Supplementary Material:
Hybrid Consistency Training with Prototype Adaptation for Few-Shot Learning

Meng Ye Xiao Lin Giedrius Burachas Ajay Divakaran Yi Yao
SRI International
{meng.ye, xiao.lin, giedrius.burachas, ajay.divakaran, yi.yao}@sri.com

A. CIPA for transductive inference

In this section we show more results on how σ and N_{iter} in CIPA algorithm affect the final performances. We did a grid search on $\sigma \in \{0.0, 0.1, ..., 1.0\}$ and $N_{iter} \in \{1, 2, 5, 10, 20, 50\}$. For each setting we evaluate the performance on 600 *mini*-ImageNet 1-shot tasks and report all their mean accuracy in Tab. 1. From the results we observe that, generally, the performance increases as larger σ and N_{iter} values are used. For larger σ values, the accuracy saturates fast in a few iterations while for smaller σ it needs more iterations to converge. In our main results, we did not exhaustively tune these two parameters and used fixed values of $\sigma = 0.2$ and $N_{iter} = 20$. Using a different set of σ and N_{iter} tuned for each experiment can potentially lead to further improved accuracy.

σ \ N_{iter}	1	2	5	10	20	50
0.0	65.96	65.96	65.96	65.96	65.96	65.96
0.1	66.44	66.90	68.44	71.05	74.81	77.61
0.2	66.88	67.90	71.12	74.88	**77.26**	78.07
0.3	67.39	69.08	73.46	76.58	77.84	78.12
0.4	67.98	70.26	75.00	77.25	77.96	78.07
0.5	68.66	71.36	75.97	77.66	78.08	78.01
0.6	69.34	72.46	76.57	77.84	78.17	77.98
0.7	70.08	73.52	76.98	77.95	78.20	78.01
0.8	70.92	74.30	77.29	78.01	78.18	78.02
0.9	71.81	74.95	77.48	78.10	78.18	78.01
1.0	72.92	75.51	77.67	78.20	78.19	78.06

Table 1. Sensitivity of CIPA to σ and N_{iter}. By default we use $\sigma = 0.2$ and $N_{iter} = 20$ (shown in the black box). Numbers in red show the configurations that lead to better performance (> 77.26) than the default setting.

B. Manifold Mixup vs. HCT

To have a thorough comparison between Manifold Mixup [2] and our proposed Hybrid Consistency Training (HCT), we train models using these two approaches with different α values. Note that α determines the distribution from which the weight λ that balances the linear combination of the two samples is drawn: $\lambda \sim Beta(\alpha, \alpha)$. We keep all other hyper-parameters exactly the same so that the changes in accuracies are only caused by the different behaviors between Manifold Mixup and HCT, *i.e.*, $\mathcal{L}_{mm}$ v.s. $\mathcal{L}_{hct}$. Tab. 2 shows the results on both the *mini*-ImageNet and CUB datasets. We can observe that, overall, HCT achieves better performance than Manifold Mixup. The improvement is more obvious on 1-shot tasks, while less noticeable on 5-shot tasks. This is reasonable since performance differences among Few-Shot Learning (FSL) methods tend to decrease as more labeled examples are used. These results prove that our proposed HCT is a better alternative than Manifold Mixup on FSL problems.

C. Semi-supervised FSL

Our proposed Calibrated Iterative Prototype Adaptation (CIPA) algorithm can not only be used for transductive inference, but also be naturally extended to the semi-supervised FSL setting, which is first proposed in SemiPN [1]. The difference between semi-supervised FSL and transductive FSL is that the former uses a separate auxiliary set of unlabeled examples to improve performance on query examples, while the latter uses query examples themselves for this purpose.

For semi-supervised FSL, we split the novel data into labeled and unlabeled sets (e.g., 60% as labeled and 40% as unlabeled). When generating test episodes, we always sample support and query examples from the labeled split (e.g., th 60% split), and sample auxiliary examples from the unlabeled split. When updating the prototypes, only auxiliary examples are used. After the class prototypes have

Method	Train				PN		SemiPN		CIPA	
	$\mathcal{L}_{ce}$	$\mathcal{L}_{mm}$	$\mathcal{L}_{hct}$	$\mathcal{L}_{rot}$	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
***mini*-ImageNet**										
Manifold Mixup (α = 0.5)	✓	✓			57.48	77.04	67.19	78.96	71.85	81.32
Manifold Mixup (α = 1.0)	✓	✓			57.07	78.09	68.25	80.12	73.69	83.06
Manifold Mixup (α = 2.0)	✓	✓			56.42	77.81	67.54	79.96	73.71	82.69
HCT (α = 0.5)	✓		✓		58.47	78.53	69.00	80.31	74.69	83.10
HCT (α = 1.0)	✓		✓		58.54	78.43	69.38	80.33	74.74	82.91
HCT (α = 2.0)	✓		✓		57.38	78.54	68.31	80.69	74.09	83.26
CUB										
Manifold Mixup (α = 0.5)	✓	✓			66.32	86.57	79.82	88.94	86.26	90.95
Manifold Mixup (α = 1.0)	✓	✓			65.78	86.53	79.26	88.84	86.12	90.95
Manifold Mixup (α = 2.0)	✓	✓			66.28	86.63	79.82	89.12	86.91	91.11
HCT (α = 0.5)	✓		✓		68.97	86.80	80.53	88.99	86.19	90.73
HCT (α = 1.0)	✓		✓		67.90	86.73	80.43	89.20	86.79	91.02
HCT (α = 2.0)	✓		✓		67.67	86.89	80.40	89.20	87.34	91.11

Table 2. Comparison between Manifold Mixup and HCT on various α values. Accuracies are averaged over 600 episodes.

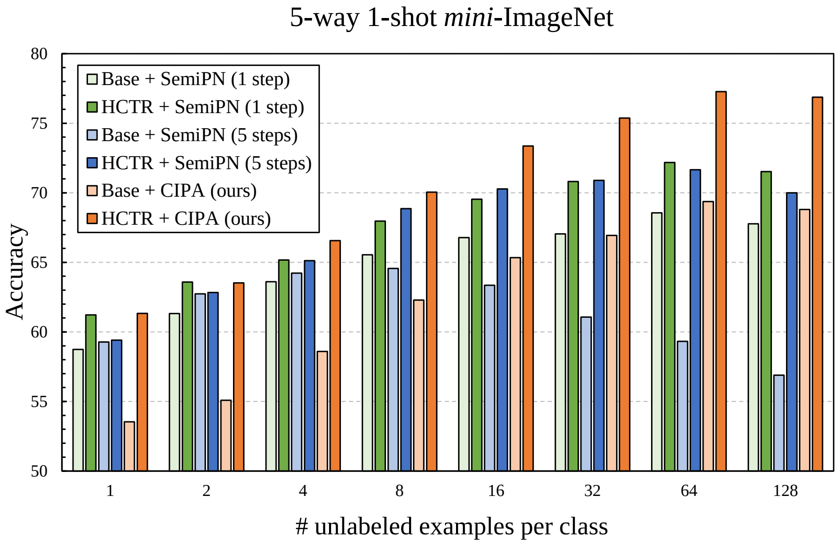

Figure 1. Bar chart of semi-supervised FSL results. Accuracies are averaged over 600 episodes. We omit the confidence intervals for clearer view.

been updated, they are used for prediction on the query examples. Since semi-supervised FSL is not transductive, statistics of query examples should not be used. Thus, for query examples, we remove the zero-mean transformation and only perform the power transformation and l_2 normalization. In each episode, we use one support example, $M = 1, 2, 4, \ldots, 128$ unlabeled examples, and 15 query examples per class. The 5-way 1-shot results on *mini*-ImageNet are shown in Fig. 1. Generally, as more unlabeled examples are used, the performance increases and saturates at certain M. For instance, the performance of HCT_R+SemiPN saturates at $M = 16$ whereas the performance of HCT_R+CIPA saturates at a later point with $M = 64$. This leaves a large room for CIPA to achieve higher performance, as indicated by an increasing lead of HCT_R+CIPA as M increases.

Overall, we observe that HCT_R and CIPA consistently outperform their counterpart: Classifier Baseline (denoted as Base in the chart to save space) and SemiPN, respectively. It is, therefore, expected that combining HCT_R and CIPA yields consistently superior performance among all methods. However, it is worth noting the interesting behaviors of the weaker combinations such as Base+CIPA and HCT_R+SemiPN as M increases. Indeed, with more unlabeled data the strength of HCT_R and CIPA starts to merge and eventually can compensate for the previously worse performance. We show two examples below.

Comparing Base+SemiPN(1 step) and Base+SemiPN(5 steps), we note that the latter one has a better performance for a smaller M while its performance saturate quickly (at $M = 4$) and starts degrading. The reason of this unexpected trend might be that, when there is more unlabeled data, the prototypes are easier to be distracted by noisy pseudo-labels in more iterations. However, when a better embedding model e.g., HCT_R is used, this trend can be fixed to certain degree. Comparing HCT_R+SemiPN(1 step) and HCT_R+ SemiPN(5 steps), the turning point is at $M = 64$. This demonstrates the robustness of the embedding leaned by HCT_R.

Comparing Base+SemiPN(1 step) and Base+CIPA, we can see that, when only a few unlabeled examples are available, CIPA produces inferior results. It catches up and achieves higher numbers as M increases. Our explanation is that, since CIPA calibrate the support data distribution and unlabeled data distribution separately, when both of them are sparse, the calibration might not work properly. This is also why it achieves better performance for larger M, where calibrating on more unlabeled data makes distance computation better. This verifies the strong adaptation capability of CIPA.

To conclude, a better embedding model (i.e., HCT_R) and a calibrated adaptive inference (i.e., CIPA) are both needed to achieve optimal FSL performance.

References

[1] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B Tenenbaum, Hugo Larochelle, and Richard S Zemel. Meta-learning for semi-supervised few-shot classification. In *ICLR*, 2018. 1

[2] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *ICML*, 2019. 1