Decoupled Optimization for Teacher–Student Semi-Supervised Learning via a Pioneer Student

Haorong Han
School of Computer Science and Technology
Beijing Jiaotong University
harohan@bjtu.edu.cn

Jidong Yuan*
School of Computer Science and Technology
Beijing Jiaotong University
yuanjd@bjtu.edu.cn

Chixuan Wei
School of Data Science and Intelligent Media
Communication University of China
chxwei@cuc.edu.cn

Yongqi Sun
School of Computer Science and Technology
Beijing Jiaotong University
yqsun@bjtu.edu.cn

Abstract

Semi-supervised learning (SSL) relies on two core mechanisms: self-training under the Teacher–Student (T–S) framework and joint optimization of labeled and unlabeled losses. Despite their effectiveness, we find both mechanisms introduce distinct optimization pathologies. First, parameter coupling enforces strict synchronization between teacher and student, where strong regularization on the student degrades the teacher's fitting ability, thereby limiting the permissible generalization intensity. Second, the imbalance in gradient update consistency between labeled and unlabeled losses drives the shared parameters to prematurely converge to labeled-dominated local minima, creating a bottleneck for global optimization. To address both issues, we propose the Pioneer Student (PiS), an auxiliary branch that operates in an independent parameter space and periodically transfers accumulated knowledge back to the T–S model. Extensive experiments show that PiS is a universal plug-and-play module that consistently improves mainstream SSL methods. Code is available at https://github.com/hhrd9/PiS.

1 Introduction

Semi-supervised learning (SSL) aims to leverage unlabeled data to improve model performance under limited supervision [1–4]. It inherently requires strong generalization, not only to extrapolate to unseen data as in supervised learning [5], but more crucially because its training relies heavily on noisy pseudo-labels. Intuitively, increasing the strength of data augmentation should improve robustness and generalization. However, empirical observations in SSL reveal a puzzling phenomenon: as shown in Table 1, while moderate augmentation improves performance, further increasing its strength leads to a clear degradation. For instance, under RandAugment [6], increasing its hyperparameters (i.e., from (2,3) to (3,5), where larger values correspond to stronger transformations) improves performance, whereas pushing them to (5,10) results in a noticeable drop.

Is this the fundamental limit of generalization capacity in SSL? Interestingly, when pseudo-labels pre-recorded offline from a complete run under the (3,5) setting are used as exogenous supervision to train a model from scratch under the (5,10) setting (with identical data ordering ensured by fixing the random seed), performance can be further improved. This reveals that the bottleneck lies not in the model's capacity to withstand strong augmentations, but in the pseudo-labeling mechanism.

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

To understand this discrepancy, we undertake a systematic investigation into the optimization mechanics of the widely adopted Teacher–Student (T–S) framework. Through our analysis (Section 3.1), we identify that the core issue stems from the parameter coupling inherent in the self-training mechanism [4]. We term this the **Fitting–Generalization Dilemma (FGD)**: the student model's pursuit of strong generalization inadvertently propagates through the coupled parameters to degrade the teacher model's fitting fidelity on the training data. This degradation lowers the quality of the generated pseudo-labels, thereby establishing a hard ceiling on the permissible generalization intensity.

Beyond the FGD, it is a standard consensus in SSL to jointly optimize the labeled and unlabeled losses within the same parameter space [4, 7, 9, 10]. However, our investigation (Section 3.2) reveals that this formulation conceals a deeper optimization conflict, which we conceptualize as the **Label-Driven Trap (LDT)**. Put simply, by introducing an *Update Consistency* metric, we observe that the gradient updates driven by the scarce labeled data are highly directed, whereas the unlabeled updates are exploratory. This asymmetry drives the shared parameters to prematurely converge to local minima that favor the labeled loss but are suboptimal for optimizing unlabeled data, thereby creating a bottleneck for global optimization.

Table 1: **Performance of FreeMatch [7] on CIFAR-100 [8] (2 labels/class) under varying augmentation intensities.** Online pseudo-labels generated by the synchronized teacher are endogenous, whereas fixed ones pre-recorded offline (e.g., from an RA(3,5) run) are exogenous.

Supervision Type	Augmentation	Acc. (%)	Pseudo-Label Acc. (%)
Endogenous	RA(2,3)	78.13	77.97
	RA(3,5)	78.46	77.92
	RA(5,10)	76.27	75.25
Exogenous	RA(5,10)	78.80	77.92

Based on our interpretations, it is intriguing that solving these two seemingly unrelated pathologies points toward the same architectural necessity: an independent parameter space. Consequently, we propose the **Pioneer Student (PiS)**, an auxiliary branch whose parameters are completely decoupled from the T–S model. Operating in this isolated parameter space, PiS achieves two forms of decoupled optimization simultaneously: it allows the student to explore extreme generalization without compromising the teacher's fitting fidelity, and it enables the autonomous optimization of the unlabeled loss, free from the interference of the labeled data. Equally important, we introduce the periodic parameter overwrite that teleports the accumulated robust knowledge from PiS back into the T–S model, ensuring that the overall framework achieves stage-wise performance breakthroughs and maximizes the utility of the pioneer branch. We summarize our contributions as follows:

- We analyze and identify two fundamental optimization coupling pathologies in conventional SSL practices: the FGD and the LDT, which restrict the generalization upper bound and create a global optimization bottleneck, respectively.
- We propose PiS, a simple yet highly effective dual-branch strategy that simultaneously resolves both the FGD and the LDT through optimization decoupling.
- As a plug-and-play module, we demonstrate through extensive experiments that PiS significantly enhances existing mainstream SSL methods and exhibits superior practical potential.

2 Related Work

Teacher–Student framework has long been intertwined with the evolution of SSL. The pioneering work Pseudo-labeling [1] establishes the fundamental logic of self-training by converting model predictions on unlabeled data into one-hot labels to supervise a parameter-sharing student network. Subsequent research has evolved along two directions: on the teacher side, Temporal Ensembling [11] improves prediction stability by aggregating historical predictions, while Mean Teacher [12] introduces EMA of weights to construct a more robust teacher. On the student side, VAT [3] creates challenging input conditions through adversarial perturbations, and Noisy Student [13] significantly pushes the generalization upper bound by injecting diverse noise strategies.

Consistency regularization with weak–strong augmentations has revitalized the T–S framework, where the teacher generates pseudo-labels from weakly-augmented views while the student learns under heavy perturbations. Building upon this, FixMatch [4] and its derivatives [14, 7, 15] introduce fixed or dynamic thresholds to filter low-confidence predictions to ensure teacher supervision fidelity. Other efforts [16, 9, 17] improve student regularization via multiple augmentation paths or auxiliary

constraints. Recently, RegMixMatch [10], related to ours, identifies that the high-entropy behavior induced by Mixup [18] can degrade teacher confidence and proposes refinements building upon prior approaches [19, 20]. Transcending the refinement of specific generalization techniques, we elucidate and reconcile the deeper FGD inherent in the T–S framework, enabling universal performance gains across mainstream SSL methods.

Labeled data's optimization role is increasingly scrutinized. MPL [21] leverages model's learning on labeled data to update teacher's parameters, guiding pseudo-label generation. Closely related to our study, FlatMatch [22] empirically identifies rapid labeled loss convergence hinders global optimization, and applies cross-sharpness penalties as mitigation. However, it neither analyzes underlying mechanism nor quantifies its impact on SSL performance, while labeled and unlabeled optimization remain entangled within shared parameters. In contrast, we formalize this pathology as the LDT, directly demonstrate its adverse effect on final accuracy (Section 5.3), and propose PiS as a structural resolution that fully decouples the two optimization into independent parameter spaces.

Dual-branch methods have also been explored in SSL and beyond. In SSL, co-training-based methods [23–26] employ two parallel T–S models to stabilize training. However, each branch still performs joint loss optimization and maintains parameter coupling, leaving both FGD and LDT unresolved while incurring roughly doubled computational overhead. Beyond SSL, the Lookahead optimizer [27] updates slow weights by *interpolating* toward fast weights to obtain a smoother parameter state, whereas our approach employs hard *overwrites* to replace the T-S weights with those of the PiS, forcefully teleporting the model away from labeled-dominated local minima.

3 Coupled Optimization Pathologies in Semi-Supervised Learning

In this section, we systematically dissect the optimization dynamics of the widely adopted T–S framework to uncover two fundamental optimization coupling pathologies: the FGD and the LDT.

3.1 Q1: Why Does SSL Degrade Under Strong Generalization?

The T–S Framework and Self-Training. The cornerstone of modern SSL is the T–S paradigm powered by weak–strong augmentations [4]. Given an unlabeled sample x_u, the student f_{θ_s} produces a prediction $p_s = f_{\theta_s}(\mathcal{A}(x_u))$ on its strongly augmented version $\mathcal{A}(x_u)$, while the teacher f_{θ_t} produces $p_t = f_{\theta_t}(\alpha(x_u))$ on the weakly augmented version $\alpha(x_u)$, from which the pseudo-label $q_u = \arg\max(p_t)$ is derived. The student is optimized via the consistency loss:

$$\mathcal{L}_u(x_u, q_u; \theta_s) = \mathcal{H}(q_u, \sigma(p_s)), \tag{1}$$

where $\sigma(\cdot)$ denotes the softmax operation and $\mathcal{H}(\cdot,\cdot)$ is the cross-entropy loss. Crucially, to form a self-training loop, teacher's parameters θ_t strictly track student's parameters θ_s via direct duplication ($\theta_t \leftarrow \theta_s$) or an EMA update ($\theta_t \leftarrow m\theta_t + (1-m)\theta_s$). This parameter coupling allows the teacher to inherit the student's evolving representations, continuously refining the pseudo-label quality.

The Cost of Parameter Coupling. This elegant self-training mechanism, however, conceals a structural flaw. Unlike supervised learning with fixed ground-truth labels, the supervision signal q_u here is an endogenous variable determined entirely by θ_t. Because θ_t is passively synchronized with θ_s, the single optimization objective $\min_{\theta_s} \mathcal{L}_u$ implicitly binds two conflicting demands: the teacher must perfectly *fit* the training data to yield reliable q_u, whereas the student must aggressively *generalize* to withstand strong augmentations $\mathcal{A}$.

Since fitting and generalization represent opposing optimization directions, enforcing strict parameter synchronization creates a zero-sum game. As shown in Table 1, strong generalization strategies significantly degrade pseudo-label accuracy, ultimately hurting performance. Conversely, when the teacher is relieved from pseudo-label generation and exogenous pseudo-labels obtained under moderate generalization are employed, a student trained with strong generalization strategies achieves better performance.

A1: The Fitting–Generalization Dilemma (FGD). Within the T–S framework, the supervision q_u is not an external ground truth but an endogenous variable that depends on θ_s through the synchronized teacher, leading to the objective $\min_{\theta_s} \mathcal{L}_u(x_u,\, q_u(\theta_s);\, \theta_s)$, where a single parameter state θ_s is forced to simultaneously act as the optimizer of the loss (demanding generalization) and the source of the supervision target $q_u(\theta_s)$ (demanding fitting). The student's pursuit of stronger

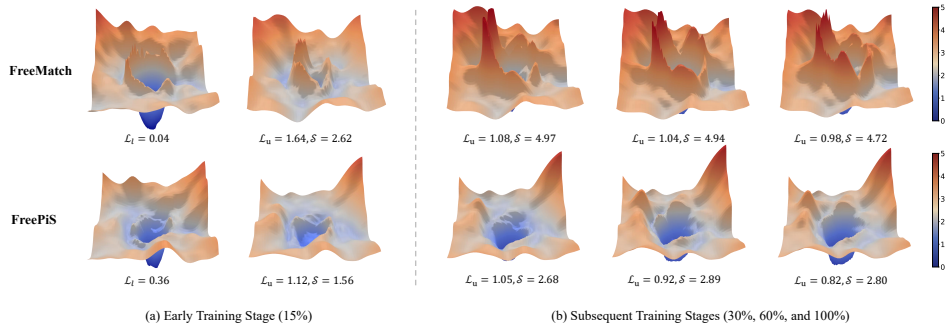


Figure 2: **Visualization of loss landscapes: FreeMatch vs ours (FreePiS).** Landscapes are obtained by perturbing the model parameters (landscape origin) along two orthogonal directions. $\mathcal{L}_l$ and $\mathcal{L}_u$ denote the labeled and unlabeled loss values at the origin, respectively, while $\mathcal{S}$ measures landscape sharpness, which reflects generalization potential (see Appendix B for formal definition).

generalization inevitably degrades the teacher's fitting fidelity, compromising pseudo-label reliability and establishing a hard ceiling on the permissible generalization intensity.

3.2 Q2: What Are the Consequences of Jointly Optimizing Both Losses?

Joint Loss Optimization. SSL methods conventionally optimize both losses within the same parameter space. Given a labeled sample $x_l \in \mathcal{X}_l$ with ground-truth label y_l, the labeled loss is:

$$\mathcal{L}_l(x_l, y_l; \theta_s) = \mathcal{H}(y_l, \sigma(f_{\theta_s}(x_l))). \tag{2}$$

The standard SSL objective then becomes:

$$\min_{\theta_s} \ \mathcal{L}_l(x_l, y_l;\ \theta_s) + \mathcal{L}_u(x_u, q_u;\ \theta_s). \tag{3}$$

Update Consistency. To characterize the optimization dynamics, we introduce *Update Consistency* (UC), which measures the directional coherence of k consecutive gradient updates:

$$\mathrm{UC} = \frac{\left\| \sum_{i=1}^{k} g^{(i)} \right\|}{\sum_{i=1}^{k} \left\| g^{(i)} \right\|}. \tag{4}$$

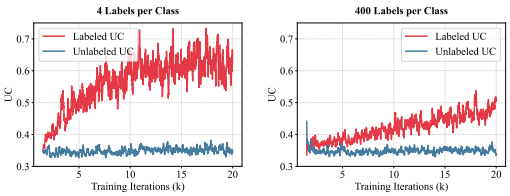


Figure 1: **UC of labeled and unlabeled losses.**

In our experiments, we set $k = 10$. A high UC indicates that gradients consistently point in the same direction, reflecting a directed and dominant optimization signal; a low UC indicates exploratory, diffuse updates. Figure 1 reports the UC of labeled and unlabeled losses in FreeMatch [7] on STL-10 [28] under 4-label and 400-label settings per class *during early training*. We observe that the labeled loss exhibits higher UC than the unlabeled loss, with the gap becoming more pronounced in the 4-label setting. This suggests that, under label scarcity, limited label supervision induces a more directed optimization path than the unlabeled loss, which is optimized over large-scale unlabeled data and thus remains exploratory.

Loss Landscape Analysis. To examine the effect of this early-stage UC imbalance within a shared parameter space, we visualize the loss landscape [29] under the 4-label setting (first row of Figure 2). Three key observations emerge. First, the labeled loss rapidly converges to a near-optimal state ($\mathcal{L}_l = 0.04$), consistent with its high UC. Second, the unlabeled loss landscape closely follows the geometry of the labeled counterpart due to shared parameters, yet remains severely under-optimized ($\mathcal{L}_u = 1.64$) and exhibits significantly higher sharpness, portending poor generalization potential [30]. Third, the unlabeled loss landscapes remains geometrically consistent while maintaining high sharpness throughout subsequent training stages ($\mathcal{S} \in [4.7, 5.0]$), indicating that parameter state shaped in early training permanently constrains the optimization trajectory.

A2: The Label-Driven Trap (LDT). Under label scarcity, the two gradient signals in Eq. 3 are highly imbalanced, with $\mathrm{UC}(\nabla_{\theta_s} \mathcal{L}_l) \gg \mathrm{UC}(\nabla_{\theta_s} \mathcal{L}_u)$, where the highly directed labeled gradient disproportionately biases the early optimization trajectory (see Appendix A for gradient magnitude

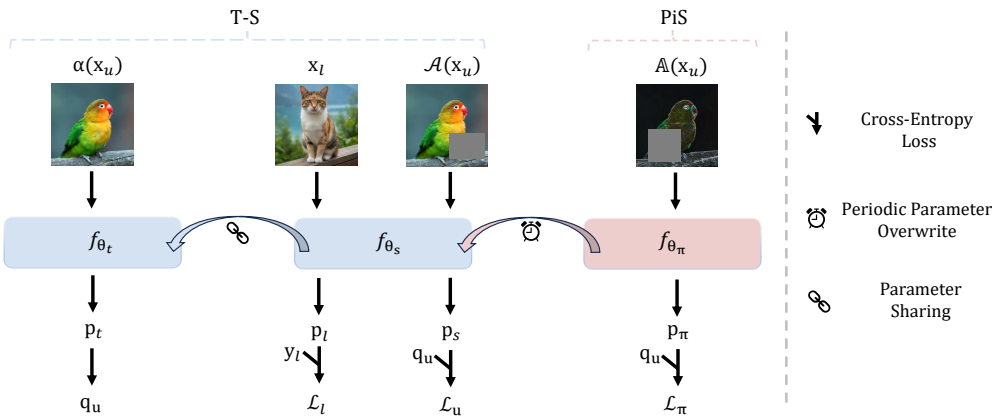


Figure 3: **Overview of T–S model integrated with *Pioneer Student*.** While maintaining the standard T–S logic, where the teacher f_{θ_t} processes weakly augmented image $\alpha(x_u)$ to generate pseudo-label q_u as supervision for the student's prediction p_s on strongly augmented sample $\mathcal{A}(x_u)$, we introduce a parameter-independent PiS model f_{θ_π}. The PiS accepts extremely augmented image $\mathbb{A}(x_u)$ and aligns its prediction p_π with the teacher's supervision. Periodically, the parameters of the T–S model θ_s are overwritten by the parameters of the pioneer student θ_π.

analysis), steering the shared parameters toward a local basin defined by the scarce labeled data. As this basin is shaped by *limited* labeled samples, it is non-global and suboptimal for minimizing the unlabeled loss. More importantly, once the parameters are trapped in this basin, the unlabeled loss can only be optimized within its neighborhood, creating a bottleneck for global optimization.

4 Method

Both FGD and LDT appear to share a common underlying issue: *a single parameter space* is required to simultaneously satisfy *two conflicting optimization demands*, resulting in optimization coupling. To address this, we augment the vanilla T–S framework with an additional, parameter-independent branch: the Pioneer Student f_{θ_π}. As illustrated in Figure 3, PiS is trained under teacher supervision and periodically teleports its accumulated knowledge back to the T–S model via parameter overwrite.

4.1 Independent Parameter Space for Decoupled Optimization

Pioneer Student Branch. Since the parameters θ_π of PiS are isolated and not instantaneously coupled with the T–S model during training, this design creates room for adopting aggressive generalization strategies. Specifically, we employ extreme augmentation $\mathbb{A}$ (see Section 5.4 for definitions) to promote robustness. Given an unlabeled sample x_u, PiS takes its extremely augmented version $\mathbb{A}(x_u)$ as input to generate the prediction $p_\pi = f_{\theta_\pi}(\mathbb{A}(x_u))$. The optimization objective is defined as:

$$\mathcal{L}_\pi(x_u, q_u; \theta_\pi) = \mathcal{H}(q_u, \sigma(p_\pi)), \tag{5}$$

where the pseudo-label supervision q_u is still provided by the teacher network f_{θ_t}. Crucially, the parameter update process of PiS is *exclusively* driven by the unlabeled loss $\mathcal{L}_\pi$:

$$\theta_\pi \leftarrow \theta_\pi - \eta \nabla_{\theta_\pi} \mathcal{L}_\pi. \tag{6}$$

Through this design, PiS simultaneously resolves both optimization couplings. Since θ_π is not instantaneously coupled with the teacher parameters, the PiS's generalization optimization does not continuously compromise the teacher's fitting fidelity, thereby alleviating the FGD. Meanwhile, since the optimization of θ_π is driven exclusively by $\mathcal{L}_\pi$, it is free from interference from non-global optimization signals induced by labeled loss, thereby mitigating the LDT.

4.2 Periodic Knowledge Transfer

Periodic Parameter Overwrite. At first glance, PiS appears capable of resolving both issues on its own. However, it remains under the supervision of the teacher. If left in a purely passive

state of supervision, the inherent biases of the teacher would gradually propagate to PiS, eventually dragging its performance toward mediocrity. To address this, we propose periodic parameter overwrite. Specifically, the training procedure is partitioned into multiple stages, each spanning a fixed proportion ρ of the total training progress. At the end of each stage, the PiS transfers its accumulated robust knowledge back to the T–S model:

$$\theta_s \leftarrow \theta_\pi. \tag{7}$$

By teleporting this robust knowledge back into the T–S framework, we expect to stage-wise raise the performance ceiling of the T–S model. This, in turn, leads to higher-quality pseudo-labels for PiS in subsequent stages, forming a positive feedback loop.

Pioneer Student Warm-Up. Since PiS is tasked with injecting its knowledge back into the T–S model, the reliability of its knowledge is important. Predictably, the convergence of PiS is relatively slow during the initial training phase because: 1) its optimization is entirely detached from the explicit guidance of ground-truth labels; and 2) the pseudo-labels q_u generated by the teacher at early stages are of low quality. Enforcing parameter overwrite during this period could lead to a collapse in the T–S model's performance. Therefore, we introduce a warm-up mechanism: during the initial ϕ fraction of the total training progress, the parameter overwrite is disabled. This ensures that PiS intervenes in the T–S model's optimization only after establishing sufficiently reliable representations. The parameter overwrite schedule with warm-up is formalized as:

$$\theta_s \leftarrow \theta_\pi \quad \text{if} \quad \tau = \phi + n\rho, \quad n = 1, 2, \ldots, \left\lfloor \frac{1-\phi}{\rho} \right\rfloor, \tag{8}$$

where $\tau \in [0, 1]$ represents the current training progress, and n denotes the index of the overwrite stage following the warm-up phase.

Parameter Update Sharing. Although the warm-up strategy addresses the unreliability of early-stage knowledge, in rare SSL scenarios where labels are highly abundant, PiS that relies solely on unlabeled data may fail to match the T–S model that fully exploits the rich supervision throughout training. To ensure the robustness and applicability of PiS across diverse regimes, we propose parameter update sharing, which leverages the architectural consistency between the T–S student and PiS. Specifically, in label-abundant settings, we record the gradient vector of the student network generated by the joint optimization loss (i.e., Equation 3) at each iteration. This gradient is then treated as an auxiliary update signal and weighted into the PiS's own gradient derived from the unlabeled loss:

$$\theta_\pi \leftarrow \theta_\pi - \eta \left(\nabla_{\theta_\pi} \mathcal{L}_\pi + \beta \nabla_{\theta_s} (\mathcal{L}_l + \mathcal{L}_u) \right), \tag{9}$$

where β is a hyperparameter controlling the intensity of the update sharing. By reusing the student's already computed gradient, PiS avoids redundant computation on labeled data, minimizing extra overhead. Meanwhile, the gradient from the optimization-leading student provides a useful optimization guidance for PiS. As shown in Figure 1, when labeled data is abundant, the UC gap between labeled and unlabeled losses is small, making it difficult for the labeled loss to dominate and form local basin. Even if such a basin emerge, *abundant* labeled data ensures they are more beneficial than detrimental to global optimization. Consequently, sharing this gradient allows PiS to keep pace with the T–S model without reintroducing the LDT.

5 Experiments

All experiments are conducted on the Unified Semi-supervised Learning Benchmark (USB) [31]. The experimental settings and training backbone [32] follow prior work [10], including teacher synchronization via direct duplication ($\theta_t \leftarrow \theta_s$). For the hyperparameters of PiS, we set the stage proportion (i.e., the parameter overwrite interval) to $\rho = 2.5\%$ and the warm-up fraction to $\phi = 8.0\%$ of the total training progress across all tasks. The update sharing intensity is fixed to $\beta = 0.6$ when activated. **Unless explicitly specified, update sharing is disabled by default for all other experiments**. To ensure reproducibility, each algorithm is executed three times with different random seeds. Detailed configurations are provided in Appendix C.

5.1 SSL with Pioneer Student

We first present the performance of eight recent SSL methods across three modalities on various datasets: CV (CIFAR-100 [8], STL-10 [28], EuroSAT [34]), audio (UrbanSound8K [35]), and

Table 2: **Top-1 accuracy (%) across CV, audio, and NLP benchmarks.** We report the mean and standard deviation over three runs. ↑ indicates performance improvement after integrating PiS. **Bold** and underline denote the best and second-best results, respectively. * denotes our re-implementation. RegMixMatch results are omitted for audio and NLP due to Mixup incompatibility in these modalities.

Modality	CV						Audio		NLP	
Dataset	CIFAR-100		STL-10		EuroSAT		UrbanSound8K		ACL-IMDb	
# Labels per Class	2	4	4	10	2	4	10	40	10	50
Baseline Methods										
FlexMatch [14]	73.24±1.12	81.76±0.36	85.60±3.11	91.83±0.78	94.83±0.57	94.42±0.81	60.77±0.96	76.30±1.66	92.18±0.77	92.59±0.38
CoMatch [16]	64.92±0.69	74.77±0.50	84.88±1.88	90.44±1.35	94.25±0.43	95.19±1.05	**70.37**±1.15	79.81±2.34	92.56±0.30	92.38±1.14
SimMatch [33]	76.22±1.08	82.94±0.78	88.23±3.20	92.45±1.86	92.34±0.60	94.73±0.89	63.37±6.10	**80.05**±2.11	92.07±0.55	92.92±0.33
SoftMatch [15]	77.33±1.32	83.16±0.66	86.45±3.16	92.16±1.72	94.25±0.62	94.10±1.42	64.68±6.59	78.57±2.37	92.24±0.58	92.03±0.72
SequenceMatch* [9]	78.52±0.46	83.69±0.23	87.77±2.61	92.16±0.55	94.12±0.72	94.99±0.69	-	-	-	-
Plug-in Evaluation (Ours)										
FixMatch* [4]	70.39±0.88	80.16±0.71	83.89±2.06	91.89±0.70	86.69±4.05	94.00±2.27	60.08±5.72	79.16±2.13	92.26±0.27	92.65±0.13
+ PiS	73.99±0.94↑	82.02±0.37↑	86.83±3.49↑	**93.65**±0.51↑	88.71±1.62↑	95.04±0.15↑	66.87±4.88↑	79.95±2.56↑	**92.59**±0.30↑	**92.98**±0.22↑
FreeMatch* [7]	78.46±0.77	83.58±0.63	87.20±2.96	91.51±0.33	93.48±0.78	94.20±0.55	60.23±4.56	75.85±0.78	91.31±0.25	92.09±0.57
+ PiS	80.83±1.64↑	**84.45**±0.44↑	90.65±0.22↑	93.13±0.15↑	94.92±1.45↑	95.89±1.07↑	66.89±3.69↑	79.40±1.87↑	92.22±0.67↑	92.60±0.65↑
RegMixMatch* [10]	80.49±0.62	83.89±0.49	89.97±3.08	92.74±0.75	95.62±0.61	**96.38**±0.74	-	-	-	-
+ PiS	**81.44**±0.97↑	84.19±0.43↑	**91.33**±1.57↑	93.62±1.10↑	**96.02**±0.20↑	96.16±0.60	-	-	-	-

NLP (ACL-IMDb [36]) under varying labeled sample counts. To verify the universality of PiS, we integrate it into three representative baselines: FixMatch [4], FreeMatch [7], and the state-of-the-art RegMixMatch [10].

As shown in Table 2, PiS consistently improves performance across diverse settings. In the CV domain, PiS brings notable gains in low-label regimes. For FixMatch, it achieves +3.60% on CIFAR-100 (2 labels) and +2.94% on STL-10 (4 labels), while reaching the best result on STL-10 (93.65%). Similarly, FreeMatch is consistently improved by PiS, including +2.37% on CIFAR-100 (2 labels) and +3.45% on STL-10 (4 labels), achieving competitive or state-of-the-art performance. Importantly, PiS also enhances the strong RegMixMatch baseline, yielding the best or second-best results across most CV settings. Beyond CV, PiS generalizes effectively to other modalities. It significantly improves performance on UrbanSound8K for FixMatch and FreeMatch (+6.79% and +6.66% at 10 labels), and further refines performance on the near-saturated ACL-IMDb benchmark. Overall, these results demonstrate that PiS is a simple yet broadly applicable plug-in for SSL.

Finally, while PiS introduces non-negligible per-iteration overhead (ranging from 20% to 36% depending on the baseline, see Appendix D for details), it significantly accelerates convergence. As shown in Figure 4, FreeMatch+PiS reaches the performance of FreeMatch at 200k iterations within only 50k (CIFAR-100) and 65k (STL-10) iterations, corresponding to a $3.1\times$–$4\times$ speedup. This improved optimization efficiency largely offsets the additional cost, making PiS a practical and cost-effective solution for SSL. Moreover, PiS maintains a clear advantage even under equal training cost (Appendix D, Table 7).

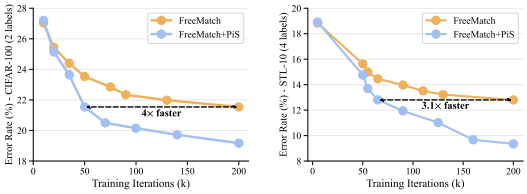


Figure 4: **PiS accelerates convergence by reaching the same performance with significantly fewer training iterations**.

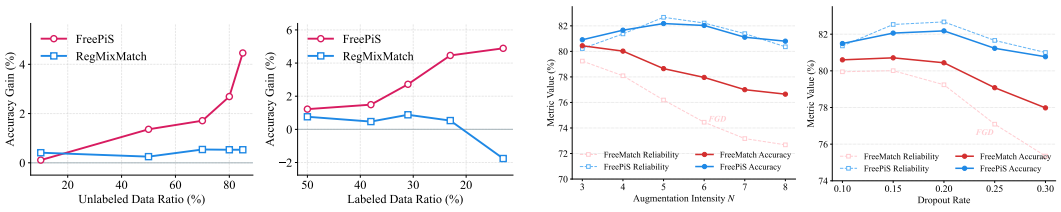
5.2 Applicability Analysis of PiS

While PiS consistently improves SSL performance across standard benchmarks, we further evaluate its behavior under a broader range of supervision densities to assess its applicability boundaries. Note that all subsequent results are based on integrating PiS into FreeMatch, which we refer to as *FreePiS*, **and are compared against other FreeMatch-based counterparts.**

Table 3: **Top-1 accuracy (%) on ImageNet.** † denotes the FreePiS w/ parameter update sharing.

Dataset	ImageNet	
# Labels per Class	10	100
FreeMatch	54.69	72.57
RegMixMatch	58.35	**73.66**
FreePiS	**60.02**	66.69
FreePiS†	–	73.24

ImageNet. Table 3 reports results on ImageNet [37] under two supervision densities. Under the 10-label-per-class setting, FreePiS achieves the best result with a substantial +5.33% gain over FreeMatch, showing its effectiveness in the low-label regime where LDT is most pronounced. Under the label-abundant setting (100 labels/class, totaling 100,000 labeled data), however, FreePiS exhibits a noticeable performance drop. This is consistent with our analysis in Section 4.2: when labeled



(a) Varying Unlabeled Data. (b) Varying Labeled Data. (c) Augmentation Intensity. (d) Dropout Rate.

Figure 5: **Comparative study of model performance under different data scales and regularization intensities.** (a–b) Performance gains across varying scales of labeled and unlabeled data. (c–d) Teacher's pseudo-label reliability and final accuracy under different augmentation intensities and dropout rates [38] (see Appendix F for numerical results; see Section 5.4 for augmentation details).

data is plentiful, a PiS branch that exclusively optimizes the unlabeled loss discards highly reliable supervision, causing it to converge slower and subsequently inject inferior knowledge into the T–S model. This failure case directly motivates the parameter update sharing mechanism. When parameter update sharing is enabled (FreePiS†), PiS is guided by the more informative joint-loss gradient of the T–S student, allowing it to keep pace and serve as a reliable knowledge source. As a result, FreePiS† recovers to 73.24%, effectively improving upon the baseline. The remaining gap relative to RegMixMatch is expected: in the label-abundant regime, the LDT effect is inherently weak, making the mitigation of FGD the primary advantage of PiS rather than the dual benefit in other settings.

Varying Labeled and Unlabeled Data. To further assess the applicability of PiS under varying data regimes, we systematically study its performance across different data scales. Specifically, on the FGVC dataset [39], we examine two complementary scenarios: 1) fixing a constant budget of labeled data at $N_l = 10$ per class while varying the unlabeled volume (Figure 5a), and 2) fixing the unlabeled data at $N_u = 33$ per class while adjusting the labeled data ratio (Figure 5b). Parameter update sharing is activated when the labeled data ratio $\geq 30\%$. We report the **performance gains** of RegMixMatch and FreePiS relative to the FreeMatch baseline.

As illustrated in Figure 5(a–b), the performance gains yielded by FreePiS become increasingly significant as the label ratio decreases or the unlabeled data scale expands. In contrast, despite employing Mixup [18], the gains from RegMixMatch are limited or even negative in certain cases. This finding validates the unique advantage of PiS: **in real-world SSL scenarios characterized by sparse labels and massive unlabeled data, the PiS demonstrates exceptional practical value.**

5.3 Dual Benefits of Pioneer Student

Overcoming the Fitting–Generalization Dilemma. Revisiting the FGD: the student's pursuit of strong generalization often comes at the expense of the teacher's fitting fidelity. In Figure 5(c–d), we illustrate the changes in teacher's reliability (i.e., pseudo-labels accuracy) and classification accuracy on the CIFAR-100 dataset (2 labels per class), where we increase the generalization intensity for the student in FreeMatch and the PiS branch in FreePiS respectively. It is observed that for FreeMatch, aggressive generalization strategies significantly weaken the teacher's reliability, thereby suppressing overall classification performance. In contrast, integrating the PiS branch allows the framework to accommodate both radical generalization strategies and high-level teacher fitting fidelity, suggesting that **PiS effectively elevates the achievable generalization upper bound for SSL.**

Avoiding the Label-Driven Trap. As shown in the second row of Figure 2a, contrary to observations from FreeMatch loss landscapes, incorporating the PiS branch moderates the convergence pace of the labeled loss during early training ($\mathcal{L}_l = 0.36$), while simultaneously steering the unlabeled loss toward a more favorable optimization state ($\mathcal{L}_u = 1.12$) with a flatter loss landscape. This implies that PiS explicitly emphasizes the role of the unlabeled loss in the global optimization process particularly in the early stages, thereby mitigating the interference of the labeled loss and **endowing the model with greater optimization potential.** Consequently, as illustrated in the second row of Figure 2b, the unlabeled loss landscape in the subsequent optimization stages exhibits better flatness ($\mathcal{S} \in [2.6, 2.9]$) compared to FreeMatch, providing direct evidence of its greater generalization.

Disentangling the Dual Advantages. Since PiS inherently addresses both issues, we design two variants to isolate their individual effects: 1) *FreePiS w. Label Loss*: The PiS is optimized with labeled

Table 4: **Ablation study on disentangling the dual advantages of FreePiS.** We evaluate two primary mechanisms: addressing the FGD (Anti-FGD) and overcoming the LDT (Anti-LDT). ✓ indicates the mechanism is **activated**. Δ represents the performance gain relative to the baseline (a). Experiments are conducted on FGVC with 5 labels per class and CIFAR-100 with 2 labels per class.

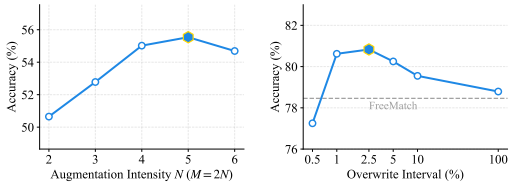
ID	Method/Variant	Mechanisms			FGVC		CIFAR-100	
		Anti-FGD	Anti-LDT	Warm-up	Acc (%)	Δ	Acc (%)	Δ
(*a*)	FreeMatch	✗	✗	–	51.65	–	78.46	–
(*b*)	FreePiS w. Label Loss	✓	✗	✓	54.66	+3.01	79.02	+0.56
(*c*)	FreePiS w. Same Aug.	✗	✓	✓	52.78	+1.13	80.57	+2.11
(*d*)	FreePiS w/o Warm-up	✓	✓	✗	52.25	+0.60	80.66	+2.20
(*e*)	**FreePiS (Full)**	✓	✓	✓	**55.54**	**+3.89**	**80.83**	**+2.37**

loss in addition to teacher supervision (targeting only the FGD); 2) *FreePiS w. Same Aug.*: The PiS adopts the same augmentation intensity as the student (targeting only the LDT). As reported in Table 4, the primary source of improvement varies across datasets. On FGVC, mitigating the FGD (comparing *a* and *b*) accounts for the majority of the performance gain (+3.01%). In contrast, on CIFAR-100, addressing the LDT (comparing *a* and *c*) provides a more significant contribution (+2.11%). Notably, comparing variants *b* and *e* reveals an interesting yet consistent finding: incorporating additional labeled loss into the PiS optimization leads to a 1.81% performance degradation on CIFAR-100. This finding offers direct empirical evidence of **the adverse impact that labeled data can exert on the SSL optimization.** Finally, comparing variants *d* and *e* highlights the necessity of a warm-up phase, with its removal leading to a substantial performance drop on FGVC (-3.29%).

5.4 Hyperparameter Sensitivity Analysis

Augmentation Intensity. SSL typically employs RandAugment [6] as the data augmentation strategy, with parameters N and M controlling the intensity. To simplify hyperparameter tuning, we fix $M = 2N$ and vary the overall intensity solely by adjusting N. Figure 6a depicts the classification accuracy versus augmentation intensity on the FGVC dataset; the model reaches its peak performance at $N = 5$. Accordingly, we apply `RandAugment(5, 10)` for the PiS branch across all tasks.[2]

Parameter Overwrite Interval. Periodic parameter overwrite requires the PiS to transfer its parameters to the teacher at specific training intervals. Intuitively, overly frequent overwriting may impair the teacher's fitting capacity and destabilize training, whereas an excessively long interval causes the knowledge acquired by the PiS to be progressively eroded by teacher supervision. As shown in Figure 6b, omitting periodic overwrite (i.e., setting the interval to 100%) results in performance close to the FreeMatch baseline (dashed line) and **substantially lower than the peak achieved at the optimal interval** (2.5%). These results suggest that parameter overwrite stage-wise raises the performance ceiling of the T–S model, leading to further overall improvement. Finally, although parameter overwrite is conceptually simple, we show in Appendix G that replacing it with softer synchronization mechanisms consistently underperforms direct overwrite, underscoring that the hard overwrite is essential for forcefully teleporting the model out of labeled-dominated local minima.



(a) Augmentation Intensity. (b) Overwrite Interval.

Figure 6: **Sensitivity analysis of FreePiS hyperparameters.** (a) PiS's data augmentation intensity N on FGVC (5 labels). (b) Parameter overwrite interval on CIFAR-100 (2 labels).

6 Conclusion

This paper identifies two optimization pathologies in conventional SSL, the FGD and the LDT, that arise from forcing conflicting optimization demands into a single parameter space. The proposed PiS addresses both issues by operating in an independent parameter space to achieve decoupled

[2]Following standard SSL practice, we use `RandAugment(3, 5)` as the default strong augmentation.

optimization, and by periodically injecting its learned knowledge into the T–S framework. Empirically, PiS consistently improves mainstream SSL methods as a plug-and-play module, with particularly pronounced gains in practical SSL scenarios characterized by sparse labels and abundant unlabeled data. We hope this work provides useful insights for the architectural design of SSL systems.

Limitations and Future Directions. As an auxiliary branch, PiS introduces non-negligible computational overhead during training, motivating future work on more cost-effective decoupled designs. Furthermore, we hope the concept of optimization decoupling can be extended to other SSL tasks (e.g., dense prediction) and even broader paradigms beyond SSL that utilize T–S architectures.

Acknowledgements and Disclosure of Funding

This work is supported in part by the Fundamental Research Funds for the Central Universities (No. 2025JBMC028), and in part by the National Natural Science Foundation of China (No. 61702030).

References

[1] Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Proceedings of the Workshop on Challenges in Representation Learning at ICML 2013*, volume 3, page 896. Atlanta, 2013.

[2] Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. Semi-supervised learning with ladder networks. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

[3] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: A regularization method for supervised and semi-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41:1979–1993, 2019.

[4] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[6] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 702–703, 2020. CVPR 2020 Workshops.

[7] Yidong Wang, Hao Chen, Qiang Heng, Wenxin Hou, Yue Fan, Zhen Wu, Jindong Wang, Marios Savvides, Takahiro Shinozaki, Bhiksha Raj, Bernt Schiele, and Xing Xie. Freematch: Self-adaptive thresholding for semi-supervised learning. In *International Conference on Learning Representations (ICLR)*, 2023.

[8] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[9] Khanh-Binh Nguyen. Sequencematch: Revisiting the design of weak-strong augmentations for semi-supervised learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 96–106, 2024.

[10] Haorong Han, Jidong Yuan, Chixuan Wei, and Zhongyang Yu. Regmixmatch: Optimizing mixup utilization in semi-supervised learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 17032–17040, 2025.

[11] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *International Conference on Learning Representations (ICLR)*, 2017.

[12] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[13] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10687–10698, 2020.

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

[15] Hao Chen, Ran Tao, Yue Fan, Yidong Wang, Jindong Wang, Bernt Schiele, Xing Xie, Bhiksha Raj, and Marios Savvides. Softmatch: Addressing the quantity-quality tradeoff in semi-supervised learning. In *International Conference on Learning Representations (ICLR)*, 2023.

[16] Junnan Li, Caiming Xiong, and Steven CH Hoi. Comatch: Semi-supervised learning with contrastive graph regularization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9475–9484, 2021.

[17] Jiacheng Zhang, Jiaming Li, Xiangru Lin, Wei Zhang, Xiao Tan, Hongbo Gao, Jingdong Wang, and Guanbin Li. Towards efficient semi-supervised object detection with detection transformer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026.

[18] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. Mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations (ICLR)*, 2018.

[19] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[20] David Berthelot, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. In *International Conference on Learning Representations (ICLR)*, 2020.

[21] Hieu Pham, Zihang Dai, Qizhe Xie, and Quoc V Le. Meta pseudo labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11557–11568, 2021.

[22] Zhuo Huang, Li Shen, Jun Yu, Bo Han, and Tongliang Liu. Flatmatch: Bridging labeled data and unlabeled data with cross-sharpness for semi-supervised learning. In *Advances in Neural Information Processing Systems*, volume 36, pages 18474–18494, 2023.

[23] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, COLT' 98, page 92–100, New York, NY, USA, 1998. Association for computing machinery.

[24] Zhanghan Ke, Daoye Wang, Qiong Yan, Jimmy Ren, and Rynson WH Lau. Dual student: Breaking the limits of the teacher in semi-supervised learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6728–6736, 2019.

[25] Jay C Rothenberger and Dimitrios I Diochnos. Meta co-training: Two views are better than one. *arXiv preprint arXiv:2311.18083*, 2023.

[26] Kaiwen Huang, Yizhe Zhang, Yi Zhou, Tianyang Xu, and Tao Zhou. Bidirectional channel-selective semantic interaction for semi-supervised medical segmentation. *arXiv preprint arXiv:2601.05855*, 2026.

[27] Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. Lookahead optimizer: K steps forward, 1 step back. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[28] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, pages 215–223, 2011.

[29] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[30] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

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

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

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

[34] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12:2217–2226, 2019.

[35] Justin Salamon, Christopher Jacoby, and Juan Pablo Bello. A dataset and taxonomy for urban sound research. In *Proceedings of the 22nd ACM International Conference on Multimedia*, pages 1041–1044, 2014.

[36] Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, 2011.

[37] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.

[38] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1): 1929–1958, 2014.

[39] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv*, 2013.

[40] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations (ICLR)*, 2017.

[41] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.

[42] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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*, pages 4171–4186, 2019.

[43] Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation for consistency training. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[44] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021.

A Gradient Magnitude Analysis Supporting the LDT

The UC metric (Eq. 4) characterizes only the directional coherence of gradient updates and is invariant to gradient magnitude. A natural question therefore arises: although the labeled gradients exhibit substantially higher UC, could their magnitudes be negligible compared to the unlabeled gradients, thereby weakening their influence on the optimization trajectory?

To address this concern, we visualize the evolution of the gradient norms $\|\nabla_{\theta_s}\mathcal{L}_l\|$ and $\|\nabla_{\theta_s}\mathcal{L}_u\|$ during the early training stage of FreeMatch on STL-10 under both the 4-label/class and 400-label/class settings (Figure 7).

We observe that, in both settings, the labeled gradient norm is slightly larger than the unlabeled counterpart at the beginning of training, and remains broadly comparable throughout the early optimization stage before the unlabeled gradient gradually becomes dominant later in training.

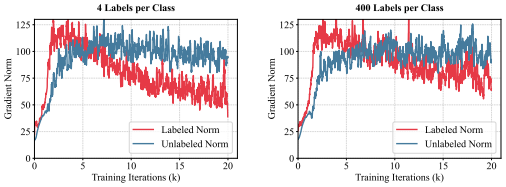


Figure 7: Gradient norms of labeled and unlabeled losses.

These observations support two important conclusions. First, the labeled gradients are not negligible in magnitude during the early stage; they contribute optimization steps of comparable scale to the unlabeled gradients. This validates that the labeled objective exerts a substantial influence on the shared parameter updates.

Second, because the gradient norms are broadly comparable during the critical early stage, the primary asymmetry between the two optimization signals lies in their directional behaviors. In this regime, optimization dynamics are increasingly governed by directional consistency rather than instantaneous magnitude. Consequently, the significantly higher UC of the labeled gradients explains why the shared parameters are preferentially steered toward a labeled-dominated local basin, giving rise to the Label-Driven Trap.

B Quantifying Loss Sharpness and Generalization

B.1 Loss Flatness and Generalization

The geometry of the loss landscape is closely related to the generalization capability of deep learning models [40, 30, 29]. A widely accepted hypothesis in optimization literature is that "flat" local minima generalize better than "sharp" ones. In a flat region, the loss value remains low even when model parameters $\boldsymbol{w}$ are subject to small perturbations $\boldsymbol{\delta}$, indicating robustness to shifts between training and testing data distributions. Conversely, a sharp minimum is characterized by a rapid increase in loss upon slight parameter changes, making it highly sensitive to noise and prone to poor generalization on unseen data.

B.2 Directional Sharpness Definition

Following previous work [29], we measure sharpness by exploring the loss surface along random directions in a reduced-dimensional subspace. Given a converged parameter point $\boldsymbol{w}$, we select two random orthogonal directions $\boldsymbol{d}_1, \boldsymbol{d}_2 \in \mathbb{R}^n$. To ensure the perturbation is scale-relevant, these directions are normalized to match the global Frobenius norm of the parameter vector $\boldsymbol{w}$. This ensures that the displacement in the parameter space is proportional to the overall magnitude of the learned weights.

We define a local perturbation $\boldsymbol{\delta}(\alpha, \beta) = \alpha \boldsymbol{d}_1 + \beta \boldsymbol{d}_2$, where α and β are scalar coefficients. The sharpness $\mathcal{S}$ is then quantified by the maximum loss increase within a local neighborhood $\mathcal{C}_\epsilon$:

$$\mathcal{S} = \max_{(\alpha,\beta)\in\mathcal{C}_\epsilon} \left(f(\boldsymbol{w} + \alpha \boldsymbol{d}_1 + \beta \boldsymbol{d}_2) - f(\boldsymbol{w}) \right) \tag{10}$$

where $\mathcal{C}_\epsilon = \{(\alpha, \beta) : -\epsilon \leq \alpha, \beta \leq \epsilon\}$ defines a square region in the 2D subspace. Note that we omit the scaling denominator $(1 + f(\boldsymbol{w}))$ used in earlier work [40] because FreeMatch [7] and FreePiS are evaluated under identical configurations, making the absolute loss increase $\mathcal{S}$ a more direct and transparent measure of the local sharpness of the loss landscape.

B.3 Implementation Details and Parameters

For the results and 3D visualizations presented in Figure 2, we set the neighborhood scale $\epsilon = 1.0$. The loss values are computed on a grid of size 50×50 within the range $\alpha, \beta \in [-1.0, 1.0]$. The directions $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$ are sampled from a Gaussian distribution, re-orthogonalized, and then scaled such that $||\boldsymbol{d}_1||_2 = ||\boldsymbol{d}_2||_2 = ||\boldsymbol{w}||_2$.

C Detailed Training Configurations and Backbones

To ensure reproducibility, we provide the comprehensive experimental settings in Table 5. Our training protocols and backbones are strictly aligned with the USB [31] and the configurations used in RegMixMatch [10]. All experiments in Table 2 are conducted on a single NVIDIA RTX 3090 GPU with 24GB memory. Each training run takes approximately 12–20 hours, depending on the dataset and configuration.

Table 5: Hyperparameter configurations and backbones for different datasets.

Dataset	CIFAR-100	STL-10	EuroSAT	FGVC-Aircraft	ImageNet-1K
Image Size	32	96	32	224	224
Backbone (ViT [41])	S-P2-32	B-P16-96	S-P2-32	B-P16-224	B-P16-224
Weight Decay	5e-4	5e-4	5e-4	5e-2	5e-2
Learning Rate	5e-4	1e-4	5e-5	5e-4	3e-4
Layer Decay Rate	0.5	0.95	1.0	0.5	0.5
Labeled Batch Size	8				32
Unlabeled Batch Size	8				32
LR Scheduler	$\eta_k = \eta_0 \cos\left(\frac{7\pi k}{16K}\right)$				
Model EMA Momentum	0.0				
Prediction EMA Momentum	0.999				
Weak Augmentation	Random Crop, Random Horizontal Flip				
Strong Augmentation	RandAugment [6]				
Pioneer Student Hyperparameters	Stage Proportion $\rho = 2.5\%$, Warm-up $\phi = 8\%$, Update Sharing Intensity $\beta = 0.6$ (if activated)				

Table 6: Computational overhead of PiS on a single NVIDIA RTX 3090 GPU. We report per-iteration training time and peak GPU memory consumption for baseline methods and their PiS-augmented variants.

Method	**Time (s/iter)**	**+PiS**	**Memory (MiB)**	**+PiS**
FixMatch	0.0944	0.1256 (+33%)	3359	4534 (+35%)
FreeMatch	0.0960	0.1306 (+36%)	3341	4543 (+36%)
RegMixMatch	0.1344	0.1613 (+20%)	4616	5446 (+18%)

For NLP tasks, we use the pre-trained BERT-Base [42] as the backbone. The strong augmentation is Back-Translation [43]. For PiS, which requires extreme augmentation, we increase the sampling temperature from 0.9 (standard strong augmentation) to 1.1. For audio tasks, we adopt HuBERT-Base [44] as the backbone. The strong augmentation includes random sub-sample, random gain, random pitch, and random speed. For the extreme augmentation in PiS, we increase the augmentation magnitude range during training. Other details strictly follow the USB benchmark.

D Computational Overhead Analysis

Although PiS substantially reduces computational cost compared to co-training methods [23–25] by reusing teacher supervision and optimizing exclusively on unlabeled data, introducing an auxiliary branch still inevitably incurs non-negligible overhead (see Table 6). Specifically, due to differences in algorithmic design, integrating PiS into FixMatch, FreeMatch, and RegMixMatch results in approximately 33%, 36%, and 20% additional per-iteration training time, respectively. In terms of memory footprint, the PiS branch increases peak GPU memory consumption by approximately 35%, 36%, and 18% compared to the respective baselines.

As discussed in Section 5.1, this per-iteration overhead is largely offset by the accelerated convergence. FreePiS requires significantly fewer iterations to reach the baseline's final performance, achieving a $3.1\times$–$4.0\times$ reduction in training iterations. Accounting for the 36% per-iteration time overhead, the corresponding wall-clock speedup is approximately $2.3\times$–$2.9\times$ in practice.

Furthermore, while the standard USB benchmark evaluates all methods under a fixed budget of 200k iterations, we also conduct experiments with a reduced iteration budget for FreePiS to match the total wall-clock training time of the FreeMatch baseline. As shown in Table 7, even under strict wall-clock time parity (i.e., training FreePiS for 147k iterations in total), FreePiS maintains a substantial performance advantage, confirming its practical efficiency.

Table 7: Performance comparison under equal wall-clock training time. FreePiS is trained for fewer iterations to match the total training time of FreeMatch (200k iterations).

Dataset	CIFAR-100		STL-10		EuroSAT	
# Labels per Class	2	4	4	10	2	4
FreeMatch (200k iters)	78.46	83.58	87.20	91.51	93.48	94.20
FreePiS (147k iters)	**80.61**	**84.27**	**90.44**	**92.85**	**94.70**	**95.84**
FreePiS (200k iters)	80.83	84.45	90.65	93.13	94.92	95.89

E Absolute Performance under Varying Data Scales

This section reports the exact sample counts, data ratios, and absolute classification accuracy corresponding to the experiments shown in Figure 5(a–b).

E.1 Performance with Fixed Labeled Samples ($N_l = 10$)

In this setting, the number of labeled samples is fixed at 10 per class, while the amount of unlabeled data (N_u) is progressively increased. The unlabeled data ratio is defined as $N_u/(N_l + N_u)$. Table 8 summarizes the absolute performance under these conditions.

Table 8: Absolute classification accuracy (%) with a fixed labeled count ($N_l = 10$). ‡ denotes that parameter update sharing is enabled.

Method	9%‡ ($N_u = 1$)	50%‡ ($N_u = 10$)	71% ($N_u = 25$)	80% ($N_u = 40$)	85% ($N_u = 56$)
FreeMatch	56.45	59.88	63.01	65.66	65.87
RegMixMatch	**56.86**	60.13	63.55	66.19	66.40
FreePiS	56.56	**61.24**	**64.72**	**68.35**	**70.33**

E.2 Performance with Fixed Unlabeled Samples ($N_u = 33$)

In this setting, the number of unlabeled samples is fixed at 33 per class, while the labeled data volume (N_l) varies. The labeled data ratio is defined as $N_l/(N_l + N_u)$. Table 9 reports the corresponding results.

Table 9: Absolute classification accuracy (%) with a fixed unlabeled count ($N_u = 33$). ‡ denotes that parameter update sharing is enabled.

Method	50%‡ ($N_l = 33$)	38%‡ ($N_l = 20$)	31%‡ ($N_l = 15$)	23% ($N_l = 10$)	13% ($N_l = 5$)
FreeMatch	83.34	74.77	70.39	62.87	49.65
RegMixMatch	84.10	75.24	71.27	63.40	47.89
FreePiS	**84.56**	**76.26**	**73.11**	**67.33**	**54.54**

F Tabulated Results for Reliability and Accuracy

For completeness, we report the pseudo-label reliability and classification accuracy shown in Figure 5(c–d) in tabular form. The corresponding numerical results are summarized in Table 10.

We note that the numerical results reported in Table 10 (and Figure 5(c–d)) are obtained from a single run with a fixed random seed (seed0) on CIFAR-100 with 2 labels per class. In contrast, the main results reported in Table 2 are averaged over three independent runs with different random seeds (seed0, seed1, seed2), following standard evaluation protocols.

As can be clearly observed, for FreeMatch, increasing the generalization intensity leads to a substantial degradation in the reliability of teacher-generated pseudo-labels. These unreliable pseudo-labels misguide the training process, ultimately resulting in a pronounced decline in classification accuracy.

Table 10: Tabulated results for classification accuracy and pseudo-label reliability under varying augmentation intensities N and dropout rates.

Metric	Method	Augmentation Intensity N ($M = 2N$)						Dropout Rate				
		3	4	5	6	7	8	0.10	0.15	0.20	0.25	0.30
Accuracy (%)	FreeMatch	80.44	80.02	78.65	77.96	77.00	76.65	80.60	80.71	80.44	79.08	77.98
	FreePiS	**80.91**	**81.66**	**82.18**	**82.03**	**81.11**	**80.80**	**81.49**	**82.06**	**82.18**	**81.23**	**80.77**
Reliability (%)	FreeMatch	79.24	78.09	76.19	74.45	73.17	72.68	79.95	80.01	79.24	77.09	75.35
	FreePiS	**80.22**	**81.37**	**82.67**	**82.22**	**81.39**	**80.36**	**81.36**	**82.53**	**82.67**	**81.66**	**81.00**

G Synchronization Mechanism Ablation

The periodic parameter overwrite in PiS transfers the pioneer student's parameters to the T–S model via a hard replacement: $\theta_s \leftarrow \theta_\pi$. While conceptually simple, this design choice is deliberate. In this appendix, we compare hard overwrite against one family of softer synchronization mechanisms to justify this design and illuminate why a hard replacement is essential for escaping labeled-dominated local minima.

G.1 Synchronization Mechanisms

We evaluate the following synchronization strategies, all applied at the same stage boundaries as the original hard overwrite (i.e., every $\rho = 2.5\%$ of total training progress, corresponding to 5,000 iterations in our standard 200,000-iteration training schedule).

Hard Overwrite (HO). The original mechanism. At the end of each stage, the T–S student parameters are *entirely replaced* by the pioneer student parameters:

$$\theta_s \leftarrow \theta_\pi. \tag{11}$$

This constitutes a one-shot, complete transfer of the pioneer student's accumulated knowledge, with no retention of the T–S student's prior state.

Linear Interpolation (LI-α). At the end of each stage, the T–S student parameters are updated as a convex combination of the current T–S student and the pioneer student:

$$\theta_s \leftarrow (1-\alpha)\,\theta_s + \alpha\,\theta_\pi, \quad \alpha \in (0, 1]. \tag{12}$$

When $\alpha = 1$, this reduces to Hard Overwrite. Smaller α values retain more of the T–S student's prior state, producing a smoother but less decisive parameter update. We evaluate $\alpha \in \{0.3, 0.5, 0.7\}$.

G.2 Results and Analysis

Table 11: Classification accuracy (%) on CIFAR-100 under different synchronization mechanisms. All PiS variants use the same stage proportion $\rho = 2.5\%$ and warm-up $\phi = 8\%$.

Method	Sync Mechanism	Accuracy (%)
FreeMatch	—	78.46
FreePiS + LI-α=0.3	Linear Interp. (α=0.3)	78.80
FreePiS + LI-α=0.5	Linear Interp. (α=0.5)	79.34
FreePiS + LI-α=0.7	Linear Interp. (α=0.7)	80.19
FreePiS + HO	**Hard Overwrite (α=1.0)**	**80.83**

All experiments are conducted on CIFAR-100 with 2 labeled samples per class using FreePiS. As shown in Table 11, hard overwrite consistently outperforms all softer synchronization alternatives. A key observation emerges from these results.

Monotonic improvement with α in Linear Interpolation. Performance increases monotonically with α, reaching its peak at $\alpha = 1.0$ (i.e., Hard Overwrite). This monotonic relationship directly supports our interpretation of the overwrite as a *parameter-space teleportation*: the more completely the T–S student's parameters are replaced by the pioneer student's, the more decisively the model escapes the labeled-dominated local minimum in which it has been trapped. A partial replacement ($\alpha < 1$) retains a residual component of the trapped parameters, partially preserving the very sharp minimum that PiS is designed to escape.

These results suggest that the seeming simplicity of hard overwrite is in fact its strength. The abrupt and complete replacement enables the T–S model to escape labeled-dominated local minima, which smooth and partial blending mechanisms fundamentally fail to achieve.

H Assets and Licensing

In compliance with the NeurIPS checklist, we provide a comprehensive summary of the datasets and codebases used in this work. All assets are used strictly for non-commercial academic research in accordance with their respective usage terms.

Table 12: Summary of datasets and codebases with usage terms.

Asset	Modality	Type	License / Terms	Reference
Datasets				
CIFAR-10/100	CV	Dataset	Non-commercial research use	[8]
STL-10	CV	Dataset	Non-commercial research use	[28]
ImageNet-1K	CV	Dataset	ImageNet Terms of Access (non-commercial use)	[37]
EuroSAT	CV	Dataset	MIT License	[34]
FGVC-Aircraft	CV	Dataset	Non-commercial research use	[39]
UrbanSound8K	Audio	Dataset	CC BY-NC-ND 4.0	[35]
ACL-IMDb	NLP	Dataset	Released for research use (non-commercial)	[36]
Codebases				
USB	Framework	Codebase	MIT License	[31]
FixMatch	Algorithm	Codebase	Open-source implementation (license varies by repository)	[4]
FreeMatch	Algorithm	Codebase	MIT License	[7]
RegMixMatch	Algorithm	Codebase	Open-source implementation (MIT-style license)	[10]

Dataset Usage Details

- **Computer Vision (CV)**: CIFAR-10/100 and STL-10 are standard benchmarks widely used in academic research under non-commercial usage conditions. EuroSAT is distributed under the MIT License. FGVC-Aircraft is used under its original non-commercial research agreement. ImageNet-1K is used under the official ImageNet Terms of Access.
- **Audio**: UrbanSound8K is used under the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND 4.0) license.
- **NLP**: The ACL-IMDb dataset is used under its original research-use-only release conditions.

Software and Codebases

- **USB (Unified SSL Benchmark)**: Built upon the USB framework, which is released under the MIT License.
- **Baseline Implementations**: We use official or widely adopted open-source implementations for all baseline methods. These implementations follow their respective repository licenses, typically MIT or similar permissive licenses.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: We clearly state the claims regarding the Fitting-Generalization Dilemma (FGD) and Label-Driven Trap (LDT) in the abstract and introduction, which strictly match our theoretical analysis and experimental results using the Pioneer Student (PiS) branch.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss the performance trade-offs in high-label scenarios (Section 5.2) that necessitate the parameter update sharing mechanism, as well as the increased single-step computational overhead (Section 6) introduced by the dual-branch architecture.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [N/A]

Justification: This paper does not present formal theoretical results such as theorems or proofs. We introduce the Update Consistency (UC) metric as a formal definition and provide empirical analysis, including loss landscape visualization, to support our claims regarding the Label-Driven Trap (LDT).

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: We provide full architectural details of the plug-and-play PiS module (Section 4). We also include key experimental configurations in Appendix C.

Guidelines:

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

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [Yes]

   Justification: We include the anonymized source code, environment dependencies, and executable scripts for reproducing the experimental results in the supplemental material.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We detail the specific hyperparameter configurations introduced by PiS (e.g., parameter overwrite interval $\rho = 2.5\%$, warmup ratio $\phi = 8.0\%$, update sharing strength $\beta = 0.6$, see Section 5), as well as standard SSL baseline settings (Appendix C).

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: We report the mean and standard deviation over three independent runs with different random seeds for all main results (see Tables 2 in Section 5). The variability arises from random initialization and data sampling, and the reported error bars correspond to standard deviation.

Guidelines:

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We provide the compute resources used in our experiments, including GPU type (RTX 3090 with 24GB memory) and training time per run (12–20 hours), as described in Appendix C.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: This fundamental algorithmic research strictly adheres to the NeurIPS Code of Ethics and does not raise ethical concerns.

   Guidelines:

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

10. **Broader impacts**

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

Answer: [N/A]

Justification: This work focuses on foundational optimization theory and architectural designs for semi-supervised learning; there are no foreseeable, direct negative societal impacts tied to specific applications.

Guidelines:

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

11. **Safeguards**

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

Answer: [N/A]

Justification: Our contribution is a general optimization framework; we do not release any high-risk assets, large pre-trained generative models, or newly scraped datasets.

Guidelines:

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

12. **Licenses for existing assets**

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

Answer: [Yes]

Justification: We properly cite all datasets and codebases used in this work and explicitly specify their licenses (see Appendix H for a complete list of asset licenses).

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

Answer: [N/A]

Justification: The paper proposes a novel optimization algorithm and architectural module; it does not introduce new standalone datasets or similar assets requiring independent documentation.

Guidelines:

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

14. **Crowdsourcing and research with human subjects**

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

Answer: [N/A]

Justification: The research relies solely on computational evaluations using standard academic benchmarks and does not involve any human subjects or crowdsourcing.

Guidelines:

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

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

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

Answer: [N/A]

Justification: The research does not involve human subjects, making IRB approval inapplicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

Answer: [N/A]

Justification: Large Language Models (LLMs) are not utilized as a component of the core Pioneer Student methodology or the semi-supervised learning algorithms proposed.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.