

Supplementary Materials for CHMATCH: Contrastive Hierarchical Matching and Robust Adaptive Threshold Boosted Semi-Supervised Learning

Jianlong Wu¹, Haozhe Yang², Tian Gan^{2*}, Ning Ding³, Feijun Jiang³, Liqiang Nie^{1*}

¹ School of Computer Science and Technology, Harbin Institute of Technology (Shenzhen)

² School of Computer Science and Technology, Shandong University ³ Alibaba Group

jluw1992@pku.edu.cn, sailist@outlook.com, gantian@sdu.edu.cn,
{yuji.dn, feijun.jiangfj}@alibaba-inc.com, nieliqiang@gmail.com

A. Detailed Experimental Settings

In Table 1, we presented the detailed settings of hyper-parameters in our semi-supervised learning experiments.

B. Generalization Analysis

To evaluate the generalization ability of our propose method, we conducted additional experiments under domain shift settings. We use the model trained on the CIFAR100 dataset under WRN-28-2 to test the results on the sketch domain. The ImageNet-Sketch dataset consists of

50,000 sketch images, 50 images for each of the 1,000 ImageNet classes. We selected the sketch images from ImageNet-Sketch that belong to the overlapped categories between CIFAR100 and ImageNet for testing. Note that one category in CIFAR100 may correspond to several classes in ImageNet. Specifically, 11,950 images that belong to 62 categories of CIFAR100 are selected. The accuracy for CHMatch, FlexMatch, and FixMatch are 15.55%, 12.31%, and 8.75%, respectively. We can see that our method also achieves much better results than these two strong baselines.

*Corresponding authors.

Table 1. Setting of parameters on various datasets.

Dataset	CIFAR-10	STL-10	CIFAR-100	CIFAR-100	ImageNet
Model	WRN-28-2	ResNet-18	WRN-28-2	WRN-28-8	ResNet-50
Batch Size	64		64		160
μ	7		7		4
Max proportion	0.95		0.8		
Initial proportion			0.05		
Dynamic duration			100		









								
cpl: \hat{q}_α^c	fish				reptiles			
y	aquarium fish	shark	ray	flatfish	turtle	dinosaur	crocodile	lizard
fpl: \hat{q}_α^f	aquarium fish	cattle	ray	turtle	turtle	crocodile	crocodile	lizard

Figure 1. Predicted results of fine-grained and coarse-grained classifications. 'cpy', y , and 'fpl' correspond to coarse-grained pseudo label, ground-truth label, and fine-grained pseudo label, respectively. The misclassified fine-grained pseudo-label can be corrected by coarse-grained pseudo-label in graph construction.

Table 2. Precision, Recall, F1-score comparison on CIFAR-100 with 400 labeled samples.

Model	WRN-28-2			WRN-28-8		
	Precision	Recall	F1-score	Precision	Recall	F1-score
FixMatch	0.5177	0.4574	0.4129	0.5615	0.54	0.4983
FlexMatch	0.4914	0.4991	0.4886	0.5930	0.6027	0.5931
CHMatch	0.5585	0.5536	0.5497	0.6641	0.6557	0.6541

Table 3. Class-wise accuracy comparison on CIFAR-10 40-label split.

Class Number	0	1	2	3	4	5	6	7	8	9
FixMatch	0.964	0.982	0.697	0.852	0.974	0.890	0.987	0.970	0.982	0.981
FlexMatch	0.967	0.980	0.921	0.866	0.957	0.883	0.988	0.975	0.982	0.968
CHMatch	0.975	0.978	0.890	0.876	0.966	0.871	0.985	0.971	0.982	0.981

C. Graph Matching Examples

As illustrated in Section 3.4, our graph matching strategy can learn an accurate affinity graph by taking advantage of coarse pseudo label graph to correct the fine pseudo label graph. Specifically, when there are many indistinguishable super classes, two samples might have the same fine-grained pseudo-label, but their coarse-grained pseudo-label are different. We show some real cases in Figure 1. We can see that both the fourth and fifth images are predicted as 'turtle' since they are very similar, while the fourth image is misclassified. However, they have different coarse pseudo labels. In this case, by our graph matching strategy, they will not be regarded as the positive pairs to guide the contrastive learning, which can well demonstrate the effect of our graph matching strategy.

D. Computational Analysis of the memory-bank based threshold learning

For the threshold computation in step 10 of the Algorithm 1, its time cost can be neglectable. Specifically, we use the function "torch.topk" in PyTorch library to select the $K\%$ largest values from the memory bank to compute the threshold. The computational complexity for this operation is $\mathcal{O}(n + t)$, where $t = n * K\%$, and n denotes the size of the memory bank. In our experiments, we set $n = 50,000$, and it only needs 2.6ms for each minibatch to compute the threshold. Moreover, this operation in our method is irrelevant to the dataset.

In comparison, FlexMatch needs the predicted probability of the whole dataset to compute the threshold for each category by the "counter" function, whose computational complexity is $\mathcal{O}(m + c)$, where m and c denote the number of training samples and classes, respectively. Since m is generally larger than n , its computational complexity is

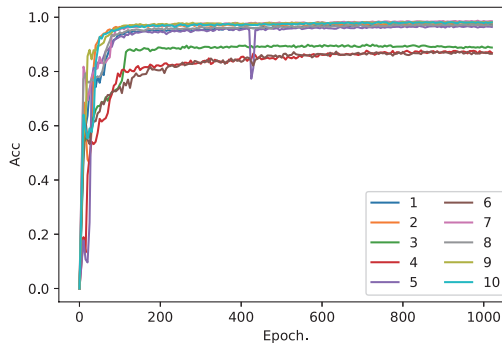


Figure 2. The class-wise accuracy convergence curve on CIFAR-10 with 40 labels.

higher than ours. Specifically, it generally needs 2.66ms on CIFAR and 109ms on ImageNet to execute this function. For the whole training, it costs extra 0.7, and 31.7 hours for such an operation on these two datasets.

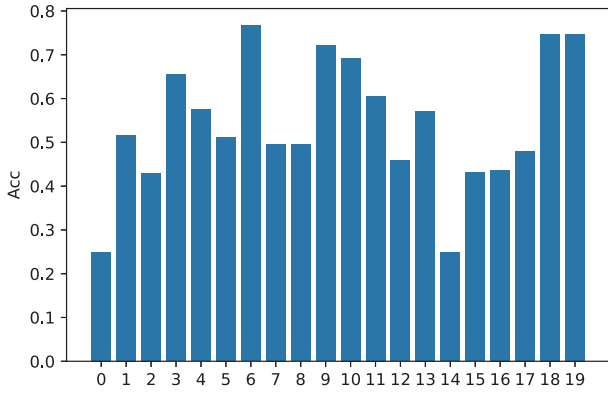
E. Detailed Results under Other Evaluation Metrics

To comprehensively evaluate the performance under various metrics, we further reported the Precision, Recall, and F1-score on CIFAR-100 with 400 labels samples under two different backbones. As shown in Table 2, CHMatch also achieves the best performance under these metrics.

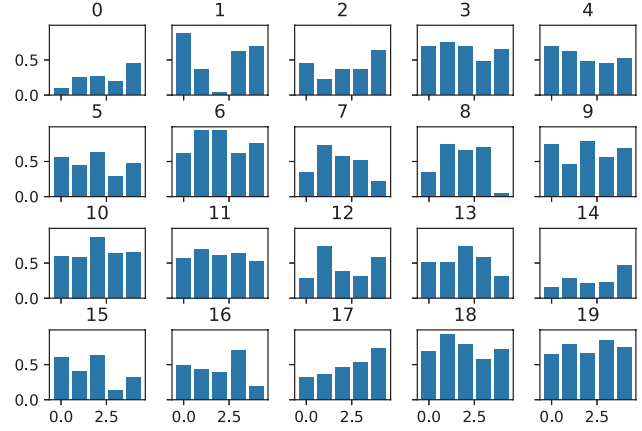
F. Class-wise Accuracy Comparison

Table 3 shows a detailed class-wise accuracy comparison on CIFAR-10. Even without assigning class-specific threshold as FlexMatch does, we still achieved competitive results on those hard-to-learn classes.

Figure 2 shows the smooth growth of the accuracy of our method for each category during the training on the CIFAR-

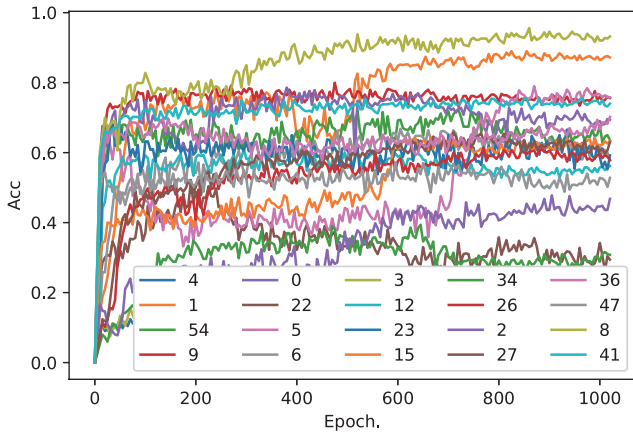


(a) Coarse-grained category accuracy

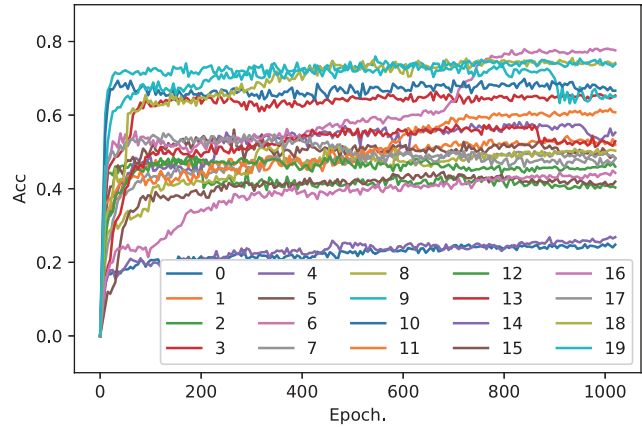


(b) Sub-category accuracy for each coarse-grained category

Figure 3. Accuracy of each category on CIFAR-100.



(a) First fine-grained class accuracy



(b) Coarse-grained class accuracy

Figure 4. The convergence curves for selected classes on CIFAR-100.

10 dataset.

We also presented the class-wise accuracy on CIFAR-100, which is shown in Figures 3 and 4. We can see that the coarse-grained classification accuracy is higher than the fine-grained results, which can provide useful information for guidance and correctness. Besides, according to Figure 4, our method can converge very fast and is very stable.

G. The Construction of ImageNet subset and visualization

The detailed construction process of the ImageNet subset can be summarized as follows:

- Step 1: The categories of the ImageNet dataset have

a tree structure. We iterate through all non-leaf nodes starting from the root node, and record the number of leaf nodes they cover on all non-leaf nodes.

- Step 2: Delete all non-leaf nodes that cover the number of leaf nodes greater than K (In the operation, we set $K=112$). These deleted non-leaf nodes have coarse granularity.
- Step 3: Delete all non-leaf nodes that have less than 10 leaf nodes.
- Step 4: Delete all non-leaf nodes whose children contain non-leaf nodes to ensure that all non-leaf nodes do not overlap.



Figure 5. Visualization of the balanced ImageNet dataset. Each row corresponds to one coarse-grained category with 10 fine-grained categories, where each fine-grained category has two images for visualization.

- Step 5: Sort these non-leaf nodes in alphabetical order based on category identity. Then take the top 20 non-leaf nodes as the coarse-grained category.
- Step 6: Sort leaf nodes under these selected 20 non-leaf nodes with the same rule. Then take the top 10 leaf nodes of each non-leaf node as their fine-grained categories.

This process finally retains 200 leaf nodes and 20 non-leaf nodes, where each leaf node corresponds to a category in ImageNet1k and each non-leaf node corresponding to a coarse-grained category. The number of 200 fine-grained and 20 coarse-grained classes are set empirically. The code for construction and detailed categories will be released after acceptance.

We also visualized the training classes of ImageNet obtained by our algorithm in Figure 5. Each row corresponds to one coarse-grained class, which contains 20 images from 10 different fine-grained classes.

H. Detailed comparison with CoMatch

CoMatch achieves very good performance for the task of semi-supervised learning. Its main contributions lie in

two aspects: memory-smoothed pseudo labeling and graph-based contrastive learning to learn better representation.

In comparison, our contributions and differences lie in two aspects, including the memory-bank based dynamic proportion strategy and hierarchical label based graph matching. Both of these two mechanisms are effective and have not been studied in the field of semi-supervised learning. First, different from FixMatch and FlexMatch which are based on a manually set threshold, our dynamic proportion strategy can select the same number of samples for training in different epochs for two independent experiments and is more robust to parameters, leading to more stable results. Second, our hierarchical labels based graph matching can construct a more accurate affinity graph for contrastive learning, contributing to more discriminative feature representation and better performance. We need to mention that our graph matching strategy is significantly different from that in CoMatch. On the one hand, our matching denotes the consistency between coarse-grained graph and fine-grained graph to get more correct supervision information, while CoMatch computes a similarity graph to constrain the feature representation. On the other hand, the way to construct the graph is also different. CoMatch adopts the similarity between probabilities to reg-

ularize the similarity between features. It also presents a memory-smoothed pseudo-labeling strategy to use similarity in feature space to update the probabilities in label space. In this case, there are so many bidirectional interactions between features and probabilities and it cannot well solve the issue claimed in CoMatch that “since the features are highly correlated with the class predictions, the same types of errors are likely to exist in both the feature space and the label space”. As a comparison, we directly use the one-hot pseudo-label to construct the graph without the above complex interactions, which can weaken the above issue.