# Image-Feature Weak-to-Strong Consistency: An Enhanced Paradigm for Semi-Supervised Learning (Appendix)

Zhiyu Wu and Jinshi Cui

National Key Laboratory of General Artificial Intelligence,
School of Intelligence Science and Technology, Peking University
wuzhiyu@pku.edu.cn, cjs@cis.pku.edu.cn

## 1 Feature-Level Perturbation Strategies

As mentioned in Sec.3.2 of the main paper, we develop a series of feature-level perturbation strategies from different perspectives, including 'dropout' (comprising channel-wise dropout and spatial-wise dropout), 'movement' (encompassing translation and shearing in both the X-axis and Y-axis), and 'value' (involving the weighted sum of the smoothed and input feature maps). In this section, we will elaborate on the implementation of these strategies. For the sake of simplicity, we maintain a consistent notation for all perturbation strategies.

$$f^{out} = \Phi(f^{in}) \tag{1}$$

$$f^{out}, f^{in} \in R^{C \times H \times W} \tag{2}$$

where $f^{in}$ and $f^{out}$ denote the input and output feature maps, $\Phi$ stands for the perturbation operation, $C$ represents the number of channels, $H$ and $W$ signifies the height and width of the feature maps.

### 1.1 Channel-Wise Dropout

This perturbation strategy applies channel-wise dropout with a drop probability of 0.5 to hidden representations. Mathematically, the feature-level perturbation using channel-wise dropout can be defined as follows.

$$f^{out} = \text{torch.nn.Dropout2d}(p = 0.5)(f^{in}) \tag{3}$$

### 1.2 Spatial-Wise Dropout

Spatial-wise dropout discards a randomly selected rectangle region across all channels, similar to the Cutout operation applied to raw images. To determine the dropped spatial region, the strategy performs the following steps. Firstly, we compute the height $h_d$ and width $w_d$ of the dropped region by rescaling the size of input feature maps. Subsequently, we randomly select the upper left point $(x, y)$ of the dropped area. With the size and the location of its top-left point in

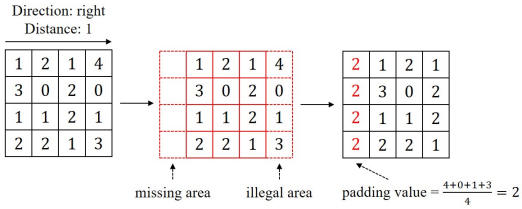



**Fig. 1:** A toy sample for feature-level translation perturbation.

place, we can uniquely define the dropped area. Mathematically speaking, the above process can be expressed as follows.

$$h, w = \text{int}(\alpha \times H), \text{int}(\alpha \times W) \tag{4}$$

$$x, y = \mathcal{U}[0, H - h], \mathcal{U}[0, W - w] \tag{5}$$

In all experiments, we set $\alpha$ to 0.5. Once the dropped spatial region is determined, we generate a 0-1 mask denoted as $m \in R^{H \times W}$ and implement spatial-wise dropout by multiplying the mask with the input feature maps. Formally, the output feature maps can be computed as follows.

$$m(i, j) = \begin{cases} 0 & \textit{if } (i, j) \textit{ in } \textit{Rectangle}(x, y, h, w) \\ 1 & \textit{otherwise} \end{cases} \tag{6}$$

$$f^{out} = f^{in} \odot m \tag{7}$$

where $\odot$ denotes the element-wise product. Note that a similar rescaling approach is applied to the remaining feature values as in channel-wise dropout.

### 1.3 Translation

The feature-level translation perturbation randomly selects a direction from the candidate set (up, down, left, and right) with equal probability and translates the input feature map along the determined direction. The length of the translation path denoted as $l$ can be computed by rescaling the size of the input feature maps. Specifically, we first draw a random factor $\alpha$ from a uniform distribution ranging from 0 to $\alpha_{max}$ and then compute the translation distance. Taking the translation along the X-axis as an example, the above process can be mathematically described as follows.

$$\alpha \sim \mathcal{U}[0, \alpha_{max}] \tag{8}$$

$$l = \text{int}(\alpha \times W) \tag{9}$$

Throughout all experiments, we set $\alpha_{max}$ to 0.5. Given the translation direction and distance, the strategy assigns the corresponding value from the input feature maps to the output feature maps. Furthermore, to address the areas left

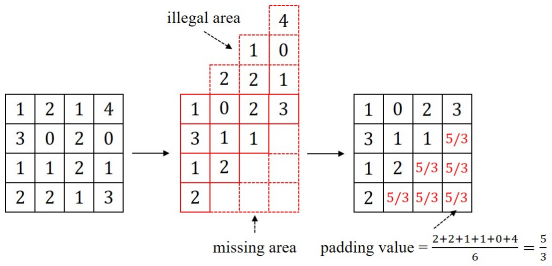



**Fig. 2:** A toy sample for feature-level shearing perturbation.

vacant due to the translation operation, we utilize the average value outside the legal region as padding, thereby avoiding significant information loss and ensuring numerical stability. For a more intuitive understanding of the translation operation, please refer to the toy example presented in Fig. 1.

### 1.4 Shearing

Similar to translation, the shearing operation randomly selects one direction from a set of candidate directions, each with equal probability. The feature map is then sheared along the chosen direction. The length of the shearing path denoted as $l$ can be calculated by rescaling the size of the input feature maps, and intermediate distances can be obtained by linear interpolation. Specifically, we first draw a random factor $\alpha$ from a uniform distribution between 0 and $\alpha_{max}$ and then compute the shearing distance. Taking the shearing operation in the X-axis as an illustrative example, the above process can be expressed as follows.

$$\alpha \sim \mathcal{U}[0, \alpha_{max}] \tag{10}$$

$$l = \text{int}(\alpha \times W) \tag{11}$$

$$[l_1, \cdots, l_W] = \text{torch.linaspace}(0, l, W) \tag{12}$$

In all experiments, we set $\alpha_{max}$ to 1.0. Subsequently, the strategy assigns corresponding values from the input feature maps to the output feature maps and employs the average value of the illegal region to pad the missing areas. Fig. 2 provides an example of the shearing operation.

### 1.5 Value Modification

The value modification operation aims to change the feature value while preserving the relationship within each feature map. To achieve this goal, we first employ a random-sized group convolution to facilitate local smoothing in each feature map. The random-sized configuration allows for various degrees of smoothness, thus comprehensively exploring the feature perturbation space. The size of the kernel $k$ is randomly sampled from the set of all odd numbers within the range

of 3 to $\min(H, W)$ with equal probability. Subsequently, we adopt the average smoothing convolution to encode the input feature maps. Finally, we draw a random factor $\alpha$ from a uniform distribution between $\alpha_{min}$ and $\alpha_{max}$ and use the weighted sum between the input and smoothed feature maps as the perturbation results. Mathematically speaking, the above process can be expressed as follows.

$$k \sim \{\mathcal{U}[3, \min(H, W)], \text{odd number}\} \tag{13}$$

$$K[i, j] = \frac{1}{k^2} \tag{14}$$

$$\alpha \sim \mathcal{U}[\alpha_{min}, \alpha_{max}] \tag{15}$$

$$f^{out} = \alpha \times (K * f^{in}) + (1 - \alpha) \times f^{in} \tag{16}$$

where $*$ denotes the convolution operation. In all experiments, we set $\alpha_{min}$ and $\alpha_{max}$ to 0.50 and 0.95, respectively.

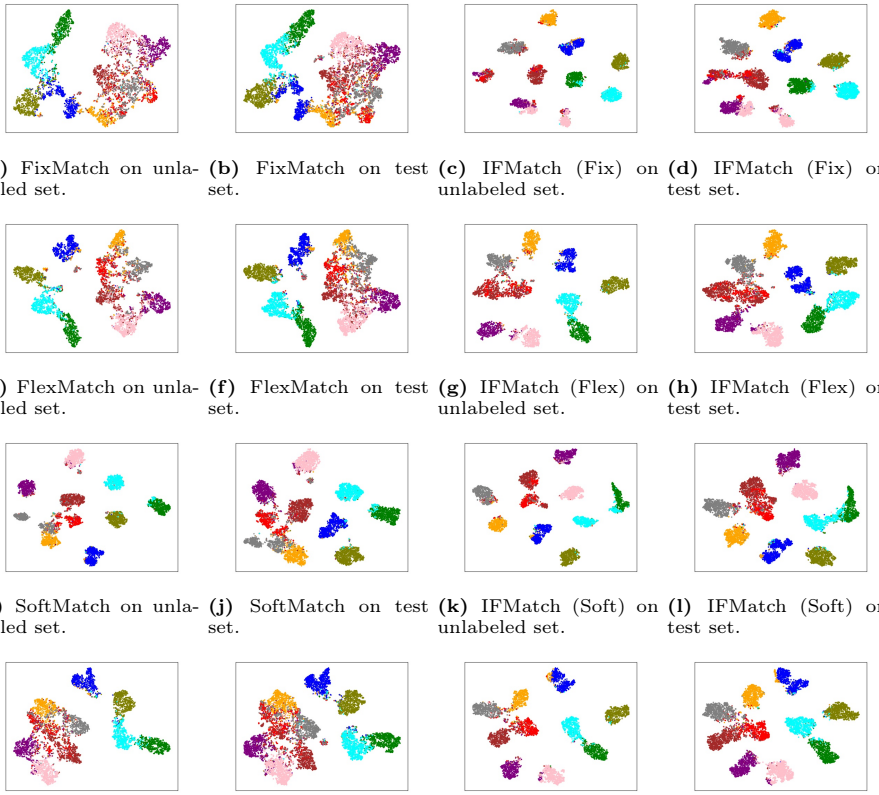
### 1.6 Conclusion for Strategies

The aforementioned perturbation strategies consider three distinct perspectives ('dropout', 'movement', and 'value'), thereby achieving diverse perturbation forms and providing a collective exploration of the feature perturbation space. Moreover, the parameter settings for each perturbation strategy are relatively simple. For example, the first four perturbation strategies all involve one single hyperparameter, and we set it to 0.5 on the first three, indicating satisfactory consistency. Note that the configuration setting $\alpha_{max}$ to 1.0 in the shearing operation leads to the same upper bound (half the size of the feature map) for the missing area as the translation operation. Additionally, the value modification operation introduces two hyperparameters, $\alpha_{min}$ and $\alpha_{max}$, and our settings keep consistent with the counterparts in strong image-level perturbation.

## 2 Visualization Analysis

### 2.1 High-Dimensional Features

To diagnose the proposed paradigm in a more intuitive manner, we visualize the high-dimensional features on STL-10 with 40 labeled samples using T-SNE. The results are presented in Fig. 3, wherein the first two and last two columns respectively depict the performance of algorithms following the old and proposed paradigms. To elaborate, algorithms adhering to the conventional approach typically generate loose and adjacent feature clusters. In contrast, the proposed paradigm achieves more separable and tightly clustered features, indicating the effectiveness of feature-level perturbation. Moreover, traditional algorithms often overfit noisy pseudo-labels, while the proposed approach positions most ambiguous samples near the decision boundary, minimizing their impacts on the model. Overall, the proposed image-level weak-to-strong consistency paradigm (IFMatch) generates easy-to-distinguish features, laying a solid foundation for a robust classifier.



**(a)** FixMatch on unlabeled set. **(b)** FixMatch on test set. **(c)** IFMatch (Fix) on unlabeled set. **(d)** IFMatch (Fix) on test set.

**(e)** FlexMatch on unlabeled set. **(f)** FlexMatch on test set. **(g)** IFMatch (Flex) on unlabeled set. **(h)** IFMatch (Flex) on test set.

**(i)** SoftMatch on unlabeled set. **(j)** SoftMatch on test set. **(k)** IFMatch (Soft) on unlabeled set. **(l)** IFMatch (Soft) on test set.

**(m)** FreeMatch on unlabeled set. **(n)** FreeMatch on test set. **(o)** IFMatch (Free) on unlabeled set. **(p)** IFMatch (Free) on test set.

**Fig. 3:** Feature visualization of four algorithms when following the old and proposed paradigms on STL-10 with 40 labeled samples.

### 2.2 Image-Level Perturbation and Feature-Level Perturbation

We present the visualization for image-level and feature-level perturbations in Fig. 4, which serves as the supplement for the counterpart in the main paper. The four parts (from left to right) in Fig. 4 correspond to the raw images, the images subjected to $\mathcal{A}^{\mathcal{I}_s}$, and the feature maps that undergo $\mathcal{A}^{\mathcal{F}_w} \circ \mathcal{A}^{\mathcal{I}_s}$ and $\mathcal{A}^{\mathcal{F}_s} \circ \mathcal{A}^{\mathcal{I}_w}$, respectively. As we can observe, a significant proportion of samples in the second part can be effortlessly classified, indicating the naive sample issue caused by the exclusive reliance on $\mathcal{A}^{\mathcal{I}_s}$. In contrast, the combination of $\mathcal{A}^{\mathcal{I}}$ and $\mathcal{A}^{\mathcal{F}}$ proposed by our paradigm effectively expand the perturbation space, thereby boosting the utilization of unlabeled samples.

**Table 1:** Running speed (sec/iter) analysis for different configurations of the proposed paradigm. We use FixMatch's threshold in the second student branch, *i.e.* the IFMatch (Fix) model.

| $\mathcal{A}^{\mathcal{F}_s}$ in Branch I | $\mathcal{A}^{\mathcal{F}_w}$ in Branch II | CIFAR-10-40 | CIFAR-100-400 | ImageNet-100k |
|---|---|---|---|---|
| | | 0.1011 | 0.2058 | 0.3914 |
| | ✓ | 0.1029 | 0.2092 | 0.3983 |
| ✓ | | 0.1114 | 0.2596 | 0.4357 |
| ✓ | ✓ | 0.1140 | 0.2640 | 0.4423 |

**Table 2:** Performance on Cityscapes. Feature map translation and shearing ('movement') are excluded from $\mathcal{A}^{\mathcal{F}}$ in segmentation tasks. The first line provides the proportion of labeled samples to the total number of available samples.

| Method (ResNet-50) | 1/16 | 1/8 | 1/4 | 1/2 |
|---|---|---|---|---|
| $U^2$PL | 70.6 | 73.0 | 76.3 | 77.2 |
| UniMatch | 75.0 | 76.8 | 77.5 | 78.6 |
| IFMatch (Fix) | **76.3** | **77.5** | **78.3** | **79.0** |

## 3 Running Speed Analysis

The proposed paradigm demonstrates substantial performance promotion when compared to the traditional approach. However, the obtained improvement is accompanied by feature-level perturbation and a triple-branch structure, inevitably resulting in a reduction in training speed. Consequently, we conduct a running speed analysis for our paradigm. The evaluated training speed is provided in Tab. 1. The baseline model, as presented in line 1, follows the old paradigm. Moreover, the adaptive weak feature-level perturbation $\mathcal{A}^{\mathcal{F}_w}$ (line 2) in the second student branch introduces negligible costs, indicating the efficiency of implementing feature-level perturbation. Additionally, the first student branch (line 3 and line 4) leverages strong feature-level perturbation $\mathcal{A}^{\mathcal{F}_s}$ to comprehensively explore the feature perturbation space, albeit incurring non-trivial costs. Overall, the proposed paradigm significantly improves the model's performance at the expense of acceptable additional computational costs.

## 4 Performance on Semi-Supervised Semantic Segmentation

In addition to semi-supervised classification, we also test the performance of the proposed paradigm on the semi-supervised semantic segmentation task. As shown in Tab. 2, IFMatch (Fix) consistently outperforms $U^2$PL and UniMatch with varying numbers of labeled samples. The impressive performance on different tasks demonstrates the effectiveness and generalization capability of our approach.

**Table 3:** Detailed training settings for balanced SSL.

| Datasets | CIFAR-10 | CIFAR-100 | SVHN | STL-10 | ImageNet |
|---|---|---|---|---|---|
| Backbone | WRN-28-2 | WRN-28-8 | WRN-28-2 | WRN-37-2 | ResNet-50 |
| Weight Decay | 5e-4 | 1e-3 | 5e-4 | 5e-4 | 3e-4 |
| $B_L$ / $B_U$ | 64 / 448 | | | | 128 / 128 |
| Initial Learning Rate | 0.03 | | | | |
| Learning Rate Scheduler | $\eta = \eta_0 cos(\frac{7\pi k}{16K})$ | | | | |
| SGD Momentum | 0.9 | | | | |
| Model EMA | 0.999 | | | | |
| $\lambda_u$ | 1.0 | | | | |
| $\mathcal{A}^{\mathcal{I}_w}$ / $\mathcal{A}^{\mathcal{I}_s}$ | Random Crop, Random Horizontal Flipping / RandAug | | | | |
| $\tau$ | 0.95 | | | | |

**Table 4:** Detailed training settings for imbalanced SSL.

| Datasets | CIFAR-10-LT | CIFAR-100-LT |
|---|---|---|
| Backbone | WRN-28-2 | |
| Weight Decay | 4e-5 | |
| $B_L$ / $B_U$ | 64 / 128 | |
| Initial Learning Rate | 2e-3 | |
| Learning Rate Scheduler | $\eta = \eta_0 cos(\frac{7\pi k}{16K})$ | |
| Optimizer | Adam | |
| Model EMA | 0.999 | |
| $\lambda_u$ | 1.0 | |
| $\mathcal{A}^{\mathcal{I}_w}$ / $\mathcal{A}^{\mathcal{I}_s}$ | Random Crop, Random Horizontal Flipping / RandAug | |
| $\tau$ | 0.95 | |

## 5 Implementation Details

The detailed training settings for balanced and imbalanced semi-supervised learning are presented in Table 3 and Table 4, respectively.

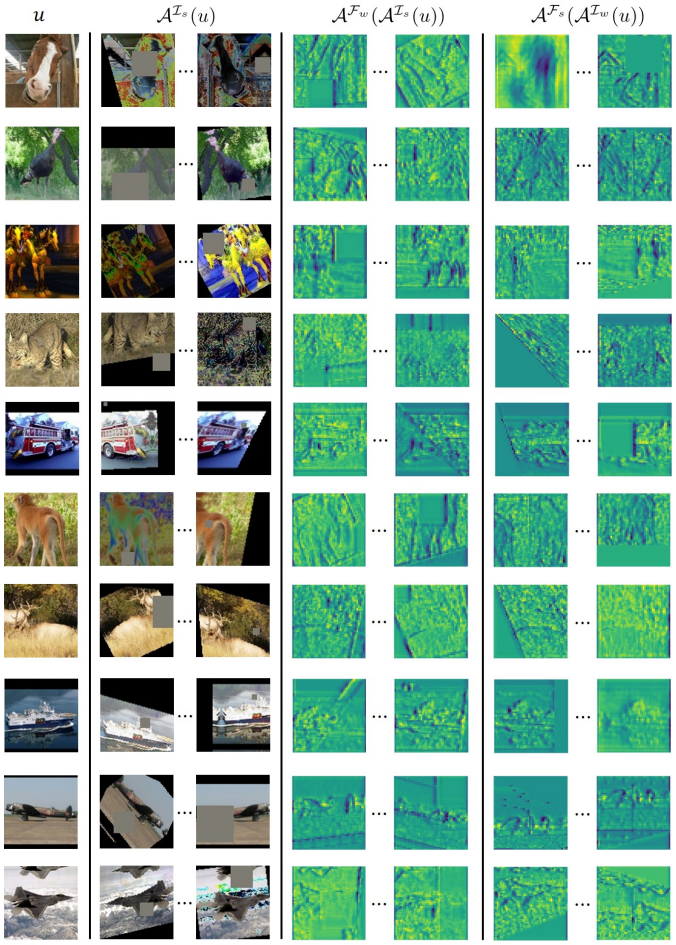



**Fig. 4:** Visualization for image-level and feature-level perturbations. We provide the feature maps for samples that undergo feature-level perturbation.