A Details of Datasets and in TorchSSL

We provide the details of datasets of TorchSSL in Table 10.

Table 10: Details of CV datasets and #labels used in TorchSSL. #Label per class represents the number of chosen labeled data per class from the training data. The test data is kept unchanged except for ImageNet where we use the validation dataset as the test dataset.

Dataset	#Label per class	#Training data	#Test data	#Class
CIFAR-10	4 / 25 / 100	50,000	10,000	10
CIFAR-100	4 / 25 / 100	50,000	10,000	100
SVHN	4 / 25 /100	604,388	26,032	10
STL-10	4 / 25 /100	100,000	10,000	10
ImageNet	100	1,281,167	50,000	1,000

B Correlation between TorchSSL and USB

Here we show the correlation between the mean error rates on TorchSSL and USB CV tasks. We take the 14 algorithms considered in the main paper and show their mean performance on TorchSSL versus that on USB CV tasks in Figure 5. Despite the fact that the Pearson correlation coefficient is 0.87, the final rank SSL algorithms is not consistent, which shows different adaptability of different methods when using pre-trained ViTs. For example, FlexMatch shows the best mean performance on USB while AdaMatch has the best mean performance on TorchSSL. Please refer to Table 8 for more detailed rankings on CV, NLP, and Audio.

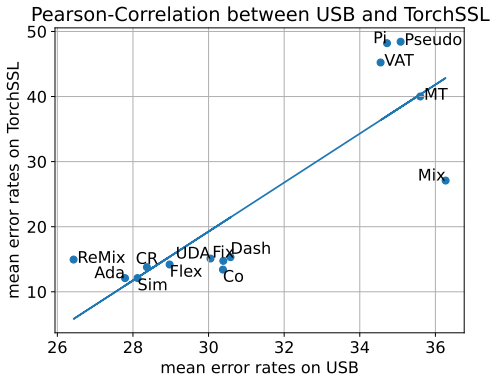

Figure 5: Correlation between TorchSSL and USB.

C Performance Results on ImageNet

Although we have excluded ImageNet from USB, we provide an evaluation on ImageNet of MAE pre-trained ViT-B, using UDA [29], FixMatch [20], FlexMatch [21], CoMatch [60], and SimMatch [47]. We train these algorithms using 10 labels per class and 100 labels per-class, i.e., a total of 10,000 labels and 100,000 labels respectively, corresponding to roughly 1% and 10% of the total labeled data in ImageNet. For learning rate and weight decay, we follow the fine-tuning protocol in MAE [33], where we use AdamW with a learning rate of 1e-3 and weight decay of 0.05. We use 16 A100 to train each algorithm and set the batch size to 256 for both labeled and unlabeled data. Other algorithmic hyper-parameters stay the same as their original implementations.

We present the results on ImageNet in Table 11. UDA and Fixmatch are near the bottom, similar to USB. SimMatch is still marked as one of the tops. Surprisingly, CoMatch does so well on ImageNet when it ranked only 9th on the USB benchmark. Also, while FlexMatch is the best on USB, it's pretty firmly behind CoMatch and SimMatch on ImageNet.

Table 11: ImageNet **accuracy** results. We use MAE pre-trained ViT-B.

Method	1w Labels	10w Labels	Rank
UDA	38.62	62.37	5
FixMatch	37.93	62.88	4
FlexMatch	39.13	63.09	3
CoMatch	44.32	65.80	2
SimMatch	46.48	67.61	1

Table 12: Swin-Transformer results on EuroSAT and Semi-AVES.

Dataset	EuroSAT		Semi-Aves
# Label	20	40	5,959
Supervised	$44.32_{\pm 1.10}$	$34.40_{\pm 1.44}$	$38.76_{\pm 0.21}$
Fully-Supervised	$1.86_{\pm 0.10}$		-
Π-Model	$42.49_{\pm 3.21}$	$30.54_{\pm 1.37}$	$38.74_{\pm 0.60}$
Pseudo-Labeling	$42.49_{\pm 3.21}$	$30.54_{\pm 1.37}$	$38.74_{\pm 0.60}$
Mean Teacher	$35.85_{\pm 1.95}$	$19.62_{\pm 3.28}$	$33.37_{\pm 0.06}$
VAT	$40.63_{\pm 2.68}$	$29.94_{\pm 1.87}$	$35.84_{\pm 0.36}$
UDA	$18.15_{\pm 5.70}$	$12.09_{\pm 1.26}$	$29.28_{\pm 0.20}$
FixMatch	$17.19_{\pm 3.46}$	$12.57_{\pm 1.28}$	$28.88_{\pm 0.22}$
Dash	$18.04_{\pm 1.21}$	$12.98_{\pm 1.27}$	$28.69_{\pm 0.39}$
CoMatch	$13.65_{\pm 1.42}$	$10.17_{\pm 0.68}$	$37.71_{\pm 0.31}$
CRMatch	$30.28_{\pm 1.64}$	$22.39_{\pm 1.41}$	$29.22_{\pm 0.21}$
FlexMatch	$10.46_{\pm 1.20}$	$9.06_{\pm 1.80}$	$30.19_{\pm 0.51}$
SimMatch	$11.19_{\pm 1.01}$	$10.65_{\pm 1.64}$	$28.55_{\pm 0.13}$

D Results with Different Pre-trained Backbones

In this section, we verify USB with different pre-trained backbones. Different pre-trained backbones do affect the performance of SSL algorithms, which makes it important to report results with multiple backbones. We will continuously update results with different backbones at `https://github.com/microsoft/Semi-supervised-learning`. Here we report several results in Table 12, Table 13, and Table 14. Across the tasks, there is a pretty clear distinction between the performance of algorithms in the first half of the ranking list and the second half of the ranking list. While switching out backbones does not change the membership of these two halves, it does seem like the relative orderings within the top half can indeed vary a bit.

To compare different backbones on CV tasks, we fine-tune pre-trained public Swin-Transformer [123] with USB. We keep all hyper-parameters the same as in Table 15, and mainly evaluate on EuroSAT (32) and Semi-Aves (224). For EuroSAT, we change the input image size of the pre-trained Swin-S from 224 to 32, and the window size from 7 to 4 to accommodate the adapted input image size. For Semi-Aves, we adopt the original Swin-S. From the results in Table 12, one can observe, that on EuroSAT (32), as we adopt 224 pre-trained Swin-S and change its input and window size, the results are inferior to ViT-32 reported in the paper, whereas on Semi-Aves (224), the results are better than ViT-S. An interesting finding is that CoMatch performs relatively better with Swin-S while CrMatch performs worse. This also shows the importance of constantly updating the backbone in the future development of USB.

For NLP tasks, we additionally experiment with RoBERTa [31]. We train RoBERTa using the same hyper-parameters reported in Table 16. RoBerta generally performs better than Bert as expected. The performance difference is both very close when using RoBerta or Bert.

Due to the fact that the audio tasks setting in the current version of USB being built upon raw waveforms, there are not many pre-trained models available to use. We report the results of HuBert [32] and Wave2Vecv2.0 [71] for audio tasks to compare different backbones. The difference between these two backbones selected mainly lies in pre-training data. Wave2Vecv2.0 is pre-trained using raw human voice data and HuBert is an improved model with a discrete clustering target. Thus we can observe from the results, that on human voice tasks Superb-KS, Wave2Vecv2.0 has better performance, whereas, on other tasks, HuBert is more robust and outperforms Wave2Vecv2.0.

Table 13: RoBERTa results on Yelp.

Dataset	Yelp	
# Labels	250	1000
Supervised	42.56±1.15	39.00±0.16
Fully-Supervised	29.15±0.12	
Pseudo-Label	48.26±0.02	40.56±0.16
MeanTeacher	49.41±0.03	44.36±1.04
Π-Model	49.16±2.04	42.93±0.88
VAT	43.04±0.02	39.24±0.06
AdaMatch	38.24±0.02	35.64±0.06
UDA	40.13±0.15	38.98±0.03
FixMatch	39.82±0.95	37.42±0.30
FlexMatch	39.11±0.02	36.84±0.01
Dash	39.86±1.01	36.23±0.21
CRMatch	40.08±1.28	35.85±0.38
CoMatch	39.95±0.86	36.89±0.22
SimMatch	38.76±0.68	36.39±0.34

Table 14: HuBert results on keyword Spotting and Wave2Vec2.0 results on FSDnoisy.

Dataset	keyword Spotting		FSDnoisy
# Label	50	400	1,772
Supervised	8.95±1.62	6.31±0.46	33.54±1.65
Fully-Supervised	2.41±0.15		-
Π-Model	87.86±2.88	72.89±3.23	35.97±0.84
Pseudo-Labeling	25.59±2.88	13.02±2.47	35.23±0.78
Mean Teacher	89.79±0.30	90.01±0.02	40.13±1.70
VAT	2.27±0.07	2.43±0.02	34.21±0.31
UDA	11.76±0.06	2.23±0.16	33.09±1.03
FixMatch	11.63±0.24	8.93±2.04	33.09±0.64
Dash	11.88±0.15	8.25±4.22	33.02±1.39
CoMatch	15.96±1.02	10.34±1.52	30.24±0.55
CRMatch	5.85±1.19	3.66±0.33	30.48±0.65
FlexMatch	10.22±1.10	5.10±3.70	32.66±4.09
SimMatch	9.43±0.63	5.47±2.72	29.57±0.52

E Details of Datasets in USB

E.1 CV Tasks

CIFAR-100 The CIFAR-100 [39] dataset is a natural image (32×32 pixels) recognition dataset consisting 100 classes. There are 500 training samples and 100 test samples per class.

STL-10 The STL-10 [40] dataset is a natural color image (96×96 pixels) recognition dataset consisting 10 classes. Particularly, each class has 500 training samples and 800 test samples. Apart from the labeled samples, STL-10 also provides 100,000 unlabeled samples. Note that the unlabeled samples contain other classes in addition to the ones in the labeled data.

EuroSat EuroSAT [43, 44] dataset is based on Sentinel-2 satellite images covering 13 spectral bands and consisting of 10 classes with 27,000 labeled and geo-referenced samples. Following [124], we use the dataset with the optical R, G, B frequency bands, thus each image is of size $64 \times 64 \times 3$. We take the first 60% images from each class as training set; the next 20% as val set, and the last 20% as test set.

TissueMNIST TissueMNIST [41, 42] is a medical dataset of human kidney cortex cells, segmented from 3 reference tissue specimens and organized into 8 categories. The total 236,386 training samples are split with a ratio of $7:1:2$ into training (165,466 images), validation (23,640 images) and test set (47,280 images). Each gray-scale image is 28×28 pixels.

Semi-Aves Semi-Aves [45] is a dataset of Aves (birds) classification, where 5,959 images of 200 bird species are labeled and 26,640 images are unlabeled. As class distribution mismatch hurts the performance [85], we do not use out-of-class unlabeled data. This dataset is challenging as it is naturally imbalanced. The validation and test set contain 10 and 20 images respectively for each of the 200 categories in the labeled set.

E.2 NLP Tasks

IMDB The IMDB [49] dataset is a binary sentiment classification dataset. There are 25,000 reviews for training and 25,000 for test. IMDB is class balanced which means the positive and negative reviews have the same number both for training and test. For USB, we draw 12,500 samples and 1,000 samples per class from training samples to form the training dataset and validation dataset respectively. The test dataset is unchanged.

Amazon Review The Amazon Review [52] dataset is a sentiment classification dataset. There are 5 classes (scores). Each class (score) contains 600,000 training samples and 130,000 test samples. For USB, we draw 50,000 samples and 5,000 samples per class from training samples to form the training dataset and validation dataset respectively. The test dataset is unchanged.

Yelp Review The Yelp Review [53] sentiment classification dataset has 5 classes (scores). Each class (score) contains 130,000 training samples and 10,000 test samples. For USB, we draw 50,000 samples and 5,000 samples per class from training samples to form the training dataset and validation dataset respectively. The test dataset is unchanged.

AG News The AG News [50] dataset is a news topic classification dataset containing 4 classes. Each class contains 30,000 training samples and 1,900 test samples. For USB, we draw 25,000 samples and 2,500 samples per class from training samples to form the training dataset and validation dataset respectively. The test dataset is unchanged.

Yahoo! Answer The Yahoo! Answer [51] topic classification dataset has 10 categories. Each class contains 140,000 training samples and 6,000 test samples. For USB, we draw 50,000 samples and 5,000 samples per class from training samples to form the training dataset and validation dataset respectively. The test dataset is unchanged.

E.3 Audio Tasks

GTZAN The GTZAN dataset is collected for music genre classification of 10 classes and 100 audio recordings for each class. The maximum length of the recordings is 30 seconds and the original sampling rate is 22,100 Hz. We split 7,000 samples for training, 1,500 for validation, and 1,500 for testing. All recordings are re-sampled at 16,000 Hz.

UrbanSound8k The UrbanSound8k dataset [54] contains 8,732 labeled sound events of urban sounds of 10 classes, with the maximum length of 4 seconds. The original sampling rate of the audio recordings is 44,100 and we re-sample it to 16,000. It is originally divided into 10 folds, where we use the first 8 folds of 7,079 samples as training set, and the last two folds as validation set of size 816 and testing set of size 837 respectively.

FSDNoisy18k The FSDNoisy18 dataset [56] is a sound event classification dataset across 20 classes. It consists of a small amount of manually labeled data - 1,772 and a large amount of noisy data - 15,813 which is treated as unlabeled data in our paper. The original sample rate is 44,100 Hz, and the length of the recordings lies between 3 seconds and 30 seconds. We use the testing set provided for evaluation, which contains 947 samples.

Keyword Spotting (Superb-KS) The Keyword spotting dataset is one of the tasks in Superb [57] for classifying the keywords. It contains speech utterances of a maximum length of 1 second and the sampling rate of 16,000. The training, validation, and testing set contain 18,538; 2,577; 2,567 recordings, respectively. For pre-processing, we remove the silence and unknown labels from the dataset.

ESC-50 The ESC-50 [55] is a dataset containing 2,000 environmental audio recordings for 50 sound classes. The maximum length of the recordings is 5 seconds and the original sampling rate is 44,100. We split 1,200 samples as training data, 400 as validation data, and 400 as testing data. We also re-sample the audio recordings to 16,000 Hz during pre-processing.

F Details of Implemented SSL algorithms in USB

Π **model [35]** is a simple SSL algorithm that forces the output probability of perturbed versions of unlabeled data be the same. Π model uses Mean Squared Error (MSE) for optimization.

Pseudo Labeling [59] turns the output probability of unlabeled data into the 'one-hot' hard one and makes the same unlabeled data to learn the pseudo 'one-hot' label. Unlike Π model, Pseudo Labeling uses CE for optimization.

Mean Teacher [36] takes the exponential moving average (EMA) of the neural model as the teacher model. With Mean Teacher, the neural model forces itself to output a similar probability to the EMA teacher. Though the later SSL algorithms will not always choose the EMA model as the teacher, they often use the EMA model for validation/test cause it decreases the risk of neural models falling into the local optima.

VAT [37] enhances the robustness of the conditional predicted label distribution around each unlabeled data against an adversarial perturbation. In other words, VAT forces the neural model to give similar predictions on unlabeled data even facing a strong adversarial perturbation.

MixMatch [28] first introduces Mixup [125] into SSL by taking the input as the mixture of labeled and unlabeled data and the output as the mixture of labels and model predictions on unlabeled data. Note that MixMatch also utilizes MSE as the unsupervised loss.

ReMixMatch [23] can be seen as the upgraded version of MixMatch. ReMixMatch improves MixMatch by (1) proposing stronger augmentation (i.e., Control Theory Augmentation (CTAugment) [23]) for unlabeled data; (2) using Augmentation Anchoring to force the model to output similar predictions to weakly augmented unlabeled data when fed strongly augmented data; (3) utilizing Distribution Alignment to encourage the marginal distribution of predictions on unlabeled data to be similar to the marginal distribution of labeled data.

UDA [29] also introduces strong augmentation (i.e., RandAugment [126]) for unlabeled data. The core idea of UDA is similar to Augmentation Anchoring [23], which forces the predictions of neural models on the strongly-augmented unlabeled data to be close to those of weakly-augmented unlabeled data. Instead of turning predictions into hard 'one-hot' pseudo-labels, UDA sharpens the prediction on unlabeled data. Thresholding technique is used to mask out unconfident unlabeled samples that are considered noise here.

FixMatch [20] is the upgraded version of Pseudo Labeling. FixMatch turns the predictions on weakly-augmented unlabeled data into hard 'one-hot' pseudo-labels and then further uses them as the learning signal of strongly-augmented unlabeled data. FixMatch finds that using a high threshold (e.g., 0.95) to filter noisy unlabeled predictions and take the rest as the pseudo-label can achieve very good performance.

Dash [24] improves the FixMatch by using a gradually increased threshold instead of a fixed threshold, which allows more unlabeled data to participate in the training at the early stage. Moreover, Dash theoretically establishes the convergence rate from the view of non-convex optimization.

CoMatch [60] firstly introduces contrastive learning into SSL. Except for consistency regularizing on the class probabilities, it is also exploited on graph-based feature representations, which impose smooth constraints on pseudo-labels generated.

CRMatch [61] proposed an improved consistency regularization framework which impose consistency and equivariance on the classification probability and the feature level.

FlexMatch [21] firstly introduces the class-specific thresholds into SSL by considering the different learning difficulties of different classes. Specifically, the hard-to-learn classes should have a low threshold to speed up convergence while the easy-to-learn classes should have a high threshold to avoid confirmation bias.

AdaMatch [62] is proposed mainly for domain adaption, but can also adapted to SSL. It is characterized by Relative Threshold and Distribution Alignment, where the relative threshold is adaptively estimated from EMA of the confidence on labeled data.

SimMatch [47] extends CoMatch [60] by considering semantic-level and instance-level consistency regularization. Similar similarity relationship of different augmented versions on the same data with respect to other instances is encouraged during training. In addition, a memory buffer consisting of predictions on labeled data is adopted to connect the two-level regularization.

G Experiment Details in USB

G.1 Setup for CV Tasks in USB

Table 15: Hyper-parameters of CV tasks in USB.

Dataset	CIFAR-100	STL-10	Euro-SAT	TissueMNIST	Semi-Aves
Image Size	32	96	32	32	224
Model	ViT-S-P4-32	ViT-B-P16-96	ViT-S-P4-32	ViT-T-P4-32	ViT-S-P16-224
Weight Decay	5e-4				
Labeled Batch size	16				
Unlabeled Batch size	16				
Learning Rate	5e-4	1e-4	5e-5	5e-5	1e-3
Layer Decay Rate	0.5	0.95	1.0	0.95	0.65
Scheduler	$\eta = \eta_0 \cos(\frac{7\pi k}{16K})$				
Model EMA Momentum	0.0				
Prediction EMA Momentum	0.999				
Weak Augmentation	Random Crop, Random Horizontal Flip				
Strong Augmentation	RandAugment [126]				

For CV tasks in USB, we use ViT models [34]. We find that directly using released ViT models leads to overfitting and one needs to fix the image resolution as the pre-trained resolution, as demonstrated in Paragraph 5.4. Instead, we pre-train our own ViT models on ImageNet-1K [8]. To match the number of parameters as the CNN models used in the classic setting, we use ViT-Tiny and ViT-Small with a patch size of 2 and image size of 32 for TissueMNIST, CIFAR-100 and EuraSAT, respectively; ViT-Small with a patch size of 16 and image size of 224 for Semi-Aves. For better transfer performance, we adopt an MLP before the final classifier during pre-training, as in [127]. For supervised pre-training on ImageNet-1K, we use Lamb optimizer with a learning rate of 0.05, and a weight decay of 0.03 for ViT-Tiny and a weight decay of 0.05 for ViT-Small. We adopt a large batch size of 4096 and train the networks for 300 epochs, with a linear learning rate warmup for the first 20 epochs. After the warmup, cosine scheduler is utilized. For augmentation, we use RandAugment [126], along with Mixup [125] and CutMix [128]. We also use label smoothing of 0.1 during pre-training. Since STL10 is a subset of ImageNet, we adopt unsupervised pre-training MAE [33] of ViT-Small with image size of 96 to avoid cheating.

For USB CV tasks, we adopt layer-wise learning rate decay as in [123]. We tune the learning rate and layer decay rate on different datasets using FixMatch, and use the best configuration to train all SSL algorithms [6]. The cosine annealing scheduler is similar to the classic setting but with total steps of $204,800$ and a warm-up of $5,120$ steps. The labeled and unlabeled batch size is both set to 16. Other algorithm-related hyper-parameters stay the same as in the original papers.

G.2 Setup for NLP Tasks in USB

We use pre-trained BERT-Base [30] for all NLP tasks in USB. We set the batch size of labeled data and unlabeled data to 4 for reducing the training time and GPU memory requirement. To fine-tune the BERT-Base under USB, we adopt AdamW optimizer with weight decay of $1\mathrm{e}{-4}$. Similarly, we conduct a grid search of the learning rate and layer decay on different datasets using FixMatch and pick the best configuration to fine-tune other SSL algorithms. We utilize the same cosine learning rate

[6] We present the full tuning results in: `https://github.com/microsoft/Semi-supervised-learning`.

scheduler as in the classic setting with the total training steps of $102,400$ and a warm-up of $5,120$ steps. We use the fine-tuned model without parameter momentum to conduct evaluations. For all datasets, we cut the long sentence to satisfy the input length requirement of BERT-Base. For data augmentation, we adopt back-translation as the strong augmentation [29, 25]. Specifically, we use De-En and Ru-En translation with WMT19.

Table 16: Hyper-parameters of NLP tasks in USB.

Dataset	AG News	Yahoo! Answer	IMDb	Amazom-5	Yelp-5
Max Length	512				
Model	Bert-Base				
Weight Decay	1e-4				
Labeled Batch size	4				
Unlabeled Batch size	4				
Learning Rate	5e-5	1e-4	5e-5	1e-5	5e-5
Layer Decay Rate	0.65	0.65	0.75	0.75	0.75
Scheduler	$\eta = \eta_0 \cos(\frac{7\pi k}{16K})$				
Model EMA Momentum	0.0				
Prediction EMA Momentum	0.999				
Weak Augmentation	None				
Strong Augmentation	Back-Translation [29]				

G.3 Setup for Audio Tasks in USB

For Audio tasks, we adopt Wav2Vec 2.0 [71] and HuBert [32] as the pre-trained model. The batch size of labeled data and unlabeled data is set to 8. We keep the sampling rate of audios as $16,000$. We adopt AdamW optimizer with a weight decay of $5\mathrm{e}{-4}$, and search the learning rate and layer decay as before. Other hyper-parameter settings are the same as NLP tasks. Mimicking RandAugment, for strong augmentation in audio tasks, we random sample 2 augmentations from the augmentation pool and random set the augmentation magnitude during training.

Table 17: Hyper-parameters of Audio tasks in USB.

Dataset	GTZAN	Keyword Spotting	UrbanSound8k	FSDNoisy	ESC-50
Sampling Rate	16,000				
Max Length	3.0	1.0	4.0	5.0	5.0
Model	Wav2Vecv2-Base		HuBERT-Base		
Weight Decay	5e-4				
Labeled Batch size	8				
Unlabeled Batch size	8				
Learning Rate	2e-5	5e-5	5e-5	5e-4	1e-4
Layer Decay Rate	1.0	0.75	0.75	0.75	0.85
Scheduler	$\eta = \eta_0 \cos(\frac{7\pi k}{16K})$				
Model EMA Momentum	0.0				
Prediction EMA Momentum	0.999				
Weak Augmentation	Random Sub-sample				
Strong Augmentation	Random Sub-sample, Random Gain, Random Pitch, Random Speed				