Supplementary Materials

Chia-Wen Kuo[†], Chih-Yao Ma[†], Jia-Bin Huang[‡], Zsolt Kira[†],
[†]Georgia Tech, [‡]Virginia Tech

albert.cwkuo@gatech.edu, cyma@gatech.edu, jbhuang@vt.edu, zkira@gatech.edu

In this supplementary document, we include additional experimental results and implementation details to complement the main paper. We will release the source code and pre-trained model for facilitating future research if accepted.

A State-of-the-art Results with other SSL Techniques

As we build upon a weaker baseline and our method is much simpler, the performance of our method on the CIFAR-10 dataset is slightly worse in some settings. However, as we claimed in Section 4.2 of the main paper, our proposed feature-based augmentation method is complementary to conventional image-based augmentation methods and can be easily integrated to further improve the performance. In Table 1, we demonstrate that by incorporating (1) distribution alignment that aligns the marginal class distribution as described in [1, 2], and (2) *Cutout* [5], an image-based augmentation method, our method indeed compares favorably against current state-of-the-art algorithms. Note that our method is still simpler when compared to state-of-the-art image-based method, e.g., ReMixMatch [2]. For example, the ReMixMatch method also incorporates self-supervsied loss, temporal ensembling of model weights, and tailored data augmentation method (CTAugment [2]), etc.

Table 1: Comparison to other state-of-the-art methods after incorporating some other modern SSL techniques (distribution alignment and Cutout). We show the results on the CIFAR-10 dataset with varying amounts of labeled samples. Numbers represent error rate across three runs.

	#Labeled samples		
Method	250	1,000	4,000
MixMatch [3]	11.08 ± 0.87	7.75 ± 0.32	6.24 ± 0.06
ReMixMatch [2]	**6.27 ± 0.34**	5.73 ± 0.16	5.14 ± 0.04
FeatMatch (Ours in the main paper)	7.50 ± 0.64	5.76 ± 0.07	**4.91 ± 0.18**
FeatMatch (Ours with other SSL techniques)	**6.00 ± 0.41**	**5.21 ± 0.08**	**4.64 ± 0.11**

B Pseudo-Labeling Accuracy Before and After *AugF*

In Section 4.4 of the main paper, we analyze other reasons that $AugF$ improves model performance. We conclude that our proposed $AugF$ module also learns to refine input feature for a better representation by attending to the prototypes. This feature refinement process by $AugF$ provides the training objectives of $\mathcal{L}_{con\text{-}g}$ (Eq. 5) and $\mathcal{L}_{con\text{-}f}$ (Eq. 6) with better pseudo-labels, which may be one of the reasons why our method can improve over image-based baseline by a larger margin. In Fig. 1 below, we can see that the accuracy of pseudo-labels from the features refined by $AugF$ is higher than those without refinement by $AugF$.

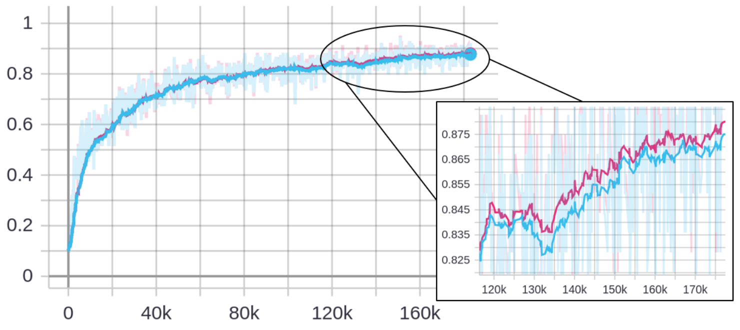

Fig. 1: We monitor the accuracy of pseudo-labeling with feature-base refinement (red curve) and without feature-based refinement (blue curve) during training. We found that the pseudo-label from the refined feature (red) has on average $0.5 - 1.0\%$ higher accuracy.

C More Analysis on Prototypes

We test the sensitivity of our method for the hyper-parameter p_k (number of prototypes per class) and I_p (the interval at which a new set of prototypes is extracted). The analysis is conducted on a held-out validation set of the CIFAR-10 dataset with 250 labels. As shown in Table 2, the final results are stable across different values of p_k. We choose the number of prototypes per class $p_k = 20$ in our method as it performs slightly better than others and has a slightly lower variance. In Table 3, we can see that the final results are also stable across different I_p. Therefore, for simplicity, we extract prototypes every epoch, which is approximately the same as $I_p = 400$.

D More Results on the DomainNet Setting

In Section 4.1, we propose a practical setting where the unlabeled data may come from other domains. We show results with different r_u, the ratio of unlabeled

Table 2: Sensitivity analysis for p_k. Numbers represent error rates in three runs.

$p_k = 1$	$p_k = 5$	$p_k = 10$	$p_k = 20$
8.14 ± 0.79	8.15 ± 0.19	8.01 ± 0.90	8.09 ± 0.58

Table 3: Sensitivity analysis for I_p. Numbers represent error rates in three runs.

$I_p = 200$	$I_p = 400$	$I_p = 600$	$I_p = 800$
8.00 ± 0.81	7.99 ± 0.74	7.99 ± 0.79	8.38 ± 1.21

data coming from the target Real domain or the shifted domains, in Section 4.2. In this section, we show additional results of $r_u = 0.25$ and $r_u = 0.75$ on both our method and the image-based baseline in Tab. 4. The results show a similar trend is similar as the Table 3 in the main paper, where the accuracy goes down as r_u goes up. Our method consistently improves over image-based semi-supervised baseline. Our method achieves comparable result even in the severe case of $r_u = 75\%$ against the image-based baseline method with clean unlabeled data of $r_u = 0\%$

Table 4: Comparison between the image-based baseline with our proposed feature-based augmentation method on DomainNet with various r_u, the ratio of unlabeled data coming from the shifted domains. For instance, $r_u = 25\%$ means 25% of the unlabeled data are coming from the shifted domains and 75% are coming from the domain same as the labeled set. Numbers are error rates across 3 runs, meaning the lower the better.

Method (5% labeled samples)	$r_u = 0\%$	$r_u = 25\%$	$r_u = 50\%$	$r_u = 75\%$
(Semi-supervised) Baseline	56.63 ± 0.17	62.44 ± 0.67 %	65.82 ± 0.07	70.50 ± 0.51
FeatMatch (Ours)	**40.66 ± 0.60**	**46.11 ± 1.15**	**54.01 ± 0.66**	**58.30 ± 0.93**
Supervised baseline (5% labeled samples, lower bound)		77.25 ± 0.52		
Supervised baseline (100% labeled samples, upper bound)		31.91 ± 0.15		

E Implementation Details

E.1 Training

We train our model with Stochastic Gradient Descent and Nesterov momentum. As the *AugF* module heavily relies on the feature representation to compute attention weights, we pre-train the model without *AugF* for 4 epochs.

We adapt the super convergence learning rate scheduler [6] to reduce the total training iterations. Specifically, in the pre-training stage, the learning rate starts from 4e-4 and linearly increase to 4e-3 in I_p iterations. After the pre-training stage, we add the *AugF* module and ramp up the learning rate linearly from 4e-3 to 4e-2 in I_c iterations, and then ramp down back to 4e-3 in another I_c iterations. In the meantime, the momentum ramps down from 0.95 to 0.85, and then ramps up back to 0.95. Finally, in the convergence stage, the learning rate ramps further down from 4e-3 to 4e-6 in I_e iterations.

We follow the guidelines in [6] to set these parameters without aggressive parameter tuning, and set $I_p = 3k$, $I_c = 75k$, and $I_e = 30k$. As the DomainNet setting has more training samples, we increase these values $I_p = 4k$, $I_c = 100k$, and $I_e = 40k$ without tuning. We only tune the peak learning rate to be 4e-4 on a held-out validation set on CIFAR-10 with 250 labels.

E.2 Hyper-parameters

All the hyper-parameters are tuned on a *held-out validation set* on CIFAR-10 with 250 labels. These hyper-parameters are shared across all settings and experiments without further tuning. Since our method is built upon the image-based baseline, we fix the hyper-parameters or select a reasonable value without tuning from the original papers.

Table 5: Hyper-parameters and their meanings.

Hyper-parameter	Description	Value
p_k	Number of prototypes per class	20
I_p	The interval at which a new set of prototypes are extracted	1 epoch
a_h	Number of attention heads in *AugF*	4
λ_g	Loss weight for $\mathcal{L}_{con\text{-}g}$	0.5
λ_f	Loss weight for $\mathcal{L}_{con\text{-}f}$	2.0
b_l	Batch size for labeled data	64
b_u	Batch size for unlabeled data	128
wd	Weight decay	2e-4

E.3 Data Augmentation Operations

We used the same sets of image transformations used in RandAugment [4]. There are two parameters in RandAugment: (1) N – number of operations applied, and (2) M – maximal magnitude of the applied augmentation. We use $N = 2$ as in RandAugment, and set M to its max value without tuning. Note that the magnitude is randomly sampled from $[-M, M]$.

References

1. Arazo, E., Ortego, D., Albert, P., O'Connor, N.E., McGuinness, K.: Pseudo-labeling and confirmation bias in deep semi-supervised learning. arXiv preprint arXiv:1908.02983 (2019)
2. Berthelot, D., Carlini, N., Cubuk, E.D., Kurakin, A., Sohn, K., Zhang, H., Raffel, C.: Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. In: Proc. International Conference on Learning Representations (ICLR) (2020)
3. Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A., Raffel, C.A.: Mixmatch: A holistic approach to semi-supervised learning. In: Advances in Neural Information Processing Systems. pp. 5050–5060 (2019)
4. Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.V.: Randaugment: Practical automated data augmentation with a reduced search space. arXiv preprint arXiv:1909.13719 (2019)
5. DeVries, T., Taylor, G.W.: Improved regularization of convolutional neural networks with cutout. arXiv preprint arXiv:1708.04552 (2017)
6. Smith, L.N., Topin, N.: Super-convergence: Very fast training of neural networks using large learning rates. In: Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications. vol. 11006, p. 1100612. International Society for Optics and Photonics (2019)