Appendix: Rethinking pseudo-labeling: Data-centric insights improve semi-supervised learning

Table of Contents

A Additional details on DIPS

A.1 DIPS algorithm

We summarize how DIPS can be incorporated into *any* pseudo-labeling algorithm as per Algorithm 1. For clarity, we highlight in red how DIPS extends the current pseudo-labeling formulations.

Algorithm 2 Plug DIPS into *any* pseudo-labeler

```
1: Train a network, $f^{(0)}$, using the samples from $\mathcal{D}_{\text{lab}}$.
2: Plug-in DIPS: set $\mathcal{D}^{(1)}_{\text{train}} = \mathcal{D}^{(1)} = r(\mathcal{D}_{\text{lab}}, f^{(0)})$    ▷ Initialization of the training set
3: for $t$ = 1..T do
4:     Initialize new network $f^{(t)}$
5:     Train $f^{(t)}$ using $\mathcal{D}^{(t)}_{\text{train}}$.
6:     Pseudo-label $\mathcal{D}_{\text{unlab}}$ using $f^{(t)}$
7:     Define $\mathcal{D}^{(t+1)}$ using the PL method's selector $s$
8:     Plug-in DIPS : Define $\mathcal{D}^{(t+1)}_{\text{train}} = r(\mathcal{D}^{(t+1)}, f^{(t)})$    ▷ Data characterization and selection, Sec. 4.3
9: end for
10: return $f_T$
```

We emphasize a key advantage of DIPS lies in its simplicity. Beyond getting the computation almost for free (no additional models to train when instantiating DIPS with learning dynamics) - i.e. **P3: Computationally cheap**, we can also plug DIPS into and augment *any* pseudo-labeling algorithm. (i.e. **P1: Plug & Play**), which makes for easier adoption.

A.2 Overview of pseudo-labeling methods

As we described in Sec. 3.1, current pseudo-labeling methods typically differ in the way the selector function s is defined. Note that s is solely used to select *pseudo-labeled samples*, among those which have not already been pseudo-labeled at a previous iteration. The general way to define s for any set $\mathcal{D}$ and function f is $s(\mathcal{D}, f) = \{(x, [\hat{y}]_k) | x \in \mathcal{D}, \hat{y} = f(x), k \in [C], [m(x,f)]_k = 1\}$, where m is such that $m(x,f) \in \mathbb{R}^C$. Alternatively stated, a selector m outputs the binary decision of selecting x and its associated pseudo-labels $\hat{y}$. Notice that we allow the multi-label setting, where $C > 1$, hence explaining why the selector m's output is a vector of size C. We now give the intuition of how widely used PL methods construct the selector m, thus leading to specific definitions of s.

Greedy pseudo-labeling (Lee et al., 2013) The intuition of greedy pseudo-labeling is to select a pseudo-label if the classifier is sufficiently confident in it. Given two positive thresholds τ_p and τ_n with $\tau_n < \tau_p$, and a classifier f, m is defined by $[m(x,f)]_k = \mathbb{1}([f(x)]_k \geq \tau_p) + \mathbb{1}([f(x)]_k \leq \tau_n)$ for $k \in [C]$.

UPS (Rizve et al., 2021) In addition to confidence, UPS considers the uncertainty of the prediction when selecting pseudo-labels. Given two thresholds $\kappa_n < \kappa_p$, UPS defines $[m(x,f)]_k = \mathbb{1}(u([f(x)]_k) \leq \kappa_p)\mathbb{1}([f(x)]_k \geq \tau_p) + \mathbb{1}(u([f(x)]_k) \leq \kappa_n)\mathbb{1}([f(x)]_k \leq \tau_n)$, where u is a proxy for the uncertainty. One could compute u using MC-Dropout (for neural networks) or ensembles.

Flexmatch (Zhang et al., 2021) FlexMatch dynamically adjusts a class-dependent threshold for the selection of pseudo-labels. The selection mechanism is defined by: $[m(x,f)]_k = \mathbb{1}(\max_j [f(x)]_j > \tau_t(\arg\max_i [f(x)]_i))$, i.e. at iteration t, an unlabeled point is selected if and only if the confidence of the most probable class is greater than its corresponding dynamic threshold.

SLA (Tai et al., 2021) and CSA (Nguyen et al., 2022a) The fundamental intuition behind SLA and CSA is to solve a linear program in order to assign pseudo-labels to unlabeled data, based on the predictions $f(x)$. The allocation of pseudo-labels considers both the rows (the unlabeled samples) and the columns (the classes), hence contrasts greedy pseudo-labeling, and incorporates linear constraints in the optimization problem. An approximate solution is found using the Sinkhorn-Knopp algorithm.

A.3 LEARNING DYNAMICS PLOT

Key to our instantiation of DIPS is the analysis of learning dynamics. We illustrate in Fig. 9 for a pseudo-labeling run in Sec. 5.5 the learning dynamics of 6 individual samples. DIPS uses these dynamics to compute the metrics of confidence and aleatoric uncertainty, as explained in Sec. 4.2. This then characterizes the samples as *Useful* or *Harmful*.

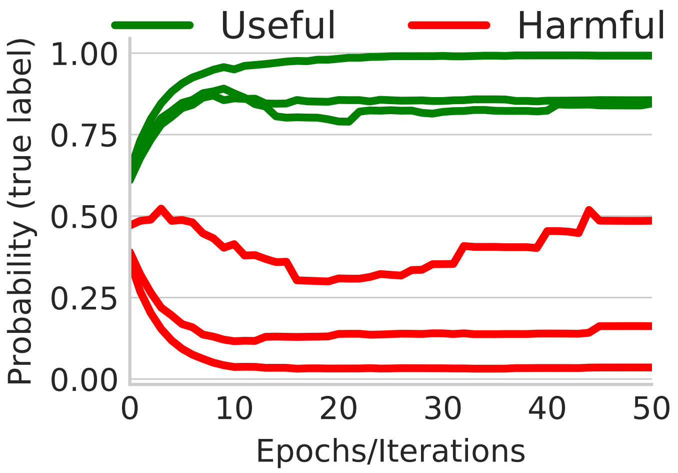


Figure 9: DIPS uses learning dynamics to compute the metrics of confidence and aleatoric uncertainty. It then characterizes labeled (and pseudo-labeled) samples based on these metrics. The y axis represents the probability assigned to the true label by the current classifier h, which is $P(\hat{Y}_{\mathcal{F}}(x, y) = 1|H = h)$.

The intuition is the following: a sample is deemed *Useful* if the classifiers at each checkpoint are confident in predicting the associated (pseudo-)label of the sample, and the associated aleatoric uncertainty is low. Failure to satisfy this criterion leads to a characterization as *Harmful*.

A.4 COMPARISON WITH SCHMARJE ET AL. (2022)

In what follows, we compare DIPS with DC3 (Schmarje et al., 2022). While both DIPS and DC3 handle the data-centric issue of issues in data and share similarities in their titles, they tackle different data-centric problems which might arise in semi-supervised learning. The main differences are along 4 different dimensions:

1. **Problem setup/Type of data-centric issue**: DIPS tackles the problems of hard noisy labels where each sample has a single label assigned in the labeled set which could be incorrect. In contrast, DC3 deals with the problem of soft labeling where each sample might have multiple annotations from different annotators which may be variable.
2. **Label noise modeling**: DIPS aims to identify the noisy labels, whereas DC3 models the inter-annotator variability to estimate label ambiguity.
3. **Integration into SSL**: DIPS is a plug-in on top of any pseudo-labeling pipeline, selecting the labeled and pseudo-labeled data. DC3 on the otherhand uses its ambiguity model (learned on the multiple annotations) to either keep the pseudo-label or use a cluster assignment.
4. **Dataset applicability**: DIPS has lower dataset requirements as it can be applied to any dataset with labeled and unlabeled samples, even if there is only a single label per sample. It does not require multiple annotations. DC3 has higher dataset requirements as it relies on having multiple annotations per sample to estimate inter-annotator variability and label ambiguity. Without multiple labels per sample, it cannot estimate ambiguity and perform joint classification and clustering. Consequently, DIPS is applicable to the standard semi-supervised learning setup of limited labeled data and abundant unlabeled data, whereas DC3 targets the specific problem of ambiguity across multiple annotators.

A.5 Connection to active learning

Active learning primarily focuses on the iterative process of selecting data samples that, when labeled, are expected to most significantly improve the model's performance. This selection is typically based on criteria such as uncertainty sampling which focuses on **epistemic uncertainty** (Mussmann & Liang, 2018; Houlsby et al., 2011; Kirsch et al., 2019; Nguyen et al., 2022b). The primary objective is to minimize labeling effort while maximizing the model's learning efficiency. In contrast, DIPS does both labeled and pseudo-labeled selection and employs the term 'useful' in a different sense. Here, 'usefulness' refers to the capacity of a data sample to contribute positively to the learning process based on its likelihood of being correctly labeled. Our approach, which leverages training dynamics based on **aleatoric uncertainty** and confidence, is designed to flag and exclude mislabeled data. This distinction is critical in our methodology as it directly addresses the challenge of data quality, particularly in scenarios where large volumes of unlabeled data are integrated into the training process. In active learning, these metrics are used to identify data points that, if labeled, would yield the most significant insights for model training. In our approach, they serve to identify and exclude data points that could potentially deteriorate the model's performance due to incorrect labeling.

B EXPERIMENTAL DETAILS

B.1 BASELINES

In our experiments, we consider the following baselines for pseudo-labeling: Greedy-PL (Lee et al., 2013), UPS (Rizve et al., 2021), Flexmatch (Zhang et al., 2021), SLA (Tai et al., 2021) and CSA (Nguyen et al., 2022a). To assess performance of DIPS for computer vision, we use FixMatch (Sohn et al., 2020).

Greedy-PL The confidence upper threshold is 0.8

UPS The confidence upper threshold is 0.8 and the threshold on the uncertainty is 0.2. The size of the ensemble is 10.

FlexMatch The upper threshold is 0.9 (which is then normalized).

CSA and SLA The confidence upper threshold is 0.8. The size of the ensemble is 20.

We use the implementation of these algorithms provided in (Nguyen et al., 2022a).

FixMatch The threshold is set to 0.95 as in (Sohn et al., 2020).

B.2 DATASETS

We summarize the different datasets we use in this paper in Table 1. The datasets vary in number of samples, number of features and domain. Recall, we use data splits with a proportion of $\mathcal{D}_{\text{lab}} : \mathcal{D}_{\text{unlab}}$ of 0.1:0.9.

Table 1: Summary of the datasets used

Name	n samples	n features	Domain
Adult Income (Asuncion & Newman, 2007)	30k	12	Finance
Agarius lepiota (Asuncion & Newman, 2007)	8k	22	Agriculture
Blog (Buza, 2013)	10k	280	Social media
Compas (Angwin et al., 2016)	5k	13	Criminal justice
Covid-19 (Baqui et al., 2020)	7k	29	Healthcare/Medicine
Credit (Taiwan) (Yeh & Lien, 2009)	30k	23	Finance
CUTRACT Prostate (PCUK, 2019)	2k	12	Healthcare/Medicine
Drug (Fehrman et al., 2017)	2k	27	Healthcare/Medicine
German-credit (Asuncion & Newman, 2007)	1k	24	Finance
Higgs (Baldi et al., 2014)	25k	23	Physics
MAGGIC (Pocock et al., 2013)	41k	29	Healthcare/Medicine
SEER Prostate (Duggan et al., 2016)	20k	12	Healthcare/Medicine

For computer vision experiments, we use CIFAR-10N (Wei et al., 2022a). The dataset can be accessed via its official release.[2]

B.3 IMPLEMENTATION DETAILS

The three key design decisions necessary for pseudo-labeling are:

1. Choice of backbone model (i.e. predictive classifier f)
2. Number of pseudo-labeling iterations — recall that it is an iterative process by repeatedly augmenting the labeled data with selected samples from the unlabeled data.
3. Compute requirements.

   We describe each in the context of each experiment. For further details on the experimental setup and process, see each relevant section of the main paper.

[2]https://github.com/UCSC-REAL/cifar-10-100n

First, let's detail some general implementation details pertinent to all experiments.

▪ Code: We will release code upon acceptance.

▪ `DIPS` thresholds: Recall that `DIPS` has two thresholds τ_{conf} and τ_{al}. We set τ_{conf} = 0.8, in order to select high confidence samples based on the mean of the learning dynamic. Note, τ_{al} is bounded between [0,0.25]. We adopt an adaptive threshold for τ_{al} based on the dataset, such that $\tau_{\text{al}} = 0.75 \cdot (\max(v_{al}(\mathcal{D}_{\text{train}})) - \min(v_{al}(\mathcal{D}_{\text{train}})))$

B.3.1 Synthetic example: Data characterization and unlabeled data improve test accuracy

1. Backbone model: We use an XGBoost, with 100 estimators similar to (Nguyen et al., 2022a). Note, XGBoost has been shown to often outperform deep learning methods on tabular data (Borisov et al., 2021; Shwartz-Ziv & Armon, 2022). That said, our framework is not restricted to XGBoost.
2. Iterations: we use $T = 5$ pseudo-labeling iterations, as in (Nguyen et al., 2022a).
3. Compute: CPU on a MacBook Pro with an Intel Core i5 and 16GB RAM.

B.3.2 Tabular data experiments: Sec 5.2, 5.3, 5.4

1. Backbone model: Some of the datasets have limited numbers, hence we have the backbone as XGBoost, with 100 estimators similar to (Nguyen et al., 2022a). Note, XGBoost has been shown to often outperform deep learning methods on tabular data (Borisov et al., 2021; Shwartz-Ziv & Armon, 2022). That said, our framework is not restricted to XGBoost.
2. Iterations: we use T=5, pseudo-labeling iterations, as in (Nguyen et al., 2022a).
3. Compute: CPU on a MacBook Pro with an Intel Core i5 and 16GB RAM.

For the tabular datasets we use splits with a proportion of $\mathcal{D}_{\text{lab}}$: $\mathcal{D}_{\text{unlab}}$ of 0.1:0.9.

B.3.3 Computer vision experiments: Sec 5.5

1. Backbone model: we use a WideResnet-18 (Zagoruyko & Komodakis, 2016) as in (Sohn et al., 2020).
2. Iterations: with Fixmatch we use T=1024k iterations as in (Sohn et al., 2020).
3. Data augmentations: Strong augmentation is done with RandAugment with random magnitude. Weak augmentation is done with a random horizontal flip.
4. Training hyperparameters: we use the same hyperparameters as in the original work (Sohn et al., 2020)
5. Compute: Nvidia V100 GPU, 6-Core Intel Xeon E5-2690 v4 with 16GB RAM.

For the experiments on CIFAR-10N, we use $\mathcal{D}_{\text{lab}}$ = 1000 samples.

C Additional experiments

C.1 Ablation study

Goal. We conduct an ablation study to characterize the importance of the different components of DIPS

Experiment. These components are:

- data characterization and selection for the **initialisation** of $\mathcal{D}^{(1)}$: DIPS defines $\mathcal{D}^{(1)} = \mathcal{D}^{(1)}_{\text{train}} = r(\mathcal{D}_{\text{lab}}, f^{(0)})$
- data characterization and selection at each **pseudo-labeling iteration**: DIPS defines $\mathcal{D}^{(i+1)}_{\text{train}} = r(\mathcal{D}^{(i+1)}, f^{(i)})$

Each of these two components can be ablated, resulting in four different possible combinations: **DIPS** (data selection both at initialization and during the pseudo-labeling iterations), **A1** (data selection only at initialization), **A2** (data selection only during the pseudo-labeling iterations), **A3** (no data selection). Note that **A3** corresponds to vanilla pseudo-labeling, and not selecting data amounts to using an identity selector.

We consider the same experimental setup as in Sec. 5.1, generating data in two quadrants with varying proportions of corrupted samples.

Results. The results are reported in Fig. 10. As we can see, each component in DIPS is important to improve pseudo-labeling: 1) the initialization of the labeled data $\mathcal{D}^{(1)}$ drives the pseudo-labeling process 2) data characterization of both labeled and pseudo-labeled samples is important.

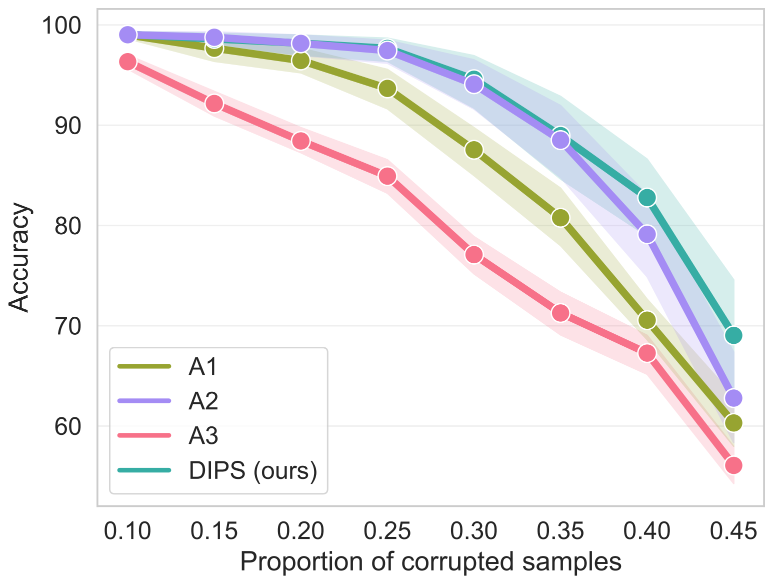


Figure 10: Ablation study: data characterization and selection is important both for the initialization of the labeled data and during the pseudo-labeling iterations, which forms the basis of DIPS.

Takeaway. It is important to characterize and select data both for the initialization of the labeled data and during the pseudo-labeling iterations.

C.2 Impact of the selector function r

Goal. We investigate variants of DIPS where we replace the selector function r with heuristics used in the LNL literature. The most commonly used is the small-loss criterion (Xia et al., 2021). Additionally, we assess Fluctuation (Wei et al., 2022b) and FINE (Kim et al., 2021) as alternative sample selection approaches.

Experiments. We use the vanilla PL algorithm as the semi-supervised backbone. We consider three different selectors in addition to the one presented in the main paper:

- Small-loss criterion (Xia et al., 2021): the quantity of interest is $\mu(x, y) = \frac{1}{E}\sum_{e=1}^{E} l(x, y, f_e)$ where l is a loss function and $f_1, ..., f_E$ are the classifiers at the different checkpoints. Intuitively, a sample with a high loss is more likely to present mislabeling issues, since it is harder to learn.
- Fluctuation criterion (Wei et al., 2022b): the quantity of interest, for two checkpoints $e_1 < e_2$ is $\beta(x, y, e_1, e_2) = 1([f_{e_1}(x)]_y > \frac{1}{2})1([f_{e_2}(x)]_y < \frac{1}{2})$, which is equal to one if the sample is correctly classified at the checkpoint e_1 and wrongly classified at e_2, 0 otherwise. Following (Wei et al., 2022b), we smooth this score with the confidence.
- FINE criterion (Kim et al., 2021): FINE creates a gram matrix of the representations in the noisy training dataset for each class. Then, FINE computes an alignment using the square of the inner product values between the representations and the first eigenvector of each gram matrix. A Gaussian mixture model is then fit on these alignment values to find clean and noisy instances.

The scores obtained by each approach are then used for sample selection, hence defining variants of the selector r.

Results. We report the results for 12 different datasets in Table 2. As we can see, the DIPS approach using learning dynamics outperforms the alternative LNL methods. This highlights the importance of a multi-dimensional data characterization, where both confidence and aleatoric uncertainty are taken into account to select samples. Moreover, the LNL methods typically operate under the assumption of a large number of labeled samples, highlighting that our DIPS approach tailored for the pseudo-labeling setting should indeed be preferred.

Table 2: DIPS outperforms heuristics used in the LNL setting by leveraging learning dynamics. Best performing method in **bold**, statistically equivalent performance <u>underlined</u>.

	DIPS (OURS)	Small-Loss (Xia et al., 2021)	Fluctuation (Wei et al., 2022b)	FINE (Kim et al., 2021)
adult	**82.66 ± 0.10**	79.52 ± 0.26	80.81 ± 0.20	24.22 ± 0.35
agaricus-lepiota	**65.03 ± 0.25**	64.45 ± 0.28	49.21 ± 1.48	35.96 ± 3.65
blog	**80.58 ± 0.10**	79.90 ± 0.28	80.05 ± 0.27	73.41 ± 1.66
credit	**81.39 ± 0.07**	78.46 ± 0.34	79.38 ± 0.29	64.58 ± 3.30
covid	<u>69.97 ± 0.30</u>	**70.09 ± 0.36**	69.03 ± 0.53	67.70 ± 0.84
compas	**65.34 ± 0.25**	62.76 ± 0.60	61.31 ± 0.64	60.20 ± 1.29
cutract	**68.60 ± 0.31**	66.33 ± 1.09	64.35 ± 1.49	61.98 ± 2.27
drug	**78.16 ± 0.26**	76.84 ± 0.61	75.66 ± 0.72	74.63 ± 1.98
German-credit	<u>69.40 ± 0.46</u>	**69.80 ± 1.00**	<u>69.60 ± 1.19</u>	<u>69.70 ± 1.19</u>
higgs	**81.99 ± 0.07**	81.08 ± 0.12	81.50 ± 0.09	73.82 ± 0.51
maggic	**67.60 ± 0.08**	65.57 ± 0.20	65.94 ± 0.20	62.28 ± 0.42
seer	**82.74 ± 0.08**	80.74 ± 0.26	82.28 ± 0.20	77.10 ± 0.76

Takeaway. A key component of DIPS is its sample selector based on learning dynamics, which outperforms methods designed for the LNL setting.

C.3 Insights into data selection

Goal. We wish to gain additional insight into the performance improvements provided by DIPS when added to the vanilla pseudo-labeling method. In particular, we examine the significant performance gain attained by DIPS for cross-country augmentation, shown in Section 5.4.

Experiment. The DIPS selector mechanism is the key differentiator as compared to the vanilla methods. Hence, we examine the samples selected samples from DIPS and vanilla. We then compare the samples to $\mathcal{D}_{\text{test}}$. Recall, we select samples from $\mathcal{D}_{\text{unlab}}$ which come from US patients, whereas $\mathcal{D}_{\text{lab}}$ and $\mathcal{D}_{\text{test}}$ are from the UK. We posit that "matching" the test distribution as closely as possible would lead to the best performance.

Results. We examine the most important features as determined by the XGBoost and compare their distributions. We find that the following 4 features in order are the most important: (1) Treatment: Primary hormone therapy, (2) PSA score, (3) Age, (4) Comorbidities. This is expected where the treatment and PSA blood scores are important predictors of prostate cancer mortality. We then compare these features in $\mathcal{D}_{\text{test}}$ vs the final $\mathcal{D}_{\text{train}}$ when using DIPS selection and using vanilla selection.

Fig. 11 shows that especially on the two most important features (1) Treatment: Primary hormone therapy and (2) PSA score; that DIPS's selection better matches $\mathcal{D}_{\text{test}}$ — which explains the improved performance.

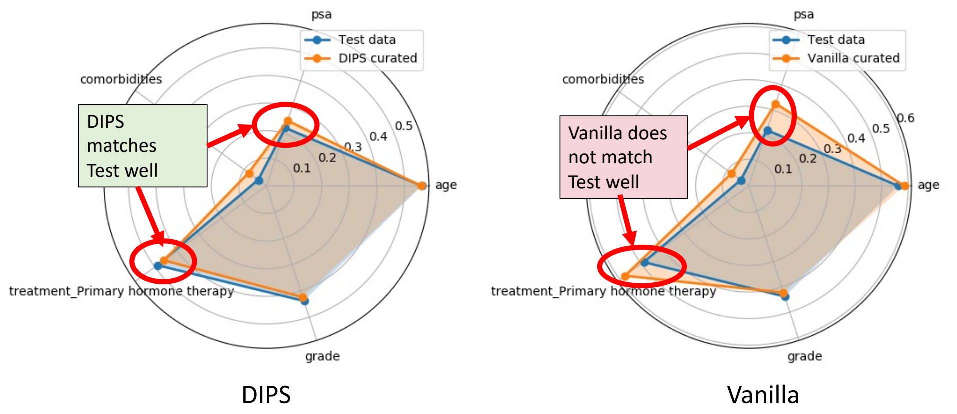


Figure 11: DIPS improves performance as its selection for $\mathcal{D}_{\text{train}}$ better matches $\mathcal{D}_{\text{test}}$ on the most important features, whereas vanilla selects samples which are different to $\mathcal{D}_{\text{test}}$

Quantitatively, we then compute the Jensen-Shannon (JS) divergence between the data selected by DIPS and vanilla as compared to $\mathcal{D}_{\text{test}}$. We find DIPS has a lower JSD of 0.0296 compared to vanilla of 0.0317, highlighting we better match the target domain's features through selection.

This behavior is reasonable, as samples that are very different will be filtered out by virtue of their learning dynamics being more uncertain.

For completeness, we also include a radar chart with vanilla PL but without any data selection in Fig. 12.

Takeaway. A significant source of DIPS's gain is that its selection mechanism selects samples that closely approximate the distribution of $\mathcal{D}_{\text{test}}$. In particular, we see this holds true for the features which are considered most important to the classifier — hence accounting for the improved downstream model performance observed when using DIPS.

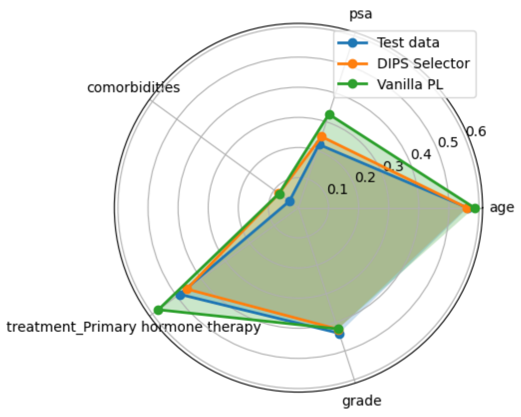


Figure 12: DIPS is closer to the test data than vanilla PL.

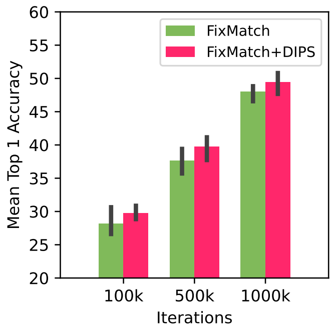
C.4 Additional experiments in computer vision

Goal. The goal of this experiment is to assess the benefit of `DIPS` in additional computer vision settings, namely:
(i) when increasing the number of classes to 100 (CIFAR-100N), with FixMatch
(ii) when the size of $\mathcal{D}_{\text{lab}}$ (n_{lab}) is small, with FixMatch and
(iii) when using a different pseudo-labeling algorithm — FreeMatch (Wang et al., 2023).

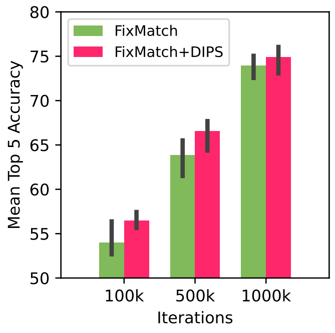
C.4.1 `DIPS` improves the performance of FixMatch on CIFAR-100N

Goal. The goal of this experiment is to further demonstrate `DIPS`'s utility in a setting with an increased number of classes.

Setup. We adopt the same setup as in Section 5.5, using the dataset CIFAR-100N (Wei et al., 2022a), which has 100 classes.



(a) Top-1 accuracy



(b) Top-5 accuracy

Figure 13: `DIPS` improves FixMatch on CIFAR-100N. We report both top-1 and top-5 accuracies.

Results. We show both top-1 and top-5 accuracies in Figure 13, for three different numbers of iterations and 3 different seeds, which highlight the performance gains obtained by using `DIPS` with FixMatch.

Takeaway. `DIPS` improves the performance of FixMatch on CIFAR-100N.

C.4.2 `DIPS` improves FixMatch for smaller n_{lab}

Setup. We consider $n_{\text{lab}} = 200$, and use the same setup as in Section 5.5, considering the dataset CIFAR-10N.

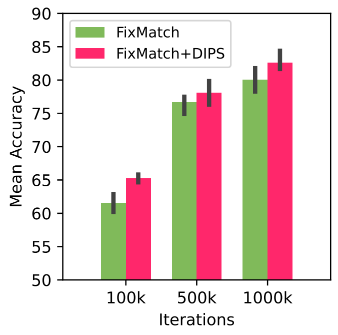


Figure 14: DIPS improves FixMatch with $n_{\text{lab}} = 200$ on CIFAR-10N

Results. We report the test accuracy in Figure 14 for three different numbers of iterations over 3 different seeds, highlighting the performance gains with DIPS.

Takeaway. DIPS improves performance of FixMatch for different sizes of D_{lab}

C.4.3 DIPS IMPROVES THE PERFORMANCE OF FREEMATCH

Goal. The goal of this experiment is to further demonstrate DIPS's utility in computer vision with an alternative pseudo-labeling method, namely FreeMatch (Wang et al., 2023).

Setup. We adopt the same setup as in Section 5.5, using the dataset CIFAR-10N.

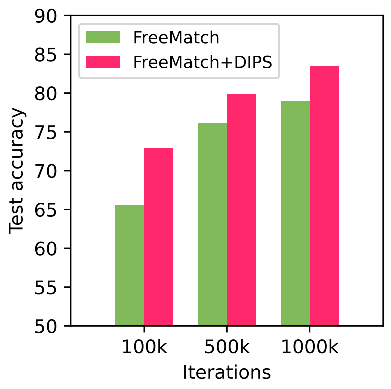


Figure 15: DIPS improves FreeMatch on CIFAR-10N

Results. We show the test accuracy in Figure 15, for three different numbers of iterations and highlight the performance gains by combining DIPS with FreeMatch to improve performance.

Takeaway. DIPS is versatile to other data modalities, improving the performance of FreeMatch on CIFAR-10N with the inclusion of DIPS.

C.4.4 ADDITIONAL DATASETS

We present in Figure 16 results for 4 additional image datasets : satellite images from Eurosat Helber et al. (2019) and medical images from TissueMNIST which form part of the USB benchmark Wang et al. (2022). Additionally, we include the related OrganAMNIST, and PathMNIST, which are part of the MedMNIST collection Yang et al. (2021). Given that these datasets are well-curated, we consider a proportion of 0.2 of symmetric label noise added to these datasets. As is shown, DIPS consistently improves the FixMatch baseline, demonstrating its generalizability.

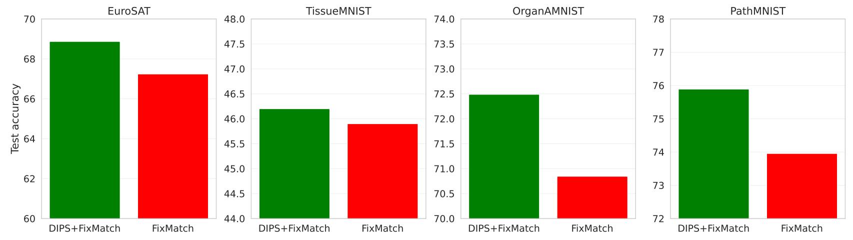


Figure 16: DIPS improves FixMatch on images datasets

C.5 Dependency between label noise level and amount of labeled data

We conduct a synthetic experiment following a similar setup as in Section 5.1 in our manuscript to investigate the dependency between label noise level and amount of labeled data. Note that the experiment is synthetic in order to be able to control the amount of label noise. We considered the same list of label noise proportions, ranging from 0. to 0.45. For each label noise proportion, we consider $n_{\text{lab}} \in \{50, 100, 1000\}$, and fix $n_{\text{unlab}} = 1000$. For each configuration we conduct the experiment 40 times. We report the results in Fig. 17. As we can see on the plots, PL+DIPS consistently outperforms the supervised baselines in almost all the configurations. When the amount of labeled data is low ($n_{lab} = 50$) and the proportion of corrupted samples is high (0.45), PL is on par with the supervised baseline. Hence pseudo-labeling is more difficult with a very low amount of labeled samples (and a high level of noise). We note, though, that DIPS consistently improves the PL baseline for reasonable amounts of label noise which we could expect in real-world settings (e.g. 0.1). The performance gap between DIPS and PL is remarkably noticeable for $n_{\text{lab}} = 1000$, i.e. when the amount of labeled samples is high.

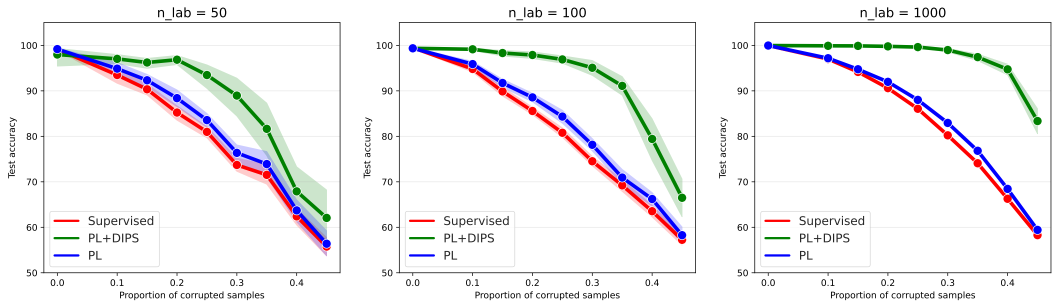


Figure 17: DIPS consistently improves upon the PL and supervised baselines, across the different label noise levels and amounts of labeled data.

C.6 Ablation on the teacher-student pseudo-labeling framework

In Section 3.1, we decided to describe the common teacher-student pseudo-labeling methodology adopted in the tabular setting. As a consequence, we used the implementation provided by Nguyen et al. (2022a), which grows the training set with pseudo-labels generated at the current iteration, thus keeping old pseudo-labels in the training dataset in subsequent iterations. In addition to adopting this practice, we investigated this choice experimentally, by comparing between two versions of confidence-based pseudo-labeling:

- Version 1): with a growing set of pseudo-labels (as followed by the implementation of Nguyen et al. (2022a) and our paper)
- Version 2): without keeping old pseudo-labels.

We evaluate these two methods in the synthetic setup described in Section 5.2, and report the test accuracy in Fig. 18. The red line corresponds to Version 1) (the implementation we used in the manuscript), while the green line corresponds to Version 2). As we can see, in this tabular setting, growing a training set by keeping the pseudo-labels generated at each iteration leads to the best results, motivating our adoption of this pseudo-labeling methodology used in the tabular setting.

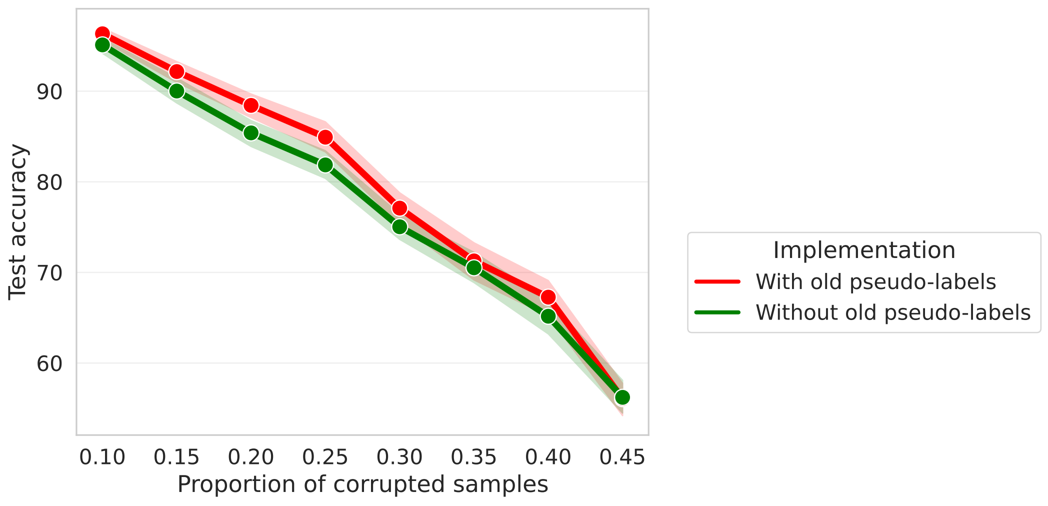


Figure 18: Growing the training set with the pseudo-labels generated at every iteration yields the best results

C.7 Ablation on the window of iterations for learning dynamics

We conduct an experiment to investigate the choice of the range of iterations used to compute the learning dynamics. We consider ignoring the first 25%/50%/75% iterations, and use the remaining iterations to compute the learning dynamics. Figure 19 shows the mean performance difference by using the truncated iteration windows versus using all the iterations, and averages the results over the 12 datasets used in Section 5.2. As we can see, it is better to use all the iterations window, as the initial iterations carry some informative signal about the hardness of samples. This motivates our choice of computing the learning dynamics over the whole optimization trajectory, a choice which we adopt for all of our experiments.

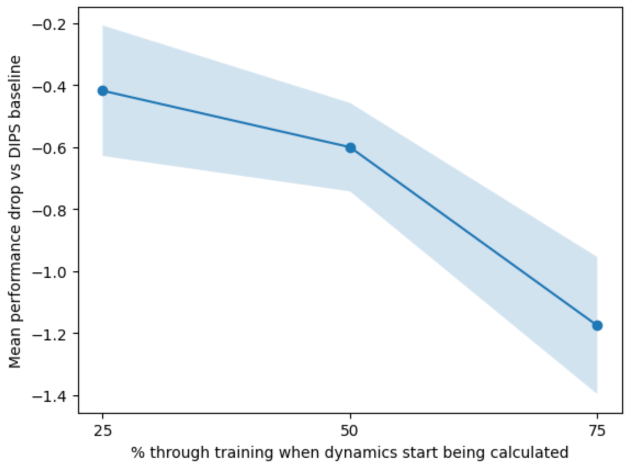


Figure 19: Computing the learning dynamics over the whole window of iterations yields the best results

C.8 Threshold choices

We conduct an experiment in the synthetic setup where we vary the thresholds used for both the confidence and the aleatoric uncertainty. In addition to our choice used in our manuscript (confidence threshold = 0.8, and adaptive threshold on the aleatoric uncertainty), we consider two baselines:

- confidence threshold = 0.9 and uncertainty threshold = 0.1 (aggressive filtering)
- confidence threshold = 0.5 and uncertainty threshold = 0.2 (permissive filtering)

We show the test accuracy for these baselines in Figure 20. As we can see, our configuration achieves a good trade-off between an aggressive filtering configuration (red line) and a permissive one (blue line), which is why we adopt it for the rest of the experiments. We empirically notice in Section 5.2 that it performs well on the 12 real-world datasets we used.

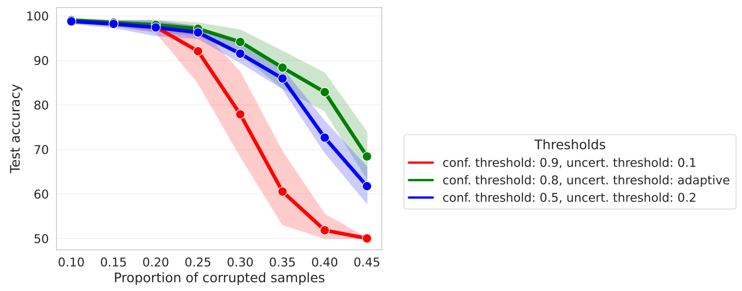


Figure 20: Our choice of thresholds performs better than an aggressive or a permissive filtering

C.9 Comparison to VIME

We compare DIPS with VIME (Yoon et al., 2020), by evaluating the two methods in the same setting as in Section 5.2. We report the results in Table 3. These results demonstrate that DIPS outperforms VIME across multiple real-world tabular datasets.

Table 3: DIPS outperforms VIME. Best performing method in **bold**, statistically equivalent performance underlined.

	DIPS (OURS)	VIME (Yoon et al., 2020)
adult	**82.66** $\pm$ **0.10**	67.69 $\pm$ 0.10
agaricus-lepiota	65.03 $\pm$ 0.25	**66.13** $\pm$ **0.01**
blog	**80.58** $\pm$ **0.10**	73.52 $\pm$ 0.01
credit	**81.39** $\pm$ **0.07**	66.91 $\pm$ 0.02
covid	**69.97** $\pm$ **0.30**	68.28 $\pm$ 0.03
compas	**65.34** $\pm$ **0.25**	63.41 $\pm$ 0.02
cutract	**68.60** $\pm$ **0.31**	60.36 $\pm$ 0.04
drug	**78.16** $\pm$ **0.26**	74.47 $\pm$ 0.03
German-credit	**69.40** $\pm$ **0.46**	62.65 $\pm$ 0.05
higgs	**81.99** $\pm$ **0.07**	71.34 $\pm$ 0.03
maggic	**67.60** $\pm$ **0.08**	64.98 $\pm$ 0.01
seer	**82.74** $\pm$ **0.08**	80.12 $\pm$ 0.01

C.10 Importance of aleatoric uncertainty

We conduct an ablation study where we remove the aleatoric uncertainty in DIPS and only keep a confidence-based selection (with threshold = 0.8). We term this confidence ablation to highlight if there is indeed value to the aleatoric uncertainty component of DIPS. We report results in Table 4, for the 12 tabular datasets used in Section 5.2, which shows the benefit of the two-dimensional selection criterion of DIPS. Of course, in some cases there might not be a large difference with respect to our confidence ablation — however we see that DIPS provides a statistically significant improvement in most of the datasets. Hence, since the computation is negligible, it is reasonable to use the 2-D approach given the benefit obtained on the noisier datasets.

Table 4: Aleatoric uncertainty is a key component of DIPS

	DIPS	Confidence Ablation
adult	**82.66** $\pm$ **0.10**	82.13 $\pm$ 0.16
agaricus-lepiota	**65.03** $\pm$ **0.25**	64.38 $\pm$ 0.23
blog	**80.58** $\pm$ **0.10**	80.22 $\pm$ 0.33
credit	**81.39** $\pm$ **0.07**	79.76 $\pm$ 0.15
covid	**69.97** $\pm$ **0.30**	69.28 $\pm$ 0.40
compas	**65.34** $\pm$ **0.25**	64.69 $\pm$ 0.25
cutract	**68.60** $\pm$ **0.31**	66.32 $\pm$ 0.12
drug	**78.16** $\pm$ **0.26**	75.37 $\pm$ 0.71
higgs	**81.99** $\pm$ **0.07**	81.42 $\pm$ 0.16
maggic	**67.60** $\pm$ **0.08**	66.26 $\pm$ 0.18
seer	**82.74** $\pm$ **0.08**	82.02 $\pm$ 0.15

D Broader impact

In this work, we delve into the essential yet often neglected aspect of labeled data quality in the application of pseudo-labeling, a semi-supervised learning technique. Our key insights stem from a data-centric approach that underscores the role of 'labeled data quality' - a facet typically overlooked due to the default assumption of labeled data being 'perfect'. In stark contrast to the traditional, algorithm-centric pseudo-labeling literature which largely focuses on refining pseudo-labeling methods, we accentuate the critical influence of the quality of labeled data on the effectiveness of pseudo-labeling.

By way of introducing the `DIPS` framework, our work emphasizes the value of characterization and selection of labeled data, consequently improving any pseudo-labeling method. Moreover, akin to traditional machine learning problems, focusing on labeled data quality in the context of pseudo-labeling promises to lessen risks, costs, and potentially detrimental consequences of algorithm deployment. This perspective opens up many avenues for applications in areas where labeled data is scarce or expensive to acquire, including but not limited to healthcare, social sciences, autonomous vehicles, wildlife conservation, and climate modeling scenarios. Our work underscores the need for a data-centric paradigm shift in the pseudo-labeling landscape.