline{F_1}$, ΔP_o, and $\Delta\sigma_{F_1}$ heatmaps.

Overall, F^{2}C generalizes well across domains. Averaged over all 80 dataset pairs, observed agreement increases by 7.49%, mean F_1 by 7.61%, and σ_{F_1} decreases by 2.94%. Moreover, positive transfers substantially outnumber negatives across all three metrics (P/N columns).

Training on story/commonsense datasets such as COPA, and StoryCloze yields the strongest average improvements in $\overline{F_1}$, while WinoGrande produces the largest average gains in P_o and the greatest reduction in σ_{F_1}. RTE and ANLI R1 also generalize reliably across many targets. Harder targets such as WSC and WiC show smaller or mixed changes in agreement, though variance typically still declines (see Appendix heatmaps 6, 5, and 7 for per-pair patterns).

	$\overline{F_1}$		P_o		σ_{F_1}	
Train Dataset	Δ	P/N	Δ	P/N	Δ	P/N
All (80 pairs)	7.613	74/6	7.491	64/16	2.941	66/14
ANLI R1	9.428	10/0	8.727	7/3	3.653	8/2
ANLI R2	5.543	10/0	6.886	8/2	2.171	8/2
ANLI R3	4.210	7/3	6.801	9/1	2.225	9/1
RTE	8.077	10/0	5.938	8/2	3.255	9/1
COPA	**11.836**	10/0	7.350	8/2	3.077	8/2
HellaSwag	2.810	7/3	8.074	8/2	2.410	7/3
StoryCloze	11.176	10/0	7.267	7/3	2.896	8/2
WinoGrande	7.823	10/0	**8.882**	9/1	**3.839**	9/1

Table 3: Cross-dataset transfer performance under F^{2}C. Each row shows the mean signed Δ relative to the base model when the *row* dataset is used for training. Columns report changes in $\overline{F_1}$, P_o, and σ_{F_1}, along with *P/N* (number of datasets with positive or negative improvement out of 10). The top "All (80 pairs)" row aggregates over all source→target pairs. Bold numbers indicate the best dataset for each metric.

5.3 Generalization to Unseen Variations

We test whether F^{2}C trained on a subset of prompt formats generalizes to *unseen* formats. We use ANLI R1/R2/R3 (15 formats each) and RTE (10 formats) due to their larger instance size and number of available variations. For RTE, we train with the first 5 formats and evaluate on the remaining 5. For each ANLI dataset, we hold out the last 5 formats for evaluation and train with the first 5, then with 10 (see Figure 2).

Across four datasets, using more training formats improves both performance and agreement on the held-out formats, and reduces across-format variance. ANLI R1 shows the largest steady gains as the number of training variations increases. ANLI R2 improves moderately but monotonically. ANLI R3 shows a small decrease with 10 variations but improves with 15 variations. RTE is strong even at 5 formats and still rises with more. Error bands (σ_{F_1} over held-out formats) shrink as we add formats, indicating higher semantic consistency.

5.4 Representation Alignment Analysis

To better understand how F^2C changes the model internally, we analyze the final-layer hidden state at the answer boundary, immediately before any answer token is generated. For each dataset instance, we extract this vector for every prompt variation, compute all pairwise cosine similarities among the prompt variations, and average them into one score for that instance. F^2C increases this within-instance similarity on 10/11 datasets, with an average gain of 0.040 (see Appendix Figure 4). The only decrease is on CB (-0.021), consistent with its weaker performance in Section 5.1. This shows that F^2C generally maps semantically equivalent prompt variants to more similar internal states, rather than only increasing output-level agreement.

Higher similarity across prompt variants alone could be caused by representation collapse. We therefore also examine whether class-level distinctions are preserved after fine-tuning with F^2C. For each instance, we average its prompt variation representations into one instance vector. We then group instances by their dataset-provided gold labels and compute the centroid of the instance vectors for each class. The within-class distance is the average cosine distance from each instance vector to its own class centroid; lower values mean that examples with the same gold label are closer together. The between-class distance is the average cosine distance between class centroids; higher values mean that different classes are more separated. This analysis is more interpretable for RTE and ANLI R1/R2/R3, where labels denote the same sentence-pair relations across examples, unlike multiple-choice tasks where label IDs often only identify answer positions. On RTE and ANLI R1/R2/R3, F^2C reduces within-class distance while maintaining or increasing between-class distance. For example, on RTE, between-class distance increases from 0.028 to 0.032, while within-class distance decreases from 0.022 to 0.018. Thus, on these NLI tasks, F^2C reduces prompt-induced variation without collapsing the class distinctions. Full results are provided in Appendix A.4.

6 Conclusion

LLMs often change their predictions when semantically equivalent prompts are phrased differently, undermining consistency and reliability. To address the problem of semantic consistency, we introduced F^2C, an unsupervised training method that uses majority voting across prompt variations to form hard pseudo-labels, then selectively aligns distributions toward confident majority voters while encouraging agreement among them. Our method relies on signals the model already treats as reliable, reinforcing them to ensure consistency while minimizing the influence of noisy or uncertain variations.

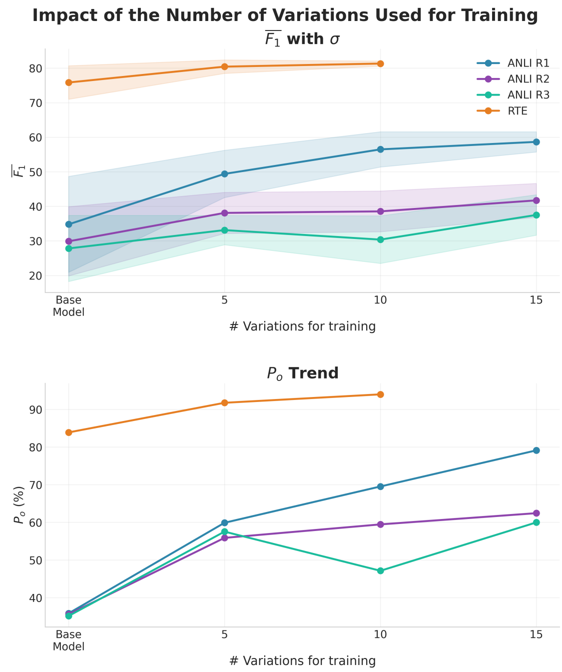

Figure 2: Top: $\overline{F_1}$ with shaded σ_{F_1} on the *held-out* prompt formats. Bottom: observed agreement P_o on the same held-out sets. The x-axis is the number of training formats (K); "Base Model" is the untrained baseline.

To validate our method, we conducted comprehensive experiments across multiple datasets and generalization scenarios. Across 11 datasets, F^2C consistently improves agreement, increases mean F_1, and reduces variance across prompt formats, outperforming both swarm distillation and the CCE-only variant. These gains persist in two generalization settings: cross-dataset transfer and generalization to unseen prompt formats, in which training on only a subset of formats still improves performance and agreement on held-out ones.

Our results suggest that much of the inconsistency from prompt phrasing can be mitigated by leveraging the model's own internal consensus, without gold labels. Future work includes extending F^2C to open-ended generation, exploring adaptive selection of high-confidence variations beyond top-K, and combining our approach with lightweight supervision when labels are available.

Limitations

While our results indicate consistent gains, several limitations should be acknowledged. First, although the first experiment is reported on two 3B instruction-tuned model families (`Qwen2.5-3B-Instruct` and `Llama-3.2-3B-Instruct`), the out-of-domain and unseen-variation generalization analyses are run only on `Qwen2.5-3B-Instruct`. Consequently, the scalability of F^2C to larger or smaller models, different pretraining corpora, or non-instruction-tuned bases is not established. Second, our evaluation is specifically focused on classification tasks with discrete labels. We do not study open-ended generation (e.g., long-form QA or chain-of-thought), for which our proposed method and evaluation metrics may require adaptation. Third, the perturbations we consider are non-adversarial template variants drawn from PromptSource. We do not test robustness to stronger or adversarial edits (character-, word-, or sentence-level changes), jailbreak-style attacks, multilingual rewrites, or heavy formatting noise. Beyond coverage, F^2C assumes multiple semantically equivalent templates per instance and uses them during training. In settings with scarce or low-quality templates, effectiveness and efficiency may degrade. Methodologically, we rely on the majority pseudo-labels and skip instances without a majority. On small or class-imbalanced datasets, the majority may be wrong, and the skip rule can bias learning toward "easier" examples, despite our confidence-aware variation selection for CC set. For model selection, we use validation $\overline{F_1}$ even though our objectives also target agreement (P_o) and dispersion (σ_{F_1}). Alternative criteria (e.g., multi-objective or worst-case) could yield different trade-offs, and we do not evaluate calibration or abstention. In terms of generalization, our target datasets are across related English NLP classification datasets, not across modalities, code, tool-use tasks, or languages. Finally, F^2C introduces several hyperparameters (e.g., CC set size cap, confidence thresholds, temperature) that we do not exhaustively tune.

Ethical Considerations

We use only publicly available datasets and open-weight models, with no new human data collection. All datasets used in this work are established benchmarks obtained via the Hugging Face `datasets` library (Lhoest et al., 2021) or their official repositories, each under its original license. These corpora are designed for evaluating language understanding and reasoning and contain de-identified, non-sensitive text drawn from newswire, Wikipedia, instructional materials, or crowdsourced fictional narratives. While some datasets may include named entities (e.g., public figures in news excerpts), to the best of our knowledge none contain contact details or other sensitive personal identifiers.

References

Norah Alzahrani, Hisham Alyahya, Yazeed Alnumay, Sultan AlRashed, Shaykhah Alsubaie, Yousef Almushayqih, Faisal Mirza, Nouf Alotaibi, Nora Al-Twairesh, Areeb Alowisheq, M Saiful Bari, and Haidar Khan. 2024. When benchmarks are targets: Revealing the sensitivity of large language model leaderboards. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13787–13805, Bangkok, Thailand. Association for Computational Linguistics.

Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, Geeta Chauhan, Anjali Chourdia, Will Constable, Alban Desmaison, Zachary DeVito, Elias Ellison, Will Feng, Jiong Gong, Michael Gschwind, and 30 others. 2024. PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation. In *29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2 (ASPLOS '24)*. ACM.

Stephen H. Bach, Victor Sanh, Zheng-Xin Yong, Albert Webson, Colin Raffel, Nihal V. Nayak, Abheesht Sharma, Taewoon Kim, M Saiful Bari, Thibault Fevry, Zaid Alyafeai, Manan Dey, Andrea Santilli, Zhiqing Sun, Srulik Ben-David, Canwen Xu, Gunjan Chhablani, Han Wang, Jason Alan Fries, and 8 others. 2022. Promptsource: An integrated development environment and repository for natural language prompts. *Preprint*, arXiv:2202.01279.

Lukas Biewald. 2020. Experiment tracking with weights and biases. Software available from wandb.com.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Bowen Cao, Deng Cai, Zhisong Zhang, Yuexian Zou, and Wai Lam. 2024. On the worst prompt performance of large language models. In *Advances in Neural Information Processing Systems*, volume 37, pages 69022–69042. Curran Associates, Inc.

Anwoy Chatterjee, H S V N S Kowndinya Renduchintala, Sumit Bhatia, and Tanmoy Chakraborty. 2024. POSIX: A prompt sensitivity index for large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 14550–14565, Miami, Florida, USA. Association for Computational Linguistics.

Jianhao Chen, Zishuo Xun, Bocheng Zhou, Han Qi, Hangfan Zhang, Qiaosheng Zhang, Yang Chen, Wei Hu, Yuzhong Qu, Wanli Ouyang, and Shuyue Hu. 2025. Do we truly need so many samples? multi-llm repeated sampling efficiently scales test-time compute. *Preprint*, arXiv:2504.00762.

Xinyun Chen, Renat Aksitov, Uri Alon, Jie Ren, Kefan Xiao, Pengcheng Yin, Sushant Prakash, Charles Sutton, Xuezhi Wang, and Denny Zhou. 2024. Universal self-consistency for large language models. In *ICML 2024 Workshop on In-Context Learning*.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Jamie Cummins. 2025. The threat of analytic flexibility in using large language models to simulate human data: A call to attention. *arXiv preprint arXiv:2509.13397*.

Arghya Datta, Subhrangshu Nandi, Jingcheng Xu, Greg Ver Steeg, He Xie, Anoop Kumar, and Aram Galstyan. 2023. Measuring and mitigating local instability in deep neural networks. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 2810–2823, Toronto, Canada. Association for Computational Linguistics.

Marie-Catherine de Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. *Proceedings of Sinn und Bedeutung*, 23(2):107–124.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychol. Bull.*, 76(5):378–382.

Junbo Fu, Guoshuai Zhao, Yimin Deng, Yunqi Mi, and Xueming Qian. 2024. Learning to paraphrase for alignment with LLM preference. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 2394–2407, Miami, Florida, USA. Association for Computational Linguistics.

Tingchen Fu and Fazl Barez. 2025. Same question, different words: A latent adversarial framework for prompt robustness. *Preprint*, arXiv:2503.01345.

Leo Gao, Tom Dupre la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. 2025. Scaling and evaluating sparse autoencoders. In *The Thirteenth International Conference on Learning Representations*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Sylvain Gugger, Lysandre Debut, Thomas Wolf, Philipp Schmid, Zachary Mueller, Sourab Mangrulkar, Marc Sun, and Benjamin Bossan. 2022. Accelerate: Training and inference at scale made simple, efficient and adaptable. `https://github.com/huggingface/accelerate`.

Neel Guha, Julian Nyarko, Daniel E. Ho, Christopher Re, Adam Chilton, Aditya Narayana, Alex Chohlas-Wood, Austin Peters, Brandon Waldon, Daniel Rockmore, Diego Zambrano, Dmitry Talisman, Enam Hoque, Faiz Surani, Frank Fagan, Galit Sarfaty, Gregory M. Dickinson, Haggai Porat, Jason Hegland, and 21 others. 2023. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *Preprint*, arXiv:1503.02531.

F K Hoehler. 2000. Bias and prevalence effects on kappa viewed in terms of sensitivity and specificity. *J. Clin. Epidemiol.*, 53(5):499–503.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2023. Large language models can self-improve. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1051–1068, Singapore. Association for Computational Linguistics.

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the*

2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, and 13 others. 2021. Datasets: A community library for natural language processing. *Preprint*, arXiv:2109.02846.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, Dublin, Ireland. Association for Computational Linguistics.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. 2022. PEFT: State-of-the-art parameter-efficient fine-tuning methods. `https://github.com/huggingface/peft`.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. In *Advances in Neural Information Processing Systems*, volume 35, pages 17359–17372. Curran Associates, Inc.

Moran Mizrahi, Guy Kaplan, Dan Malkin, Rotem Dror, Dafna Shahaf, and Gabriel Stanovsky. 2024. State of what art? a call for multi-prompt LLM evaluation. *Transactions of the Association for Computational Linguistics*, 12:933–949.

Anthony Moi and Nicolas Patry. 2023. HuggingFace's Tokenizers.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, San Diego, California. Association for Computational Linguistics.

Grigor Nalbandyan, Rima Shahbazyan, and Evelina Bakhturina. 2025. SCORE: Systematic COnsistency and robustness evaluation for large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 3: Industry Track)*, pages 470–484, Albuquerque, New Mexico. Association for Computational Linguistics.

Lilian Ngweta, Kiran Kate, Jason Tsay, and Yara Rizk. 2025. Towards llms robustness to changes in prompt format styles. *Preprint*, arXiv:2504.06969.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

OpenAI. 2025. Gpt-5 system card. Technical report, OpenAI. Accessed: September 28, 2025.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. WiC: the word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1267–1273, Minneapolis, Minnesota. Association for Computational Linguistics.

Felipe Maia Polo, Ronald Xu, Lucas Weber, Mírian Silva, Onkar Bhardwaj, Leshem Choshen, Allysson Flavio Melo de Oliveira, Yuekai Sun, and Mikhail Yurochkin. 2024. Efficient multi-prompt evaluation of LLMs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Yao Qiang, Subhrangshu Nandi, Ninareh Mehrabi, Greg Ver Steeg, Anoop Kumar, Anna Rumshisky, and Aram Galstyan. 2024. Prompt perturbation consistency learning for robust language models. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 1357–1370, St. Julian's, Malta. Association for Computational Linguistics.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Harsh Raj, Vipul Gupta, Domenic Rosati, and Subhabrata Majumdar. 2025. Semantic consistency for assuring reliability of large language models. *Preprint*, arXiv:2308.09138.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. 2011. Choice of Plausible Alternatives: An Evaluation of Commonsense Causal Reasoning. In *AAAI Spring Symposium on Logical Formalizations of Commonsense Reasoning*, Stanford University.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Winogrande: An adversarial winograd schema challenge at scale. *Preprint*, arXiv:1907.10641.

Abel Salinas and Fred Morstatter. 2024. The butterfly effect of altering prompts: How small changes and jailbreaks affect large language model performance. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 4629–4651, Bangkok, Thailand. Association for Computational Linguistics.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2024. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. In *The Twelfth International Conference on Learning Representations*.

Rulin Shao, Shuyue Stella Li, Rui Xin, Scott Geng, Yiping Wang, Sewoong Oh, Simon Shaolei Du, Nathan Lambert, Sewon Min, Ranjay Krishna, Yulia Tsvetkov, Hannaneh Hajishirzi, Pang Wei Koh, and Luke Zettlemoyer. 2026. Spurious rewards: Rethinking training signals in rlvr. *Preprint*, arXiv:2506.10947.

Lujia Shen, Yuwen Pu, Shouling Ji, Changjiang Li, Xuhong Zhang, Chunpeng Ge, and Ting Wang. 2024. Improving the robustness of transformer-based large language models with dynamic attention. In *Proceedings 2024 Network and Distributed System Security Symposium*, NDSS 2024. Internet Society.

Karan Singhal, Tao Tu, Juraj Gottweis, Rory Sayres, Ellery Wulczyn, Mohamed Amin, Le Hou, Kevin Clark, Stephen R Pfohl, Heather Cole-Lewis, Darlene Neal, Qazi Mamunur Rashid, Mike Schaekermann, Amy Wang, Dev Dash, Jonathan H Chen, Nigam H Shah, Sami Lachgar, Philip Andrew Mansfield, and 16 others. 2025. Toward expert-level medical question answering with large language models. *Nat. Med.*, 31(3):943–950.

Jiuding Sun, Chantal Shaib, and Byron C Wallace. 2024. Evaluating the zero-shot robustness of instruction-tuned language models. In *The Twelfth International Conference on Learning Representations*.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. 2024. Steering language models with activation engineering. *Preprint*, arXiv:2308.10248.

Anton Voronov, Lena Wolf, and Max Ryabinin. 2024. Mind your format: Towards consistent evaluation of in-context learning improvements. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 6287–6310, Bangkok, Thailand. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*, pages 3266–3280, Red Hook, NY, USA. Curran Associates, Inc.

Weixuan Wang, Barry Haddow, Alexandra Birch, and Wei Peng. 2024. Assessing factual reliability of large language model knowledge. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 805–819, Mexico City, Mexico. Association for Computational Linguistics.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Huggingface's transformers: State-of-the-art natural language processing. *Preprint*, arXiv:1910.03771.

Tianyi Yan, Fei Wang, James Y. Huang, Wenxuan Zhou, Fan Yin, Aram Galstyan, Wenpeng Yin, and Muhao Chen. 2024. Contrastive instruction tuning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 10288–10302, Bangkok, Thailand. Association for Computational Linguistics.

Jingyuan Yang, Dapeng Chen, Yajing Sun, Rongjun Li, Zhiyong Feng, and Wei Peng. 2024. Enhancing semantic consistency of large language models through model editing: An interpretability-oriented approach. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3343–3353, Bangkok, Thailand. Association for Computational Linguistics.

Jingyuan Yang, Rongjun Li, Weixuan Wang, Ziyu Zhou, Zhiyong Feng, and Wei Peng. 2025. Lf-steering: Latent feature activation steering for enhancing semantic consistency in large language models. *Preprint*, arXiv:2501.11036.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Yukun Zhao, Lingyong Yan, Weiwei Sun, Guoliang Xing, Shuaiqiang Wang, Chong Meng, Zhicong

Cheng, Zhaochun Ren, and Dawei Yin. 2024. Improving the robustness of large language models via consistency alignment. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 8931–8941, Torino, Italia. ELRA and ICCL.

Chunting Zhou, Junxian He, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Prompt consistency for zero-shot task generalization. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2613–2626, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

A Appendix

A.1 Why swarm distillation moves all students towards their average?

Theorem A.1 (Mixture-teacher decomposition)**.** *Let Y be finite. For a fixed input with K semantically equivalent formats, let $q_1, \dots, q_K \in \Delta^{|Y|}$ be teacher distributions (treated as constants) and let $p_j(\theta) \in \Delta^{|Y|}$ be the student for format j. For weights $w_i^{(j)} \geq 0$ with $\sum_{i=1}^{K} w_i^{(j)} = 1$, define*

$$\bar{q}^{(j)} := \sum_{i=1}^{K} w_i^{(j)} q_i.$$

Then, for each j,

$$\sum_{i=1}^{K} w_i^{(j)} \operatorname{KL}\big(q_i \,\|\, p_j(\theta)\big) = \operatorname{KL}\big(\bar{q}^{(j)} \,\|\, p_j(\theta)\big) + \sum_{i=1}^{K} w_i^{(j)} \operatorname{KL}\big(q_i \,\|\, \bar{q}^{(j)}\big). \tag{12}$$

and the last sum is constant in θ. Hence ∇_θ of the left side equals $\nabla_\theta \operatorname{KL}(\bar{q}^{(j)} \| p_j(\theta))$. In the uniform case $w_i^{(j)} = \frac{1}{K}$, every p_j is pulled toward the same average $\bar{q} = \frac{1}{K}\sum_i q_i$.

Proof. Use $\log \frac{q_i}{p_j} = \log \frac{q_i}{\bar{q}^{(j)}} + \log \frac{\bar{q}^{(j)}}{p_j}$, sum over i with weights $w_i^{(j)}$, and note that $\sum_i w_i^{(j)} q_i = \bar{q}^{(j)}$. The term $\sum_i w_i^{(j)} \operatorname{KL}(q_i \| \bar{q}^{(j)})$ contains no θ. $\square$

A.2 Datasets

We derive our validation split from the original training data (the original train is partitioned into our train and validation). Unless the original split is smaller, we hold out up to 1,000 examples for validation (600 for HellaSwag). For low-resource datasets, we set the derived validation size to match the size of the official validation split. When the remaining training pool is large, we cap it at 10,000 examples via uniform random sampling with a fixed seed (see Table 4). Before stratification, we deduplicate base examples whose *rendered* prompt text would otherwise repeat across splits.

All datasets used in this work are publicly available under research-oriented licenses. Specifically, RTE, CB, WSC, COPA, WiC, WinoGrande, HellaSwag, and ANLI are accessible via the Hugging Face `datasets` library (Lhoest et al., 2021) and retain their original license terms (some permit commercial use, while others restrict use to non-commercial research). StoryCloze 2016 is available from the official ROCStories website for research use only. We do not redistribute any dataset; instead, users may obtain them directly from their original sources or through Hugging Face. Each dataset preserves its original licensing and citation requirements, and all usage in this work complies with those terms.

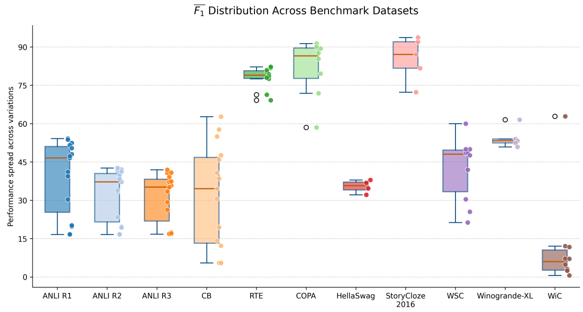

Figure 3: Per-dataset distribution of F_1 across prompt variations for the `Qwen2.5-3B-Instruct`.

Dataset	Train	Val	Test	#Formats	#Labels
ANLI R1	10,000	1,000	1,000	15	3
ANLI R2	10,000	1,000	1,000	15	3
ANLI R3	10,000	1,000	1,200	15	3
CB	194	56	56	15	3
RTE	1,488	1,000	277	10	2
COPA	300	100	100	8	2
HellaSwag	10,000	600	600	4	4
StoryCloze	871	1,000	1,871†	5	2
WSC	430	95	95	10	2
WinoGrande	10,000	1,000	1,267	5	2
WiC	4,428	1,000	638	10	2

Table 4: "Test" denotes the official *validation* set used for evaluation because most tasks do not release test labels. #Formats is the number of PromptSource (Bach et al., 2022) templates used to construct each dataset's prompt variations. †StoryCloze has no public train split; we use the 2016 validation file for train/val and the 2016 test file for evaluation.

Dataset	Val	Strict-majority (%)	Pseudo-label F_1
ANLI R1	1,000	93.0	66.89
ANLI R2	1,000	92.6	69.33
ANLI R3	1,000	91.6	59.34
CB	56	78.6	35.82
RTE	1,000	97.0	83.10
COPA	100	97.0	92.78
HellaSwag	600	67.7	37.83
StoryCloze	1,000	100.0	89.18
WSC	95	91.6	38.71
WinoGrande	1,000	100.0	57.60
WiC	1,000	99.6	1.20
Aggregate	7,851	93.9	59.50

Table 5: Majority pseudo-label quality for `Qwen2.5-3B-Instruct` on validation splits. Aggregate F_1 is averaged across datasets by the number of strict-majority examples.

A.3 Implementation Details

Per-variation scoring. Let $y_c^{(v)}$ tokenize into $T_{i,v,c}$ answer tokens. We define the average token log-probability of choosing label ℓ_c under template v:

$$\mathrm{LL}_i[v,c] = \frac{1}{T_{i,v,c}} \sum_{t=1}^{T_{i,v,c}} \log p_\theta\big(y_t \mid x_i^{(v)}, y_{<t}\big), \tag{13}$$

where y_t is the t-th answer token of $y_c^{(v)}$. These scores induce a per-variation distribution over labels:

$$\pi_{i,v,c} = \frac{\exp(\mathrm{LL}_i[v,c])}{\sum_{c'=1}^{L} \exp(\mathrm{LL}_i[v,c'])}. \tag{14}$$

The predicted label for variation v is

$$\hat{y}_{i,v} = \arg \max_{c \in \{1,\ldots,L\}} \pi_{i,v,c}. \tag{15}$$

LoRA Configuration. We use LoRA (Hu et al., 2022) to reduce the number of trainable parameters and the compute required per step. We apply LoRA adapters in every transformer block to the attention and MLP projections, keeping backbone weights frozen. In our setup, we use rank r=16, scaling α=32, and dropout 0.05.

Model Selection. We select checkpoints by the highest $\overline{F_1}$ on the validation set. The $\overline{F_1}$ decreases under overfitting to particular prompts and when the majority label diverges from the gold label, and provides a robust target while P_o and σ_{F_1} quantify consistency.

Dataset	Model	λ_{CCE}	$\tau_{\text{unanimous}}$	$k_{\max}$	$f_{\min}$	$f_{\max}$	t	β_{jsd}
ANLI R1	Both	1.0	1.5	4	0.001	0.015	2.0	0.05
ANLI R2	Both	1.5	1.2	4	0.05	0.30	2.0	0.30
ANLI R3	Both	1.0	1.2	5	0.05	0.35	2.0	0.20
CB	Both	1.0	9	7	0.01	0.10	2.0	0.01
RTE	Both	0.4	1.4	3	0.05	0.35	2.2	0.35
COPA	Both	1.0	1.0	2	0.005	0.05	3.0	0.05
HellaSwag	Both	1.0	0.5	2	0.01	0.10	2.0	0.01
StoryCloze	Both	1.0	1.0	3	0.01	0.10	2.0	0.05
WSC	Both	1.0	2.5	3	0.20	0.60	2.0	1.00
WinoGrande	Both	1.0	2.2	2	0.05	0.20	2.0	0.15
WiC	Qwen	1.0	12	3	0.01	0.10	2.0	0.05
	Llama	1.0	0.65	7	0.10	1.00	1.0	0.20

Table 6: F^{2}C hyperparameters used for the first experiment. "Both" means the same values were used for `Qwen2.5-3B-Instruct` and `Llama-3.2-3B-Instruct`.

Majority Pseudo-Label Quality. F^{2}C uses majority agreement across prompt variations as a pseudo-labeling signal. To measure how reliable this signal is in our setting, we evaluate the strict-majority labels produced by the base `Qwen2.5-3B-Instruct` model. For each validation example, the pseudo-label is the label predicted by more than half of the prompt variations; examples without a strict majority are excluded from the F_1 calculation. Table 5 reports validation-set size, strict-majority ratio, and pseudo-label macro/binary F_1. Overall, strict-majority labels are available for 7,372 of 7,851 examples (93.9%). Averaging the per-dataset F_1 scores by the number of strict-majority examples gives 59.5.

F^{2}C Hyperparameters. The F^{2}C hyperparameters were chosen separately for each dataset. We did not conduct an exhaustive per-dataset search. Instead, we used a small set of manual configurations and selected checkpoints by validation $\overline{F_1}$ as described above. Table 6 reports the values that produced the F^{2}C rows in Tables 1 and 2.

Compute, Infrastructure, and Packages. We fine-tune two open-weight 3B instruction-tuned models. The first experiment is conducted on both `Qwen2.5-3B-Instruct` (Qwen et al., 2025) and `Llama-3.2-3B-Instruct` (Grattafiori et al., 2024), while the out-of-domain and unseen-variation generalization experiments use `Qwen2.5-3B-Instruct`. Fine-tuning is performed using the Hugging Face `transformers` (Wolf et al., 2020) and `peft` (Mangrulkar et al., 2022) libraries with LoRA adapters (see section A.3).

Training was conducted on a GPU cluster running Ubuntu 22.04.5 LTS with NVIDIA Container Toolkit. Smaller datasets with fewer

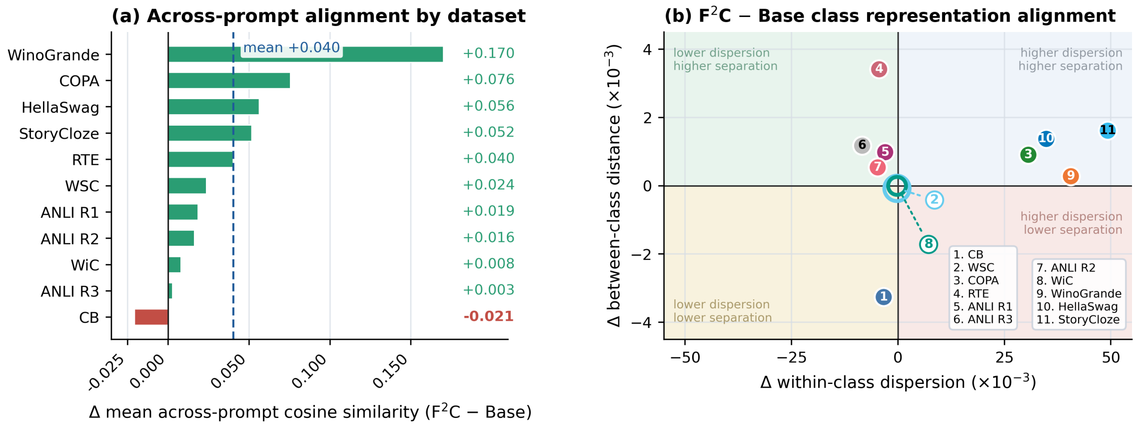

Figure 4: Representation analysis for `Qwen2.5-3B-Instruct`. In panel (a), each bar shows how much the average cosine similarity between prompt formats of the same example changes after F^{2}C. In panel (b), the x-axis shows the change in within-class distance, where negative values mean examples with the same label move closer together. The y-axis shows the change in between-class distance, where positive values mean different label groups move farther apart. Both axes show F^{2}C minus Base.

prompt variations were trained on a pool of seven NVIDIA RTX A6000 GPUs (48 GB VRAM each). Larger datasets were trained on a single NVIDIA A100 GPU (80 GB). Depending on dataset size and number of prompt formats, total training time ranged from approximately 3 to 23 hours per dataset.

To maintain reproducibility and efficiency, we use mixed-precision (`bf16`) training, gradient accumulation, and uniform random seeds across runs. Experiments are orchestrated via custom shell scripts and Weights & Biases (Biewald, 2020) logging for monitoring. We did not use model parallelism or distributed fine-tuning beyond single-node multi-GPU setups.

We use Python 3.9 with PyTorch 2.8.0 (Ansel et al., 2024), `transformers` 4.56.1 (Wolf et al., 2020), `peft` 0.17.1 (Mangrulkar et al., 2022), `accelerate` 1.10.1 (Gugger et al., 2022), `datasets` 4.1.0 (Lhoest et al., 2021), and `tokenizers` 0.22.0/`sentencepiece` 0.2.1 (Kudo and Richardson, 2018; Moi and Patry, 2023).

A.4 Representation Alignment Details

For each example and prompt format, we extract the final-layer hidden state immediately before the answer begins. This is the model state used to start generating the answer. For the class-level analysis, we first average prompt variations' representations of each example into one vector. We then group these example vectors by the dataset label. The within-class distance is the average cosine distance from an example vector to the mean vector of its own label group. The between-class distance is the average cosine distance between the mean vectors of different label groups. Figure 4 visualizes the changes after F^{2}C, and Table 7 reports the corresponding values.

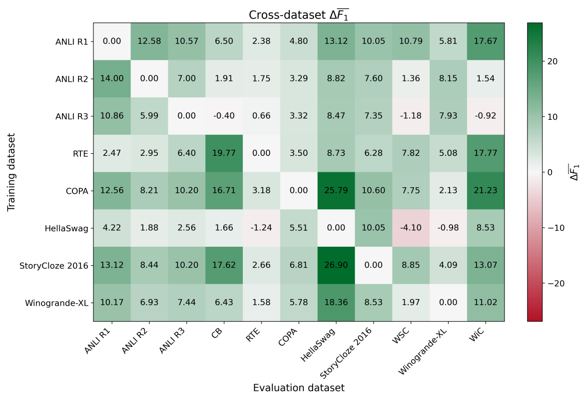

Figure 5: $\Delta\overline{F_1}$: Cross-dataset transfer under F^{2}C. Each cell shows the change relative to the base model when training on the row dataset and evaluating on the column dataset. Green indicates improvement; red indicates degradation.

	Across-prompt similarity			Within-class distance			Between-class distance		
Dataset	Base	F^{2}C	Δ	Base	F^{2}C	Δ	Base	F^{2}C	Δ
ANLI R1	0.879	0.897	**+0.019**	0.017	0.014	**-0.003**	0.026	0.027	**+0.001**
ANLI R2	0.879	0.895	**+0.016**	0.018	0.013	**-0.005**	0.024	0.024	**+0.001**
ANLI R3	0.884	0.887	**+0.003**	0.022	0.014	**-0.008**	0.016	0.017	**+0.001**
CB	0.895	0.874	-0.021	0.014	0.010	**-0.003**	0.018	0.015	-0.003
RTE	0.947	0.988	**+0.040**	0.022	0.018	**-0.004**	0.028	0.032	**+0.003**
COPA	0.881	0.957	**+0.076**	0.032	0.063	+0.031	0.004	0.005	**+0.001**
HellaSwag	0.873	0.929	**+0.056**	0.022	0.057	+0.035	0.003	0.004	**+0.001**
StoryCloze	0.933	0.985	**+0.052**	0.032	0.081	+0.049	0.004	0.005	**+0.002**
WSC	0.875	0.899	**+0.024**	0.012	0.012	0.000	0.001	0.001	0.000
WinoGrande	0.805	0.975	**+0.170**	0.037	0.077	+0.041	0.000	0.000	0.000
WiC	0.851	0.859	**+0.008**	0.005	0.005	0.000	0.002	0.002	0.000

Table 7: Full representation-alignment results. Across-prompt similarity is the mean cosine similarity between hidden states from different prompt formats of the same example. For the class-distance columns, each example is represented by averaging the representations of its prompt formats. Δ columns report F^{2}C minus Base; blue indicates the favorable direction (higher across-prompt cosine, lower within-class distance, or higher between-class distance), and orange indicates the opposite.

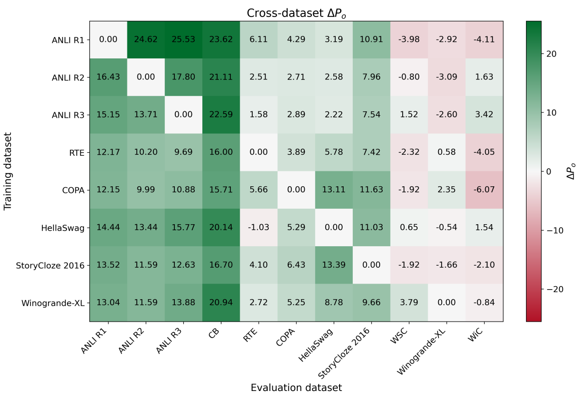

Figure 6: Cross-dataset transfer for observed agreement P_o. Each cell shows ΔP_o relative to the base model (train on rows, evaluate on columns). Green indicates improvement; red indicates degradation.

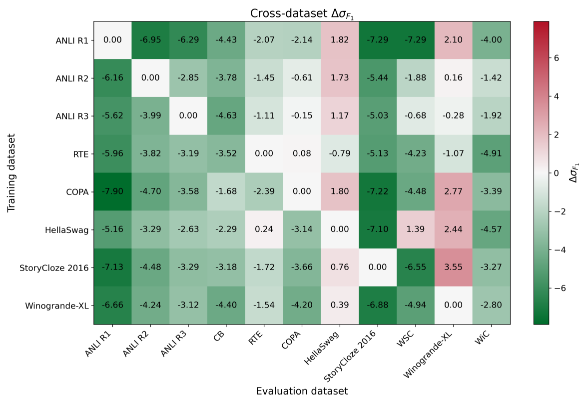

Figure 7: $\Delta\sigma_{F_1}$ (lower is better): Cross-dataset transfer under F^{2}C. Each cell shows the change relative to the base model when training on the row dataset and evaluating on the column dataset. Green indicates improvement; red indicates degradation.