SequenceMatch
Revisiting the design of weak-strong augmentations for Semi-supervised learning

Khanh-Binh Nguyen
Sungkyunkwan University
South Korea
n.k.binh00@gmail.com

Abstract

Semi-supervised learning (SSL) has become popular in recent years because it allows the training of a model using a large amount of unlabeled data. However, one issue that many SSL methods face is the confirmation bias, which occurs when the model is overfitted to the small labeled training dataset and produces overconfident, incorrect predictions. To address this issue, we propose SequenceMatch, an efficient SSL method that utilizes multiple data augmentations. The key element of SequenceMatch is the inclusion of a medium augmentation for unlabeled data. By taking advantage of different augmentations and the consistency constraints between each pair of augmented examples, SequenceMatch helps reduce the divergence between the prediction distribution of the model for weakly and strongly augmented examples. In addition, SequenceMatch defines two different consistency constraints for high and low-confidence predictions. As a result, SequenceMatch is more data-efficient than ReMixMatch, and more time-efficient than both ReMixMatch ($\times 4$) and CoMatch ($\times 2$) while having higher accuracy. Despite its simplicity, SequenceMatch consistently outperforms prior methods on standard benchmarks, such as CIFAR-10/100, SVHN, and STL-10. It also surpasses prior state-of-the-art methods by a large margin on large-scale datasets such as ImageNet, with a 38.46% error rate. Code is available at https://github.com/beandkay/SequenceMatch.

1. Introduction

Deep Neural Networks (DNNs) have made significant strides in recent years, achieving an extraordinary performance on many tasks such as image recognition [12], speech recognition [1], and natural language processing [37]. The state-of-the-art performance of DNNs is achieved through supervised learning, which requires labeled data. The empirical observation shows that training DNNs on larger labeled datasets produces a better performance [13, 14, 20, 29, 30]. However, the labeled data is limited in quantity and significantly costly due to the hand-labeling process which must be done by experts.

An impressive approach for training models with a large amount of unlabeled data is semi-supervised learning (SSL). In recent years, SSL has received much attention due to its advantages in leveraging a large amount of unlabeled data. Since the unlabeled data can be obtained easily without the need for human labor, using SSL results in comparable performance to the supervised learning methods but with a lower cost. This success has led to the development of many SSL methods [3, 4, 16, 17, 40, 43].

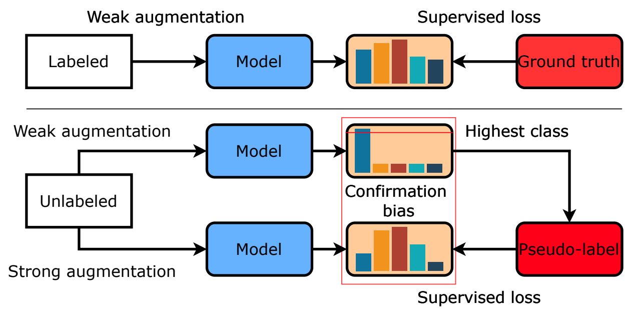


Figure 1. Example scheme where the prediction distribution of weakly and strongly augmented examples have high KL divergence. This high divergence happens when the model suffers from the confirmation bias issue.

There are two popular SSL methods which are pseudo-labeling [17] (also called self-training [32, 43]) and consistency regularization [2, 16, 35]. While the pseudo-labeling-based approaches use model predictions as labels to train the unlabeled data, the consistency-regularization-based approaches use loss functions such as mean squared error (MSE) or Kullback-Leibler divergence (KL divergence) to minimize the difference between the prediction distribution of different augmented inputs. However, they are still en-

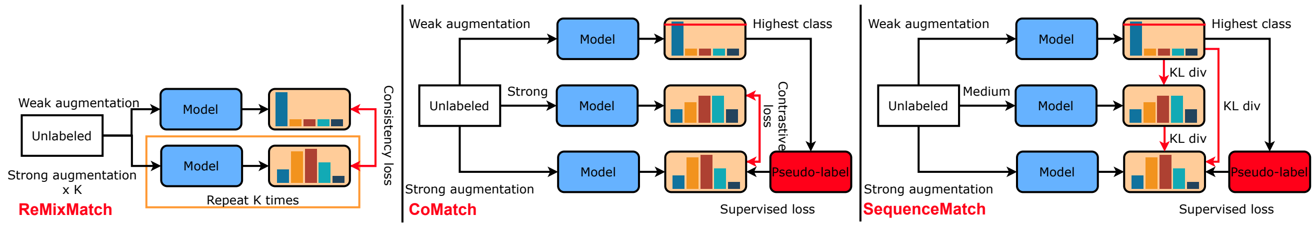


Figure 2. Differences between SequenceMatch versus ReMixMatch and CoMatch methods for multi augmentations.

countering the confirmation bias issue because of the small labeled training dataset. Hence, during training, when a confirmation bias issue occurs, the performance stops improving and could become worse.

Based on the finding from [5] that we could utilize KL divergence with multiple augmentations to increase model invariance and generalization, we propose a simple SSL pipeline, SequenceMatch. The idea of using multiple data augmentations for SSL is not new since it has been introduced by [3] and [18]. ReMixMatch [3] uses a technique called Augmentation Anchoring (AA). AA anchors a weak augmentation then makes $\mathbf{K}$ strong augmentations and encourages each output to be close to the anchoring prediction. Similarly, CoMatch [18] generates two strongly augmented versions for each unlabeled sample to construct the embedding graph. However, we argue that using multiple strong augmentations can result in disparate predictions, and thus may not be a meaningful target. Particularly, ReMixMatch [3] found that using stronger augmentations in MixMatch resulted in high divergence, and the training would not converge if we replace the weak with strong augmentation, resulting in very poor performance. SequenceMatch also uses multiple data augmentations but in a different manner. Specifically, we introduce a medium augmentation, then minimize the KL divergence between prediction distributions for each pair of inputs, thus minimizing the discrepancy between the representation of weak and strong augmented predictions. Therefore, by minimizing these divergences, we assume that the learned representation of the strong augmentation would align with the one from the weak augmentation by using medium augmentation as an anchor. The medium augmentation also works like a Teacher Assistant (TA) to distill the knowledge, similar to a TA in [23]. As a result, SequenceMatch encourages the similarity of the network outputs to produce more reliable pseudo-labels for unlabeled data during training, reduces overconfident pseudo-labels, and optimizes data utilization for the unlabeled dataset.

The benefit of SequenceMatch is found in all datasets. For instance, on the STL-10 dataset, SequenceMatch achieves 15.45%, and 5.56% error rates when the label amount is 40, and 1000, respectively. Moreover, SequenceMatch shows its superiority on imbalanced datasets such as SVHN and ImageNet. On the SVHN dataset, SequenceMatch achieves 1.96%, 1.89%, and 1.79% error rate when the label amount is 40, 250, and 1000, respectively. For ImageNet, SequenceMatch achieves 38.46% error rate, surpassing FlexMatch of 41.85%, FixMatch of 43.66%, CoMatch of 42.17%, and FreeMatch of 40.57%. In addition, SequenceMatch achieves high performance even though it does not introduce as many augmentations as ReMixMatch and does not need to store the embedded graph like CoMatch. To sum up, this paper makes the following contributions:

- We propose SequenceMatch, a SSL training pipeline that helps reduce the divergence between the prediction distributions of different augmented versions of the same input. Therefore, SequenceMatch helps reduce the overconfident predictions and the distribution discrepancy between weakly and strongly augmented predictions.
- SequenceMatch leverages the whole unlabeled dataset, including high-confidence and low-confidence predictions, thus optimizing the data utilization.
- We verify our hypothesis that reducing the confirmation bias issue of the trained model and reducing the divergence between the prediction distributions would yield better results. Hence, SequenceMatch significantly achieves state-of-the-art results on many datasets with different numbers of labels.

2. Analysis of high-confidence and low-confidence pseudo-label

Following [26], to investigate the role of predictions with low confidence during training, we separately train FixMatch using “hard” and “soft” pseudo-labels. The “hard” pseudo-label training refers to the traditional FixMatch method that utilizes high-confidence predictions. On the other hand, “soft” pseudo-label training involves training the model solely on low-confidence predictions. In particular, we select low-confidence predictions from weakly-augmented examples as the pseudo-label, instead of the usual high-confidence predictions. These predictions are then sharpened using a temperature of $\mathbf{T} = \mathbf{0.5}$. Following this, we calculate the KL divergence with the predictions from strongly-augmented examples.

Table 1. Comparison of error rates between high-confidence and low-confidence predictions using FixMatch on CIFAR-10 with splits of 40, 250, and 1000 labels

DATASET	HIGH-CONFIDENCE	LOW-CONFIDENCE
CIFAR-10-40	7.47	28.88
CIFAR-10-250	4.86	8.07
CIFAR-10-4000	4.21	8.04

The experimental results presented in Table 1 indicate that training the model solely with low-confidence predictions can yield performance that is competitive with that achieved using high-confidence predictions on the CIFAR-10 dataset.

This suggests that the traditional method of setting a high threshold and discarding a significant portion of unlabeled data during training is not the most efficient approach, as it fails to fully utilize the unlabeled data.

Therefore, in this study, we propose a different approach. Instead of relying solely on high-confidence predictions, we combine the advantages of both high-confidence and low-confidence predictions.

3. Background

We give a brief introduction to unsupervised data augmentation (UDA) [42] and FixMatch [38], which are mostly related to our work. Let B be the batch size of labeled data, μ be the ratio of unlabeled data to labeled data, and p_m represent the output probability of the model. A_w and A_s are weakly and strongly augmentation functions, respectively. The unsupervised loss term in UDA is formulated as:

$$\frac{1}{\mu B}\sum_{b=1}^{\mu B}\mathbb{1}\left(\max\left(q_b\right)\geq\tau\right)H\left(q_s,p_m\left(y\mid\mathcal{A}_s\left(u_b\right)\right)\right), \quad (1)$$

where τ is the constant pre-defined threshold, $q_s = \frac{\exp\left(q_b/\mathbf{T}\right)}{\sum_k \exp\left(q_k/\mathbf{T}\right)}$ is the sharpen predictions by temperature $\mathbf{T}$, $q_b = p_m\left(y\mid\mathcal{A}_w\left(u_b\right)\right)$ is the logit of label y for input $\mathcal{A}_w\left(u_b\right)$. Unlike UDA, FixMatch leverages this consistency regularization with strong augmentation to achieve a competitive performance. The unsupervised loss term becomes:

$$\frac{1}{\mu B}\sum_{b=1}^{\mu B}\mathbb{1}\left(\max\left(q_b\right)\geq\tau\right)H\left(\hat{q}_b,p_m\left(y\mid\mathcal{A}_s\left(u_b\right)\right)\right), \quad (2)$$

where $\hat{q}_b = \arg\max\left(q_b\right)$ is the pseudo-label of q_b. Following FixMatch, FlexMatch uses the same loss term with a dynamic threshold τ_t for each class, thus improving the per-class sampling rate and making the model learn equally.

FixMatch shows that using a high-confidence threshold with ”hard” labels can eliminate the noise pseudo-labels, thus enhancing the performance of the whole SSL framework. In addition, FixMatch also claims that using the high-confidence threshold with the ”soft” pseudo-labels does not show a significant difference in performance.

4. SequenceMatch

We propose SequenceMatch, a simple SSL pipeline that aims at balancing the prediction distribution of unlabeled data. The distinction with FixMatch is that we consider both ”hard” and ”soft” pseudo-labels. The main novelty comes from the additional medium augmentations for unlabeled data. With the additional mediumly augmented examples, SequenceMatch helps reduce the divergence between the prediction distributions of the weakly and strongly augmented data.

The intuition is to make the prediction distribution of weakly, mediumly, and strongly augmented examples similar to each other while maintaining the correct pseudo-labels, thus reducing the overfitting of the model on labeled data and reducing the confirmation bias issue. The medium augmentation is made up of weak augmentation, a transformation chosen at random from the list of strong augmentation transformations, and cutout [10]. This makes the mediumly augmented samples look different from the weakly augmented ones, but not as distorted as the strongly augmented ones because the induced distortions could severely change the image structures, and thus the transformed images cannot maintain the identity of the original instances. We visualize the differences of three kinds of augmentation in Appendix **??**.

4.1. SequenceMatch Pipeline

In this section, we present the pipeline of SequenceMatch method as shown in Figure 3. First, similar to other SSL methods, we train the model on the labeled data. Then, for the unlabeled data, instead of using only weakly and strongly augmented examples, we create three versions of augmented input: weakly, mediumly, and strongly augmented examples. Finally, for each pair of the prediction distribution such as weak-medium, medium-strong, and weak-strong, a Kullback-Leibler divergence loss function is used to measure the divergence of each pair. The KL divergence losses will be optimized during the training process to minimize the divergence.

4.2. Loss Function

The loss function for SequenceMatch consists of two different loss terms. One is the supervised loss, which is a standard cross-entropy loss ($\mathcal{L}_s^{\mathrm{CE}}$) for the labeled data. The other one is the unsupervised loss, including the Kullback-Leibler divergence ($\mathcal{L}_{KL}$) between the prediction distributions and the standard cross-entropy loss ($\mathcal{L}_u^{\mathrm{CE}}$) for strongly augmented data with pseudo-labels.

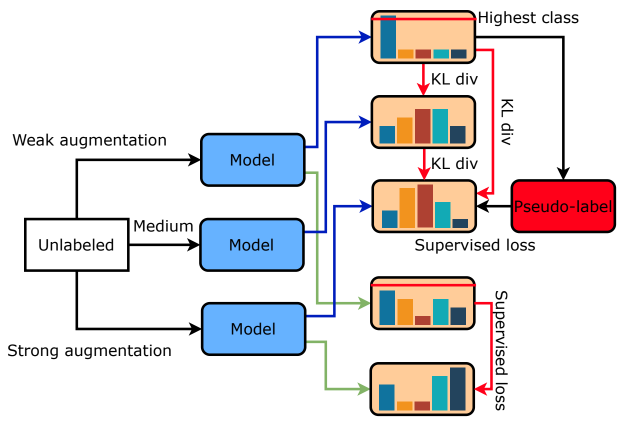


Figure 3. SequenceMatch pipeline. Unlike other SSL methods that use only two types of augmented versions for unlabeled data, we propose a "mediumly augmented" version for unlabeled data. The blue and green arrows indicate high-confidence and low-confidence predictions, respectively. In addition, we measure the Kullback-Leibler divergence losses between the weakly, mediumly, and strongly augmented versions of the same input, then we minimize them during training.

For an L-class classification problem, let $\mathcal{X} = \{(x_b, y_b) : b \in (1, \ldots, B)\}$ be a batch of B labeled examples, where x_b is the training examples and y_b is one-hot labels. Let $\mathcal{U} = \{u_b : b \in (1, \ldots, \mu B)\}$ be a batch of μB unlabeled examples where μ is a hyperparameter that determines the relative sizes of $\mathcal{X}$ and $\mathcal{U}$. Let $p_m(y|x)$ is the predicted class distribution of the model for input x, $H(p, q)$ denotes the "hard" label cross entropy between two probability distributions p and q. The loss function for SSL is defined as:

$$\mathcal{L}_{\text{SSL}} = \mathcal{L}_s^{\text{CE}} + \lambda_u \mathcal{L}_u, \tag{3}$$

where λ_u represents the constant weight for the loss of unlabeled data. The term $\mathcal{L}_s^{\text{CE}}$ refers to the standard cross entropy loss applied to weakly augmented labeled data:

$$\mathcal{L}_s^{\text{CE}} = \frac{1}{B} \sum_{b=1}^{B} \text{H}\left(y_b, p_m\left(y \mid \mathcal{A}_w\left(x_b\right)\right)\right) \tag{4}$$

Then, let $\mathcal{A}_w$, $\mathcal{A}_m$, $\mathcal{A}_s$ be the weakly, mediumly, and strongly augmentations for unlabeled data. $\mathcal{L}_u$ is defined as a total of the standard cross-entropy loss ($\mathcal{L}_u^{\text{CE}}$) and the Kullback-Leibler divergence loss ($\mathcal{L}_{\text{KL}}$). $\mathcal{L}_u^{\text{CE}}$ has two parts: first is the cross-entropy loss between pseudo-labels and the strongly augmented predictions; second is the cross-entropy loss between sharpen predictions of weakly and strongly augmented samples. $\mathcal{L}_{KL}$ is the KL divergence of the prediction distribution between each pair of augmented examples $\mathcal{A}_w - \mathcal{A}_m$, $\mathcal{A}_w - \mathcal{A}_s$, $\mathcal{A}_m - \mathcal{A}_s$:

$$\mathcal{L}_u = \mathcal{L}_u^{\text{CE}} + \mathcal{L}_{\text{KL}}^{w-m} + \mathcal{L}_{\text{KL}}^{m-s} + \mathcal{L}_{\text{KL}}^{w-s} \tag{5}$$

$$\mathcal{L}_u^{\text{CE}} = \frac{1}{\mu B} \sum_{b=1}^{\mu B} \left(\mathbb{1}\left(\max\left(q_b^w\right) \geq \tau\right) \text{H}\left(\hat{q}_b, p_m\left(y \mid \mathcal{A}_s\left(u_b\right)\right)\right)\right.$$
$$\left. + \ \mathbb{1}\left(\max\left(q_b^w\right) < \tau\right) \text{H}\left(q_s \mid p_m\left(y \mid \mathcal{A}_s\left(u_b\right)\right)\right)\right) \tag{6}$$

$$\mathcal{L}_{\text{KL}}^{w-m} = \frac{1}{\mu B} \sum_{b=1}^{\mu B} \mathbb{1}\left(\max\left(q_b^w\right) \geq \tau\right) D_{\text{KL}}\left(q_s^w \mid p_m\left(y \mid \mathcal{A}_m\left(u_b\right)\right)\right) \tag{7}$$

$$\mathcal{L}_{\text{KL}}^{m-s} = \frac{1}{\mu B} \sum_{b=1}^{\mu B} \mathbb{1}\left(\max\left(q_b^m\right) \geq \tau\right) D_{\text{KL}}\left(q_s^m \mid p_m\left(y \mid \mathcal{A}_s\left(u_b\right)\right)\right) \tag{8}$$

$$\mathcal{L}_{\text{KL}}^{w-s} = \frac{1}{\mu B} \sum_{b=1}^{\mu B} \mathbb{1}\left(\max\left(q_b^w\right) \geq \tau\right) D_{\text{KL}}\left(q_s^w \mid p_m\left(y \mid \mathcal{A}_s\left(u_b\right)\right)\right) \tag{9}$$

where $\hat{q}_b = \arg\max\left(q_b\right)$ is the pseudo-label with $q_b = p_m\left(y \mid \Omega\left(u_b\right)\right)$, Ω is the corresponding augmentation function, $q_s = \frac{\exp\left(q_b/\mathbf{T}\right)}{\sum_k \exp\left(q_k/\mathbf{T}\right)}$ is the sharpen predictions, τ is the fixed threshold for choosing pseudo-labels, D_{KL} denotes the KL divergence function, and T is the temperature for sharpening. We use the fixed τ for high-confidence KL loss to reduce the divergence of overconfident predictions over three augmentations. Notably, we only enforce the consistency loss for low-confidence on weak-strong predictions pair since the low-confidence predictions from medium and strong augmented predictions are unreliable. Compared with prior methods, the use of unlabeled data by KL loss is more reasonable, as KL loss will not bring negative supervisory information due to the wrong predictions but just emphasize the distribution consistency between the weakly, mediumly, and strongly augmented images.

5. Experiments

We evaluate SequenceMatch on common datasets: CIFAR-10/100 [15], SVHN [25], STL-10 [7], and ImageNet [9], and extensively investigate the performance under various labeled data amounts. We mainly compare our proposed method with fully SSL methods without using self-supervised pre-trained weights such as UDA [42], FixMatch [38], FlexMatch [46], FreeMatch [41], etc, since they all include a pre-defined threshold, and they are currently the state-of-the-art in the field. To gain a deeper understanding of the results from Semi-Supervised Learning (SSL) methods, we have also conducted a fully-supervised experiment for each dataset. We implement our proposed method and evaluate all methods using USB framework[1].

To ensure a fair comparison, we maintain consistent hyperparameters across the UDA, FixMatch, and FlexMatch methods. All experiments employ standard stochastic gradient descent (SGD) with a momentum of 0.9 as the optimizer, as suggested by [28, 39]. Across all datasets, we initiate with a learning rate of 0.03, coupled with a cosine anneal-

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

Table 2. Error rates on CIFAR-10/100, SVHN, and STL-10 datasets on 5 different folds.

DATASET	CIFAR-10			CIFAR-100			SVHN			STL-10	
# LABEL	40	250	4000	400	2500	10000	40	250	1000	40	1000
Π MODEL [31]	74.34±1.76	46.24±1.29	13.13±0.59	86.96±0.80	58.80±0.66	36.65±0.00	67.48±0.95	13.30±1.12	7.16±0.11	74.31±0.85	32.78±0.40
PSEUDO LABEL [17]	74.61±0.26	46.49±2.20	15.08±0.19	87.45±0.85	57.74±0.28	36.55±0.24	64.61±5.60	15.59±0.95	9.40±0.32	74.68±0.99	32.64±0.71
VAT [24]	74.66±2.12	41.03±1.79	10.51±0.12	85.20±1.40	46.84±0.79	32.14±0.19	74.75±3.38	4.33±0.12	4.11±0.20	74.74±0.38	37.95±1.12
MEANTEACHER [40]	70.09±1.60	37.46±3.30	8.10±0.21	81.11±1.44	45.17±1.06	31.75±0.23	36.09±3.98	3.45±0.03	3.27±0.05	71.72±1.45	33.90±1.37
MIXMATCH [4]	36.19±6.48	13.63±0.59	6.66±0.26	67.59±0.66	39.76±0.48	27.78±0.29	30.60±8.39	4.56±0.32	3.69±0.37	54.93±0.96	21.70±0.68
REMIXMATCH [3]	9.88±1.03	6.30±0.05	4.84±0.01	42.75±1.05	$\underline{26.03\pm0.35}$	**20.02±0.27**	24.04±9.13	6.36±0.22	5.16±0.31	32.12±6.24	6.74±0.14
UDA [42]	10.62±3.75	5.16±0.06	4.29±0.07	46.39±1.59	27.73±0.21	22.49±0.23	5.12±4.27	$\underline{1.92\pm0.05}$	$\underline{1.89\pm0.01}$	37.42±8.44	6.64±0.17
FIXMATCH [38]	7.47±0.28	$\underline{4.86\pm0.05}$	4.21±0.08	46.42±0.82	28.03±0.16	22.20±0.12	3.81±1.18	2.02±0.02	1.96±0.03	35.97±4.14	6.25±0.33
DASH [44]	8.93±3.11	5.16±0.23	4.36±0.11	44.82±0.96	27.15±0.22	21.88±0.07	2.19±0.18	2.04±0.02	1.97±0.01	34.52±4.30	6.39±0.56
MPL [27]	6.62±0.91	5.76±0.24	4.55±0.04	46.26±1.84	27.71±0.19	21.74±0.09	9.33±8.02	2.29±0.04	2.28±0.02	35.76±4.83	6.66±0.00
FLEXMATCH [46]	4.97±0.06	4.98±0.09	4.19±0.01	39.94±1.62	26.49±0.20	21.90±0.15	8.19±3.20	6.59±2.29	6.72±0.30	29.15±4.16	5.77±0.18
FREEMATCH [41]	$\underline{4.90\pm0.04}$	4.88±0.18	**4.10±0.02**	$\underline{37.98\pm0.42}$	26.47±0.20	21.68±0.03	$\underline{1.97\pm0.02}$	1.97±0.01	1.96±0.03	$\underline{15.56\pm0.55}$	$\underline{5.63\pm0.15}$
SEQUENCEMATCH	**4.80±0.01**	**4.75±0.05**	$\underline{4.15\pm0.01}$	**37.86±1.07**	**25.99±0.22**	$\underline{20.10\pm0.04}$	**1.96±0.23**	**1.89±0.31**	**1.79±0.02**	**15.45±1.40**	**5.56±0.35**
FULLY-SUPERVISED		4.62±0.05			19.30±0.09			2.13±0.02		NONE	

ing learning rate scheduler [19], for a total of 2^{20} training iterations. We also implement an exponential moving average with a momentum of 0.999. The batch size for labeled data is set at 64, except for ImageNet. For CIFAR-10/100, SVHN, and STL-10, we set μ to 7, while for ImageNet, it is set to 1. In the case of UDA, τ is set to 0.8, whereas for FixMatch, FlexMatch, and SequenceMatch, it is set to 0.95. These configurations follow the original papers [38, 42, 46]. The medium and strong augmentation in our experiments is RandAugment [8] with a different number of augmentations (1 for medium augmentation and 3 for strong augmentation; we study the choices for medium augmentation and visualize the differences in the Appendix). We use ResNet-50 [15] for the ImageNet dataset and Wide-ResNet (WRN) [45] for other datasets.

5.1. CIFAR-10/100, STL-10, SVHN

We evaluate the best error rate of each method by averaging results from five runs with distinct random seeds. The classification error rates on CIFAR-10/100, STL-10, and SVHN datasets are recorded in Table 2.

For the CIFAR-10 and SVHN datasets, we use Wide-ResNet-28-2 [45] as a backbone model, Wide-ResNet-28-8 for the CIFAR-100 dataset, and Wide-ResNet-37-2 for the STL-10 dataset. As shown in Table 2, the proposed method outperforms all other methods on most datasets with varying numbers of labels. According to FlexMatch study [46], FlexMatch performs less favorably on imbalanced datasets such as SVHN. SequenceMatch, on the other hand, not only achieves high performance across all datasets but also performs well on the SVHN dataset. This shows that our proposed method has the effect of reducing overfitting, which usually appears when training on a small and imbalanced dataset.

Precision, Recall, F1 Score and AUC results on CIFAR-10

To comprehensively evaluate the performance of all methods in a classification setting, we further report the precision, recall, F1-score, and AUC (area under curve) results on the CIFAR-10 dataset. As shown in Table 3, we see that in addition to the reduced error rates, SequenceMatch also has the best performance in precision, recall, F1 score, and AUC. These metrics, together with error rates (or accuracy), show the strong performance of our proposed method.

STL-10 Confusion Matrix

The confusion matrix of FixMatch, FlexMatch, and SequenceMatch on the STL-10 dataset with a 40-label split is visualized in Figure 4. Compared with FlexMatch, SequenceMatch improves the performance for classes 2, 4, and 6. In addition, FixMatch is overfitted for class number 1, and it failed to recognize class number 3 and 7.

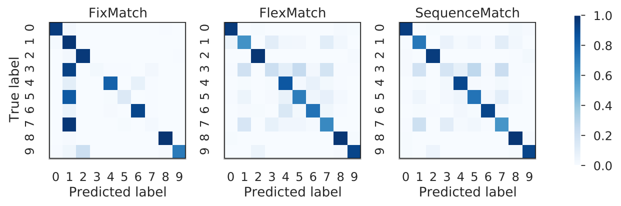


Figure 4. Confusion matrix of FixMatch, FlexMatch, and SequenceMatch features on STL-10 dataset with 40-label.

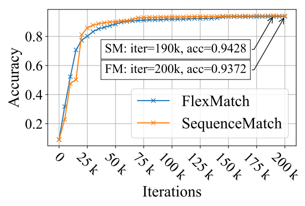
Convergence Speed

Our proposed SequenceMatch outperforms FlexMatch when the number of labels is limited. We visualize a validation loss and a top-1 accuracy of both FlexMatch and SequenceMatch on the CIFAR-10 dataset with 40 labels within the first 200k iterations. As we can see in Figure 5, SequenceMatch achieves over 80% of accuracy within the first 25k

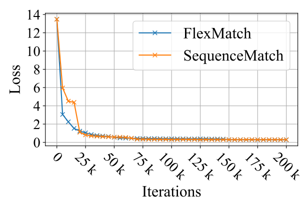
Table 3. Precision, recall, F1-score and AUC results on CIFAR-10.

Label Amount	40 labels				4000 labels			
Criteria	Precision	Recall	F1 Score	AUC	Precision	Recall	F1 Score	AUC
FixMatch	0.9333	0.9290	0.9278	0.9910	0.9571	0.9571	0.9569	0.9984
FlexMatch	0.9506	0.9507	0.9506	0.9975	0.9580	0.9581	0.9580	0.9984
FreeMatch	0.9510	0.9512	0.9510	-	0.9568	0.9568	0.9567	-
SequenceMatch	**0.9519**	**0.9521**	**0.9519**	**0.9976**	**0.9590**	**0.9591**	**0.9590**	**0.9986**

iterations when FlexMatch nearly hits 80%. After 200k iterations, SequenceMatch achieves up to 94.28% accuracy while FlexMatch can only achieve 93.72% of accuracy. Moreover, the loss of our proposed SequenceMatch also decreases as fast and smoothly as FlexMatch, even though we add extra augmented data.



(a) Top-1 accuracy



(b) Loss

Figure 5. Convergence analysis vs FlexMatch. (a) and (b) depict the loss and top-1 accuracy on CIFAR-10 with 40 labels. ”SM” and ”FM” denote SequenceMatch and FlexMatch, respectively.

Class-wise accuracy on CIFAR-10 40-label

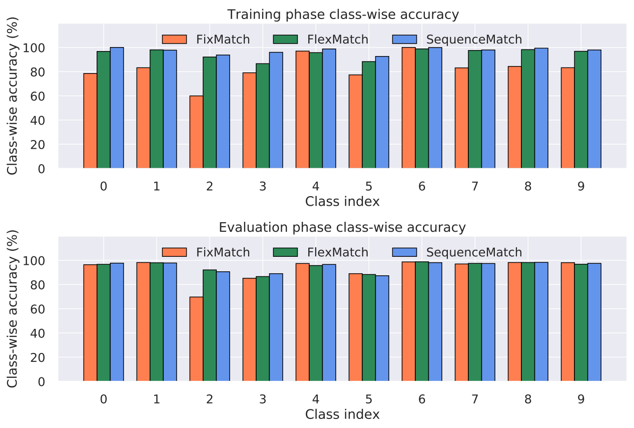


Figure 6. Class-wise accuracy comparison on CIFAR-10 40-label split at the best iteration of FixMatch, FlexMatch and SequenceMatch.

We report a detailed comparison for class-wise accuracy in Table 4. Our proposed SequenceMatch not only retains a high accuracy in easy-to-learn classes but also improves the accuracy of hard-to-learn classes. The final class-wise accuracy of SequenceMatch is balanced between classes, including hard-to-learn classes (*e.g.* class 2, 3). Especially for the evaluation phase, the performance of hard-to-learn classes surpasses FixMatch by a large margin.

The class-wise accuracy from the training phase, as shown in Figure 6, also indicates that SequenceMatch can help reduce the confirmation bias issue. It can be seen that the training phase accuracy of SequenceMatch is not only higher than FixMatch and FlexMatch but also balanced between classes. SequenceMatch prevents the trained model from overfitting toward easy-to-learn classes.

Figure 7 shows the accuracy of the pseudo-label during training on the CIFAR-10 40-label split. We can see that SequenceMatch mitigates the overfitting and overconfidence issues, therefore achieving a higher pseudo-label accuracy.

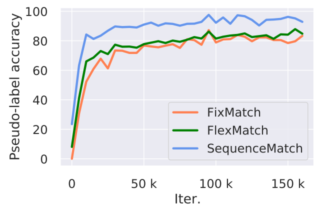
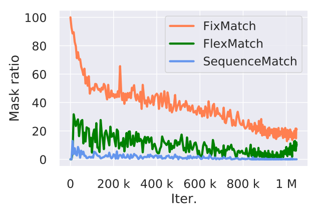
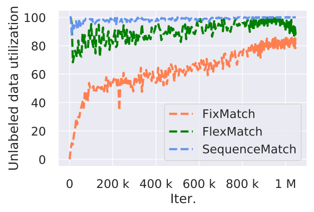


Figure 7. Pseudo-label accuracy on CIFAR-10-40.



(a) Mask ratio



(b) Data utilization ratio

Figure 8. Unlabeled data utilization and mask ratio on CIFAR-100 dataset with 400-label split.

Data utilization and mask ratio

We present the unlabeled data utilization and mask ratio of FixMatch, FlexMatch, and SequenceMatch on the CIFAR-100 dataset with a 400-label split in Figure 8a, 8b. SequenceMatch significantly reduces the mask-out data ratio and is

Table 4. Class-wise accuracy comparison on CIFAR-10 40-label split.

Class Number	0	1	2	3	4	5	6	7	8	9
FixMatch	0.964	0.982	0.697	0.852	**0.974**	0.890	0.987	0.970	0.982	**0.981**
FlexMatch	0.967	0.980	0.921	0.866	0.957	0.883	**0.988**	0.975	0.982	0.968
FreeMatch	0.962	0.984	**0.923**	0.874	0.963	**0.894**	0.979	**0.977**	0.980	0.976
SequenceMatch	**0.977**	**0.984**	0.922	**0.890**	0.966	0.889	0.981	0.974	**0.985**	0.980

stable throughout the training process. It also can be seen that the mask ratio of SequenceMatch fluctuates less than that of FixMatch and FlexMatch. Furthermore, the data utilization ratio of SequenceMatch surpasses that of FixMatch and FlexMatch by a large margin.

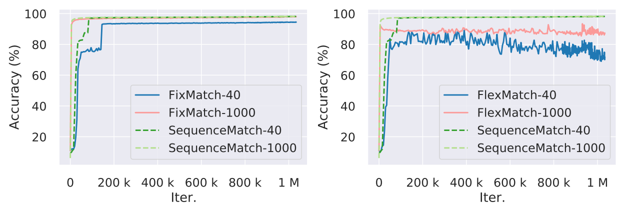
5.2. ImageNet

We further evaluate SequenceMatch on the ImageNet [9] dataset to verify the performance on the large and complex dataset. We compare the proposed SequenceMatch with FixMatch, FlexMatch, CoMatch, and SimMatch. All of the models are trained on 100k of the training data as labeled. The rest of the data is treated as unlabeled data. Furthermore, because the ImageNet dataset is large and complex, we set the τ threshold to 0.7 to improve the capture of samples with the correct pseudo-label. The batch size is set to 128 and the weight decay is set to 0.0003.

Table 5. Error rate results on ImageNet.

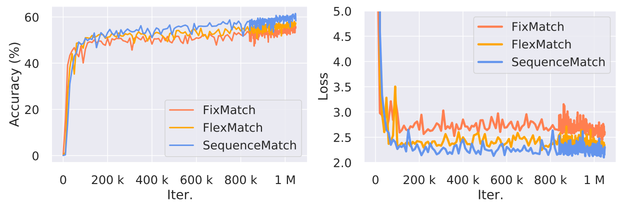
Method	Top-1	Top-5	Top-1	Top-5
	100k		10%	
FixMatch [38]	43.66	21.80	28.50	10.90
FlexMatch [46]	41.85	19.48	-	-
CoMatch [18]	42.17	19.64	26.30	8.60
SimMatch [47]	-	-	25.60	8.40
FreeMatch [41]	40.57	18.77	-	-
SequenceMatch	**38.46**	**17.38**	**25.20**	**8.10**

As reported in Table 5, SequenceMatch outperforms FlexMatch with 38.46% and 17.38% for top-1 and top-5 error rates, respectively. The top-1 error rate result is 3.39% lower than FlexMatch and 5.20% lower than FixMatch. This result strongly indicates that when the task is complicated and the dataset is imbalanced (in the ImageNet dataset, the number of images per class ranges between 732 and 1300), our proposed SequenceMatch can help boost the performance. We also compare SequenceMatch with CoMatch and SimMatch using their source code on 10% labeled data. SequenceMatch outperforms FixMatch with and without using self-supervised pre-trained weights. Compared with CoMatch and SimMatch, SequenceMatch achieves higher performance while having a fewer number of parameters.



(a) vs FixMatch (b) vs FlexMatch

Figure 9. Accuracy comparison of Figure 9a: FixMatch vs SequenceMatch and Figure 9b: Flexmatch vs SequenceMatch for first 150k iterations on SVHN dataset with 40-label and 1000-label



(a) Top-1 accuracy (b) Eval loss

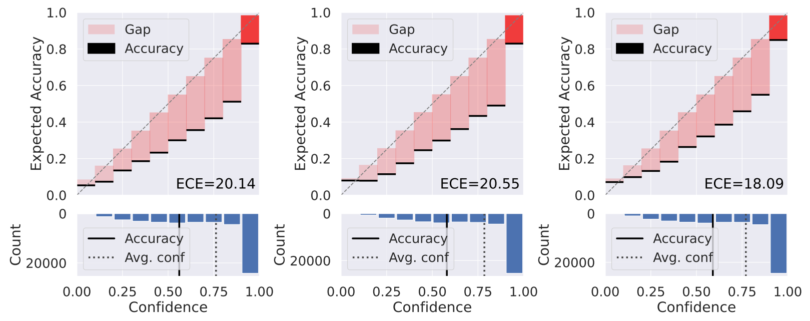
Figure 10. Accuracy and loss comparison of Fixmatch, FlexMatch, and SequenceMatch on Imagenet dataset.

5.3. Imbalance dataset

In Figure 9 and 10, we show the performance of FixMatch, FlexMatch, and SequenceMatch on the SVHN and ImageNet datasets. For instance, our proposed SequenceMatch results show superiority over FixMatch and FlexMatch when dealing with imbalanced data problems such as SVHN and ImageNet datasets. According to Table 2, Table 5, our results are identical to FlexMatch results; however, FlexMatch fails on the SVHN dataset since CPL may generate low final thresholds for the tail classes that allow noisy pseudo-labeled samples to be trusted and learned. SequenceMatch solves this problem by maintaining the consistency of the model throughout the training process and mitigating the overfitting issue. Furthermore, SequenceMatch results on the ImageNet dataset outperform FixMatch and FlexMatch without additional modules.

5.4. Calibration of SSL

The study by [6] proposes tackling confirmation bias from a calibration standpoint. We evaluate the calibration



(a) FixMatch (b) FlexMatch (c) SequenceMatch

Figure 11. Reliability diagrams (top) and confidence histograms (bottom) for ImageNet dataset.

of FixMatch, FlexMatch, and SequenceMatch, which are trained on the ImageNet dataset with 100k labels [2]. Several standard calibration metrics are employed, including the Expected Calibration Error (ECE), confidence histogram, and reliability diagram. As depicted in Figure 11, despite FlexMatch exhibiting higher accuracy than FixMatch, it has a larger ECE value of 20.55 compared to FixMatch's 20.14, suggesting less accurate probability estimation. Conversely, SequenceMatch not only achieves superior accuracy but also a lower ECE value of 18.09. This demonstrates its ability to mitigate confirmation bias and yield a well-calibrated model.

6. Related Work

Self-training is a concept that has been around for decades [22, 36]. Self-training (i.e., utilizing a prediction distribution to generate pseudo-labels for unlabeled data) has been employed in a variety of areas, including natural language processing [21], object recognition [33], image classification [17,43], domain adaption [48], etc. Pseudo-labeling [17] is a pioneering SSL method that uses "hard" artificial labels converted from model predictions. Pseudo-labeling is often paired with confidence-based thresholding, a method that only retains unlabeled samples when the predictions are highly confident. This approach is commonly used in various studies [33, 38, 42, 46].

[2] introduced consistency regularization, which was later popularized [16,35]. Consistency regularization utilizes unlabeled data by relying on the assumption that the model should output similar predictions when perturbed versions of the same image are fed. Data augmentation [11], stochastic regularization [16, 34], and adversarial perturbations [24] have all been used to generate random perturbations. It has recently been demonstrated that applying significant data augmentation can improve outcomes [42].

FixMatch [38] proposed a combination of both pseudo-labeling and consistency regularization methods for SSL. FixMatch's thresholded pseudo-labeling produces a sharpening-like effect that encourages the model to deliver high-confidence predictions. FixMatch could be considered a combination version of UDA and ReMixMatch, in which two common strategies (pseudo-labeling and consistency regularization) are integrated while many components are removed (sharpening, training signal annealing from UDA, distribution alignment, and the rotation loss from ReMixMatch, etc.). FlexMatch [46] introduces a Curriculum Pseudo Labeling (CPL) method, which enables conventional SSL to train with a dynamic threshold for each class. CPL can be considered a dynamic thresholding approach since it dynamically adjusts the threshold for each class after each iteration, thus enabling higher performance for each class. FlexMatch outperforms most state-of-the-art SSL across a wide range of datasets.

Lately, CoMatch [18] is introduced, which combines two contrastive representations on unlabeled data. However, CoMatch is extremely sensitive to hyperparameter settings and requires a large memory bank during training to store the embedded features. Recently work of [47] considers the semantic similarity and instance similarity during training. It shows that forcing consistency on both the semantic-level and instance-level can bring an improvement, thus achieving state-of-the-art benchmarks.

7. Conclusion

In this paper, we introduce SequenceMatch, an SSL pipeline that sequentially matches predictions to reduce the divergence between the predicted class distributions for different augmented versions of the same input. SequenceMatch introduces a medium augmentation for unlabeled data, which helps reduce the divergence between the prediction distributions while maintaining the correct pseudo-label. Furthermore, SequenceMatch also helps reduce the overfitting phenomenon that most SSL methods are facing. SequenceMatch achieves state-of-the-art performance on a variety of SSL benchmarks and works well for all datasets.

[2]https://github.com/hollance/reliability-diagrams

References

[1] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning*, pages 173–182. PMLR, 2016. 1

[2] Philip Bachman, Ouais Alsharif, and Doina Precup. Learning with pseudo-ensembles. In *NIPS*, 2014. 1, 8

[3] David Berthelot, Nicholas Carlini, Ekin Dogus Cubuk, Alexey Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring. In *ICLR*, 2020. 1, 2, 5

[4] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. *Advances in neural information processing systems*, 32, 2019. 1, 5

[5] Aleksander Botev, Matthias Bauer, and Soham De. Regularising for invariance to data augmentation improves supervised learning. *arXiv preprint arXiv:2203.03304*, 2022. 2

[6] Mingcai Chen, Yuntao Du, Yi Zhang, Shuwei Qian, and Chongjun Wang. Semi-supervised learning with multi-head co-training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6278–6286, 2022. 7

[7] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223. JMLR Workshop and Conference Proceedings, 2011. 4

[8] Ekin Dogus Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical automated data augmentation with a reduced search space. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3008–3017, 2020. 5

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, K. Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 4, 7

[10] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 3

[11] Geoffrey French, Michal Mackiewicz, and Mark Fisher. Self-ensembling for visual domain adaptation. *arXiv preprint arXiv:1706.05208*, 2017. 8

[12] 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*, pages 770–778, 2016. 1

[13] Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md. Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically, 2017. 1

[14] Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. Exploring the limits of language modeling, 2016. 1

[15] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 4, 5

[16] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017. 1, 8

[17] Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896, 2013. 1, 5, 8

[18] 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*, pages 9475–9484, 2021. 2, 7, 8

[19] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 5

[20] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining, 2018. 1

[21] David McClosky, Eugene Charniak, and Mark Johnson. Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 152–159, 2006. 8

[22] Geoffrey J McLachlan. Iterative reclassification procedure for constructing an asymptotically optimal rule of allocation in discriminant analysis. *Journal of the American Statistical Association*, 70(350):365–369, 1975. 8

[23] Seyed Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 5191–5198, 2020. 2

[24] 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. 5, 8

[25] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011. 4

[26] Khanh-Binh Nguyen and Joon-Sung Yang. Boosting semi-supervised learning by bridging high and low-confidence predictions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1028–1038, 2023. 2

[27] 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*, pages 11557–11568, 2021. 5

[28] Boris T Polyak. Some methods of speeding up the convergence of iteration methods. *Ussr computational mathematics and mathematical physics*, 4(5):1–17, 1964. 4

[29] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019. 1

[30] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and

Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2020. 1

[31] Antti Rasmus, Harri Valpola, Mikko Honkala, Mathias Berglund, and Tapani Raiko. Semi-supervised learning with ladder networks. *arXiv preprint arXiv:1507.02672*, 2015. 5

[32] Chuck Rosenberg, Martial Hebert, and Henry Schneiderman. Semi-supervised self-training of object detection models. In *WACV/MOTION*, pages 29–36, 2005. 1

[33] Chuck Rosenberg, Martial Hebert, and Henry Schneiderman. Semi-supervised self-training of object detection models. *2005 Seventh IEEE Workshops on Applications of Computer Vision (WACV/MOTION'05) - Volume 1*, 1:29–36, 2005. 8

[34] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. *Advances in neural information processing systems*, 29:1163–1171, 2016. 8

[35] Mehdi S. M. Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *NIPS*, 2016. 1, 8

[36] Henry Scudder. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*, 11(3):363–371, 1965. 8

[37] Richard Socher, Yoshua Bengio, and Christopher D Manning. Deep learning for nlp (without magic). In *Tutorial Abstracts of ACL 2012*, pages 5–5. 2012. 1

[38] 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. *Advances in neural information processing systems*, 33:596–608, 2020. 3, 4, 5, 7, 8

[39] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147. PMLR, 2013. 4

[40] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NIPS*, 2017. 1, 5

[41] Yidong Wang, Hao Chen, Qiang Heng, Wenxin Hou, Marios Savvides, Takahiro Shinozaki, Bhiksha Raj, Zhen Wu, and Jindong Wang. Freematch: Self-adaptive thresholding for semi-supervised learning. *arXiv preprint arXiv:2205.07246*, 2022. 4, 5, 7

[42] Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation for consistency training. *Advances in Neural Information Processing Systems*, 33:6256–6268, 2020. 3, 4, 5, 8

[43] Qizhe Xie, Minh-Thang Luong, Eduard H. Hovy, and Quoc V. Le. Self-training with noisy student improves imagenet classification. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020*, pages 10684–10695, 2020. 1, 8

[44] Yi Xu, Lei Shang, Jinxing Ye, Qi Qian, Yu-Feng Li, Baigui Sun, Hao Li, and Rong Jin. Dash: Semi-supervised learning with dynamic thresholding. In *International Conference on Machine Learning*, pages 11525–11536. PMLR, 2021. 5

[45] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *ArXiv*, abs/1605.07146, 2016. 5

[46] Bowen Zhang, Yidong Wang, Wenxin Hou, Hao Wu, Jindong Wang, Manabu Okumura, and Takahiro Shinozaki. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. *Advances in Neural Information Processing Systems*, 34:18408–18419, 2021. 4, 5, 7, 8

[47] 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*, pages 14471–14481, 2022. 7, 8

[48] Yang Zou, Zhiding Yu, BVK Kumar, and Jinsong Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *Proceedings of the European conference on computer vision (ECCV)*, pages 289–305, 2018. 8