International Journal of Computer Vision (2023) 131:626–643
https://doi.org/10.1007/s11263-022-01723-4

Revisiting Consistency Regularization for Semi-Supervised Learning

Yue Fan[1] · Anna Kukleva[1] · Dengxin Dai[1] · Bernt Schiele[1]

Received: 14 February 2022 / Accepted: 18 November 2022 / Published online: 7 December 2022

Abstract
Consistency regularization is one of the most widely-used techniques for semi-supervised learning (SSL). Generally, the aim is to train a model that is invariant to various data augmentations. In this paper, we revisit this idea and find that enforcing invariance by decreasing distances between features from differently augmented images leads to improved performance. However, encouraging equivariance instead, by increasing the feature distance, further improves performance. To this end, we propose an improved consistency regularization framework by a simple yet effective technique, FeatDistLoss, that imposes consistency and equivariance on the classifier and the feature level, respectively. Experimental results show that our model defines a new state of the art across a variety of standard semi-supervised learning benchmarks as well as imbalanced semi-supervised learning benchmarks. Particularly, we outperform previous work by a significant margin in low data regimes and at large imbalance ratios. Extensive experiments are conducted to analyze the method, and the code will be published.

Keywords Semi-supervised learning · Consistency regularization · Representation learning · Classification

1 Introduction

Deep learning requires large-scale and annotated datasets to reach state-of-the-art performance (Russakovsky et al. 2015; Lin et al. 2014). As labels are not always available or expensive to acquire a wide range of semi-supervised learning (SSL) methods have been proposed to leverage unlabeled data (Tarvainen and Valpola 2017; Laine and Aila 2017; Miyato et al. 2018; Verma et al. 2019; Berthelot et al. 2019; Sohn et al. 2020; Xie et al. 2020; Berthelot et al. 2020; Arazo et al. 2020; Lee 2013; Pham et al. 2020; French et al. 2020; Bachman et al. 2019; Chen et al. 2020b).

Consistency regularization (Bachman et al. 2014; Laine and Aila 2017; Sajjadi et al. 2016) is one of the most widely-used SSL methods. Recent work (Sohn et al. 2020; Xie et al. 2020; Kuo et al. 2020) achieves strong performance by utilizing unlabeled data in a way that model predictions should be invariant to input perturbations. However, when using advanced and strong data augmentation schemes, we question if the model should be invariant to such strong perturbations. In Fig. 1 we illustrate that strong data augmentation leads to perceptually highly diverse images. Thus, we argue that improving equivariance on such strongly augmented images can provide even better performance rather than making the model invariant to all kinds of augmentations. Moreover, existing works apply consistency regularization either at the feature level or at the classifier level. We find empirically that it is more beneficial to introduce consistency on both levels. To this end, we propose a simple yet effective technique, Feature Distance Loss (FeatDistLoss), to improve data-augmentation-based consistency regularization.

We formulate our FeatDistLoss as to explicitly encourage invariance or equivariance between features from different augmentations while enforcing the same semantic class label. Figure 2 shows the intuition behind the idea. Specifically, encouragement of equivariance for the same image but different augmentations (increase distance between stars and circles of the same color) pushes representations apart from each other, thus, covering more space for the class. Impos-

Communicated by Christian Bauckhage.

✉ Yue Fan
yfan@mpi-inf.mpg.de

Anna Kukleva
akukleva@mpi-inf.mpg.de

Dengxin Dai
ddai@mpi-inf.mpg.de

Bernt Schiele
schiele@mpi-inf.mpg.de

[1] Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany

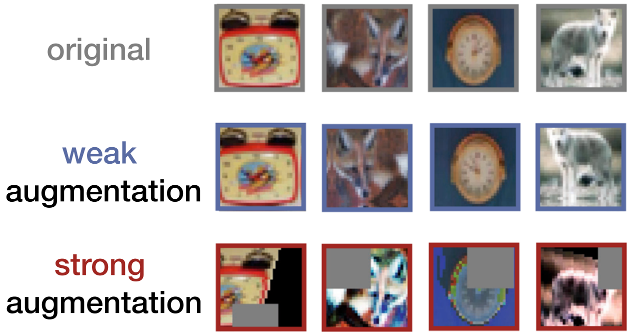

Fig. 1 Examples of strongly and weakly augmented images from CIFAR-100 (please refer to Sect. 3.3 for details of strong and weak augmentation). The visually large difference between them indicates that it can be more beneficial if they are treated differently

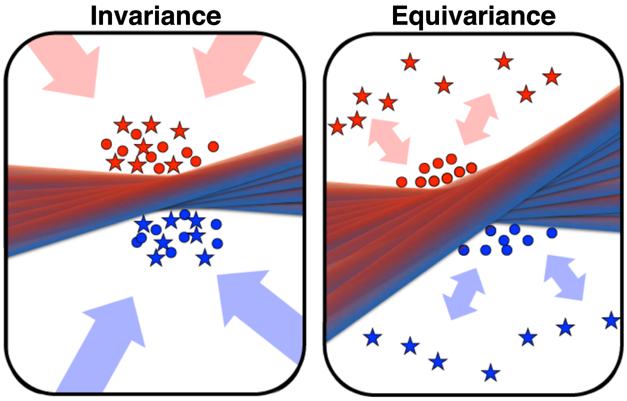

Fig. 2 Binary classification task. Stars are features of strongly augmented images and circles are of weakly augmented images (please refer to Sect. 3.3 for details of strong and weak augmentation). While encouraging invariance by decreasing distance between features from differently augmented images gives good performance (left), encouraging equivariant representations by increasing the distance regularizes the feature space more, leading to even better generalization performance

ing invariance, on the contrary, makes the representations of the same semantic class more compact. In this work, we empirically find that increasing equivariance to differently augmented versions of the same image can lead to better performance especially when rather few labels are available per class (see Sect. 4.3).

This paper introduces the method *CR-Match* which combines FeatDistLoss with other strong techniques defining a new state-of-the-art across a wide range of settings of standard SSL benchmarks, including CIFAR-10, CIFAR-100, SVHN, STL-10, and Mini-Imagenet. More specifically, our contribution is fourfold. (1) We improve data-augmentation-based consistency regularization by a simple yet effective technique for SSL called *FeatDistLoss* which regularizes the distance between feature representations from differently augmented images of the same class as well as the classifier simultaneously. (2) We show that while encouraging invariance results in good performance, encouraging equivariance to differently augmented versions of the same image consistently results in even better generalization performance. (3) We provide comprehensive ablation studies on different distance functions and different augmentations with respect to the proposed FeatDistLoss. (4) In combination with other strong techniques, we achieve *new state-of-the-art results* on most standard semi-supervised learning benchmarks as well as imbalanced semi-supervised learning benchmarks. In particular, our method outperforms previous methods by a significant margin in low data regimes and at large imbalance ratios.

A preliminary version of this work has been published in Fan et al. (2021). In this work, we extend Fan et al. (2021) in three aspects: (1) We extend the existing standard SSL settings by providing evaluations on wider range of the datasets and showing the benefit of the proposed technique on top of various SSL methods. In particular, combining with the recently published method FlexMatch (Zhang et al. 2021), we can push the state-of-the-art even further under the standard settings. Moreover, we evaluate our method on ImageNet to verify that the method scales to larger datasets as well. (2) We evaluate our methods under a more realistic and challenging setting: imbalanced SSL, where the training data is not only partially annotated but also exhibits long-tailed class distribution. We achieve new state-of-the-art results on multiple imbalanced SSL benchmarks across a wide range of settings. (3) To give more in-depth insight into our method, we provide pseudo-code and more analysis of the method, especially the robustness against important hyper-parameters.

2 Related Work

SSL is a broad field aiming to exploit both labeled and unlabeled data. Consistency regularization is a powerful method for SSL (Rasmus et al. 2015; Sajjadi et al. 2016; Bachman et al. 2014). The idea is that the model should output consistent predictions for perturbed versions of the same input. Many works explored different ways to generate such perturbations. For example, Tarvainen and Valpola (2017) uses an exponential moving average of the trained model to produce another input; Sajjadi et al. (2016) and Laine and Aila (2017) use random max-pooling and Dropout (Srivastava et al. 2014); Xie et al. (2020), Berthelot et al. (2020), Sohn et al. (2020) and Kuo et al. (2020) use advanced data augmentation; Berthelot et al. (2019), Verma et al. (2019) and Berthelot et al. (2020) use MixUp regularization (Zhang et al. 2018), which encourages convex behavior “between” examples; Gong et al. (2021) enforces label consistency with alpha-divergence. Another spectrum of popular approaches is pseudo-labeling (Scudder 1965; Nesterov 1983; Lee 2013), where the model is trained with artificial labels. Arazo et al. (2020) trained the model with "soft" pseudo-labels from network predictions; Pham et al. (2020) proposed a meta learning method that deploys a teacher model to adjust the pseudo-label alongside the training of the student; Sohn et al. (2020) and Lee (2013) learn from "hard" pseudo-labels and only retain a pseudo-label if the largest class probability is above a predefined threshold; Zhang et al. (2021) further refines the thresholding mechanism by adaptively adjusting thresholds for different classes according to the learning effect of each class. Furthermore, there are many excellent works around generative models (Kingma et al. 2014; Odena 2016; Denton et al. 2016) and graph-based methods (Luo et al. 2018; Liu et al. 2019; Bengio et al. 2006; Joachims 2003). We refer to Chapelle et al. (2009), Zhu (2005) and Zhu (2009) for a more comprehensive introduction of SSL methods.

Noise injection plays a crucial role in consistency regularization (Xie et al. 2020). Thus advanced data augmentation, especially combined with weak data augmentation, introduces stronger noise to unlabeled data and brings substantial improvements (Berthelot et al. 2020; Sohn et al. 2020). Sohn et al. (2020) proposes to integrate pseudo-labeling into the pipeline by computing pseudo-labels from weakly augmented images, and then uses the cross-entropy loss between the pseudo-labels and strongly augmented images. Besides the classifier level consistency, our model also introduces consistency on the feature level, which explicitly regularizes representation learning and shows improved generalization performance. Moreover, self-supervised learning is known to be beneficial in the context of SSL. He et al. (2020), Chen et al. (2020a), Chen et al. (2020b) and Rebuffi et al. (2020), self-supervised pre-training is used to initialize SSL. However, these methods normally have several training phases, where many hyper-parameters are involved. We follow the trend of Zhai et al. (2019) and Berthelot et al. (2020) to incorporate an auxiliary self-supervised loss alongside training. Specifically, we optimizes a rotation prediction loss (Gidaris and Komodakis 2018).

Another paradigm of SSL is to first perform self-supervised pre-training on unlabeled data and then fine-tune the pretrained model with labeled data. In particular, contrastive learning based methods are gaining popularity and achieve good performance recently (Chen et al. 2020b; He et al. 2020; Caron et al. 2020; Grill et al. 2020; Bardes et al. 2022; Chen and He 2021). The goal of contrastive representation learning is to learn an embedding space in which different versions of the same image stay close to each other while features of different images are far apart. Different to this stream of works, our FeatDistLoss with equivariance pushes apart features from different augmentations of the same image while enforcing the same semantic label, which leads to both more expressive representation and more powerful classifier. Moreover, FeatDistLoss does not have the collapse problem (Chen and He 2021) due to the availability of labeled data.

Equivariant representations are recently explored by capsule networks (Sabour et al. 2017; Hinton et al. 2018). They replaced max-pooling layers with convolutional strides and dynamic routing to preserve more information about the input, allowing for preservation of part-whole relationships in the data. It has been shown, that the input can be reconstructed from the output capsule vectors. Another stream of work on group equivariant networks (Cohen and Welling 2016; Weiler and Cesa 2019; Cohen and Welling 2016) explores various equivariant architectures that produce transform in a predictable linear manner under transformations of the input. Different from previous work, our work explores equivariant representations in the sense that differently augmented versions of the same image are represented by different points in the feature space despite the same semantic label. As we will show in Sect. 4.3, information like object location or orientation is more predictable from our model when features are pushed apart from each other.

Imbalanced semi-supervised learning While SSL has been extensively studied, the setting of class-imbalanced semi-supervised is rather under-explored. Most successful methods from standard SSL do not generalize well to this more realistic scenario without addressing the data imbalance explicitly. Hyun et al. (2020) proposed a suppressed consistency loss to suppress the loss on minority classes. Kim et al. (2020) proposed Distribution Aligning Refinery (DARP) to refine raw pseudo-labels via convex optimization. Wei et al. (2021) found that the raw SSL methods usually have high recall and low precision for head classes while the reverse is true for the tail classes and further proposed a reverse sampling method for unlabeled data based on that. BiS (He et al. 2021) implements a novel sampler which is helpful for the encoder in the beginning but classifier in the end. DASO (Oh et al. 2022) refines pseudo-labels by two complementary classifiers. ABC (Lee et al. 2021) introduces an auxiliary classifier which is trained in a balanced way to help the model while sharing the same backbone. As is shown in Sect. 5, we examine the effectiveness of our method on top of state-of-the-art imbalanced SSL frameworks and show improved results.

3 CR-Match

Consistency regularization is highly-successful and widely-adopted technique in SSL (Bachman et al. 2014; Laine and Aila 2017; Sajjadi et al. 2016; Sohn et al. 2020; Xie et al. 2020; Kuo et al. 2020). In this work, we aim to leverage and improve it by even further regularizing the feature space. To this end, we present a simple yet effective technique FeatDistLoss to explicitly regularize representation learning and classifier learning at the same time. We describe our SSL method, called CR-Match, which shows improved performance across many different settings, especially in scenarios with few labels. In this section, we first describe our technique FeatDistLoss and then present CR-Match that combines FeatDistLoss with other regularization techniques inspired from the literature.

3.1 Feature Distance Loss

Background The idea of consistency regularization (Bachman et al. 2014; Laine and Aila 2017; Sajjadi et al. 2016) is to encourage the model predictions to be invariant to input perturbations. Given a batch of n unlabeled images $\mathbf{u}_i, i \in (1, ..., n)$, consistency regularization can be formulated as the following loss function:

$$\frac{1}{n}\sum_{i=1}^{n} \|f(\mathcal{A}(\mathbf{u}_i)) - f(\alpha(\mathbf{u}_i))\|_2^2 \tag{1}$$

where f is an encoder network that maps an input image to a d-dimensional feature space; $\mathcal{A}$ and α are two stochastic functions which are, in our case, strong and weak augmentations, respectively (details in Sect. 3.3). By minimizing the L_2 distance between perturbed images, the representation is therefore encouraged to become more invariant with respect to different augmentations, which helps generalization. The intuition behind this is that a good model should be robust to data augmentations of the images.

FeatDistLoss As shown in Fig. 3, we extend the above consistency regularization idea by introducing consistency on the classifier level and invariance or equivariance on the feature level. FeatDistLoss thus allows to apply different types of control for these levels. In particular, when encouraging to reduce the feature distance, it becomes similar to classic consistency regularization, and encourages invariance between differently augmented images. As argued above, making the model predictions invariant to input perturbations gives good generalization performance. Instead, in this work we find it is more beneficial to treat images from different augmentations differently because some distorted images are largely different from their original images as demonstrated visually in Fig. 1. Therefore, the final model (CR-Match) uses FeatDistLoss to increase the distance between image features from augmentations of different intensities while at the same time enforcing the same semantic label for them. Note that in Sect. 4.3, we conduct an ablation study on the choice of distance function, where we denote CR-Match as CR-Equiv, and the model that encourages invariance as CR-Inv.

The final objective for the FeatDistLoss consists of two terms: $\mathcal{L}_{Dist}$ (on the feature level), that explicitly regularizes feature distances between embeddings, and a standard cross-entropy loss $\mathcal{L}_{PseudoLabel}$ (on the classifier level) based on pseudo-labeling.

With $\mathcal{L}_{Dist}$ we either decrease or increase the feature distance between weakly and strongly augmented versions of the same image in a low-dimensional space projected from the original feature space to overcome the curse of dimensionality (Bellman 1966). Let $d(\cdot, \cdot)$ be a distance metric and z be a linear layer that maps the high-dimensional feature into a low-dimensional space. Given an unlabeled image $\mathbf{u}_i$, we first extract features with strong and weak augmentations by $f(\mathcal{A}(\mathbf{u}_i))$ and $f(\alpha(\mathbf{u}_i))$ as shown in Fig. 3a, and then FeatDistLoss is computed as:

$$\mathcal{L}_{Dist}(\mathbf{u}_i) = d(z(f(\mathcal{A}(\mathbf{u}_i))), z(f(\alpha(\mathbf{u}_i)))) \tag{2}$$

Different choices of performing $\mathcal{L}_{Dist}$ are studied in Sect. 4.3, where we find empirically that applying $\mathcal{L}_{Dist}$ at (a) using cosine distance in Fig. 3 gives the best performance. The use of the projection head z does not only reduce the computation burden as the original feature space is high-dimensional, but also brings additional performance improvements as shown in Chen et al. (2020a) and Chen et al. (2020b).

At the same time, images from strong and weak augmentations should have the same class label because they are essentially generated from the same original image. Inspired by Sohn et al. (2020), given an unlabeled image $\mathbf{u}_i$, a pseudo-label distribution is first generated from the weakly augmented image by $\hat{\mathbf{p}}_i = g(f(\alpha(\mathbf{u}_i)))$, and then a cross-entropy loss is computed between the pseudo-label and the prediction for the corresponding strongly augmented version as:

$$\mathcal{L}_{PseudoLabel}(\mathbf{u}_i) = \ell_{CE}(\hat{\mathbf{p}}_i, g(f(\mathcal{A}(\mathbf{u}_i)))) \tag{3}$$

where ℓ_{CE} is the cross-entropy, g is a linear classifier that maps a feature representation to a class distribution, and $\mathcal{A}(\mathbf{u}_i)$ denotes the operator for strong augmentations.

Putting it all together, FeatDistLoss processes a batch of unlabeled data $\mathbf{u}_i, i \in (1, ..., B_u)$ with the following loss:

$$\mathcal{L}_U = \frac{1}{B_u}\sum_{i=1}^{B_u} \mathbb{1}\{c_i > \tau\}(\mathcal{L}_{Dist}(\mathbf{u}_i) + \mathcal{L}_{PseudoLabel}(\mathbf{u}_i)) \tag{4}$$

where $c_i = max\ \hat{\mathbf{p}}_i$ is the confidence score, and $\mathbb{1}\{\cdot\}$ is the indicator function which outputs 1 when the confidence score is above a threshold. This confidence thresholding mechanism ensures that the loss is only computed for unlabeled images for which the model generates a high-confidence prediction. Therefore, it controls the trade-off between the quality and the quantity of contributing unlabeled samples. As is shown in Sect. 4.2, a higher threshold τ is normally preferred because it alleviates the instability early in the training by eliminating less confident unlabeled samples. As training progresses, the model produces more confident predictions

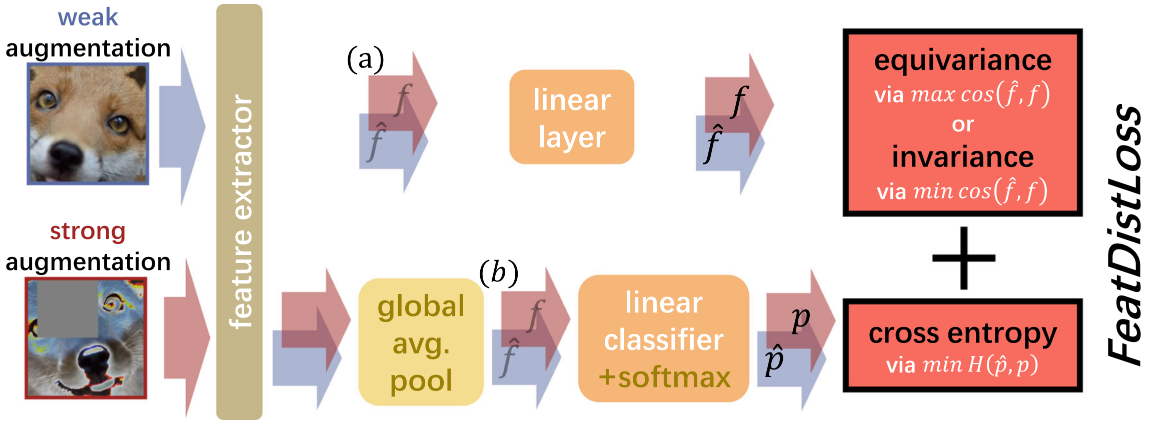

Fig. 3 The proposed FeatDistLoss utilizes unlabeled images in two ways: on the classifier level, different versions of the same image should generate the same class label, whereas on the feature level, representations are encouraged to become either more equivariant (pushing away) or invariant (pulling together). f and $\hat{f}$ denote strong and weak features; p and $\hat{p}$ are predicted class distributions from strong and weak features; **a**, **b** denote features before and after the global average pooling layer. Our final model takes features from a) and encourages equivariance to differently augmented versions of the same image. An ablation study of other choices is in Sect. 4.3

and more samples will contribute to the final loss, which also provides a natural curriculum to balance labeled and unlabeled losses (Sohn et al. 2020). Moreover, the thresholding mechanism is applied for both the feature level consistency and the classifier level consistency so that the two losses are well-synchronized.

As mentioned before, depending on the function d, FeatDistLoss can decrease the distance between features from different data augmentation schemes (when d is a distance function, thus pulling the representations together), or increase it (when d is a similarity function, thus pushing the representations apart). As shown in Table 5, we find that both cases results in an improved performance. However, increasing the distance between weakly and strongly augmented examples consistently results in better generalization performance. We conjecture that the reason lies in the fact that FeatDistLoss by increasing the feature distance explores equivariance properties (differently augmented versions of the same image having distinct features but the same label) of the representations. It encourages the model to have more distinct weakly and strongly augmented images while still imposing the same label, which leads to both more expressive representation and more powerful classifier. As we will show in Sect. 4.3, information like object location or orientation is more predictable from models trained with FeatDistLoss that pushes the representations apart. Additional ablation studies of other design choices such as the distance function and the linear projection z are also provided in Sect. 4.3.

3.2 Overall CR-Match

Now we describe our SSL method called CR-Match leveraging the above FeatDistLoss. Pseudo-code for processing a batch of labeled and unlabeled examples is shown in Algorithm 1.

Algorithm 1

```
Require: Labeled batch $\mathcal{X} = \{(\mathbf{x}_i, \mathbf{p}_i) : i \in (1, \dots, B_s)\}$, unlabeled
    batch $\mathcal{U} = \{\mathbf{u}_i : i \in (1, \dots, B_u)\}$, confidence threshold $\tau$, Feat-
    DistLoss weight $\lambda_u$, rotation prediction loss weight $\lambda_r$, classifier $g$,
    distance metric $d$, FeatDistLoss head $z$, rotation prediction head $h$.
▷ Cross-entropy loss for labeled data
$\mathcal{L}_S = \frac{1}{B_s} \sum_{i=1}^{B_s} \ell_{CE}(\mathbf{p}_i, g(\alpha(\mathbf{x}_i)))$
for $i = 1$ to$B_u$ do
    ▷ Extract representation from weak data augmentation
    $\mathbf{u}_i^w = f(\alpha(\mathbf{u}_i))$
    ▷ Extract representation from strong data augmentation
    $\mathbf{u}_i^s = f(\mathcal{A}(\mathbf{u}_i))$
    ▷ Compute confidence score from the weakly augmented image
    $c_i = max\ g(\mathbf{u}_i^w)$
end for
▷ Cross-entropy loss with pseudo-label for unlabeled data
$\mathcal{L}_{Pseudo} = \frac{1}{B_u} \sum_{i=1}^{B_u} \mathbb{1}\{c_i > \tau\}\, \ell_{CE}(g(\mathbf{u}_i^w), \mathbf{u}_i^s)$
▷ Increase the feature distance for unlabeled data
$\mathcal{L}_{Dist} = \frac{1}{B_u} \sum_{i=1}^{B_u} \mathbb{1}\{c_i > \tau\}\ - d(z(\mathbf{u}_i^w), z(\mathbf{u}_i^s))$
▷ rotation prediction loss
$\mathcal{L}_{Rot} = \frac{1}{4B_u} \sum_{i=1}^{B_u} \sum_{r \in \mathbb{R}} \ell_{CE}(r, h(R(\mathbf{u}_i^w, r)))$ return $\mathcal{L}_S +$
    $\lambda_u(\mathcal{L}_{Pseudo} + \mathcal{L}_{Dist}) + \lambda_r \mathcal{L}_{Rot}$
```

Given a batch of labeled images with their labels as $\mathcal{X} = \{(\mathbf{x}_i, \mathbf{p}_i) : i \in (1, ..., B_s)\}$ and a batch of unlabeled images as $\mathcal{U} = \{\mathbf{u}_i : i \in (1, ..., B_u)\}$.[1] CR-Match minimizes the following learning objective:

$$\mathcal{L}_S(\mathcal{X}) + \lambda_u \mathcal{L}_U(\mathcal{U}) + \lambda_r \mathcal{L}_{Rot}(\mathcal{U}) \tag{5}$$

[1] In practice, unlabeled data includes all labeled data without labels.

where $\mathcal{L}_S$ is the supervised cross-entropy loss for labeled images with weak data augmentation regularization; $\mathcal{L}_U$ is our novel feature distance loss for unlabeled images which explicitly regularizes the distance between weakly and strongly augmented images in the feature space; and $\mathcal{L}_{Rot}$ is a self-supervised loss for unlabeled images and stands for rotation prediction from Gidaris and Komodakis (2018) to provide an additional supervisory and regularizing signal.

Fully supervised loss for labeled data We use cross-entropy loss with weak data augmentation regularization for labeled data:

$$\mathcal{L}_S = \frac{1}{B_s} \sum_{i=1}^{B_s} \ell_{CE}(\mathbf{p}_i, g(f(\alpha(\mathbf{x}_i)))) \tag{6}$$

where ℓ_{CE} is the cross-entropy loss, $\alpha(\mathbf{x}_i)$ is the extracted feature from a weakly augmented image $\mathbf{x}_i$, g is the same linear classifier as in equation 2, and $\mathbf{p}_i$ is the corresponding label for $\mathbf{x}_i$.

Self-supervised loss for unlabeled data Rotation prediction (Gidaris and Komodakis 2018) (RotNet) is one of the most successful self-supervised learning methods, and has been shown to be complementary to SSL methods (Zhai et al. 2019; Berthelot et al. 2019; Rebuffi et al. 2020). Here, we create four rotated images by 0°, 90°, 180°, and 270° for each unlabeled image $\mathbf{u}_i$ for $i \in (1, ..., \mu B)$. Then, classification loss is applied to train the model predicting the rotation as a four-class classification task:

$$\mathcal{L}_{Rot} = \frac{1}{4B_u} \sum_{i=1}^{B_u} \sum_{r \in \mathbb{R}} \ell_{CE}(r, h(\alpha(R(\mathbf{u}_i, r)))) \tag{7}$$

where $\mathbb{R}$ is $\{0°, 90°, 180°, 270°\}$ and r refers to one of the four rotations, h denotes a three-layer MLP with its hidden dimension the same as the input dimension. Using a predictor head is shown to be beneficial for such an auxiliary loss (Chen et al. 2020a, b). Note that rotation prediction, though commonly used, might also have adverse effects. For example, numbers six and nine in most print fonts are centrosymmetric, rotating one upside down gives the other.

3.3 Implementation Details

Data augmentation As mentioned above, CR-Match adopts two types of data augmentations: weak augmentation and strong augmentation from Sohn et al. (2020). Specifically, the weak augmentation α corresponds to a standard random cropping and random mirroring with probability 0.5, and the strong augmentation $\mathcal{A}$ is a combination of RandAugment (Cubuk et al. 2020) and CutOut (DeVries and Taylor 2017). At each training step, we uniformly sample two operations for the strong augmentation from a collection of transformations and apply them with a randomly sampled magnitude from a predefined range. The complete table of transformation operations for the strong augmentation is provided in the supplementary material.

Other implementation details For our results in Sects. 4 and 5, we minimize the cosine similarity in FeatDistLoss, and use a fully-connected layer for the projection layer z, which maps the feature from the original un-flattened 8192-dimension space into a 128-dimension space, the same dimension as the feature dimension for classification. The dimension of the original feature space and the patch size are fixed and depend on the architecture, which is chosen following the previous conventions (Oliver et al. 2018; Berthelot et al. 2019, 2020; Sohn et al. 2020). In our case, $8192 = 8 \times 8 \times 128$, where the patch size is 8×8, and there are 128 feature maps. The predictor head h in rotation prediction loss consists of two fully-connected layers and a ReLU as non-linearity. We use the same $\lambda_u = \lambda_r = 1$ in all experiments since CR-Match shows good robustness within a range of loss weights in our preliminary experiments. We train our model for 512 epochs on CIFAR-10, CIFAR-100, and SVHN. On STL-10 and Mini-ImageNet, we train the model for 300 epochs. Other hyper-parameters are from Sohn et al. (2020) for the compatibility. Specifically, the confidence thresholds τ for pseudo-label selection is 0.95. We use SGD with momentum 0.9 and cosine learning rate schedule from Sohn et al. (2020) starting from 0.03, batch size B_s is 64 for labeled data, and B_u is $7 \times B_s$. The final performance is reported using an exponential moving average of model parameters as recommended by Tarvainen and Valpola (2017). As a common practice, we repeat each experiment with five different data splits and report the mean and the standard deviation of the error rate.

4 Experimental Results

Following protocols from previous work (Berthelot et al. 2019; Sohn et al. 2020), we conduct experiments on several commonly used SSL image classification benchmarks to test the efficacy of CR-Match. We show our main results in Sect. 4.1, where we achieve state-of-the-art error rates across all settings on SVHN (Netzer et al. 2011), CIFAR-10 (Krizhevsky and Hinton 2009), CIFAR-100 (Krizhevsky and Hinton 2009), STL-10 (Coates et al. 2011), and mini-ImageNet (Ravi and Larochelle 2017). In our ablation study in Sect. 4.2 we analyze the effect of FeatDistLoss and RotNet across different settings. Finally, in Sect. 4.3 we extensively analyse various design choices for our FeatDistLoss.

Table 1 Error rates on CIFAR-10, and CIFAR-100

	CIFAR-10			CIFAR-100		
Per class labels	4 labels	25 labels	400 labels	4 labels	25 labels	100 labels
Mean teacher (Tarvainen and Valpola 2017)	-	32.32 ± 2.30*	9.19 ± 0.19*	-	53.91 ± 0.57*	35.83 ± 0.24*
MixMatch (Berthelot et al. 2019)	47.54 ± 11.50*	11.08 ± 0.87	6.24 ± 0.06	67.61 ± 1.32*	39.94 ± 0.37*	25.88 ± 0.30
UDA (Xie et al. 2020)	29.05 ± 5.93*	5.43 ± 0.96	4.32 ± 0.08*	59.28 ± 0.88*	33.13 ± 0.22*	24.50 ± 0.25*
ReMixMatch (Berthelot et al. 2020)	19.10 ± 9.64*	6.27 ± 0.34	5.14 ± 0.04	44.28 ± 2.06*	27.43 ± 0.31*	23.03 ± 0.56*
FixMatch (RA) (Sohn et al. 2020)	13.81 ± 3.37	*5.07* ± 0.65	*4.26* ± 0.05	48.85 ± 1.75	28.29 ± 0.11	22.60 ± 0.12
FixMatch (CTA) (Sohn et al. 2020)	11.39 ± 3.35	*5.07* ± 0.33	4.31 ± 0.15	49.95 ± 3.01	28.64 ± 0.24	23.18 ± 0.11
FeatMatch (Kuo et al. 2020)	-	6.00 ± 0.41	4.64 ± 0.11	-	-	-
FlexMatch (Zhang et al. 2021)	**5.19** ± 0.05	5.33 ± 0.12	4.47 ± 0.09	45.91 ± 1.76	28.11 ± 0.20	23.04 ± 0.28
CR-Match	10.70 ± 2.91	**5.05** ± 0.12	**3.96** ± 0.16	*39.45* ± 1.69	*25.43* ± 0.14	**20.40** ± 0.08
CR-Match§	*5.52* ± 0.32	5.21 ± 0.06	*4.26* ± 0.19	**35.72** ± 0.50	**24.61** ± 0.37	*20.91* ± 0.24

A Wide ResNet-28-2 (Zagoruyko and Komodakis 2016) is used for CIFAR-10 and a Wide ResNet-28-8 with 135 filters per layer (Berthelot et al. 2019) is used for CIFAR-100. We use the same code base as Sohn et al. (2020) (i.e., same network architecture and training protocol) to make the results directly comparable. The best number is in bold and the second best number is in italic

*Numbers are generated by Sohn et al. (2020). CR-Match§ refers to CR-Match combined with CPL (Zhang et al. 2021) from FlexMatch

Table 2 *Left:* error rates on STL-10 and SVHN

Per class labels	STL-10	SVHN		
	100 labels	4 labels	25 labels	100 labels
Mean Teacher (Tarvainen and Valpola 2017)	21.34 ± 2.39*	-	3.57 ± 0.11*	3.42 ± 0.07*
MixMatch (Berthelot et al. 2019)	10.18 ± 1.46	42.55 ± 14.53*	3.78 ± 0.26	3.27 ± 0.31
UDA (Xie et al. 2020)	7.66 ± 0.56*	52.63 ± 20.51*	2.72 ± 0.40	*2.23* ± 0.07
ReMixMatch (Berthelot et al. 2020)	6.18 ± 1.24	*3.34* ± 0.20*	3.10 ± 0.50	2.83 ± 0.30
FixMatch (RA) (Sohn et al. 2020)	7.98 ± 1.50	3.96 ± 2.17	*2.48* ± 0.38	2.28 ± 0.11
FixMatch (CTA) (Sohn et al. 2020)	*5.17* ± 0.63	7.65 ± 7.65	2.64 ± 0.64	2.36 ± 0.19
FeatMatch (Kuo et al. 2020)	–	–	3.34 ± 0.19†	3.10 ± 0.06†
FlexMatch (Zhang et al. 2021)	6.15 ± 0.25	20.81 ± 5.26	17.32 ± 2.07	12.90 ± 2.68
CR-Match	**4.89** ± 0.17	**2.79** ± 0.93	**2.35** ± 0.29	**2.08** ± 0.07

Per class labels	Mini-ImageNet	
	40 labels	100 labels
Mean Teacher (Tarvainen and Valpola 2017)	72.51 ± 0.22	57.55 ± 1.11
Label Propagation (Iscen et al. 2019)	70.29 ± 0.81	57.58 ± 1.47
PLCB (Arazo et al. 2020)	56.49 ± 0.51	46.08 ± 0.11
FeatMatch (Kuo et al. 2020)	*39.05* ± 0.06	*34.79* ± 0.22
CR-Match	**34.87** ± 0.99	**32.58** ± 1.60

A Wide ResNet-28-2 and a Wide ResNet-37-2 (Zagoruyko and Komodakis 2016) is used for SVHN and STL-10, repectively. The same code base is adopted as Sohn et al. (2020) to make the results directly comparable. Notations follow Table 1. *Right:* error rates on Mini-ImageNet with 40 labels and 100 labels per class. All methods are evaluated on the same ResNet-18 architecture

*Numbers are generated by Sohn et al. (2020)

†Numbers are produced without CutOut. The best number is in bold and the second best number is in italic

4.1 Main Results

In the following, each dataset subsection includes two paragraphs. The first provides technical details and the second discusses experimental results.

CIFAR-10, CIFAR-100, and SVHN We follow prior work (Sohn et al. 2020) and use 4, 25, and 100 labels per class on CIFAR-100 and SVHN without extra data. For CIFAR-10, we experiment with settings of 4, 25, and 400 labels per class. We create labeled data by random sampling, and the remaining images are regarded as unlabeled by discarding their labels. Following Berthelot et al. (2019), Sohn et al. (2020) and Berthelot et al. (2020), we use a Wide ResNet-28-2 (Zagoruyko and Komodakis 2016) with 1.5M parameters on CIFAR-10 and SVHN, and a Wide ResNet-28-8 with 135 filters per layer (26M parameters) on CIFAR-100.

As shown in Tables 1 and 2, our method improves over previous methods across all settings, and defines a new state-of-the-art. Most importantly, we improve error rates in low data regimes by a large margin (e.g., with 4 labeled examples per class on CIFAR-100, we outperform FlexMatch and the second best method by 10.19 and 8.56% in absolute value respectively). Prior works (Sohn et al. 2020; Berthelot et al. 2019, 2020) have reported results using a larger network architecture on CIFAR-100 to obtain better performance. On the contrary, we additionally evaluate our method on the small network used in CIFAR-10 and find that our method is more than 17 times ($17 \approx 26/1.5$) parameter-efficient than FixMatch. We reach 46.05% error rate on CIFAR-100 with 4 labels per class using the small model, which is still slightly better than the result of FixMatch using a larger model.

STL-10 STL-10 contains 5000 labeled images of size 96-by-96 from 10 classes and 100,000 unlabeled images. The dataset pre-defines ten folds of 1000 labeled examples from the training data, and we evaluate our method on five of these ten folds as in Sohn et al. (2020) and Berthelot et al. (2020). Following Berthelot et al. (2019), we use the same Wide ResNet-37-2 model (comprising 5.9M parameters), and report error rates in Table 2.

Our method achieves state-of-the-art performance with 4.89% error rate. Note that FixMatch with error rate 5.17% used the more advanced CTAugment (Berthelot et al. 2020), which learns augmentation policies alongside model training. When evaluated with the same data augmentation (RandAugment) as we use in CR-Match, our result surpasses FixMatch by 3.09% (3.09% = 7.98−4.89%), which indicates that CR-Match itself induces a strong regularization effect.

Mini-ImageNet We follow Iscen et al. (2019), Arazo et al. (2020) and Kuo et al. (2020) to construct the mini-ImageNet training set. Specifically, 50,000 training examples and 10,000 test examples are randomly selected for a predefined list of 100 classes (Ravi and Larochelle 2017) from

Table 3 Error rates on ImageNet after 2^{20} iterations

Method	Top-1	Top-5
FixMatch (Sohn et al. 2020)	43.66*	21.80*
FlexMatch (Zhang et al. 2021)	41.85*	19.48*
CR-Match§	**40.69**	**18.44**

CR-Match§ refers to CR-Match combined with CPL (Zhang et al. 2021) from FlexMatch
*Numbers are from Zhang et al. (2021)
The best number is in bold

ILSVRC (Deng et al. 2009). Following Kuo et al. (2020), we use a ResNet-18 network (He et al. 2016) as our model and experiment with settings of 40 labels per class and 100 labels per class.

As shown in Table 2, our method consistently improves over previous methods and achieves a new state-of-the-art in both the 40-label and 100-label settings. Especially in the 40-label case, CR-Match achieves an error rate of 34.87% which is 4.18% higher than the second best result. Note that our method is 2 times more data efficient than the second best method FeatMatch (Kuo et al. 2020) (FeatMatch, using 100 labels per class, reaches a similar error rate as our method with 40 labeled examples per class).

ImageNet To verify the effectiveness of our method on large scale datasets, we conduct experiments on ImageNet-1k. Following Zhang et al. (2021), we take $\sim 10\%$ (100,000) training images as the labeled set and construct unlabeled set using the rest of the images. The validation setting remains the same. We train a ResNet-50 (He et al. 2016) with the same hyper-parameters from Zhang et al. (2021). Note that FixMatch and FlexMatch use different protocols on ImageNet, and we follow the setup from FlexMatch therefore the numbers are directly comparable.

Table 3 shows the error rate comparison after running 2^{20} iterations. Our method outperforms the previous state-of-the-art by 1.04% absolute top-5 error rate, which demonstrates the efficacy of the proposed method at large scale dataset.

4.2 Ablation Study

In this section, we analyze how FeatDistLoss and RotNet influence the performance across different settings, particularly when there are few labeled samples. We conduct experiments on a single split on CIFAR-10, CIFAR-100, and SVHN with 4 labeled examples per class, and on MiniImageNet with 40 labels per class. Specifically, we remove the $\mathcal{L}_{Dist}$ from Eq. (4) and train the model again using the same training scheme for each setting. We do not ablate $\mathcal{L}_{Pseudo}$ and $\mathcal{L}_{S}$ due to the fact that removing one of them leads to a divergence of training.

We report final test error rates in Table 4. We see that both RotNet and FeatDistLoss contribute to the final performance while their proportions can be different depending on the setting and dataset. For MiniImageNet, CIFAR-100 and SVHN, the combination of both outperforms the individual losses. For CIFAR-10, FeatDistLoss even outperforms the combination of both. This suggests that RotNet and FeatDistLoss are both important components for CR-Match to achieve the state-of-the-art performance. Note that RotNet can be replaced by other types of self-supervision as well. We opt RotNet due to its superior performance in our initial experiments. On CIFAR-100 with 4 labels per class, CRMatch with SimCLR achieves an error rate of 42.50% compared to that of 39.22% from CRMatch with RotNet. More details of the experiment are provided in the supplementary material.

Figure 4 shows a more detailed analysis of the training process on CIFAR-100 with 4 labels per class for CR-Match and CR-Match without FeatDistLoss. The confidence threshold in CR-Match filters out unconfident predictions during training. Therefore, at each training step only images with confidence scores above the threshold contribute to the loss. We observe that CR-Match improves pseudo-labels for the unlabeled data, as it achieves a lower error rate of all unlabeled images as well as contributing unlabeled images during the training while maintaining the percentage of contributing images. The increasing of the pseudo-label error rate in Fig. 4 middle is due to the increasing of the percentage of contributing pseudo-labels and the prediction confidence. At the beginning of the training, the contributing pseudo-labels are mostly correct as only a small number of samples are highly confident and, thus, selected. However, during the course of the training, the overall prediction confidence increases, resulting in more unlabeled data being used, which introduces more errors in pseudo-labels.

Effect of different confidence thresholds For the main results in Sect. 4.1, we use a confidence threshold of 0.95 following Sohn et al. (2020). We now study the model robustness against different confidence thresholds. Experiments are conducted on a single split with 4 labeled examples from CIFAR-100 on a Wide ResNet-28-2. Figure 5 shows the error rate of CR-Match when using a confidence threshold from 0.90 to 0.99. In general, the thresholding mechanism provides the model a relatively smooth transition between learning from labeled data and learning from unlabeled data. A low percentage of the contributing unlabeled data at the beginning of the training can alleviate the potential error introduced by the low-quality pseudo-labels. This suggests that the quality of pseudo-labels is more important than the quantity for reaching a high accuracy at the early stage. As the model learns from the labeled data, the error rate of the pseudo-label decreases, and the model becomes more confident about its predictions. Then, the number of unlabeled data that contribute to the final loss gradually increases, which

Table 4 Ablation studies across different settings

RotNet	FeatDistLoss	MiniImageNet@40	CIFAR10@4	CIFAR100@4	SVHN@4
		35.13	11.86	46.22	2.42
	✓	34.14	**10.33**	43.48	2.34
✓		34.64	11.27	41.48	2.21
✓	✓	**33.82**	10.92	**39.22**	**2.09**

The best number is in bold
Error rates are reported for a single split

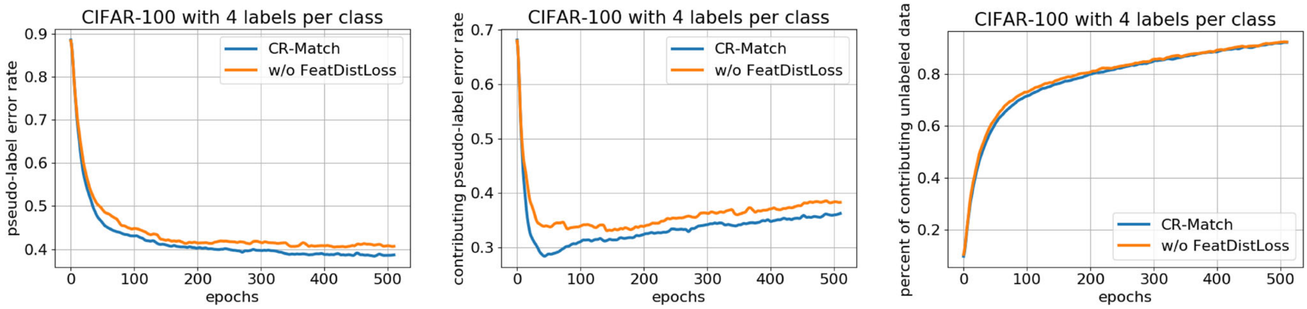

Fig. 4 Ablation study of our best model on CIFAR-100 with 4 labels per class. *Left:* CR-Match has a lower pseudo-label error rate. *Middle:* if only the confident predictions are taken into account, CR-Match outperforms the other with a even larger margin in terms of pseudo-label error rate. *Right:* in spite of a better pseudo-label error rate on contributing unlabeled images, the percentage of contributing unlabeled images is maintained the same for CR-Match

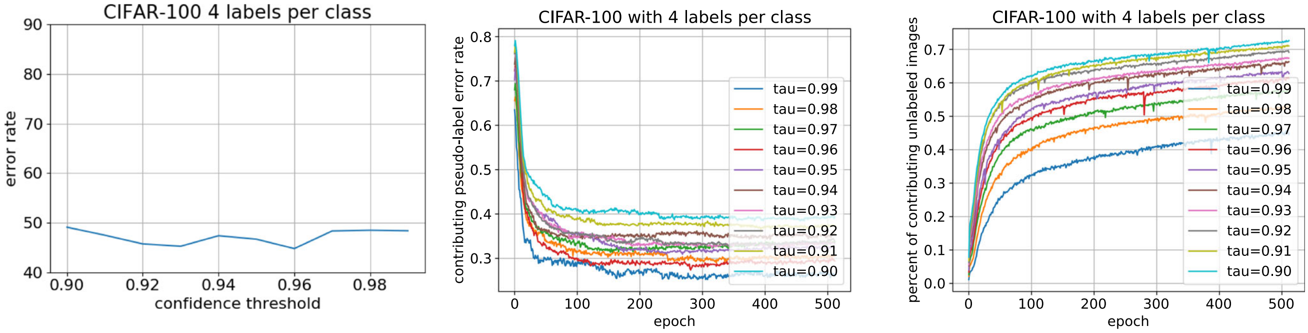

Fig. 5 *Left:* effect of different confidence thresholds on error rate. We run experiments on a single split of CIFAR-100 with 4 labels per class. The model is a Wide-ResNet-28-2. Our model shows good robustness against small changes in the confidence threshold. *Middle:* effect of different confidence thresholds on pseudo-label error rate during the training. *Right:* effect of different confidence thresholds on the number of unlabeled training samples

allows the model to continue learning from unlabeled data. Figure 5 left also implies that our model is quite robust against small changes in the confidence threshold.

4.3 Influence of Feature Distance Loss

In this section, we analyze different design choices for FeatDistLoss to provide additional insights of how it helps generalization. We focus on a single split with 4 labeled examples from CIFAR-100 and report results for a Wide ResNet-28-2 (Zagoruyko and Komodakis 2016). For fair comparison, the same 4 random labeled examples for each class are used across all experiments in this section.

Different distance metrics for FeatDistLoss Here we discuss the effect of different metric functions d for FeatDistLoss. Specifically, we compare two groups of functions in Table 5: metrics that increase the distance between features, including cosine similarity, negative JS divergence, and L2 similarity (i.e. normalized negative L2 distance); metrics that decrease the distance between features, including cosine distance, JS divergence, and L2 distance. We find that both increasing and decreasing distance between features of different augmentations give reasonable perfor-

Table 5 Effect of different distance functions for FeatDistLoss

Metric		Error rate
Impose equivariance	Cosine similarity	**45.52**
	L_2 similarity	46.22
	Negative JS div.	46.46
Impose invariance	Cosine distance	46.98
	L_2 distance	48.74
	JS divergence	47.48
CR-Match w/o FeatDistLoss		48.89

The best number is in bold

The same split on CIFAR-100 with 4 labels per class and a Wide ResNet-28-2 is used for all experiments. Metrics that pull features together performs worse than those that push features apart. The error rate of CR-Match without FeatDistLoss is shown at the bottom

Table 6 Error rates of binary classification (whether a specific augmentation is applied) on the features from CR-Equiv (increasing the cosine distance) and CR-Inv (decreasing the cosine distance)

Transformations	Feature extractor	
	CR-Equiv	CR-Inv
Translation	**33.22** ± 0.28	36.80 ± 0.30
Scaling	**11.09** ± 0.66	14.87 ± 0.40
Rotation	**15.05** ± 0.33	21.92 ± 0.32
ColorJittering	**31.04** ± 0.50	35.99 ± 0.27

We evaluate translation, scaling, rotation, and color jittering. Lower error rate indicates more equivariant features. Results are averaged over 10 runs

mance. However, increasing the distance always performs better than the counterpart (e.g., cosine similarity is better than cosine distance). We conjecture that decreasing the feature distance corresponds to an increase of the invariance to data augmentation and leads to ignorance of information like rotation or translation of the object. In contrast, increasing the feature distance while still imposing the same label makes the representation equivariant to these augmentations, resulting in more descriptive and expressive representation with respect to augmentation. Moreover, a classifier has to cover a broader space in the feature space to recognize rather dissimilar images from the same class, which leads to improved generalization. In summary, we found that both increasing and decreasing feature distance improve over the model which only applies consistency on the classifier level, whereas increasing distances shows better performance by making representations more equivariant. Please refer to the supplementary material for experiments of combining both invariant and equivariant loss in FeatDistLoss.

Invariance and equivariance Here we provide an additional analysis to demonstrate that increasing the feature distance provides equivariant features while the other provides invariant features. Based on the intuition that specific transformations of the input image should be more predictable from equivariant representations, we quantify the equivariance by how accurate a linear classifier can distinguish between features from augmented and original images. Specifically, we compare two models from Table 5: the model trained with cosine similarity denoted as *CR-Equiv* and the model trained with cosine distance denoted as *CR-Inv*. We train a linear SVM to predict whether a certain transformation is applied for the input image. 1000 test images from CIFAR-100 are used for training and the rest (9000) for validation. The binary classifier is trained by an SGD optimizer with an initial learning rate of 0.001 for 50 epochs, and the feature extractor is fixed during training. We evaluate translation, scaling, rotation, and color jittering in Table 6. All augmentations are from the standard PyTorch library. The SVM has a better error rate across all augmentations when trained on CR-Equiv features, which means information like object location or orientation is more predictable from CR-Equiv features, suggesting that CR-Equiv produces more equivariant features than CR-Inv. Furthermore, if the SVM is trained to classify strongly and weakly augmented image features, CR-Equiv achieves a 0.27% test error while CR-Inv is 46.18%.

Regularization on the classifier level As we described in Sect. 3, FeatDistLoss contains two levels of regularization: On the feature level, representations are encouraged to become more equivariant. On the classifier level, the same class label is imposed on different versions of the same image via pseudo-labeling. Here we provide more insights into the regularization on the classifier level in Table 7. Specifically, we conduct experiments on replacing or complementing the CE loss with Jensen–Shannon divergence. First, we can see that removing the classifier loss and using only the equivariant loss on the feature level leads to a significant drop on performance (from 45.52% to 91.53%). This is because $\mathcal{L}_{Dist}$ alone will just make the model aware of the difference between augmentations but does not help the classifier to distinguish between classes of unlabeled data, making the classifier unable to benefit from the usage of unlabeled data. Thus, the performance is on par with the model trained on labeled data only (91.28% error rate) Second, complementing the cross-entropy loss on the classifier level with Jensen-Shannon divergence, improves the performance (45.01%) while replacing it leads to inferior performance (76.83%).

Different data augmentations for FeatDistLoss. In our main results in Sect. 4.1, FeatDistLoss is computed between features generated by weak augmentation and strong augmentation. Here we investigate the impact of FeatDistLoss with respect to different types of data augmentations. Specifically, we evaluate the error rate of CR-Inv and CR-Equiv under three augmentation strategies: weak-weak pair indicates that FeatDistLoss uses two weakly augmented images, weak-strong pair indicates that FeatDistLoss uses a weak

Table 7 Effect of different regularization techniques on the classifier level

Classifier level	Feature level	Error rate
None	None	91.28
None	Equiv.	91.53
CE	Equiv.	**45.52**
JSD	Equiv.	76.83
CE + JSD	Equiv.	**45.01**

The best number is in bold

CE denotes cross-entropy loss. JSD denotes Jensen-Shannon divergence. Equiv. denotes the equivariance version of $\mathcal{L}_{Dist}$. Note that the chance level is 99%. None + None represents the model trained with labeled data only. The same split on CIFAR-100 with 4 labels per class and a Wide ResNet-28-2 is used for all experiments

Table 8 Effect of combinations of weak and strong augmentation in FeatDistLoss on a Wide ResNet-28-2 for CR-Inv and CR-Equiv

Error rate	CR-Inv	CR-Equiv
Weak-weak	48.88	48.51
Weak-strong	**46.98**	**45.52**
Strong-strong	48.57	48.05

The best number is in bold

augmentation and a strong augmentation, and strong-strong pair indicates that FeatDistLoss uses two strongly augmented images.

As shown in Table 8, using either CR-Inv or CR-Equiv using weak-strong pairs conistently outperforms the other augmentation settings (weak-weak and strong-strong). Additionally, CR-Equiv consistently achieves better generalization performance across all three settings. In particular, in the case advocated in this paper, namely using weak-strong pairs, CR-Equiv outperforms CR-Inv by 1.46%. Even in the other two settings, CR-Equiv leads to improved performance even though only by a small margin. This suggests that, on the one hand, that it is important to use different types of augmentations for our FeatDistLoss. And on the other hand, maximizing distances between images that are inherently different while still imposing the same class label makes the model more robust against changes in the feature space and thus gives better generalization performance.

Linear projection and confidence threshold in FeatDistLoss As mentioned in Sect. 3, we apply $\mathcal{L}_{Dist}$ at (a) in Fig. 3 with a linear layer mapping the feature from the encoder to a low-dimensional space before computing the loss, to alleviate the curse of dimensionality. Also, the loss only takes effect when the model's prediction has a confidence score above a predefined threshold τ. Here we study the effect of other design choices in Table 9. While features after the global average pooling (i.e. (b)) gives a better result than the ones directly from the feature extractor, (b) performs worse than (a) when additional projection heads are added. Thus, we use features from the feature extractor in CR-Match.

The error rate increases from 45.52 to 48.37 and 47.52% when removing the linear layer and replacing the linear layer by a MLP (two fully-connected layers and a ReLU activation function), respectively. This suggests that a lower dimensional space serves better for comparing distances, but a non-linear mapping does not give further improvement. Moreover, when we apply FeatDistLoss for all pairs of input images by removing the confidence threshold, the test error increases from 45.52 to 46.94%, which suggests that regularization should be only performed on features that are actually used to update the model parameters, and ignoring those that are also ignored by the model.

FeatDistLoss improves decision boundaries As suggested by Fig. 2, models trained with FeatDistLoss tend to have improved decision boundaries. Here we take two models from Sect. 4.2, CR-Match (39.22% error rate) and CR-Match without FeatDistLoss (41.48% error rate), and plot t-SNE plots of features extracted from unlabeled images. As shown in Fig. 6, CR-Match with FeatDistLoss produces better separation between classes. For example, CR-Match forms two clearer clusters for caterpillar and butterfly, while CR-Match without FeatDistLoss mostly mixes them up. Another example is that the overlap between crab, bowl, and pear is much less for CR-Match compared to CR-Match without FeatDistLoss. Moreover, the improved decision boundaries also lead to better per-class error rate. The standard deviation of per-class error rates for CR-Match is 4.34% lower than that from CR-Match without FeatDistLoss (30.83% v.s. 26.49%).

Additional analysis on FeatDistLoss. To further verify the importance of FeatDistLoss, we show in Fig. 7 the contribution of FeatDistLoss compared to other losses. The model is CR-Equiv. trained on CIFAR-100 with 4 labels per class. We can see that during the training, the two components of FeatDistLoss, $\mathcal{L}_{Dist}$ and $\mathcal{L}_{PseudoLabel}$, account for a large portion of the overall loss, thus, the gradient. Note that $\mathcal{L}_{Dist}$ is the negative cosine distance, thus, ranging from 1 to -1.

5 Experiments on Imbalanced SSL

In this section, we go beyond the standard setting and evaluate the efficacy of our method under imbalanced SSL settings where both labeled and unlabeled data follow class imbalanced distributions. We first present the problem setup of imbalanced SSL. Then, we introduce the construction of the datasets before showing the final evaluation results.

Problem setup and notations For a K-class classification problem, there is a labeled set $\mathcal{X} = \{(\mathbf{x}_n, y_n) : n \in (1, ..., N)\}$ and an unlabeled set $\mathcal{U} = \{\mathbf{u}_m : m \in (1, ..., M)\}$, where $\mathbf{x}_n, \mathbf{u}_m \in \mathbb{R}^d$ are training examples and $y_n \in \{1, ..., K\}$ are class labels for labeled examples. N_k and

Table 9 Effect of the projection head z, and the place to apply $\mathcal{L}_{Dist}$

Features taken from Fig. 3 at	Feature	Feature + linear	Feature + MLP
(a)	48.37	**45.52**	47.52
(b)	47.37	46.10	47.15

The best number is in bold
(a) denotes un-flattened features taken from the feature extractor directly
(b) denotes features after the global average pooling. MLP has 2 FC layers and a ReLU. Removing the linear projection head harms the test error, and a non-linear projection head does not improve the performance further

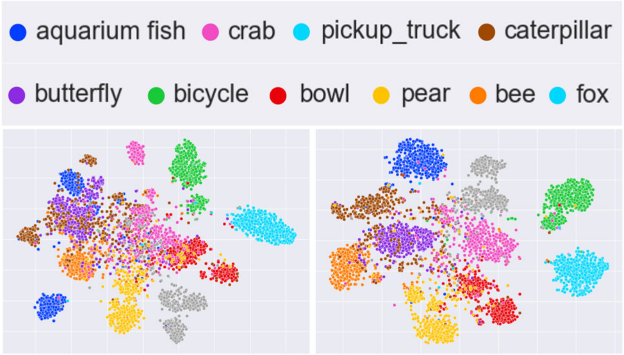

Fig. 6 We plot t-SNE of input image features extracted by a CR-Match model trained without FeatDistLoss (left) and a CR-Match model with it (right). The better separation from CR-Match suggests that FeatDistLoss improves decision boundaries

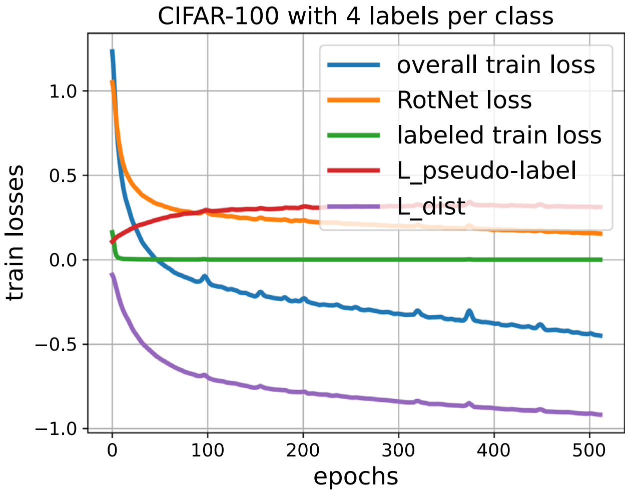

Fig. 7 The amount of the contribution of the regularization term in the loss. The model is CR-Equiv. trained on CIFAR-100 with 4 labels per class

M_k denote the numbers of labeled and unlabeled examples in class k, respectively, i.e., $\sum_{k=1}^{K} N_k = N$ and $\sum_{k=1}^{K} M_k = M$. Without loss of generality, we assume the classes are sorted by the number of training samples in descending order, i.e., $N_1 \geq N_2 \geq ... \geq N_k$. The goal is to train a classifier $f : \mathbb{R}^d \rightarrow \{1, ..., K\}$ on $\mathcal{X} \cup \mathcal{U}$ that generalizes well on a class-balanced test set.

Datasets We consider three common datasets in the field to evaluate the efficacy of CRMatch for imbalanced SSL: CIFAR10-LT (Krizhevsky and Hinton 2009), CIFAR100-LT (Krizhevsky and Hinton 2009), and Semi-Aves (Su and Maji 2021).

For CIFAR-10-LT and CIFAR100-LT, we follow the convention (Kim et al. 2020; Wei et al. 2021) and randomly select some training images for each class determined by a pre-defined imbalance ratio γ as the labeled and the unlabeled set. Specifically, we set $N_k = N_1 \cdot \gamma^{-\frac{k-1}{K-1}}$ for labeled data and $M_k = M_1 \cdot \gamma^{-\frac{k-1}{K-1}}$ for unlabeled data. We use $N_1 = 1500$; $M_1 = 3000$ for CIFAR-10 and $N_1 = 150$; $M_1 = 300$ for CIFAR-100, respectively. Following Kim et al. (2020) and Wei et al. (2021), we report results with imbalance ratio $\gamma = 50$, 100 and 150 for CIFAR10-LT and $\gamma = 20$, 50 and 100 for CIFAR100-LT. Therefore, the number of labeled samples for the least class is 10 and 1 for CIFAR-10 with $\gamma = 150$ and CIFAR-100 with $\gamma = 100$, respectively.

Semi-Aves is a subset of bird species from the Aves kingdom of the iNaturalist 2018 dataset. There are 200 in-class and 800 out-of-class categories. The dataset consists of a labeled set L_{in} with 3,959 labeled images, an in-class unlabeled set U_{in} with 26,640 images, an out-of-class unlabeled set U_{out} with 122,208 images, a validation set L_{val} of 2,000 images, and 8,000 test images. The training data in L_{in}, U_{in}, and U_{out} has imbalanced distributions, specifically L_{in} has 5 to 43 images and U_{in} has 16 to 229 images per class. The validation data and test data have a uniform distribution with 40 and 10 images per class, repectively. In our experiments, we use L_{in} or $L_{in} \cup L_{val}$ as the labeled set and U_{in} as the unlabeled set. We do not use unlabeled images from U_{out} since out-of-class unlabeled images are found empirically harmful to the final performance (Oliver et al. 2018) and making good use of out-of-class unlabeled images is out of the scope of this paper. More details on the class distribution can be found in Su and Maji (2021).

Implementation details Due to the performance superiority of $\mathcal{L}_{U-equiv}$ over $\mathcal{L}_{U-inv}$, we use CR-Equiv throughout this section. For all experiments in this section, we use the same hyper-parameters and design choices from the CIFAR experiments in Sect. 4.1. We deploy FixMatch (Sohn et al. 2020) as the base SSL method due to its superiority under

Table 10 Class-wise precision and recall (%) on the balanced test set of CIFAR-10-LT

Class index		1	2	3	4	5	6	7	8	9	10	Avg.
Recall	FixMatch + CReST+	**98.6**	99.3	85.8	77.4	84.4	63.4	77.2	55.5	37.4	34.6	71.3
	CR-Match + CReST+	**98.6**	**99.6**	**88.8**	**82.5**	**86.7**	**67.8**	**78.7**	**57.0**	**42.7**	**40.6**	**74.3**
Precision	FixMatch + CReST+	53.3	61.4	71.4	57.9	77.0	82.4	93.0	**97.1**	97.6	**98.0**	78.9
	CR-Match + CReST+	**56.1**	**62.9**	**75.1**	**63.7**	**78.6**	**83.3**	**94.5**	**97.1**	**98.1**	97.1	**80.7**

The best number is in bold

Models are trained with imbalance ratio $\gamma = 150$

Table 11 Classification accuracy (%) on CIFAR-10-LT using a Wide ResNet-28-2 under the uniform test distribution of three different class-imbalance ratios γ

	CIFAR-10-LT		
	γ=50	γ=100	γ=150
Vanilla	65.2±0.05*	58.8±0.13*	55.6±0.43*
Long-tailed recognition methods			
Re-sampling (Japkowicz 2000)	64.3±0.48*	55.8±0.47*	52.2±0.05*
LDAM-DRW (Cao et al. 2019)	68.9±0.07*	62.8±0.17*	57.9±0.20*
cRT (Kang et al. 2020)	67.8±0.13*	63.2±0.45*	59.3±0.10*
SSL methods			
FixMatch (Sohn et al. 2020)	81.58 ± 0.34	74.74 ± 1.35	70.04 ± 0.77
ReMixMatch (Berthelot et al. 2020)	82.79 ± 0.17	76.81 ± 0.23	72.53 ± 1.16
FlexMatch (Zhang et al. 2021)	81.89 ± 0.25	74.94 ± 0.96	70.09 ± 0.42
CR-Match	82.87 ± 0.04	76.54 ± 0.87	72.14 ± 0.76
FixMatch + DARP (Kim et al. 2020)	82.46 ± 0.30	76.51 ± 0.50	71.88 ± 1.02
ReMixMatch + DARP (Kim et al. 2020)	82.88 ± 0.23	76.77 ± 0.29	72.90 ± 0.95
FlexMatch + DARP (Kim et al. 2020)	81.93 ± 0.22	74.84 ± 0.66	70.46 ± 0.58
CR-Match + DARP	83.22 ± 0.27	77.32 ± 0.29	73.44 ± 0.06
FixMatch + CReST+ (Wei et al. 2021)	82.25 ± 0.08	76.31 ± 0.23	71.70 ± 0.83
ReMixMatch + CReST+ (Wei et al. 2021)	83.71 ± 0.17	79.13 ± 0.19	75.17 ± 0.31
FlexMatch + CReST+ (Wei et al. 2021)	82.75 ± 0.25	77.23 ± 0.35	72.21 ± 0.11
CR-Match + CReST+	84.11 ± 0.32	78.55 ± 0.55	74.21 ± 0.11
FixMatch + CoSSL (Fan et al. 2022)	86.63 ± 0.24	83.10 ± 0.48	80.15 ± 0.59
ReMixMatch + CoSSL (Fan et al. 2022)	*87.55 ± 0.06*	*84.15 ± 0.65*	*81.28 ± 0.95*
FlexMatch + CoSSL (Fan et al. 2022)	86.30 ± 0.30	81.61 ± 0.74	78.80 ± 0.73
CR-Match + CoSSL (Fan et al. 2022)	**88.11 ± 0.17**	**84.80 ± 0.54**	**82.29 ± 0.33**

The numbers are averaged over 5 different folds. We use the same code base as Kim et al. (2020) for fair comparison following Oliver et al. (2018). Numbers with * are taken from the original papers. The best number is in bold and the second best number is in italic

the standard SSL settings. A Wide ResNet-28-2 (Zagoruyko and Komodakis 2016) is used as the backbone as recommended by Oliver et al. (2018). We base our implementation on the public codebases of each methods. Therefore, method-specific hyper-parameters follow the same as in their original papers (Kim et al. 2020; Wei et al. 2021). For example, all experiments on CIFAR-LT are trained with batch size 64 using Adam optimizer (Kingma and Ba 2015) with a constant learning rate of 0.002 without any decay. We train the models for 500 epochs, each of which has 500 steps, resulting in a total number of 2.5×10^5 training iterations. On Semi-Aves, we follow the hyper-parameters from Oh et al. (2022). For example, the models are trained for 90 epochs with a batch size of 256, and the optimizer is SGD with a learning rate of 0.04. For all experiments, we report the average test accuracy of the last 20 epochs following Oliver et al. (2018).

Results on CIFAR-10 and CIFAR-100 Tables 11 and 12 compare our method with various SSL algorithms and long-tailed recognition algorithms on CIFAR-10-LT and CIFAR-100-LT with various imbalance ratios γ. Adding our method shows improved performance in most of settings. Our method combining with CoSSL (Fan et al. 2022) achieves the best or comparable performance across all settings. In particular, CRMatch + CoSSL outperforms others at large imbalance ratios (82.29% v.s. the second best 81.28% on CIFAR-10 at imbalance ratio $\gamma = 150$), which indi-

Table 12 Classification accuracy (%) on CIFAR-100-LT under the uniform test distribution of three different class-imbalance ratios γ

	CIFAR-100-LT		
	γ = 20	γ = 50	γ = 100
FixMatch (Sohn et al. 2020)	49.58 ± 0.90	42.10 ± 0.38	37.46 ± 0.48
ReMixMatch (Berthelot et al. 2020)	51.46 ± 0.51	44.37 ± 0.62	39.29 ± 0.59
FlexMatch (Zhang et al. 2021)	51.00 ± 0.75	42.86 ± 0.42	37.20 ± 0.51
CR-Match	52.03 ± 0.42	44.37 ± 0.57	39.32 ± 0.31
FixMatch + DARP (Kim et al. 2020)	50.89 ± 0.86	43.12 ± 0.61	38.19 ± 0.47
ReMixMatch + DARP (Kim et al. 2020)	51.95 ± 0.40	45.24 ± 0.46	39.50 ± 0.58
FlexMatch + DARP (Kim et al. 2020)	50.78 ± 0.71	42.81 ± 0.36	36.99 ± 0.66
CR-Match + DARP	49.33 ± 0.32	44.13 ± 0.38	39.18 ± 0.80
FixMatch + CReST+ (Wei et al. 2021)	51.87 ± 0.11	45.25 ± 0.06	40.41 ± 0.35
ReMixMatch + CReST+ (Wei et al. 2021)	51.22 ± 0.38	45.91 ± 0.33	41.24 ± 0.79
FlexMatch + CReST+ (Wei et al. 2021)	51.16 ± 0.63	43.12 ± 0.57	38.09 ± 0.58
CR-Match + CReST+	53.77 ± 0.36	46.44 ± 0.58	40.94 ± 0.43
FixMatch + CoSSL (Fan et al. 2022)	53.99 ± 0.87	47.78 ± 0.53	42.87 ± 0.61
ReMixMatch + CoSSL (Fan et al. 2022)	**55.92 ± 0.69**	**49.10 ± 0.59**	44.10 ± 0.68
FlexMatch + CoSSL (Fan et al. 2022)	53.46 ± 0.79	46.83 ± 0.80	41.42 ± 0.58
CR-Match + CoSSL (Fan et al. 2022)	*55.34 ± 0.43*	*48.83 ± 0.87*	**44.21 ± 0.61**

The numbers are averaged over 5 different folds. We reproduce all numbers using the same codebase from Kim et al. (2020) for a fair comparison. The best number is in bold and the second best number is in italic

Table 13 Classification accuracy (%) on Semi-Aves under the uniform test distribution

	Semi-Aves	
	$\mathcal{X} = L_{in} \cup L_{val}$	$\mathcal{X} = L_{in}$
FixMatch (Sohn et al. 2020)	53.15	42.46
ReMixMatch (Berthelot et al. 2020)	51.28	40.10
FlexMatch (Zhang et al. 2021)	52.78	43.50
CRMatch	*54.53*	44.42
FixMatch + CoSSL (Fan et al. 2022)	54.15	*44.58*
ReMixMatch + CoSSL (Fan et al. 2022)	54.13	43.97
FlexMatch + CoSSL (Fan et al. 2022)	53.98	44.09
CRMatch + CoSSL (Fan et al. 2022)	**54.90**	**45.81**

L_{train} and $L_{in} \cup L_{val}$ have imbalance ratio $\gamma \approx 9$ and $\gamma \approx 4$, respectively. The best number is in bold and the second best number is in italic

cates the superiority of our method in handling severe dataset imbalance.

To analyze how the improvement is obtained, we compare the class-wise precision and recall of CReST+ and CReST+ with our method in Table 10. Both models are trained with imbalance ratio $\gamma = 150$ on CIFAR-10-LT using the same data split. The class indices are sorted according to the number of samples in descending order, i.e., class 1 has the largest number of data. For CReST+, the head classes tend to have higher precision but lower recall while the tail classes have lower precision but higher recall. By adding our method, the recall on the tail classes can be significantly improved without sacrificing much precision, which leads to the overall better performance. Similarly, the precision of the head classes is improved while the recall remains at the same level.

Results on Semi-Aves As Semi-Aves is naturally imbalanced ($\gamma \approx 9$ and 4 for L_{train} and $L_{in} \cup L_{val}$, respectively), we compare CRMatch with other methods using different numbers of labeled data. We report the raw performance of backbone algorithms as well as the performance with CoSSL (Fan et al. 2022) considering its superior performance on CIFAR-10-LT and CIFAR-100-LT. From Table 13, we can see that CRMatch outperforms other backbone algorithms by a large margin in both settings. While CoSSL leads to improvement in all methods, CRMatch still achieves the best performance, which demonstrates the effectiveness of our method in realistic settings.

6 Conclusion

The idea of consistency regularization gives rise to many successful works for SSL (Bachman et al. 2014; Laine and Aila 2017; Sajjadi et al. 2016; Sohn et al. 2020; Xie et al. 2020; Kuo et al. 2020). While making the model invariant against input perturbations induced by data augmentation gives improved performance, the scheme tends to be suboptimal when augmentations of different intensities are used. In this work, we propose a simple yet effective improvement, called FeatDistLoss. It introduces consistency regularization on both the classifier level, where the same class label is imposed for versions of the same image, and the feature level, where distances between features from augmentations of different intensities is increased. By encouraging the representation to distinguish between weakly and strongly augmented images, FeatDistLoss encourages more equivariant representations, leading to improved classification boundaries, and a more robust model.

Through extensive experiments we show the superiority of our training framework, and define a new state-of-the-art on both standard and imbalanced semi-supervised learning benchmarks. Particularly, our method outperforms previous methods in low data regimes by significant margins, e.g., on CIFAR-100 with 4 annotated examples per class, our error rate (39.45%) is 4.83% better than the second best (44.28%). In future work, we are interested in integrating more prior knowledge and stronger regularization into SSL to further push the performance in low data regimes.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11263-022-01723-4.

Funding Open Access funding enabled and organized by Projekt DEAL.

Declarations

Conflict of interest The authors have no competing interests to declare that are relevant to the content of this article.

References

Arazo, E., Ortego, D., Albert, P., O'Connor, N. E., & McGuinness, K. (2020). *Pseudo-labeling and confirmation bias in deep semi-supervised learning*. In *International joint conference on neural networks (IJCNN)*. IEEE.

Bachman, P., Alsharif, O., & Precup, D. (2014). Learning with pseudo-ensembles. In *Advances in neural information processing systems*.

Bachman, P., Hjelm, R. D, & Buchwalter, W. (2019). Learning representations by maximizing mutual information across views. In *Advances in neural information processing systems*.

Bardes, A., Ponce, J., & LeCun, Y. (2022). VICReg: Variance-invariance-covariance regularization for self-supervised learning. In *International conference on learning representations*.

Bellman, R. (1966). Dynamic programming. *Science, 153*(3731), 34–37.

Bengio, Y., Delalleau, O., & Le Roux, N. (2006). 11 label propagation and quadratic criterion, 193–216.

Berthelot, D., Carlini, N., Cubuk, E. D., Kurakin, A., Sohn, K., Zhang, H., & Raffel, C. (2020). Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring. In *8th international conference on learning representations*. ICLR.

Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A., & Raffel, C. A. (2019). Mixmatch: A holistic approach to semi-supervised learning. In *Advances in neural information processing systems*.

Cao, K., Wei, C., Gaidon, A., Arechiga, N., & Ma, T. (2019). Learning imbalanced datasets with label-distribution-aware margin loss. In *Advances in neural information processing systems*.

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., & Joulin, A. (2020). Unsupervised learning of visual features by contrasting cluster assignments. In *Advances in neural information processing systems*.

Chapelle, O., Scholkopf, B., & Zien, A. (2009). Semi-supervised learning (Chapelle, O. et al., eds.; 2006) [book reviews]. *IEEE Transactions on Neural Networks*, *20*(3), 542.

Chen, X., & He, K. (2021). Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020a). A simple framework for contrastive learning of visual representations. arXiv preprint arXiv:2002.05709.

Chen, T., Kornblith, S., Swersky, K., Norouzi, M., & Hinton, G. (2020b) Big self-supervised models are strong semi-supervised learners. arXiv preprint arXiv:2006.10029.

Coates, A., Ng, A., & Lee, H. (2011). An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the 14th international conference on artificial intelligence and statistics*.

Cohen, T. S., & Welling, M. (2016). Steerable CNNs. arXiv preprint arXiv:1612.08498.

Cohen, T.,& Welling, M. (2016). Group equivariant convolutional networks. In *International conference on machine learning*.

Cubuk, E. D., Zoph, B., Shlens, J., & Le, Q. V. (2020) Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *IEEE conference on computer vision and pattern recognition*.

Denton, E., Gross, S., & Fergus, R. (2016). Semi-supervised learning with context-conditional generative adversarial networks. arXiv preprint arXiv:1611.06430.

DeVries, T., & Taylor, G. W. (2017). Improved regularization of convolutional neural networks with cutout. arXiv preprint arXiv:1708.04552.

Fan, Y., Dai, D., Kukleva, A., & Schiele, B. (2022). Cossl: Co-learning of representation and classifier for imbalanced semi-supervised learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

Fan, Y., Kukleva, A., & Schiele, B. (2021). Revisiting consistency regularization for semi-supervised learning. In *Pattern recognition: 43nd DAGM German conference, DAGM GCPR 2021*.

French, G., Oliver, A., & Salimans, T. (2020). Milking cowmask for semi-supervised image classification. CoRR arXiv:2003.12022

Gidaris, S., Singh, P., & Komodakis, N. (2018). Unsupervised representation learning by predicting image rotations. In *International conference on learning representations*.

Gong, C., Wang, D., & Liu, Q. (2021). Alphamatch: Improving consistency for semi-supervised learning with alpha-divergence. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., et al. (2020). Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems, 33*, 21271–212184.

He, K., Fan, H., Wu, Y., Xie, S., & Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

He, J., Kortylewski, A., Yang, S., Liu, S., Yang, C., Wang, C., & Yuille, A. (2021). Rethinking re-sampling in imbalanced semi-supervised learning. arXiv preprint arXiv:2106.00209.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Hinton, G. E., Sabour, S., & Frosst, N. (2018). Matrix capsules with em routing. In *International conference on learning representations*.

Hyun, M., Jeong, J., & Kwak, N. (2020). Class-imbalanced semi-supervised learning. arXiv preprint arXiv:2002.06815.

Iscen, A., Tolias, G., Avrithis, Y., & Chum, O. (2019). Label propagation for deep semi-supervised learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Japkowicz, N. (2000). The class imbalance problem: Significance and strategies. In *Proceedings of the international conference on artificial intelligence*.

Joachims, T. (2003). Transductive learning via spectral graph partitioning. In *Proceedings of the 20th international conference on machine learning (ICML)*.

Kang, B., Xie, S., Rohrbach, M., Yan, Z., Gordo, A., Feng, J., & Kalantidis, Y. (2020). Decoupling representation and classifier for long-tailed recognition. In *International conference on learning representations*.

Kim, J., Hur, Y., Park, S., Yang, E., Hwang, S., & Shin, J. (2020). Distribution aligning refinery of pseudo-label for imbalanced semi-supervised learning. In *Advances in neural information processing systems*.

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *International conference on learning representations*.

Kingma, D. P., Mohamed, S. , Rezende, D. J., & Welling, M. (2014). Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*.

Krizhevsky, A., & Hinton, G. (2009). *Learning multiple layers of features from tiny images*. Toronto: ON, Canada.

Kuo, C.-W., Ma, C.-Y., Huang, J.-B., & Kira, Z. (2020). Featmatch: Feature-based augmentation for semi-supervised learning. In *Computer vision—ECCV 2020* (pp. 1–19).

Laine, S., & Aila, T. (2017). Temporal ensembling for semi-supervised learning. In *5th international conference on learning representations*. ICLR.

Lee, D.-H. (2013). *Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks*. ICML: In Workshop on challenges in representation learning.

Lee, H., Shin, S., & Kim, H. (2021). Abc: Auxiliary balanced classifier for class-imbalanced semi-supervised learning. *Advances in Neural Information Processing Systems, 34*, 7082–7094.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision*.

Liu, B., Wu, Z., Hu, H., Lin, S. (2019). Deep metric transfer for label propagation with limited annotated data. In *Proceedings of the IEEE international conference on computer vision workshops*.

Luo, Y., Zhu, J., Li, M., Ren, Y., & Zhang, B. (2018). Smooth neighbors on teacher graphs for semi-supervised learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Miyato, T., Maeda, S.-I., Koyama, M., & Ishii, S. (2018). Virtual adversarial training: A regularization method for supervised and semi-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 41*(8), 1979–1993.

Nesterov, Y. (1983). *A method of solving a convex programming problem with convergence rate* $o(k^2)$. Doklady Akademii Nauk.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., & Ng, A. Y. (2011). Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*.

Odena, A. (2016). Semi-supervised learning with generative adversarial networks. arXiv preprint arXiv:1606.01583.

Oh, Y., Kim, D.-J., & Kweon. I. S. (2022). Distribution-aware semantics-oriented pseudo-label for imbalanced semi-supervised learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

Oliver, A., Odena, A., Raffel, C. A., Cubuk, E. D., & Goodfellow, I. (2018). Realistic evaluation of deep semi-supervised learning algorithms. In *Advances in neural information processing systems*.

Pham, H., Xie, Q., Dai, Z., & Le, Q. V. (2020). Meta pseudo labels. arXiv preprint arXiv:2003.10580.

Rasmus, A., Berglund, M., Honkala, M., Valpola. H., & Raiko. T. (2015). Semi-supervised learning with ladder networks. In *Advances in neural information processing systems*.

Ravi, S., & Larochelle, H. (2017). Optimization as a model for few-shot learning. In *5th international conference on learning representations*. ICLR.

Rebuffi, S.-A., Ehrhardt, S., Han, K., Vedaldi, A., & Zisserman, A. (2020). Semi-supervised learning with scarce annotations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., et al. (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision, 115*(3), 211–252.

Sabour, S., Frosst, N., & Hinton, G. E. (2017). Dynamic routing between capsules. In *Advances in neural information processing systems*.

Sajjadi, M., Javanmardi, M., & Tasdizen, T. (2016). Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *Advances in neural information processing systems*.

Scudder, H. (1965). Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory, 11*(3), 363–371.

Sohn, K., Berthelot, D., Li, C.-L., Zhang, Z., Carlini, N., Cubuk, E. D., Kurakin, A., Zhang, H., & Raffel, C. (2020). Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *Advances in neural information processing systems*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research, 15*(1), 1929–1958.

Su, J.-C., & Maji, S. (2021). The semi-supervised iNaturalist-Aves challenge at FGVC7 workshop. arXiv preprint arXiv:2103.06937.

Tarvainen, A., & Valpola, H. (2017). Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in neural information processing systems*.

Verma, V., Lamb, A., Kannala, J., Bengio, Y., & Lopez-Paz, D. (2019). Interpolation consistency training for semi-supervised learning. In *Proceedings of the 28th international joint conference on artificial intelligence*. IJCAI.

Weiler, M., & Cesa, G. (2019). General e(2)-equivariant steerable CNNs. In *Advances in neural information processing systems*.

Wei, C., Sohn, K., Mellina, C., Yuille, A., & Yang, F. (2021). Crest: A class-rebalancing self-training framework for imbalanced semi-supervised learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

Xie, Q., Dai, Z., Hovy, E., Luong, T., & Le, Q. (2020). Unsupervised data augmentation for consistency training. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., & Lin, H. (eds.) *Advances in neural information processing systems*, 33, 6256–6268.

Zagoruyko, S., & Komodakis, N. (2016). Wide residual networks. In *Proceedings of the British machine vision conference (BMVC)*.

Zhai, X., Oliver, A., Kolesnikov, A., & Beyer, L. (2019) S4l: Self-supervised semi-supervised learning. In *Proceedings of the IEEE international conference on computer vision*.

Zhang, H., Cisse, M., Dauphin, Y. N., & Lopez-Paz, D. (2018). mixup: Beyond empirical risk minimization. In *6th international conference on learning representations*. ICLR.

Zhang, B., Wang, Y., Hou, W., Wu, H., Wang, J., Okumura, M., & Shinozaki, T. (2021). Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. *Advances in Neural Information Processing Systems, 34*, 18408–18419.

Zhu, X. (2005). *Semi-supervised learning literature survey* (Technical Report 1530, Computer Sciences, University of Wisconsin-Madison).

Zhu, X., & Goldberg, A. B. (2009). Introduction to semi-supervised learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning, 3*(1), 1–130.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.