Supplementary Material of "Iterative label cleaning for transductive and semi-supervised few-shot learning"

Michalis Lazarou[1] Tania Stathaki[1] Yannis Avrithis[2]
[1]Imperial College London
[2]Inria, Univ Rennes, CNRS, IRISA

A. Datasets

***mini*ImageNet** This is a widely used few-shot image classification dataset [61, 49]. It contains 100 randomly sampled classes from ImageNet [28]. These 100 classes are split into 64 training (base) classes, 16 validation (novel) classes and 20 test (novel) classes. Each class contains 600 examples (images). We follow the commonly used split proposed in [49]. All images are resized to 84×84.

***tiered*ImageNet** This is also sampled from ImageNet [28] but has a hierarchical structure. Classes are partitioned into 34 categories, organized into 20 training, 6 validation and 8 test categories, containing 351, 97 and 160 classes, respectively. This ensures that training classes are semantically distinct from test classes, which is more realistic. We follow the common split of [9]. Again, all images are 84×84.

CUB This is a fine-grained classification dataset consisting of 200 classes, each corresponding to a bird species. We follow the split defined by [10, 15], with 100 training, 50 validation and 50 test classes. To compare fairly with competitors, we *use* bounding boxes on ResNet features following [60] to compare with [63] but we *do not use* bounding boxes on WRN [52] features to compare with [19].

CIFAR-FS This dataset is derived from CIFAR-100 [27], consisting of 100 classes with 600 examples per class. We follow the split provided by [10], with 64 training, 16 validation and 20 test classes. To compare fairly with competitors, we use the original image resolution of 32×32 on WRN features to compare with [19] but we resize images to 84×84 on ResNet features to compare with [63].

B. Feature pre-processing

ResNet-12A ResNet-12A is the pre-trained backbone network used in [63]. For all of our transductive and semi-supervised experiments using this network, we adopt exactly the same pre-processing as [63], which is ℓ_2-normalization on the output features.

WRN-28-10 WRN-28-10 is the pre-trained network used in [38] and [19]. To provide fair comparisons with PT+MAP [19] we adopt exactly the same pre-processing as [19]. In the transductive experiments, we apply power transform, ℓ_2-normalization and centering on the output features. In the semi-supervised experiments, we applied centering by calculating the mean and variance from the support set, S, and the unlabeled set, U, not taking into consideration the query set, Q, since this is an inductive setting.

ResNet-12B ResNet-12B is the pre-trained network used in MCT [29]. For the experiments in Table 6, we adopt exactly the same pre-processing as [29], that is, ℓ_2-normalization on the output features.

C. Hyperparameters

Table 11 shows the best hyperparameters k (1) and α (4) for every dataset, network and number of support examples per class $K \in \{1, 5\}$. The hyperparameters are optimized on the validation set separately for each experiment. We carried out experiements in the transductive setting for $k \in \{5, 8, 10, 15, 20, 25, 30, 40, 50, 60\}$ and $\alpha \in \{0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$ and select the combination resulting in the best mean validation accuracy. In the semi-supervised setting, we use the same optimal values.

D. Confidence weights

The Sinkhorn-Knopp algorithm iteratively normalizes a $M \times N$ positive matrix P to a row-wise sum $\mathbf{p} \in \mathbb{R}^M$ and column-wise sum $\mathbf{q} \in \mathbb{R}^N$. We experiment with two ways of setting the value of $\mathbf{p}$:

1. **Uniform.** Interpreting the i-th row of P as a class probability distribution for the i-th query, it should be normalized to one, such that $p_i = 1$ uniformly.

2. **Entropy.** Because we do not have the same confidence for each prediction, we use the entropy of the predicted

PARAM	*m*IN		*t*IN		CFS		CUB	
K (shot)	1	5	1	5	1	5	1	5
ResNet-12A								
k (1)	15	25	15	60	15	15	10	8
α (4)	0.8	0.4	0.5	0.8	0.8	0.4	0.6	0.6
ResNet-12B								
k (1)	15	15	-	-	-	-	-	-
α (4)	0.9	0.9	-	-	-	-	-	-
WRN-28-10								
k (1)	20	30	20	20	20	25	25	25
α (4)	0.8	0.2	0.8	0.8	0.4	0.5	0.2	0.5

Table 11. Selected hyperparameters. *m*IN: *mini*ImageNet. *t*IN: *tiered*ImageNet. CFS: CIFAR-FS.

METHOD	RESNET-12A		WRN-28-10	
	1-shot	5-shot	1-shot	5-shot
uniform	**69.79**$\pm$0.99	**79.82**$\pm$0.55	**83.05**$\pm$0.79	**88.82**$\pm$0.42
entropy	66.94$\pm$1.01	78.34$\pm$0.58	81.05$\pm$0.90	88.43$\pm$0.44

Table 12. Comparison between ways of setting confidence weights $\mathbf{p}$; transductive inference on *mini*ImageNet. Uniform: $p_i = 1$. Entropy: $p_i = \omega_i$ (16).

class probability distribution of each example to quantify its uncertainty. Following [22], we associate to each example x_{L+i} for $i \in [M]$ a weight

$$\omega_i := 1 - \frac{H(\hat{\mathbf{z}}_\mathbf{i})}{\log(N)}, \tag{16}$$

where N is the number of classes and $\hat{z}_i$ is the ℓ_1-normalized i-th row of Z (4), that is, $\hat{z}_{ij} := z_{ij} / \sum_{k=1}^{N} z_{ik}$. We then set the confidence weights $p_i = \omega_i$. Note that ω_i takes values in $[0, 1]$ because $\log(N)$ is the maximum possible entropy.

Given $\mathbf{p}$ and assuming balanced classes, $\mathbf{q}$ is defined by (7), that is, $q_j = \frac{1}{N} \sum_{i=1}^{M} p_i$ for $j \in [N]$. In the special case of $p_i = 1$, this simplifies to $q_j = \frac{M}{N}$.

Table 12 compares the two approaches. Even though using non-uniform confidence weights is a reasonable choice, uniform weights are superior in all settings. This can be attributed to the fact that examples with small weight tend to be ignored in the balancing process, hence their class distribution and consequently their predictions are determinded mostly by other examples with large weight. For this reason, examples with small weight may get more incorrect predictions in the case of entropy.

METHOD	INFERENE TIME
LR+ICI [63]	0.89
PT+MAP [19]	**0.57**
iLPC	1.20

Table 13. Average inference time (in seconds) for the 1-shot tasks in *mini*ImageNet dataset.

E. Inference time

We conduct inference time experiments to investigate the computational efficiency of our iLPC compared with PT+MAP [19] and LR+ICI [63]. Using the WRN-28-10 backbone, we calculate the inference time required for a single 5-way, 1-shot task, averaged over 1000 tasks. For each task there are 15 queries per class. The results can be seen on Table 13.

F. Flaws in evaluation

Throughout our investigations we observed that comparisons are commonly published that are not under the same settings. In this section we highlight such problems.

1. In multiple works such as [51, 19, 65, 11], comparisons between state-of-the-art methods are made without explicitly differentiating between inductive and transductive methods. This is unfair since transductive methods perform better by leveraging query data.

2. Comparisons use different networks without mentioning so. For example, Table 1 of [66] does not indicate what network each method uses. [66] uses WRN-28-10, while [36] uses a 4-layer convolutional network.

3. In the semi-supervised setting, comparisons use different numbers of unlabelled data without mentioning so. In Table 4 of [51] for example, [51] uses 100 unlabelled examples while [32] uses 30 for 1-shot and 50 for 5-shot, [50] and [36] use 20 for 1-shot and 20 for 5-shot. In Table 1 of [63], the best model of [63] uses an 80/80 split for 1/5-shot while other methods such as [32] use a 30/50 split. In Table 1 of [66], [66] uses 100 or 200 unlabelled examples while [50, 36] use 20/20 split for 1/5-shot.

4. Some methods use different dataset settings when comparing with other methods without explicitly stating so. In Table 1 of [63] for instance, [63] uses the bounding box provided for CUB while other methods such as [10, 31] do not.

5. Comparisons using the same network is made but this network has been trained using a different training regimes. Unless the novelty of the work lies in the

training regime, this is unfair. As shown in [38], a better training regime can increase the performance significantly.

6. There are several different variants of the benchmark datasets, coming from different sources. The two most common variants are [10], which uses original image files, and [31], which uses pre-processed tensors stored in `pkl` files. Testing a network on a different variant than the one it was trained on may result in performance drops as large as 5%.

We believe that highlighting these evaluation flaws will help researchers avoid making such mistakes and move towards a fairer evaluation. We encourage the community to compare different methods against the same settings and if otherwise, state clearly the differences. As a contribution towards a fairer evaluation, we intend to make our code publicly available along with the pre-trained networks used in this work.