LLM-Guided Co-Training for Text Classification

Md Mezbaur Rahman
Computer Science
University of Illinois Chicago
mrahma56@uic.edu

Cornelia Caragea
Computer Science
University of Illinois Chicago
cornelia@uic.edu

Abstract

In this paper, we introduce a novel weighted co-training approach that is guided by Large Language Models (LLMs). Namely, in our co-training approach, we use LLM labels on unlabeled data as target labels and co-train two encoder-only based networks that train each other over multiple iterations: first, all samples are forwarded through each network and historical estimates of each network's confidence in the LLM label are recorded; second, a dynamic importance weight is derived for each sample according to each network's belief in the quality of the LLM label for that sample; finally, the two networks exchange importance weights with each other—each network back-propagates all samples weighted with the importance weights coming from its peer network and updates its own parameters. By strategically utilizing LLM-generated guidance, our approach significantly outperforms conventional SSL methods, particularly in settings with abundant unlabeled data. Empirical results show that it achieves state-of-the-art performance on 4 out of 5 benchmark datasets and ranks first among 14 compared methods according to the Friedman test. Our results highlight a new direction in semi-supervised learning—where LLMs serve as knowledge amplifiers, enabling backbone co-training models to achieve state-of-the-art performance efficiently.

1 Introduction

Semi-Supervised Learning (SSL) has gained substantial attention for its ability to improve model performance by leveraging a small set of labeled data alongside a large pool of unlabeled data (Wang et al., 2022; Van Engelen and Hoos, 2020). A widely used SSL strategy is pseudo-labeling that relies on the model itself to generate pseudo-labels for unlabeled data and uses these pseudo-labels during training. FixMatch exemplifies this by selecting high-confidence predictions exceeding a fixed threshold as pseudo-labels. However, this rigid threshold often excludes diverse examples with lower confidence scores. To address this, recent methods either: (1) adaptively adjust class-wise thresholds to incorporate more examples (Zhang et al., 2021), or (2) assign confidence-based weights to all unlabeled samples (Chen et al., 2023). While these strategies improve data utilization, they remain dependent on the model's own potentially unreliable predictions, introducing noisy pseudo-labels that can harm generalization, especially due to overconfident yet incorrect predictions.

Recent advances have highlighted the potential of LLMs in producing high-quality pseudo-labels for SSL. The VerifyMatch framework (Park and Caragea, 2024) incorporates LLM-generated pseudo-labels into the SSL setup and tackles their noise by first identifying noisy samples using a verifier network. The verifier flags low-confidence pseudo-labels (based on softmax probability) and applies MixUp (Zhang et al., 2018) to blend them with human-annotated data, thereby mitigating noise and improving label quality. Although this approach presents a strong case in combining LLM-generated pseudo-labels into the SSL setup, it uses only the confidence scores from the current training iteration from a single verifier model to distinguish between potentially correct and noisy samples. This limited view misses opportunities to capture richer training dynamics across time or across multiple models equipped with complementary knowledge and learning abilities, which can further enhance the robustness of pseudo-labeling.

In this paper, we propose a novel co-training framework that leverages LLM-generated pseudo-labels without modifying or discarding any samples. Our approach diverges from VerifyMatch (Park and Caragea, 2024), which mitigates noise by altering low-confidence pseudo-labels (e.g., via MixUp with human-labeled data). Instead, we pre-

---

We make our implementation publicly available at https://github.com/mezbaur-rahman/Lg-CoTrain

serve all pseudo-labels and reweight them based on their estimated reliability, allowing the model to learn from the entire data while reducing the influence of incorrect pseudo-labels. Moreover, while VerifyMatch uses a single verifier model to estimate the label quality, we employ a dual model-based weighting mechanism. Our framework also diverges from traditional co-training methods (Blum and Mitchell, 1998) that exchange the most confident pseudo-labels between models; we instead exchange importance weights that reflect each model's belief in the quality of the LLM pseudo-labels. Leveraging the insights from Swayamdipta et al. (2020), our framework goes beyond relying solely on confidence from the current training epoch. Instead, it utilizes historical training dynamics and the consistency and disagreements of the two models' predictions across epochs to assign higher weights to likely correct samples and lower weights to potentially incorrect ones.

Our contributions are as follows:

- We propose LG-CoTRAIN, a novel co-training method that leverages pseudo-labels generated by LLMs alongside a small set of human-annotated labels. Two peer networks are trained jointly, each guiding the other using its training dynamics to maintain divergence and promote complementary learning.

- LG-CoTRAIN consistently outperforms all standard SSL baselines across five benchmark classification datasets and exceeds the performance of the LLMs themselves on four out of five datasets. Remarkably, it also achieves the top-1 rank in the Friedman test among 14 methods, reducing the mean error rate by 1.56% compared to the second-best baseline.

- We conduct extensive ablation studies and qualitative analyses to validate the effectiveness of our proposed method in both balanced and imbalanced data distributions. We also evaluate the effectiveness of LG-CoTRAIN in several natural language understanding tasks.

2 Related Works

2.1 Large Language Models

Large Language Models (LLMs) have demonstrated remarkable capabilities across a wide range of NLP tasks, including text generation, question answering, summarization, and dialogue generation (Zhang et al., 2020; Touvron et al., 2023; Chen

et al., 2024). Their generalization ability and scalability have made them a key component in modern NLP pipelines. Earlier approaches such as BERT (Devlin et al., 2019) relied on task-specific fine-tuning, requiring labeled data and weight updates for each new task. However, as LLMs have grown in size and training scope, prompting has emerged as a more flexible alternative (Brown et al., 2020; Wei et al., 2022). Prompting enables LLMs to perform tasks by conditioning on task descriptions or a few in-context examples written in natural language, allowing for zero-shot and few-shot learning. In our work, we leverage LLM prompting to generate high-quality pseudo-labels for unlabeled data. We also include prompting-based baselines in our evaluations to assess the effectiveness of our method relative to direct LLM predictions.

2.2 Semi Supervised Learning

Semi-Supervised Learning (SSL) has led to a surge of research based on self-training and pseudo-labeling (Xie et al., 2020b; Sohn et al., 2020; Sosea and Caragea, 2022; Sadat and Caragea, 2022; Sosea and Caragea, 2023; Chen et al., 2023; Zou and Caragea, 2023; Zou et al., 2023; Gyawali et al., 2024; Hosseini and Caragea, 2023), where models are trained using a mix of labeled and pseudo-labeled data. Popular methods like FixMatch (Sohn et al., 2020) and FlexMatch (Zhang et al., 2021) rely on confidence thresholding to select pseudo-labels, but it often results in discarding a large portion of the unlabeled data. To improve data utilization, Chen et al. (2023) proposed SoftMatch, which retains all samples by assigning lower weights to low-confidence samples during training. However, whether through hard thresholding or confidence-based weighting, methods that rely solely on a single model for pseudo-labeling remain vulnerable to assigning high weights to incorrect pseudo-labels, especially in the early stages of training.

2.3 Learning with Noisy Labels

The challenge of learning from noisy labels has been explored for decades (Angluin and Laird, 1988), with renewed focus in deep learning due to neural networks' tendency to memorize mislabeled data (Arpit et al., 2017). Noisy labels—corrupted deviations from ground truth labels—can significantly degrade model performance. To combat this, several strategies have been proposed to distinguish between reliable and unreliable training samples. MentorNet (Jiang et al., 2018) proposed an auxil-

iary network to select clean samples, but its reliance on a single model introduces confirmation bias. Co-teaching (Han et al., 2018a) addressed this by training two peer networks that exchange low-loss samples, minimizing mutual errors. DivideMix (Li et al., 2020) treated noisy label learning as a semi-supervised task, using Gaussian Mixture Models to split data into confident and unconfident subsets and co-trained two models on complementary partitions. VerifyMatch (Park and Caragea, 2024) used LLM-generated pseudo-labels, which are generally of high quality but still noisy. VerifyMatch introduces a verifier network to estimate label confidence: low-confidence, noisy samples are refined using MixUp (Zhang et al., 2018) and human-annotated data, while high-confidence samples are used directly.

In contrast, we propose a dual-model semi-supervised framework that integrates LLM-generated pseudo-labels while training two classifiers in parallel. Unlike VerifyMatch (Park and Caragea, 2024), which employs a single-model setup with a verifier, our approach leverages a dual-model architecture and dynamically weights training samples based on cross-model training dynamics. Another key distinction is our use of multi-epoch training dynamics to assess model behavior consistency over time—an aspect VerifyMatch overlooks. Furthermore, our method differs from DivideMix (Li et al., 2020) and Co-Teaching (Han et al., 2018a), which rely on label exchange or sample selection during training. Instead, we retain all pseudo-labels without discarding or modifying them, and guide the training process through historical training dynamics based weighting. This strategy encourages model divergence and enhances robustness while preserving the integrity of the pseudo-labeled data.

3 Proposed Approach

In this section, we present LG-COTRAIN, our proposed approach for semi-supervised learning. We begin by defining the notation used throughout the method. We then describe the pseudo-label generation process using LLMs (§3.1). Next, we describe the weighting metrics to estimate the reliability of each pseudo-labeled sample (§3.2). Finally, we present our weighted co-training strategy, where two peer models are trained jointly using these weights, which are continuously updated during training (§3.3).

Notation. We consider a training dataset D consisting of a small labeled set $\mathcal{D}_l = \{(x_i^l, y_i^l)\}_{i=1}^{N_l}$ and a much larger unlabeled set $\mathcal{D}_u = \{x_j^u\}_{j=1}^{N_u}$. Here, x denotes an input sample (e.g., a text sequence), and $y \in \{1, \ldots, C\}$ denotes its corresponding class label. We define N_l, N_u, and C as the number of labeled samples, unlabeled samples, and class categories, respectively, where $N_u \gg N_l$.

3.1 Pseudo-Label Generation with LLMs

We utilize LLMs to generate pseudo-labels for unlabeled texts through task-specific prompting designed for classification. We use the LLMs in zero-shot or few-shot settings to generate pseudo-labels for the unlabeled samples.

Given an unlabeled input $x_j^u \in \mathcal{D}_u$, the prompt $P(x_j^u)$ formats the input into a query suitable for the LLM. The model then generates an output token sequence, from which a label word is extracted and mapped to class c, resulting in the pseudo-label $\tilde{y}_j = c$. This yields the pseudo-labeled set:

$$\mathcal{D}_{LG} = \{(x_j^u, \tilde{y}_j) \mid x_j^u \in \mathcal{D}_u, \tilde{y}_j \in \{1, \ldots, C\}\}$$

This process ensures that $\mathcal{D}_{LG}$ contains samples with valid class-aligned pseudo-labels. These samples serve as additional supervision during the co-training stage. The prompt templates and the prompting strategies are provided in Appendix **??**.

3.2 Weight Generation

The two networks (i.e., the classifiers) in our co-training setup are denoted as θ_1 and θ_2. At each training epoch, we assign importance weights to the samples from $\mathcal{D}_{LG}$ based on the training dynamics of each classifier. The goal is twofold: (1) to emphasize reliable pseudo-labeled examples, and (2) to maintain divergence between the classifiers, enabling them to learn complementary signals and suppress the effects of noisy supervision.

To assess sample reliability, we make use of two metrics—**confidence** and **variability**—as proposed in Dataset Cartography (Swayamdipta et al., 2020). These metrics capture the historical behavior of a model on each sample during training. **Confidence** reflects how confidently a model predicts the assigned label over iterations. For example, for a sample $(x, \tilde{y})$, the confidence of a classifier θ across e training iterations is computed as:

$$c_\theta(x, \tilde{y}) = \frac{1}{e} \sum_{t=1}^{e} p(\tilde{y}|x; \theta) \tag{1}$$

Variability captures how consistently a model predicts the assigned label over iterations, i.e., the standard deviation of these predictions, indicating how stable the model's belief in the label is:

$$v_\theta(x, \tilde{y}) = \sqrt{\frac{1}{e} \sum_{t=1}^{e} \left(p(\tilde{y}|x; \theta_t) - c_\theta(x, \tilde{y}) \right)^2} \quad (2)$$

While these metrics individually signal sample quality, they do not directly promote complementary learning between the two classifiers. To address this, inspired by prior works (Poesina et al., 2024; Sadat and Caragea, 2024), we use two asymmetric importance weighting schemes—λ_1 and λ_2—which leverage the confidence and variability of each sample in opposing ways. These weights are used to supervise the *peer* network, thereby encouraging divergence and suppressing mutual error reinforcement. Specifically, for each pseudo-labeled sample $(x^i, \tilde{y}^i)$ and each classifier, θ_1 and θ_2, we define:

$$\lambda_1^i = c_{\theta_1}(x^i, \tilde{y}^i) + v_{\theta_1}(x^i, \tilde{y}^i) \quad (3)$$

$$\lambda_2^i = c_{\theta_2}(x^i, \tilde{y}^i) - v_{\theta_2}(x^i, \tilde{y}^i) \quad (4)$$

As a result, examples that are easy to classify by both classifiers (i.e., high confidence and low variability) produce high values for both λ_1^i and λ_2^i. Conversely, hard examples (i.e., low confidence and low variability) yield low values for both λ_1^i and λ_2^i. For ambiguous examples, defined by high variability and moderate confidence by both classifiers, the weights differ between the classifiers. Specifically, the weight for θ_1 (i.e., λ_2^i) is low due to the classifiers showing only moderate confidence in these examples. That is, a confidence of 0.5 and a variability of 0.4 result in a low weight of 0.1. In contrast, the weight for θ_2 (i.e., λ_1^i) remains high even with moderate confidence, when the variability is high. That is, a confidence of 0.5 and variability of 0.4 result in a high weight of 0.9.

This weighting strategy gives higher weight to easy, likely correct examples, while down-weighting harder, noisier ones to reduce their impact on both classifiers. For ambiguous examples, where the classifiers exhibit uncertainty about the assigned labels, one classifier receives high weights for these examples, while the other receives low weights. This approach preserves divergence between the classifiers and allows them to learn complementary information from the data.

Algorithm 1 LG-CoTrain

1: **Input:** Labeled set $\mathcal{D}_l = \{(\mathbf{x}^i, y^i)\}_{i=1}^{N_l}$;
   Unlabeled set $\mathcal{D}_u = \{\mathbf{x}^i\}_{i=1}^{N_u}$; Classifiers θ_1, θ_2;
   Weight generation epochs T, Max co-training epochs E;
   Probability matrices $\mathcal{P}_1, \mathcal{P}_2 \in \mathbb{R}^{T \times |\mathcal{D}_{LG}|}$
2: Construct $\mathcal{D}_{LG} = \{(\mathbf{x}^i, \tilde{y}^i)\}_{i=1}^{N_{LG}} \leftarrow$ Sec. 3.1
3: Divide $\mathcal{D}_l$ into two balanced subsets $\mathcal{D}_{l_1}$ and $\mathcal{D}_{l_2}$
4: **for** iter $t \in \{1, 2, \ldots, T\}$ **do**
5:    Train θ_1 and θ_2 using $\mathcal{D}_{l_1}$ and $\mathcal{D}_{l_2}$, respectively;
6:    **for** each $i \in \mathcal{D}_{LG}$ **do**
7:       $\mathcal{P}_1[t, i] \leftarrow p_t(y^i \mid \mathbf{x}^i; \theta_1)$
8:       $\mathcal{P}_2[t, i] \leftarrow p_t(\tilde{y}^i \mid \mathbf{x}^i; \theta_2)$
9:    **end for**
10: **end for**
11: **for** each sample $i \in D_{LG}$ **do**
12:    $c_{\theta_1}^i, c_{\theta_2}^i \leftarrow$ GETCONF$(\mathcal{P}_1, i)$, GETCONF$(\mathcal{P}_2, i)$
       $v_{\theta_1}^i, v_{\theta_2}^i \leftarrow$ GETVAR$(\mathcal{P}_1, i)$, GETVAR$(\mathcal{P}_2, i)$
13:    $\lambda_1^i \leftarrow c_{\theta_1}^i + v_{\theta_1}^i$; $\lambda_2^i \leftarrow c_{\theta_2}^i - v_{\theta_2}^i$.
14: **end for**
15: Re-initialize θ_1, θ_2.
16: **for** iter $\epsilon \in \{1, 2, \ldots, E\}$ **do**
17:    **for** each mini-batch $B \in D_{LG}$ **do**
18:       $\mathcal{L}_1 \leftarrow \frac{1}{|B|} \sum_{i=0}^{|B|} \lambda_2^i * H(\tilde{y}^i, p_d(\mathbf{x}^i; \theta_1))$
19:       $\mathcal{L}_2 \leftarrow \frac{1}{|B|} \sum_{i=0}^{|B|} \lambda_1^i * H(\tilde{y}^i, p_d(\mathbf{x}^i; \theta_2))$
20:       **for** each example $i \in B$ **do**
21:          Update $c_{\theta_k}^i, v_{\theta_k}^i$ via Eq. (1), (2) for $k = 1, 2$
22:          $\lambda_1^i \leftarrow c_{\theta_1}(\mathbf{x}^i, \tilde{y}^i) + v_{\theta_1}(\mathbf{x}^i, \tilde{y}^i)$
23:          $\lambda_2^i \leftarrow c_{\theta_2}(\mathbf{x}^i, \tilde{y}^i) - v_{\theta_2}(\mathbf{x}^i, \tilde{y}^i)$
24:       **end for**
25:    **end for**
26:    $\theta_1 \leftarrow \theta_1 - \eta * \nabla\mathcal{L}_1$; $\theta_2 \leftarrow \theta_2 - \eta * \nabla\mathcal{L}_2$.
27: **end for**
28: Fine-tune θ_1 and θ_2 using $\mathcal{D}_{l_1}$ and $\mathcal{D}_{l_2}$ respectively

---

3.3 Weighted Co-Training

The weighted co-training module consists of three main steps, which are outlined below:

Initial Weight Assignment To initialize the importance weights for the samples in the pseudo-labeled set $\mathcal{D}_{LG}$, the classifiers θ_1 and θ_2 are first trained on different, equally sized subsets of the labeled dataset $\mathcal{D}_l$, denoted as $\mathcal{D}_{l_1}$ and $\mathcal{D}_{l_2}$, respectively. These subsets are constructed to maintain similar class distributions, ensuring a fair training process.

Next, we define two probability matrices, $\mathcal{P}_1, \mathcal{P}_2 \in \mathbb{R}^{T \times |\mathcal{D}_{LG}|}$, where each entry stores the predicted probabilities $p(\tilde{y}_i \mid x_i; \theta_1)$ and $p(\tilde{y}_i \mid x_i; \theta_2)$ for each sample i in $\mathcal{D}_{LG}$ across epochs $t = 1, \ldots, T$. These probabilities are recorded at each training epoch.

For each sample in $\mathcal{D}_{LG}$, we compute confidence and variability using Eq. (1) and Eq. (2), respectively, based on the corresponding probability matrix. These values are then substituted into Eq. (3) and Eq. (4) to determine the initial importance weights λ_1 and λ_2.

Co-Training In this step, we begin by re-initializing the two classifiers, θ_1 and θ_2. Both classifiers are then co-trained using the pseudo-labeled dataset $\mathcal{D}_{LG}$. After each training epoch, for model θ_1, each sample's loss is scaled by the weight metric λ_2; similarly, for model θ_2, each sample's loss is scaled by the weight metric λ_1. Specifically, for each mini-batch $B \subseteq \mathcal{D}_{LG}$, the losses $\mathcal{L}_1$ and $\mathcal{L}_2$ for θ_1 and θ_2, respectively, are computed as follows:

$$\mathcal{L}_1 = \frac{1}{|B|} \sum_{i=1}^{|B|} \lambda_2^i * H(\tilde{y}^i, p_d(x^i; \theta_1)). \quad (5)$$

$$\mathcal{L}_2 = \frac{1}{|B|} \sum_{i=1}^{|B|} \lambda_1^i * H(\tilde{y}^i, p_d(x^i; \theta_2)). \quad (6)$$

Here, p_d represents the probability distribution over the labels predicted by a classifier for a pseudo-labeled sample x^i in B, and H denotes the standard cross-entropy loss.

After calculating the loss for both classifiers, the confidence and variability metrics are updated using Eq. (1) and Eq. (2). It is important to note that the probabilities predicted by the classifiers during the final epoch of Step 1 are used as the initial probabilities for both classifiers. These initial probabilities are included in the calculation of the mean and standard deviation throughout the co-training epochs. Subsequently, both sets of weights are recalculated using Eq. (3) and Eq. (4), based on the updated confidence and variability metrics, to be applied in the next epoch.

This iterative process allows the classifiers to co-train by exchanging complementary information derived from their respective training dynamics across epochs. Crucially, each model guides the other by dynamically adjusting the influence of each training sample using the weight metric λ.

Fine-tuning Upon completing the co-training process after a predefined maximum of E epochs, the classifiers θ_1 and θ_2 are fine-tuned using the labeled subsets $\mathcal{D}_{l_1}$ and $\mathcal{D}_{l_2}$, respectively.

During inference, we ensemble the two models by averaging their softmax outputs and taking the argmax to get the final prediction.

4 Experiments

In this section, we describe the datasets used for evaluation (§4.1), present our experimental setup and baselines (§4.2), and discuss the results (§4.3).

4.1 Datasets

We conduct experiments on five widely used classification datasets: IMDB (Maas et al., 2011), AG News (Zhang et al., 2015), Yahoo! Answers (Chang et al., 2008), Amazon Reviews (McAuley and Leskovec, 2013), and Yelp Reviews (Yelp.com), along with three natural language understanding (NLU) tasks: QQP (Iyer et al., 2017), SWAG (Zellers et al., 2018), and MNLI (Williams et al., 2018). The first five datasets are used to evaluate performance against SSL baselines, while the remaining three assess generalization on NLU tasks. For the classification datasets, we use the pre-processed splits (labeled, unlabeled, validation, and test) provided by Wang et al. (2022),[1] released under the MIT license. For the NLU tasks, we use the official dev set for validation and construct the labeled and unlabeled sets by randomly sampling from their respective training sets. Further dataset details are provided in Appendix A.

4.2 Experimental Setup & Baselines

We evaluate our method using both BERT-base and RoBERTa-base as backbone classifiers to demonstrate generality across architectures. All experiments are conducted with three independent runs, and results are reported using *mean error rate* and standard deviation. In addition, we compute the *Friedman rank* (Friedman, 1940) of each algorithm to assess overall ranking across tasks: $\text{rank}_F = \frac{1}{m} \sum_{i=1}^{m} \text{rank}_i$, where $m = 15$ is the number of evaluation setups and rank_i is the algorithm's rank in the i-th setup. Our implementation details are presented in Appendix B.

For LLMs pseudo-labels, we experiment with three open-source LLMs—PHI-3-MEDIUM-4K, MISTRAL-7B, and LLAMA-3.1-8B. We use both zero-shot and few-shot prompting strategies with all three LLMs. Among all settings, PHI-3 zero-shot—where PHI-3 is also the largest model in terms of parameter count (14B)—achieves the best performance across all test sets; hence, we report the PHI-3 zero-shot baseline and the corresponding LG-COTRAIN variant that uses pseudo-labels generated by PHI-3 zero-shot in the main paper. We show the complete set of prompts for all LLMs in Appendix C, with extensive experimental results for both zero-shot and few-shot scenarios being provided in Appendix D.

---

[1] https://github.com/microsoft/Semi-supervised-learning/tree/main

Dataset # Label	IMDB 20	100	AG News 40	200	Amazon Review 250	1000	Yahoo! Answer 500	2000	Yelp Review 250	1000	Mean Err.	Fried. rank	Final rank
Sup (full)	$5.69_{0.15}$		$5.73_{0.11}$		$36.38_{0.01}$		$24.86_{0.07}$		$31.98_{0.20}$		21.07	-	-
Sup (lb)	$20.31_{2.79}$	$14.02_{1.22}$	$15.06_{1.08}$	$14.25_{0.97}$	$52.31_{1.28}$	$47.53_{0.69}$	$37.43_{0.29}$	$33.26_{0.10}$	$51.22_{0.98}$	$46.71_{0.37}$	33.31	-	-
Sup (lb+ps)	$10.68_{1.17}$	$7.88_{0.01}$	$11.88_{0.07}$	$10.92_{0.25}$	$47.77_{0.53}$	$42.23_{0.42}$	$33.38_{0.34}$	$32.15_{0.52}$	$44.44_{0.49}$	$39.57_{0.23}$	28.09	9.8	11
UDA	$49.97_{0.04}$	$50.0_{0.0}$	$41.02_{4.96}$	$53.68_{30.15}$	$60.76_{13.61}$	$68.38_{16.44}$	$71.3_{26.45}$	$70.5_{27.58}$	$69.33_{15.08}$	$66.95_{18.46}$	60.19	15	15
Dash	$8.34_{0.86}$	$7.55_{0.35}$	$31.67_{13.19}$	$13.76_{1.67}$	$47.1_{0.74}$	$43.09_{0.6}$	$35.26_{0.33}$	$31.19_{0.29}$	$45.24_{2.02}$	$40.14_{0.79}$	30.33	11.6	14
FixMatch	$7.72_{0.33}$	$7.33_{0.13}$	$30.17_{1.87}$	$11.71_{1.95}$	$47.61_{0.83}$	$43.05_{0.54}$	$33.03_{0.49}$	$30.51_{0.53}$	$46.52_{0.94}$	$40.65_{0.46}$	29.83	10.2	13
Flexmatch	$7.82_{0.77}$	$7.41_{0.38}$	$16.38_{3.94}$	$12.08_{0.73}$	$45.73_{1.6}$	$42.25_{0.33}$	$35.61_{1.08}$	$31.13_{0.18}$	$43.35_{0.69}$	$40.51_{0.34}$	28.23	10.0	12
Softmatch	$7.76_{0.58}$	$7.97_{0.72}$	$11.9_{0.27}$	$11.72_{1.58}$	$45.29_{0.95}$	$42.21_{0.2}$	$33.07_{0.31}$	$30.44_{0.62}$	$44.09_{0.5}$	$39.76_{0.13}$	27.42	8.3	8
Simmatch	$7.93_{0.55}$	$7.08_{0.33}$	$14.26_{1.51}$	$12.45_{1.37}$	$45.91_{0.95}$	$42.21_{0.3}$	$33.06_{0.2}$	$30.16_{0.21}$	$46.12_{0.48}$	$40.26_{0.62}$	27.94	8.95	10
Crmatch	$8.96_{0.88}$	$7.16_{0.09}$	$12.28_{1.43}$	$11.08_{1.24}$	$45.49_{0.98}$	$43.07_{0.5}$	$32.51_{0.4}$	$29.98_{0.07}$	$45.71_{0.63}$	$40.62_{0.28}$	27.69	8.70	9
Adamatch	$8.09_{0.99}$	$7.11_{0.2}$	$11.73_{0.17}$	$11.22_{0.95}$	$46.72_{0.72}$	$42.27_{0.25}$	$32.75_{0.35}$	$30.44_{0.31}$	$45.4_{0.96}$	$40.16_{0.49}$	27.59	8.15	7
VerifyMatch $_R$	$7.58_{0.61}$	$7.38_{0.43}$	$11.95_{0.18}$	$11.64_{0.21}$	$39.97_{0.29}$	$40.94_{0.44}$	$32.03_{0.08}$	$32.14_{0.71}$	$37.63_{0.10}$	$37.16_{2.23}$	25.84	6.7	6
CoTeach $_R$	$9.04_{2.07}$		$11.04_{0.48}$		$39.38_{0.56}$		$31.31_{0.18}$		$36.35_{1.27}$		25.42	6.7	5
DivideMix $_R$	$7.67_{0.17}$		$11.05_{0.14}$		$39.34_{0.44}$		$30.83_{0.66}$		$35.34_{0.55}$		24.85	5.0	3
Phi-3	**4.78**		12.67		39.32		33.53		34.62		24.98	6.3	4
LG-CoTr $_B$	$7.65_{0.05}$	$7.63_{0.08}$	$11.36_{0.06}$	$10.77_{0.06}$	$38.12_{0.06}$	$37.66_{0.27}$	$29.38_{0.16}$	$\mathbf{28.14^{\dagger}_{0.23}}$	$33.87_{0.14}$	$33.52_{0.12}$	23.81	3.1	2
LG-CoTr $_R$	$6.77_{0.32}$	$6.58_{0.15}$	$11.35_{0.17}$	$\mathbf{10.41_{0.20}}$	$\mathbf{37.15^{\dagger}_{0.19}}$	$\mathbf{37.04^{\dagger}_{0.13}}$	$\mathbf{29.31^{\dagger}_{0.11}}$	$28.16_{0.06}$	$\mathbf{33.15^{\dagger}_{0.27}}$	$\mathbf{32.93^{\dagger}_{0.53}}$	23.29	1.5	1

Table 1: Error rate (%) and Rank across datasets and label sizes. The first 8 rows are from Wang et al. (2022). Subsequent rows show results using pseudo-labels generated by Phi-3 with zero-shot prompting, including our **LG-CoTr**. Subscripts $_R$ and $_B$ indicate RoBERTa and BERT backbones, respectively. Best results are **bolded**. Superscript † indicates a statistically significant improvement ($p < 0.05$) over the best baseline using a paired t-test.

We benchmark our method against a diverse set of SSL baselines using varying amounts of labeled and unlabeled data, following the standardized setup from Wang et al. (2022). These baselines include FixMatch (Sohn et al., 2020), FlexMatch (Zhang et al., 2021), SoftMatch (Chen et al., 2023), SimMatch (Zheng et al., 2022), CRMatch (Fan et al., 2023), AdaMatch (Berthelot et al., 2022), Dash (Xu et al., 2021), and UDA (Xie et al., 2020a), along with a supervised-only baseline for reference (as an upper bound), a supervised baseline that uses only the small labeled data, and a supervised baseline that uses the combination of small labeled data and LLM pseudo-labeled data.

To ensure a fair comparison with our dual-classifier architecture, we also include two-network-based methods such as Co-teaching (Han et al., 2018b) and DivideMix (Li et al., 2020), commonly used in noisy label learning. Additionally, we compare against VerifyMatch (Park and Caragea, 2024), a recent LLM-guided SSL framework that directly handles noisy pseudo-labels.

4.3 Results & Observations

The comprehensive results of our main experiments, along with comparisons to other baselines, are presented in Table 1. Based on these results, we highlight the following key observations.

LG-CoTrain vs. Other SSL Baselines: LG-COTRAIN outperforms traditional SSL baselines across all five datasets. While standard methods rely on self-generated pseudo-labels, often amplifying errors, our approach integrates LLM-generated labels and adaptively weights them using training dynamics. Compared to BERT-based results from Wang et al. (2022), our BERT-based model (LG-CoTr$_B$) achieves a 3.78% lower average error than AdaMatch, the strongest of the SSL baselines, demonstrating the effectiveness of our framework.

LG-CoTrain vs. VerifyMatch: VerifyMatch is a semi-supervised learning method that, like our approach, incorporates LLM-generated pseudo-labels into the training process. Thus, it serves as a critical baseline for evaluating the effectiveness of our framework. As shown in Table 1, LG-COTRAIN outperforms VerifyMatch across all five datasets, with substantial improvements in four of them. On IMDB, our method still achieves a lower error rate, although the margin is relatively small. Overall, LG-COTRAIN achieves an average error rate improvement of 2.55%, further demonstrating the advantage of our method.

LG-CoTrain vs. CoTeach and DivideMix: Our LG-COTRAIN method consistently surpasses CoTeach and DivideMix, both of which are designed to mitigate the impact of noisy labels. On average, LG-COTRAIN achieves a 1.56% lower mean error compared to DivideMix and a 2.13% lower mean error compared to CoTeach, demonstrating the effectiveness of training dynamics-based weighting methods in noisy settings.

LG-CoTrain vs. Supervised Settings: We

Dataset	Amazon Review		Yahoo! Answer		Yelp Review		Mean Error Rate (%)	Friedman Rank	Final Rank
# Label	250	1000	500	2000	250	1000			
LG-CoTRAIN$_{oracle}$	$34.68_{0.06}$	$34.71_{0.03}$	$24.27_{0.04}$	$24.27_{0.02}$	$30.33_{0.05}$	$30.52_{0.24}$	29.80	-	-
ST-RANDOM	$37.55_{0.14}$	$38.08_{0.21}$	$30.37_{0.14}$	$29.17_{0.21}$	$34.14_{0.29}$	$34.05_{0.49}$	33.89	3.67	4
FT-ENSEMBLED	$41.83_{0.19}$	$39.83_{0.10}$	$33.60_{0.15}$	$32.08_{0.10}$	$39.18_{0.14}$	$36.74_{0.08}$	37.21	5.33	5
VANILLA-COTRAIN	$50.37_{0.53}$	$49.00_{0.66}$	$33.10_{0.33}$	$29.29_{0.82}$	$43.68_{0.29}$	$39.23_{0.63}$	40.78	5.67	6
LG-COTRAIN$_{SingleSet}$	$37.86_{0.12}$	$37.70_{0.25}$	$30.03_{0.12}$	$28.90_{0.04}$	$33.39_{0.40}$	$33.01_{0.35}$	33.48	2.17	2
LG-COTRAIN$_{cc}$	$38.35_{0.17}$	$38.06_{0.21}$	$30.15_{0.26}$	$29.15_{0.28}$	$33.83_{0.26}$	$33.86_{0.11}$	33.9	3.17	3
LG-COTRAIN	$\mathbf{37.15}^{\dagger}_{0.19}$	$\mathbf{37.04}^{\dagger}_{0.13}$	$\mathbf{29.31}^{\dagger}_{0.11}$	$\mathbf{28.16}_{0.06}$	$\mathbf{33.15}_{0.27}$	$\mathbf{32.93}_{0.53}$	32.96	1	1

Table 2: Ablation Study for different componenets of LG-COTRAIN. The best-performing results among all baselines are highlighted in bold. Superscript † indicates a statistically significant improvement ($p < 0.05$) over the best baseline using a paired t-test.

compare three baselines to LG-COTRAIN (BERT-base): (i) supervised training on the full labeled set (SUP (FULL)), serving as an empirical upper bound; (ii) supervised training on a small labeled subset (SUP (LB)); and (iii) supervised training on labeled data augmented with LLM pseudo-labels (SUP (LB+PS)). As shown in Table 1, as expected, SUP (FULL) achieves the lowest error (2.22%), while SUP (LB) performs much worse (33.31%). In case of (SUP (LB+PS)), adding pseudo-labeled data with the small labeled set helps reduce the mean error by 5.12 points, and LG-COTRAIN achieves a further 4.28% reduction in mean error, totaling a 9.40% mean error reduction over SUP (LB).

LG-CoTrain vs. LLMs: Among the three LLMs evaluated—PHI-3-14B, MISTRAL-7B, and LLAMA-3.1-8B—PHI-3 achieves the strongest overall performance in the zero-shot setting, while LLAMA-3 performs best under few-shot prompting. Despite these strong baselines, our LG-COTRAIN approach consistently outperforms both the zero-shot and few-shot results of all three LLMs across the test splits of all datasets, with the sole exception of IMDB (see the comparison with Phi-3 in Table 1 and the other LLM comparisons in Appendix D). Notably, our best variant achieves a 1.76% lower mean error than PHI-3 zero-shot and a 2.93% lower mean error than LLAMA-3 few-shot. Comprehensive results with both zero-shot and few-shot approaches are reported in Appendix D.

5 Analysis

We conduct our analysis through four complementary investigations: (1) ablation studies examining the contributions of LG-COTRAIN's core components; (2) assessment of LG-COTRAIN's generalizability across natural language understanding (NLU) tasks; (3) analysis of class imbalance and pseudo-label skew effects; and (4) qualitative evaluation of the weighting mechanism's capacity to distinguish clean from noisy samples.

5.1 Ablations

We conducted an ablation study and show the results in Table 2. The goal of this study is to understand the effectiveness of different components within our framework. Although LG-COTRAIN is a cohesive design where multiple components interact synergistically, and it is not always straightforward to isolate the contribution of each part, we aim to analyze their individual impact by systematically removing or modifying specific elements. The results from these controlled variants are reported and analyzed below.

LG-CoTrain vs. Finetuning: We compare LG-COTRAIN with a finetuning-based baseline, where two RoBERTa-base classifiers are trained on the combined dataset $\mathcal{D}_{LG} \cup \mathcal{D}_l$, and their predictions are ensembled during inference. This baseline, referred to as FT-ENSEMBLED, results in a 4.25% higher mean error rate, underscoring the effectiveness of our weighted co-training strategy—especially in scenarios where the training data includes noisy pseudo-labels generated by LLMs.

LG-CoTrain vs. Self-Training with Random Weighting: To highlight the importance of using dual classifiers and training-dynamics-guided weighting, we compare against ST-RANDOM, a self-training baseline where a single classifier is trained using pseudo-labeled samples. For each sample, either λ_1 or λ_2—as defined in Eq. (3) and Eq. (4)—is randomly selected and used as its training weight. This setup results in a 0.93% higher mean error rate, reinforcing the effectiveness of our dual-classifier design and training dynamics-guided weighting strategy.

LG-CoTrain vs. Vanilla CoTraining: VANILLA-COTRAIN relies only on the most confident predictions exchanged between classifier pairs, without leveraging LLM-generated

Dataset	QQP		SWAG		MNLI-m		MNLI-mm		Mean	Fried.	Final
# Label	100	500	200	1000	150	750	150	750	Error	Rank	Rank
Phi-3	24.66		23.33		19.36		21.53		22.22	2.38	2
CoTeach	$24.51_{0.64}$		$23.49_{0.28}$		$20.62_{0.46}$		$21.75_{0.40}$		22.59	2.88	3
Vanilla Cotrain	$32.01_{0.25}$	$23.26_{1.28}$	$33.57_{0.38}$	$27.49_{0.52}$	$63.62_{1.40}$	$48.72_{6.44}$	$59.91_{3.27}$	$49.10_{9.01}$	42.21	3.75	4
LG-CoTrain	$\mathbf{20.22}^{\dagger}_{0.77}$	$\mathbf{17.48}^{\dagger}_{0.25}$	$\mathbf{22.53}^{\dagger}_{0.27}$	$\mathbf{20.91}^{\dagger}_{0.30}$	$\mathbf{16.82}^{\dagger}_{1.0}$	$\mathbf{15.29}^{\dagger}_{0.29}$	$\mathbf{17.32}^{\dagger}_{0.94}$	$\mathbf{15.19}^{\dagger}_{0.22}$	18.22	1	1

Table 3: Performance evaluation on the QQP, SWAG, and MNLI development sets. Here, MNLI-m refers to MNLI matched, while MNLI-mm denotes MNLI mismatched. Best results are **bolded**. † indicates a statistically significant improvement ($p < 0.05$) over the best baseline using a paired t-test.

pseudo-labels. This results in a 7.82% increase in mean error rate, demonstrating the value of incorporating LLM pseudo labels and systematically incorporating them into the training process.

LG-CoTrain vs. LG-CoTrain$_{CC}$: LG-COTRAIN$_{CC}$ is a variant that uses only confidence-based weighting (averaged across iterations), completely ignoring variability. This design reduces divergence between classifiers and weakens performance, leading to a 0.94% increase in mean error.

LG-CoTrain vs. LG-CoTrain (SingleSet): In LG-COTRAIN (SINGLESET), both classifiers are trained on the same labeled subset, removing the complementary learning effect gained from disjoint splits. This results in a 0.52% increase in mean error, indicating that diversity in supervision is important for robust co-training.

LG-CoTrain in an Oracle Setting: In Table 2, **LG-COTRAIN$_{oracle}$** shows an ideal case where we use only gold-standard (human-annotated) labels instead of LLM-generated pseudo-labels for training. This helps us measure the maximum possible gain from perfect pseudo-labels—resulting in a 3.16% lower error rate compared to our regular LG-COTRAIN that uses LLM-generated labels.

5.2 Evaluation on Other NLU Tasks

We evaluate LG-COTRAIN on a range of challenging natural language understanding (NLU) tasks under different SSL settings. Specifically, we assess its performance on paraphrase detection using the QQP dev set, natural language inference (NLI) using both the MNLI matched and mismatched dev sets, and commonsense reasoning using the SWAG dev set. For each task, we experiment with two SSL settings by varying the number of labeled examples while treating the remaining training data as unlabeled. We compare LG-COTRAIN against several baselines, including COTEACH, VANILLA CO-TRAINING, and zero-shot LLM prompting. Across all tasks, LG-COTRAIN consistently outperforms the baselines, demonstrating its robustness and ef-

Dataset	Most (count)	Least (count)	Skew (Most/Least)
IMDB	12,303	10,685	1.15
AG News	25,545	24,144	1.06
Yahoo	63,659	31,465	2.02
Amazon	62,213	30,759	2.02
Yelp	67,796	32,420	2.09

Table 4: **LLM pseudo-label distribution**. For each dataset we report the most and least frequent pseudo-label counts and the induced skew ratio.

fectiveness in NLU tasks under limited supervision. Detailed results are provided in Table 3.

5.3 Effect of Class Imbalance and Pseudo-Label Skew

As specified in Appendix A, our main experiments use class-balanced labeled and unlabeled training splits. We use the same dataset splits as in Wang et al. (2022), which ensures fair and direct comparison with several strong SSL baselines. While the ground-truth label distribution of the training data is balanced in these settings, the *LLM-generated pseudo-labels* over the unlabeled set exhibit varying degrees of skew across datasets: IMDB and AG News maintain near-uniform distributions, while Amazon Review, Yelp Review, and Yahoo Answers exhibit moderate skew with the most frequent class appearing approximately twice as often as the least frequent class (see Table 4). This pseudo-label imbalance arises from prediction error by the LLMs. In this scenario—*balanced true labels but imbalanced pseudo-labels*—LG-CoTrain remains robust, as its weighting mechanism down-weights erroneous pseudo-labels and up-weights correct ones (Appendix F). Thus, the gains in Table 1 hold even when pseudo-labels are skewed.

Long-tailed true label distributions. To further stress-test robustness, we evaluate a more challenging scenario where both labeled and unlabeled sets follow a long-tailed ground-truth distribution with an imbalance ratio (IR) of 10 (max-to-min class frequency ratio). This creates simultaneous challenges

Dataset	Amazon Review		Yahoo Answers		Yelp Review		Mean	Fried.	Final
# Label	250	1000	500	2000	250	1000	Error	Rank	Rank
VerifyMatch (imb)	$40.71_{0.76}$	$39.48_{0.62}$	$32.75_{0.52}$	$32.26_{0.48}$	$37.38_{0.98}$	$37.17_{0.23}$	36.62	2	2
LG-CoTrain (imb)	$\mathbf{37.77}_{0.11}$	$\mathbf{39.01}_{0.40}$	$\mathbf{30.71}_{0.22}$	$\mathbf{29.46}_{0.23}$	$\mathbf{34.92}_{0.30}$	$\mathbf{35.73}_{0.12}$	34.60	1	1

Table 5: Error rates (%) under long-tailed imbalance (IR=10). Both labeled and unlabeled sets follow a long-tailed ground-truth distribution. Lower is better. Means$_{std}$ over three runs. Best results are **bolded**. Mean error is averaged across all setups. Friedman rank (Fried.) is the average rank across setups, and Final Rank is the overall ranking.

of (i) true-class imbalance and (ii) pseudo-label noise from LLM predictions. We compare LG-CoTRAIN against VERIFYMATCH, a competing method that also utilizes LLM-generated pseudo-labels. The results in Table 5 show that across three datasets (Amazon, Yelp, Yahoo) and two labeled-data budgets, LG-CoTRAIN consistently outperforms VERIFYMATCH. This maintains the performance trends observed in Table 1 and demonstrates superior robustness to real-world distribution skew.

5.4 Qualitative Analysis

To better understand the behavior of our sample weighting strategy, we analyze how the learned weights correlate with the correctness of LLM-generated pseudo-labels. In our SSL framework, pseudo-labels remain fixed throughout training, making it essential that correct labels are emphasized while incorrect ones are down-weighted. Figure 1 presents kernel density estimates (KDE) of the two weighting metrics, λ_1 and λ_2, corresponding to the two co-training models trained on YAHOO! ANSWERS dataset. For each plot, we separate samples into two groups: those where the LLM-generated pseudo-label matches the ground truth (Match, shown in blue), and those where it does not (Mismatch, shown in orange).

We observe a clear separation between the two distributions in both λ_1 and λ_2. Correctly labeled samples (Match) tend to have higher weights, as indicated by the rightward shift of the blue curves. In contrast, incorrectly labeled samples (Mismatch) exhibit a leftward shift, reflecting lower assigned weights. This confirms that the dynamic weighting effectively distinguishes between high-quality and noisy pseudo-labels, promoting reliable learning signals during co-training. Similar trends are observed across other datasets (see Appendix F).

6 Conclusion

In this paper, we introduced LG-CoTRAIN, a novel semi-supervised learning (SSL) framework that leverages Large Language Model (LLM)-guided
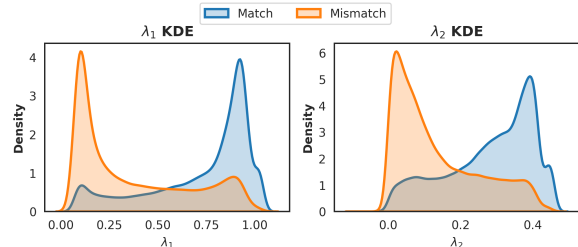

Figure 1: KDE plots of λ_1 and λ_2 for matched and mismatched samples of Yahoo! Answers dataset.

pseudo-labeling with weighted error mitigation to enhance co-training. Our approach addresses the key SSL challenge of pseudo-label quality by leveraging LLM-generated labels and refining them with a dynamic weighting mechanism based on training dynamics. Unlike conventional SSL methods that discard low-confidence pseudo-labels, LG-CoTRAIN utilizes all pseudo-labels, assigning appropriate importance to each to maximize data efficiency. Through extensive experiments on five benchmark datasets, LG-CoTRAIN consistently outperforms prior SSL baselines, setting new state-of-the-art results. Additionally, we validate its effectiveness across multiple NLU tasks, demonstrating its strong generalization capability. Our findings demonstrate the feasibility of integrating LLMs into SSL frameworks efficiently, opening new avenues for research in semi-supervised NLP. In the future, it would also be interesting to explore approaches where the LLM-generated pseudo-labels are updated over iterations.

Acknowledgement

Limitations

One limitation of our approach is the slightly increased computational overhead compared to standard SSL baselines. This is primarily due to the use of a dual-network architecture, which requires training and updating the parameters of two models simultaneously. A detailed comparison of training times is provided in Appendix E.

References

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, and 1 others. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*.

Dana Angluin and Philip Laird. 1988. Learning from noisy examples. *Machine learning*, 2:343–370.

Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and 1 others. 2017. A closer look at memorization in deep networks. In *International conference on machine learning*, pages 233–242. PMLR.

David Berthelot, Rebecca Roelofs, Kihyuk Sohn, Nicholas Carlini, and Alexey Kurakin. 2022. Adamatch: A unified approach to semi-supervised learning and domain adaptation. In *International Conference on Learning Representations*.

Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100.

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.

Ming-Wei Chang, Lev-Arie Ratinov, Dan Roth, and Vivek Srikumar. 2008. Importance of semantic representation: Dataless classification. In *Aaai*, volume 2, pages 830–835.

Hao Chen, Ran Tao, Yue Fan, Yidong Wang, Jindong Wang, Bernt Schiele, Xing Xie, Bhiksha Raj, and Marios Savvides. 2023. Softmatch: Addressing the quantity-quality trade-off in semi-supervised learning. *arXiv preprint arXiv:2301.10921*.

Yuhan Chen, Shuqi Li, and Rui Yan. 2024. FlexiQA: Leveraging LLM's evaluation capabilities for flexible knowledge selection in open-domain question answering. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 56–66, St. Julian's, Malta. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yue Fan, Anna Kukleva, Dengxin Dai, and Bernt Schiele. 2023. Revisiting consistency regularization for semi-supervised learning. *International Journal of Computer Vision*, 131(3):626–643.

Milton Friedman. 1940. A comparison of alternative tests of significance for the problem of m rankings. *The annals of mathematical statistics*, 11(1):86–92.

Nikesh Gyawali, Iustin Sirbu, Tiberiu Sosea, Sarthak Khanal, Doina Caragea, Traian Rebedea, and Cornelia Caragea. 2024. GunStance: Stance detection for gun control and gun regulation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12027–12044, Bangkok, Thailand. Association for Computational Linguistics.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. 2018a. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in neural information processing systems*, 31.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. 2018b. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Mahshid Hosseini and Cornelia Caragea. 2023. Semi-supervised domain adaptation for emotion-related tasks. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5402–5410, Toronto, Canada. Association for Computational Linguistics.

Shankar Iyer, Nikhil Dandekar, and Kornél Csernai. 2017. Quora question pairs.

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. 2018. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International conference on machine learning*, pages 2304–2313. PMLR.

Diederik P Kingma. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Junnan Li, Richard Socher, and Steven CH Hoi. 2020. Dividemix: Learning with noisy labels as semi-supervised learning. In *International Conference on Learning Representations*.

Yinhan Liu. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 364.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. 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, Portland, Oregon, USA. Association for Computational Linguistics.

Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 165–172.

Meta. 2024. Introducing llama 3.1: Our most capable models to date. https://ai.meta.com/blog/meta-llama-3-1/. Accessed: December 9, 2024.

Mistral AI. 2024. Announcing Mistral-7B. https://mistral.ai/news/announcing-mistral-7b/. Accessed: December 9, 2024.

Seo Yeon Park and Cornelia Caragea. 2024. Verify-Match: A semi-supervised learning paradigm for natural language inference with confidence-aware MixUp. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 19319–19335, Miami, Florida, USA. Association for Computational Linguistics.

Eduard Poesina, Cornelia Caragea, and Radu Ionescu. 2024. A novel cartography-based curriculum learning method applied on RoNLI: The first Romanian natural language inference corpus. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 236–253, Bangkok, Thailand. Association for Computational Linguistics.

Mobashir Sadat and Cornelia Caragea. 2022. Learning to infer from unlabeled data: A semi-supervised learning approach for robust natural language inference. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4763–4776, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Mobashir Sadat and Cornelia Caragea. 2024. Co-training for low resource scientific natural language inference. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2538–2550, Bangkok, Thailand. Association for Computational Linguistics.

Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. 2020. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in neural information processing systems*, 33:596–608.

Tiberiu Sosea and Cornelia Caragea. 2022. Leveraging training dynamics and self-training for text classification. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4750–4762, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Tiberiu Sosea and Cornelia Caragea. 2023. Marginmatch: Improving semi-supervised learning with pseudo-margins. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15773–15782.

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

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Jesper E Van Engelen and Holger H Hoos. 2020. A survey on semi-supervised learning. *Machine learning*, 109(2):373–440.

Yidong Wang, Hao Chen, Yue Fan, Wang Sun, Ran Tao, Wenxin Hou, Renjie Wang, Linyi Yang, Zhi Zhou, Lan-Zhe Guo, and 1 others. 2022. Usb: A unified semi-supervised learning benchmark for classification. *Advances in Neural Information Processing Systems*, 35:3938–3961.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M.

Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020a. Unsupervised data augmentation for consistency training. *Advances in neural information processing systems*, 33:6256–6268.

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. 2020b. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10687–10698.

Yi Xu, Lei Shang, Jinxing Ye, Qi Qian, Yu-Feng Li, Baigui Sun, Hao Li, and Rong Jin. 2021. Dash: Semi-supervised learning with dynamic thresholding. In *International conference on machine learning*, pages 11525–11536. PMLR.

Yelp.com. Yelp dataset challenge. https://www.yelp.com/dataset.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104, Brussels, Belgium. Association for Computational Linguistics.

Bowen Zhang, Yidong Wang, Wenxin Hou, Hao Wu, Jindong Wang, Manabu Okumura, and Takahiro Shinozaki. 2021. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. *Advances in Neural Information Processing Systems*, 34:18408–18419.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Yiming Zhang, Shi Feng, and Chenhao Tan. 2022. Active example selection for in-context learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9134–9148, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.

Mingkai Zheng, Shan You, Lang Huang, Fei Wang, Chen Qian, and Chang Xu. 2022. Simmatch: Semi-supervised learning with similarity matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14471–14481.

Henry Peng Zou and Cornelia Caragea. 2023. JointMatch: A unified approach for diverse and collaborative pseudo-labeling to semi-supervised text classification. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7290–7301, Singapore. Association for Computational Linguistics.

Henry Peng Zou, Yue Zhou, Weizhi Zhang, and Cornelia Caragea. 2023. DeCrisisMB: Debiased semi-supervised learning for crisis tweet classification via memory bank. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6104–6115, Singapore. Association for Computational Linguistics.

A Dataset Details

As previously mentioned, we adopt the same dataset splits as Wang et al. (2022) to ensure fair comparison with existing benchmarks. Additionally, to evaluate the generalizability of our approach, we incorporate widely-used datasets—MNLI, QQP, and SWAG—from the Hugging Face Datasets Library (Lhoest et al., 2021). Below, we provide detailed descriptions of each dataset used in our study.

IMDB The IMDB dataset (Maas et al., 2011) is a binary sentiment classification dataset consisting of 25,000 reviews for training and 25,000 for testing. The dataset is balanced, with an equal number of positive and negative reviews in both the training and test sets. For consistency, we rely on the official splits released by Wang et al. (2022), which contain 12,500 samples and 1,000 samples per class in the training and validation datasets, respectively. The test set remains unchanged.

Amazon Review The Amazon Review dataset (McAuley and Leskovec, 2013) is a sentiment classification dataset with five classes (ratings). Each class contains 600,000 training samples and 130,000 test samples. For consistency, we rely on the official splits released by Wang et al. (2022), which contain 50,000 samples and 5,000 samples

per class in the training and validation datasets, respectively. The test set remains unchanged.

Yelp Review The Yelp Review dataset (Yelp.com) is a sentiment classification dataset with five classes. It includes 130,000 training samples and 10,000 test samples per class. For consistency, we rely on the official splits released by Wang et al. (2022), which contain 50,000 samples and 5,000 samples per class in the training and validation datasets, respectively. The test set remains unchanged.

AG News The AG News dataset (Zhang et al., 2015) is a news topic classification dataset with four classes. Each class contains 30,000 training samples and 1,900 test samples. For consistency, we rely on the official splits released by Wang et al. (2022), which contain 25,000 samples and 2,500 samples per class in the training and validation datasets, respectively. The test set remains unchanged.

Yahoo! Answer The Yahoo! Answer dataset (Chang et al., 2008) is a topic classification dataset with ten categories. It includes 140,000 training samples and 6,000 test samples per class. For consistency, we rely on the official splits released by Wang et al. (2022), which contain 50,000 samples and 5,000 samples per class in the training and validation datasets, respectively. The test set remains unchanged.

QQP The Quora Question Pairs (QQP) dataset (Iyer et al., 2017) is a binary classification dataset designed to determine whether two questions are semantically equivalent. For our work, we use the training and development splits from the Hugging Face Datasets library, which consist of approximately 363,846 training samples and 40,430 development samples.

SWAG Dataset The Situations With Adversarial Generations (SWAG) dataset (Zellers et al., 2018) is a multiple-choice dataset for commonsense reasoning. For our work, we use the training and development splits from the Hugging Face Datasets library, which consist of 73,546 training samples and 20,006 development samples.

MNLI The Multi-Genre Natural Language Inference (MNLI) dataset (Williams et al., 2018) is a textual entailment dataset with three labels: entailment, contradiction, and neutral. For our work, we use the training and development splits from the Hugging Face Datasets library, which consist of 392,702 training samples and 9815 matched and mismatched developement samples.

B Implementation Details

We utilized Llama-3.1-8B (Meta, 2024), Phi-3-medium-4k (Abdin et al., 2024), and Mistral-7B-Instruct (Mistral AI, 2024) as our large language models (LLMs) to generate pseudo-labels for unlabeled samples. Throughout this paper, any mention of these models specifically refers to the variants described here. For the backbone models for co-training, we used the base variant of ROBERTA (Liu, 2019) and BERT (Devlin et al., 2019). For all methods, including baselines and our proposed approach, we set the batch size to 32. The models are trained using the Adam optimizer (Kingma, 2014) with a learning rate of $2e - 5$. The maximum number of epochs is set to 10 for co-training and 100 for fine-tuning. We apply early stopping with a patience of 5 to prevent overfitting. Each experiment is run three times with different random seeds to ensure the reliability of results.

C Prompts Used for Pseudo Label Generation

In Table 6, we present the base zero-shot prompts used for generating pseudo-labels for each dataset. For few-shot scenarios, we augment these prompts with randomly selected labeled examples from the training set. Specifically, we use 4 examples per class for AG News, 3 for Yahoo! Answers, and 2 for IMDB, Amazon Review, and Yelp Review.

Additionally, prompts are customized for each LLM by incorporating model-specific system instructions or formatting conventions to ensure compatibility and optimal performance.

D Additional Results

This section presents extended experimental results incorporating three additional LLM pseudo-labelers: **LLaMA-3**, **Mistral**, and **Phi-3**. We evaluate these models under both **zero-shot** and **few-shot** prompting configurations. Table 7 provides a comprehensive comparison across datasets and label sizes, reporting both error rates and ranking metrics to complement the main paper's findings.

Notably, we observe that few-shot in-context learning occasionally underperforms compared to zero-shot settings. We attribute this phenomenon to the random sampling of few-shot examples from the training set, as prior work (Zhang et al., 2022) demonstrates that exemplar selection critically impacts in-context learning performance. The suboptimal performance in some few-shot scenarios

Dataset	Prompt
AG News	News Article: {`news_article`}. Based on the content of the news article provided, which of the following categories would it best fit under: World, Sports, Business, or Science/Technology? Just select one of these four options. No explanation required.
IMDB	Movie Review: {`movie_review`}. Based on the content of the movie review provided, determine the category for the movie review: Positive or Negative. Select only one of these options. No explanation required.
Yahoo! Answer	Question: {`question`}. Based on the content of the question provided, which of the following categories would it best fit under: Society & Culture, Science & Mathematics, Health, Education & Reference, Computers & Internet, Sports, Business & Finance, Entertainment & Music, Family & Relationships, or Politics & Government? Just select one of these ten options. No explanation required.
Amazon Reviews	Review Text: {`review_text`}. Based on the content of the review text provided, determine the rating for the product review: (1 star, 2 stars, 3 stars, 4 stars, or 5 stars). Just select one of these five options. No explanation required.
Yelp Reviews	Review Text: {`review_text`}. Based on the content of the review text provided, determine the rating for the restaurant review: (1 star, 2 stars, 3 stars, 4 stars, or 5 stars). Just select one of these five options. No explanation required.
QQP	Question 1: {`question1`}. Question 2: {`question2`}. Based on the context, phrasing, and intent of the two questions provided, determine if the two questions are semantically equivalent (duplicates). Select only one of these options: duplicate or not duplicate. No explanation required.
SWAG	Startphrase: {`startphrase`}. Ending 0: {`ending0`}. Ending 1: {`ending1`}. Ending 2: {`ending2`}. Ending 3: {`ending3`}. Based on the startphrase and potential endings provided, determine the most likely ending to the startphrase. Respond with only one of these options: 0, 1, 2, or 3. No explanation required.
MNLI	Premise: {`premise`}. Hypothesis: {`hypothesis`}. Based on the premise and hypothesis provided, determine the most likely relationship between the two. Respond with only one of these options: entailment, contradiction, or neutral. No explanation required.

Table 6: Example Prompts used to generate pseudo-labels

suggests that randomly selected examples may introduce noise or bias that degrades pseudo-labeling quality.

E Training Cost

To ensure a fair comparison across all methods, we standardized the training configuration for each experiment. Table 8 summarizes the compute budget used for each method and dataset, including the number of training hours, training settings per run, and the number of random seeds. The total GPU hours reported assume a single GPU per run. All experiments were conducted using NVIDIA V100 GPUs.

E.1 Fairness in Parameter and Inference Cost

Our proposed framework requires training two models jointly and performing inference with both at test time. To ensure fair comparisons, we further evaluated *self-ensembled* baselines, where each baseline (e.g., AdaMatch, SoftMatch) is trained twice independently and their predictions are ensembled at inference. This matches our framework

in terms of parameter count and test-time computation.

The results, shown in Table 9, indicate that our method consistently outperforms these self-ensembled variants, demonstrating that our co-training design provides benefits beyond simply increasing model size or applying post-hoc ensembling.

F Distribution of Pseudo-Label Weights

In this section, we provide kernel density estimates (KDE) of the weighting metrics λ_1 and λ_2 across all five benchmark datasets to evaluate the consistency and effectiveness of our sample weighting strategy. For each dataset, we visualize the distribution of weights assigned to samples where the LLM-generated pseudo-labels match the ground truth (Match) and where they do not (Mismatch), as shown in Figure 2.

Across four out of five datasets: AG NEWS, Amazon Reviews, Yahoo! Answers, and Yelp Reviews—we observe a clear separation between the Match and Mismatch distributions. Correctly la-

Dataset	IMDB		AG News		Amazon Review		Yahoo! Answer		Yelp Review		Mean Err.	Fried. rank	Final rank
# Label	20	40	40	200	250	1000	500	2000	250	1000			
LG-CoTr $_{P\text{-}FS}$	$7.11_{0.15}$	$6.99_{0.20}$	$13.22_{1.24}$	$11.29_{0.50}$	$\mathbf{36.76}^{\dagger}_{0.05}$	$\mathbf{36.70}^{\dagger}_{0.09}$	$29.46_{0.07}$	$28.62_{0.09}$	$33.46_{0.52}$	$32.85_{0.61}$	23.65	5.55	4
LG-CoTr $_{M\text{-}FS}$	$7.11_{0.06}$	$6.83_{0.11}$	$11.96_{0.80}$	$11.35_{0.23}$	$37.53_{0.23}$	$36.85_{0.34}$	$30.22_{0.13}$	$28.75_{0.15}$	$33.80_{0.14}$	$33.04_{0.29}$	23.74	6.35	6
LG-CoTr $_{L\text{-}FS}$	$6.63_{0.35}$	$6.43_{0.32}$	$11.12_{0.11}$	$10.70_{0.27}$	$37.47_{0.24}$	$36.71_{0.18}$	$\mathbf{28.67}^{\dagger}_{0.19}$	$\mathbf{27.97}^{\dagger}_{0.14}$	$33.60_{0.13}$	$32.91_{0.17}$	23.22	3.10	1
LG-CoTr $_{P\text{-}ZS}$	$6.77_{0.32}$	$6.58_{0.15}$	$11.35_{0.17}$	$10.41_{0.20}$	$37.15_{0.19}$	$37.04_{0.13}$	$29.31_{0.11}$	$28.16_{0.06}$	$\mathbf{33.15}^{\dagger}_{0.27}$	$32.93_{0.53}$	23.29	3.70	2
LG-CoTr $_{M\text{-}ZS}$	$6.78_{0.23}$	$6.94_{0.04}$	$11.19_{0.12}$	$\mathbf{10.01}_{0.30}$	$37.91_{0.33}$	$37.21_{0.10}$	$30.20_{0.09}$	$28.96_{0.20}$	$33.68_{0.27}$	$\mathbf{32.76}^{\dagger}_{0.07}$	23.56	5.00	3
LG-CoTr $_{L\text{-}ZS}$	$6.81_{0.27}$	$6.68_{0.26}$	$12.29_{0.52}$	$10.77_{0.81}$	$38.49_{0.07}$	$37.53_{0.23}$	$29.18_{0.16}$	$28.29_{0.06}$	$33.85_{0.48}$	$33.28_{0.12}$	23.71	6.00	5
VerifyMatch $_{P\text{-}ZS}$	$7.58_{0.61}$	$7.38_{0.43}$	$11.95_{0.18}$	$11.64_{0.21}$	$39.97_{0.29}$	$40.94_{0.44}$	$32.03_{0.08}$	$32.14_{0.71}$	$37.63_{0.10}$	$37.16_{2.23}$	25.84	12.30	11
VerifyMatch $_{M\text{-}ZS}$	$8.29_{0.69}$	$8.00_{0.33}$	$13.33_{1.13}$	$14.22_{1.93}$	$40.41_{0.77}$	$40.03_{1.24}$	$34.52_{0.51}$	$34.17_{0.36}$	$35.45_{0.53}$	$35.74_{0.19}$	26.42	14.40	15
VerifyMatch $_{L\text{-}ZS}$	$7.57_{0.66}$	$7.90_{0.85}$	$15.27_{1.55}$	$14.31_{1.36}$	$45.88_{1.26}$	$43.86_{0.74}$	$33.78_{0.25}$	$33.43_{0.51}$	$37.79_{0.44}$	$39.58_{1.23}$	27.94	16.30	20
CoTeach $_{P\text{-}ZS}$	$9.04_{2.07}$		$11.04_{0.48}$		$39.38_{0.56}$		$31.31_{0.18}$		$36.35_{1.27}$		25.42	11.0	9
CoTeach $_{M\text{-}ZS}$	$10.44_{2.95}$		$16.76_{1.53}$		$39.72_{0.84}$		$33.67_{0.34}$		$35.94_{0.73}$		27.31	15.50	17
CoTeach $_{L\text{-}ZS}$	$8.30_{1.75}$		$15.74_{2.30}$		$43.72_{1.23}$		$33.45_{0.52}$		$36.98_{0.50}$		27.64	15.70	18
DivideMix $_{P\text{-}ZS}$	$7.67_{0.17}$		$11.05_{0.14}$		$39.34_{0.44}$		$30.83_{0.66}$		$35.34_{0.55}$		24.85	9.30	8
DivideMix $_{M\text{-}ZS}$	$9.98_{1.90}$		$12.80_{0.59}$		$38.82_{0.05}$		$31.99_{0.99}$		$34.83_{0.22}$		25.68	11.30	10
DivideMix $_{L\text{-}ZS}$	$8.45_{1.29}$		$13.06_{0.79}$		$41.95_{1.73}$		$30.81_{0.31}$		$35.09_{0.79}$		25.87	12.50	12
Phi-3 $_{FS}$	11.84		18.07		37.5		38.52		34.14		28.01	14.20	14
Mistral $_{FS}$	5.79		19.62		41.67		39.45		39.18		29.14	16.30	21
Llama-3 $_{FS}$	6.37		13.09		40.14		33.96		37.2		26.15	12.80	13
Phi-3 $_{ZS}$	4.78		12.67		39.32		33.53		34.62		24.98	8.40	7
Mistral $_{ZS}$	5.08		14.39		44.74		36.27		38.51		27.80	15.10	16
Llama-3 $_{ZS}$	$\mathbf{4.68}$		25.7		45.67		39.17		38.32		30.71	16.20	19

Table 7: Error rate (%) and Rank across datasets and label sizes. Subscripts $_P$, $_M$, and $_L$ denote the LLMs used: Phi-3, Mistral, and LLaMA-3, respectively. Subscripts $_{FS}$ and $_{ZS}$ indicate few-shot and zero-shot prompting settings. Best results are **bolded**. † indicates a statistically significant improvement ($p < 0.05$) over the best baseline using a paired t-test. Grayed rows are also reported in the main paper.

beled samples consistently receive higher weights, as shown by the rightward shift of the blue curves, while mislabeled ones are assigned lower weights. These trends align with our overall performance improvements in these datasets.

The only exception is IMDB, where the Match and Mismatch distributions overlap more significantly, suggesting reduced discriminative capacity in weighting. Correspondingly, this is also the only dataset where our method does not outperform LLM baselines, reinforcing the link between effective sample weighting and performance gains.

G Variability vs. Accuracy

LG-CoTrain is designed to exploit complementary behavior between two jointly trained models, which is reflected in the performance gains reported in Table 2 of the main paper. In our framework, each model assigns a per-sample weighting factor λ when computing the loss. For model 1, the weight is defined as $\lambda_1 = \text{confidence} + \text{variability}$, while for model 2 it is $\lambda_2 = \text{confidence} - \text{variability}$. This asymmetry in weighting causes the two models to prioritize different examples during training, leading them to learn distinct yet complementary representations.

To further understand the benefits of this design, we include the **ST-Random** baseline, where a single model is trained using a randomly selected $\lambda \in \{\lambda_1, \lambda_2\}$ for each sample. This baseline consistently underperforms LG-CoTrain, underscoring the importance of the dual-model structure in which each model makes unique and complementary contributions to the final prediction.

G.1 Distributional Analysis of δ_λ

To quantify the divergence between the two models, we analyze the difference in sample weights:

$$\delta_\lambda = \lambda_1 - \lambda_2.$$

A larger range of δ_λ indicates stronger divergence in how the models prioritize samples, suggesting that the two models are being trained differently and thus forming distinct perspectives.

We find that **Yahoo Answers** and **Yelp Reviews** exhibit the widest ranges of δ_λ (approximately 0 to 0.8), followed by **Amazon Reviews** (approximately 0 to 0.6). In contrast, **AG News** and **IMDB** show narrower ranges (approximately 0 to 0.2 and 0 to 0.1, respectively), suggesting more similar sample prioritization between the models.

Method	Dataset	Hours × Settings × Seeds	Total GPU Hours
USB SSL	IMDB	$8 \times 2 \times 3$	222 GPU Hours (9 GPU Days)
	AG News	$6 \times 2 \times 3$	
	Amazon Review	$8 \times 2 \times 3$	
	Yahoo! Answer	$7 \times 2 \times 3$	
	Yelp Review	$8 \times 2 \times 3$	
VerifyMatch	IMDB	$1 \times 2 \times 3$	195 GPU Hours (8 GPU Days)
	AG News	$2.5 \times 2 \times 3$	
	Amazon Review	$8 \times 2 \times 3$	
	Yahoo! Answer	$13 \times 2 \times 3$	
	Yelp Review	$8 \times 2 \times 3$	
LG-CoTrain	IMDB	$1 \times 2 \times 3$	294 GPU Hours (12 GPU Days)
	AG News	$4 \times 2 \times 3$	
	Amazon Review	$12 \times 2 \times 3$	
	Yahoo! Answer	$20 \times 2 \times 3$	
	Yelp Review	$12 \times 2 \times 3$	

Table 8: Comparison of GPU Budget and Training Setup for Each Method and Dataset

Dataset	Amazon		Yahoo		Yelp	
# Label	250	1000	500	2000	250	1000
AdaMatch-En	$46.24_{0.52}$	$41.27_{0.09}$	$30.53_{0.40}$	$28.39_{0.24}$	$44.31_{0.43}$	$37.96_{0.19}$
SoftMatch-En	$42.97_{0.35}$	$41.15_{0.29}$	$30.24_{0.17}$	$28.48_{0.12}$	$41.71_{0.35}$	$37.44_{0.18}$
Ours	$\mathbf{38.12}_{0.06}$	$\mathbf{37.66}_{0.27}$	$\mathbf{29.38}_{0.16}$	$\mathbf{28.14}_{0.23}$	$\mathbf{33.87}_{0.14}$	$\mathbf{33.52}_{0.12}$

Table 9: **Error rates (%) of ensemble variants.** Comparison between AdaMatch-Ensembled, SoftMatch-Ensembled, and our method on three datasets. Lower is better. Means$_{std}$ over three runs. Best results are **bolded**.

Correspondingly, LG-CoTrain achieves the largest performance improvements over baselines on Yahoo, Yelp, and Amazon, while the gains are smaller on AG News and IMDB. These trends support the hypothesis that greater divergence—and thus more unique contributions from each model—leads to improved overall performance. To better illustrate this relationship, we added Figure 3, which visualizes the distributions of δ_λ across all datasets.
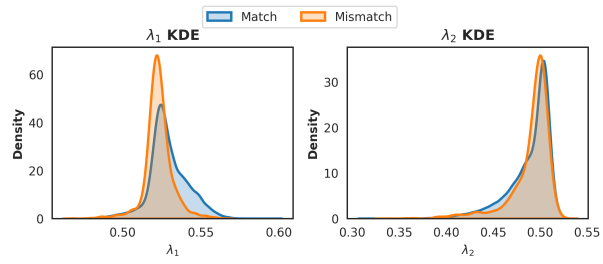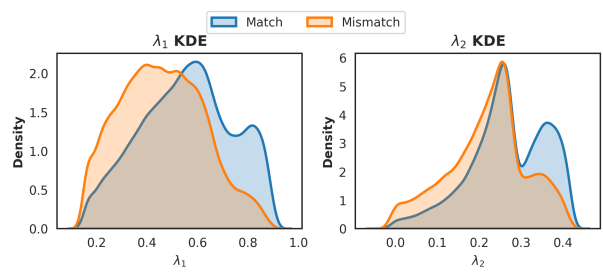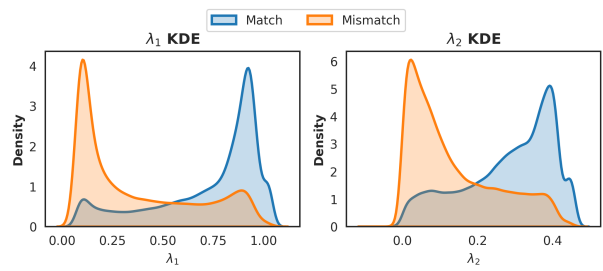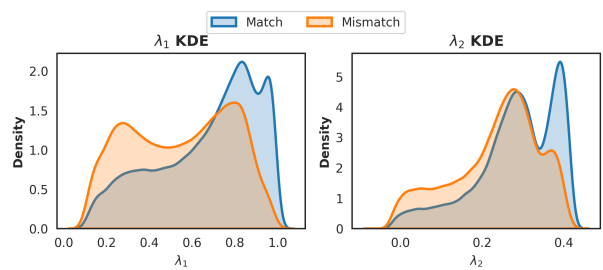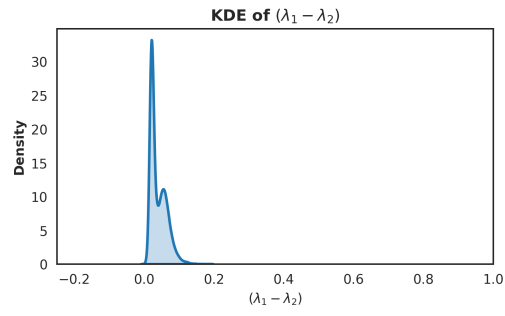
(a) IMDB
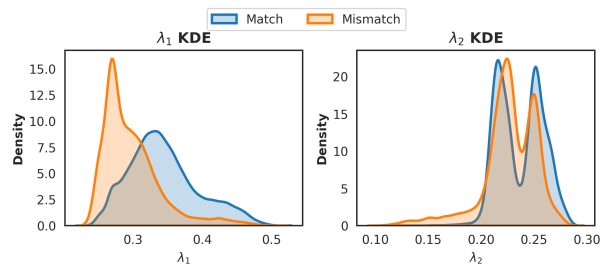
(b) AG NEWS

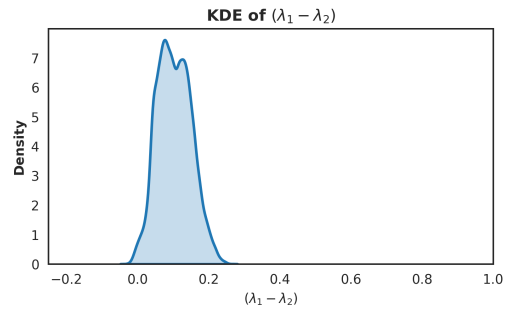

(c) AMAZON REVIEW

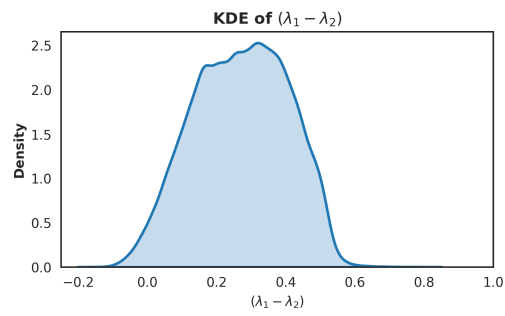(d) YAHOO! ANSWERS

(e) YELP REVIEW

Figure 2: Kernel density plots of λ_1 and λ_2 weights for label-matched and mismatched samples across five datasets.
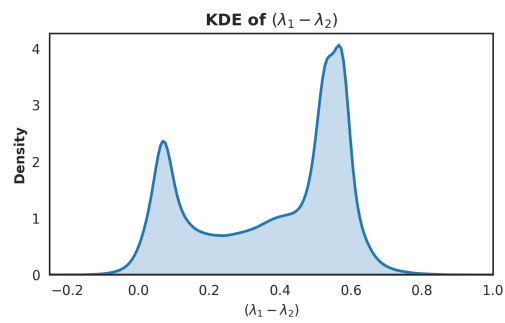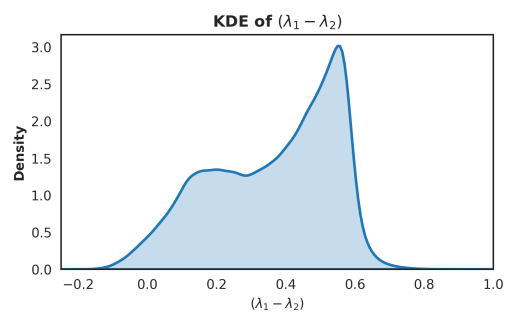
(a) IMDB

(b) AG NEWS

(c) AMAZON REVIEW

(d) YAHOO! ANSWERS

(e) YELP REVIEW

Figure 3: Kernel density plots of $\delta_\lambda = (\lambda_1 - \lambda_2)$ for all samples across five datasets.