Iterative label cleaning for transductive and semi-supervised few-shot learning
Michalis Lazarou, Tania Stathaki, Yannis Avrithis

To cite this version:
Michalis Lazarou, Tania Stathaki, Yannis Avrithis. Iterative label cleaning for transductive and semi-supervised few-shot learning. ICCV 2021 - IEEE/CVF International Conference on Computer Vision, Oct 2021, Virtual, France. pp.1-10. hal-03530170

HAL Id: hal-03530170
https://inria.hal.science/hal-03530170
Submitted on 17 Jan 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Iterative label cleaning for transductive and semi-supervised few-shot learning

Michalis Lazarou\textsuperscript{1} Tania Stathaki\textsuperscript{1} Yannis Avrithis\textsuperscript{2}
\textsuperscript{1}Imperial College London
\textsuperscript{2}Inria, Univ Rennes, CNRS, IRISA

Abstract

Few-shot learning amounts to learning representations and acquiring knowledge such that novel tasks may be solved with both supervision and data being limited. Improved performance is possible by transductive inference, where the entire test set is available concurrently, and semi-supervised learning, where more unlabeled data is available.

Focusing on these two settings, we introduce a new algorithm that leverages the manifold structure of the labeled and unlabeled data distribution to predict pseudo-labels, while balancing over classes and using the loss value distribution of a limited-capacity classifier to select the cleanest labels, iteratively improving the quality of pseudo-labels. Our solution surpasses or matches the state of the art results on four benchmark datasets, namely miniImageNet, tieredImageNet, CUB and CIFAR-FS, while being robust over feature space pre-processing and the quantity of available data. The publicly available source code can be found in \url{https://github.com/MichalisLazarou/iLPC}

1. Introduction

Few-shot learning \cite{61, 56} is challenging the deep learning paradigm in that, not only supervision is limited, but data is limited too. Despite the initial promise of meta-learning \cite{39, 12}, transfer learning \cite{10, 59} is becoming increasingly successful in decoupling representation learning from learning novel tasks on limited data. Semi-supervised learning \cite{30, 5} is one of the dominant ways of dealing with limited supervision and indeed, its few-shot learning counterparts \cite{50, 66} are miniature versions where both labeled and unlabeled data are limited proportionally, while representation learning may be decoupled. These methods are closer to transductive inference \cite{36, 51}, which was a pillar of semi-supervised learning before deep learning \cite{8}.

Predicting pseudo-labels on unlabeled data \cite{30} is one of the oldest ideas in semi-supervised learning \cite{54}. Graph-based methods, in particular label propagation \cite{68, 67}, are prominent in transductive inference and translate to inductive inference in deep learning exactly by predicting pseudo-labels \cite{22}. However, with the representation being fixed, the quality of pseudo-labels is critical in few-shot learning \cite{63, 29}. At the same time, in learning with noisy labels \cite{3, 21, 57}, it is common to clean labels based on the loss value statistics of a small-capacity classifier.

In this work, we leverage these ideas to improve transductive and semi-supervised few-shot learning. As shown in Figure 1, focusing on transduction, a set of labeled support examples $S$ and unlabeled queries $Q$ are given, represented in a feature space by mapping $f$. By label propagation \cite{67}, we obtain a matrix that associates examples to classes. The submatrix corresponding to unlabeled examples, $P$, is normalized over examples and classes using the Sinkhorn-Knopp algorithm \cite{24}, assuming a uniform distribution over classes. We extract pseudo-labels from $P$, which we clean following O2U-Net \cite{21}, keeping only one example per class. Finally, inspired by \cite{26}, we move these examples from $Q$ to $S$ and iterate until $Q$ is empty.

Figure 1. Overview of the proposed method. See text for details.

2. Related work and contributions

2.1. Few-shot learning

Meta-learning This is a popular paradigm, where the training set is partitioned in episodes resembling in struc-
ture the novel tasks [12, 25, 56, 39, 61]. Model-based methods rely on the properties of specific model architectures, such as recurrent and memory-augmented networks [39, 53, 40]. Optimization-based methods attempt to learn model parameters that are able to adapt fast in novel tasks [12, 48, 70, 49, 41, 31, 6]. Metric-based methods attempt to learn representations that are appropriate for comparisons [56, 25, 58, 61]. Of course, metric learning is a research area on its own [62, 23] and modern ideas are commonly effective in few-shot learning [34].

Predicting weights, data augmentation Also based on meta-learning, it is possible to predict new parameters or even data. For instance, it is common to learn to predict data-dependent network parameters in the last layer (classifier) [13, 45, 47] or even in intermediate convolutional layers [7]. Alternatively, one can learn to generate novel-task data in the feature space [14] or in the input (image) space [64, 2]. The quality can be improved by translating images, similar to style transfer [35]. Such learned data augmentation is complementary to other ideas.

Transfer learning More recently, it is recognized that learning a powerful representation on the entire training set is more effective than sampling few-shot training episodes that resemble novel tasks [13, 10, 59, 38]. In doing so, one may use standard loss functions [13, 10], knowledge distillation [59] or other common self-supervision and regularization methods [38]. We follow this transfer learning approach, which allows us to decouple representation learning from the core few-shot learning idea and provide clearer comparisons with the competition.

2.2. Using unlabeled data

Leveraging unlabeled data is of interest due to the ease of obtaining such data. Two common settings are transductive inference and semi-supervised learning.

Transductive inference In this setting, all novel-class unlabeled query examples are assumed available at the same time at inference [36, 33, 46, 18, 19, 51]. These examples give additional information on the distribution of novel classes on top of labeled support examples.

Common transductive inference solutions are adapted for few-shot classification, notably label propagation [36] and embedding propagation [51], which smooths embeddings as in image segmentation [4]. Both operations are also used at representation learning, as in meta-learning. Using dimensionality reduction, TAFSSL [33] learns a task-specific feature subspace that is highly discriminant for novel tasks. Meta-confidence transduction (MCT) [29] meta-learns a data-dependent scaling function to normalize every example and iteratively updates class centers. PT+MAP [19] uses a similar iterative process but also balances over classes. Cross-attention [16], apart from aligning feature maps by correlation, leverages query examples by iteratively making predictions and using the most confident ones to update the class representation.

Semi-supervised learning In this case, labeled novel-class support examples and additional unlabeled data are given. A classifier may be learned on both to make predictions on novel-class queries [50, 63, 66].

One of the first contributions uses unlabelled examples to adapt prototypical networks [56], while discriminating from distractor classes [50]. Common semi-supervised solutions are also adapted to few-shot classification, for instance learning to self-train [32], which adapts pseudo-label [30] and TransMatch [66], which is an adaptation of MixMatch [5]. Instance credibility inference [63] predicts pseudo-labels iteratively, using a linear classifier to select the most likely to be correct and then augmenting the support set. Adaptive subspaces [55] are learned from labeled and unlabeled data, yielding a discriminative subspace classifier that maximizes the margin between subspaces.

2.3. Contributions

In this work, focusing on the transfer learning paradigm to learn novel tasks given a fixed representation [59, 38], we make the following contributions:

1. We combine the power of predicting pseudo-labels in semi-supervised learning [30, 32] with label cleaning in learning from noisy labels [21].
2. According to manifold assumption, we use label propagation [67, 36] to infer pseudo-labels, while balancing over classes [24, 19] and iteratively re-use pseudo-labels in the propagation process [26, 63].
3. We achieve new state of the art in both transductive and semi-supervised few-shot learning.

3. Method

3.1. Problem formulation

At representation learning, we assume access to a labeled dataset $D_{base}$ with each example having a label in one of the classes in $C_{base}$. This dataset is used to learn a mapping $f : \mathcal{X} \rightarrow \mathbb{R}^d$ from an input space $\mathcal{X}$ to a $d$-dimensional feature or embedding space.

The knowledge acquired at representation learning is used to solve novel tasks, assuming access to a dataset $D_{novel}$ with each example being associated with one of the classes $C_{novel}$, where $C_{novel}$ is disjoint from $C_{base}$. Examples in $D_{novel}$ may be labeled or not.

In few-shot classification [61], a novel task is defined by sampling a support set $S$ from $D_{novel}$, consisting of $N$ classes with $K$ labeled examples per class, for a total of $L := NK$ examples. Given the mapping $f$ and the support set $S$, the problem is to learn an $N$-way classifier that makes
predictions on unlabeled queries also sampled from $D_{\text{novel}}$. Queries are treated independently of each other. This is referred to as inductive inference.

In transductive inference, a query set $Q$ consisting of $M$ unlabeled examples is also sampled from $D_{\text{novel}}$. Given the mapping $f$, $S$ and $Q$, the problem is to make predictions on $Q$, without necessarily learning a classifier. In doing so, one may leverage the distribution of examples in $Q$, which is important because $M$ is assumed greater than $L$.

In semi-supervised few-shot classification, an unlabeled set $U$ of $M$ unlabeled examples is also sampled from $D_{\text{novel}}$. Given $f$, $S$ and $U$, the problem is to learn to make predictions on new queries from $D_{\text{novel}}$, as in inductive inference. Again, $M > L$ and we may leverage the distribution of $U$.

In this work, we focus on transductive inference and semi-supervised classification, given $f$. The performance of $f$ on inductive inference is one of our baselines. We develop our solution for transductive inference. In the semi-supervised case, we follow the same solution with $Q$ replaced by $U$. Using the predictions on $U$, we then proceed as in the inductive case, with $S$ replaced by $S \cup U$.

### 3.2. Nearest-neighbor graph

We are given the mapping $f$, the labeled support set $S := \{(x_i, y_i)\}_{i=1}^{L}$ and the query set $Q := \{x_{L+i}\}_{i=1}^{M}$, where $y_i \in [N] := \{1, \ldots, N\}$. We embed all examples from $S$ and $Q$ into $V = \{v_1, \ldots, v_T\} \subset \mathbb{R}^d$ and $\ell_1$-normalize them, where $T := L + M$ and $v_i := f(x_i)$ for $i \in [T]$. Following [22], we construct a $k$-nearest neighbour graph of the features in $V$, represented by a sparse $T \times T$ nonnegative affinity matrix $A$, with

$$A_{ij} := \begin{cases} \gamma & \text{if } i \neq j \wedge v_i \in \text{NN}_k(v_j) \\ 0 & \text{otherwise} \end{cases}$$

(1)

for $i \in [T]$, $j \in [N]$, where $\text{NN}_k(v)$ are the $k$-nearest neighbors of $v$ in $V$ and $\gamma > 1$ is a hyperparameter. Finally, we obtain the symmetric $T \times T$ adjacency matrix $W := \frac{1}{2}(A + A^\top)$ and we symmetrically normalize it as

$$W := D^{-1/2}WD^{-1/2},$$

(2)

where $D = \text{diag}(W1_T)$ is the $T \times T$ degree matrix of $W$.

### 3.3. Label propagation

Following [67], we define the $T \times N$ label matrix $Y$ as

$$Y_{ij} := \begin{cases} 1, & \text{if } i \leq L \wedge y_i = j \\ 0, & \text{otherwise} \end{cases}$$

(3)

for $i \in [T]$, $j \in [N]$. Matrix $Y$ has one column per class and one row per example, which is an one-hot encoded label for $S$ and a zero vector for $Q$. Label propagation amounts to solving $N$ linear systems

$$Z := (I - \alpha W)^{-1}Y,$$

(4)

where $\alpha \in [0, 1)$ is a hyperparameter. The resulting $T \times N$ matrix $Z$ can be used to make predictions by taking the maximum element per row [67]. However, before making predictions, we balance over classes.

### 3.4. Class balancing

We focus on the $M \times N$ submatrix

$$P := Z_{L+1:T,:}$$

(5)

(the last $M$ rows) of $Z$ that corresponds to unlabeled queries. We first perform an element-wise power transform

$$P_{ij} \leftarrow P_{ij}^\tau$$

(6)

for $i \in [M]$, $j \in [N]$, where $\tau > 1$, encouraging hard predictions. Parameter $\tau$ is analogous to the scale (or inverse temperature) of logits in softmax-based classifiers [13, 45, 42], only here the elements of $P$ are proportional to class probabilities rather than logits.

Inspired by [19], we normalize $P$ to a given row-wise sum $p \in \mathbb{R}^M$ and column-wise sum $q \in \mathbb{R}^N$. Each element $p_i \in [0, 1]$ of $p$ represents a confidence of example $x_{L+i}$ for $i \in [M]$; it can be a function of the $i$-th row of $P$ or set to 1. Each element $q_j \geq 0$ of $q$ represents a weight of class $j$ for $j \in [N]$. In the absence of such information, we set

$$q := \frac{1}{N}(p^\top 1_M)1_N,$$

(7)

assuming a uniform distribution of queries over classes.

The normalization itself is a projection of $P$ onto the set $S(p, q)$ of nonnegative $M \times N$ matrices having row-wise sum $p$ and column-wise sum $q$.

$$S(p, q) := \{X \in \mathbb{R}^{M \times N} : X1_N = p, X^\top 1_M = q\}.$$  (8)

We use the Sinkhorn-Knopp algorithm [24] for this projection, which alternates between rescaling the rows of $P$ to sum to $p$ and its columns to sum to $q$,

$$P \leftarrow \text{diag}(p) \text{diag}(P1_N)^{-1}P \quad \text{(9)}$$

$$P \leftarrow P \text{ diag}(P^\top 1_M)^{-1} \text{diag}(q), \quad \text{(10)}$$

until convergence. Finally, for each query $x_{L+i}$, $i \in [M]$, we predict the pseudo-label

$$\hat{y}_{L+i} := \arg \max_{j \in [N]} P_{ij}$$

(11)

that corresponds to the maximum element of the $i$-th row of the resulting matrix $P$, for $i \in [M]$. 

3.5. Label cleaning

The predicted pseudo-labels are not necessarily correct, yet a classifier can be robust to such noise. This is the case when enough data is available to adapt the representation [30, 22], such that the quality of pseudo-labels improves with training. Since data is limited here, we would like to select pseudo-labeled queries in $Q$ that are most likely to be correct, treating them as truly labeled and add them to the support set $S$. Iterating this process is an alternative way of improving the quality of pseudo-labels.

We interpret this problem as learning with noisy labels, leveraging recent advances in label cleaning [3, 21, 57]. Assuming that the classifier does not overfit the data, e.g. with small capacity, high learning rate or few iterations, the principle is that examples with clean labels exhibit less loss than examples with noisy labels.

In particular, given the labeled support set $S := \{(x_i, y_i)\}_{i=1}^L$ and the pseudo-labeled query set $Q := \{(x_{L+i}, y_{L+i})\}_{i=1}^M$, we train an $N$-way classifier $g$ using a weighted cross-entropy loss

$$\ell := -\sum_{i=1}^L \log g(x_i)_{y_i} - \sum_{i=1}^M p_i \log g(x_{L+i})_{y_{L+i}}, \quad (12)$$

where $p_i$ is the confidence weight of example $x_{L+i}$. Here, the classifier $g$ is assumed to yield a vector of probabilities over classes using softmax and $g(x)_y$ refers to element $y \in [N]$ of $g(x)$. In practice, it is obtained by a linear classifier on top of embedding $f$, optionally allowing the adaptation of the last layers of the network implementing $f$.

The loss term $\ell_i := -p_i \log g(x_{L+i})_{y_{L+i}}$, corresponding to the pseudo-labeled query $x_{L+i}$, is used for selection. Following O2U-Net [21], we use large learning rate and collect the average loss $\bar{\ell}_i$ over all epochs, for $i \in [M]$. In learning with noisy labels, it is common to detect noisy labels based on statistics of this loss for clean and noisy labels [3, 57]. However, this does not work well with predicted pseudo-labels [1], hence we select queries having the least average loss [21, 1]. Since we iterate the process, we take the extreme case of selecting one query example per class:

$$I := \left\{ \arg \min_{j} \bar{\ell}_i : j \in [N] \right\}. \quad (13)$$

Finally, we augment the support set $S$ with the selected queries and their pseudo-labels, while at the same time removing the selected queries from $Q$:

$$S \leftarrow S \cup \{(x_{L+i}, y_{L+i})\}_{i \in I} \quad (14)$$

$$Q \leftarrow Q \setminus \{x_{L+i}\}_{i \in I} \quad (15)$$

3.6. Iterative inference

Although label propagation and class balancing make predictions on the entire unlabeled query set $Q$, we apply cleaning to only keep one pseudo-labeled query per class, which we move from $Q$ to the support set $S$. We iterate the entire process, selecting one pseudo-labeled query per class at a time, until $Q$ is empty and $S$ is augmented with all pseudo-labeled queries. Assuming that the selections are correct, the idea is that treating them as truly labeled in $S$ improves the quality of the pseudo-labels.

Algorithm 1: Iterative label propagation and cleaning (iLPC).

```
input : embedding $f$
input : labeled support set $S$ with $|S| = L$
input : unlabeled query set $Q$ with $|Q| = M$
output : augmented support set $S$ with $|S| = L + M$
1 repeat
   2 $\mathcal{W} \leftarrow \text{GRAPH}(f, S; \gamma, k)$ \triangleright adjacency matrix (1),(2)
   3 $Y \leftarrow \text{LABEL}(S)$ \triangleright label matrix (3)
   4 $Z \leftarrow \text{LP}(\mathcal{W}; Y; \alpha)$ \triangleright label propagation (4)
   5 $P \leftarrow Z_{L+1:L+M}$ \triangleright unlabeled submatrix (5)
   6 $P \leftarrow \text{POWER}(P; \tau)$ \triangleright power transform (6)
   7 $(p, q) \leftarrow \text{BALANCE}(P)$ \triangleright class balance (7)
   8 $P \leftarrow \text{Sinkhorn}(P; p, q)$ \triangleright Sinkhorn-Knopp (9),(10)
   9 $Y \leftarrow \text{PREDICT}(P)$ \triangleright pseudo-labels (11)
10 $\mathcal{I} \leftarrow \text{CLEAN}(f, S, Q, Y, p)$ \triangleright label cleaning (12),(13)
11 $(S, Q) \leftarrow \text{AUGMENT}(S, Q, \mathcal{I})$ \triangleright augment support (14),(15)
12 until $Q = \emptyset$ \triangleright all queries are predicted
```

Algorithm 1 summarizes this process, called iterative label propagation and cleaning (iLPC). Given $S$, $Q$ and the embedding $f$, we construct the nearest neighbor graph represented by the normalized adjacency matrix $\mathcal{W}$ (1),(2) and we perform label propagation on the current label matrix $Y$ (4). Focusing on the unlabeled submatrix $P$ of the resulting matrix $Z$, we perform power transform (6) and row/column normalization to balance over classes (9),(10). We predict pseudo-labels $\hat{Y}$ from the normalized $P$ (11), which we use along with $S$ and $Q$ to train a linear classifier on top of $f$ with cross entropy loss (12) and a cyclical learning rate schedule [21]. We select one query per class with the least average loss over all epochs (13), which we move from $Q$ to $S$ as labeled (14),(15). With $Q$, $S$ redefined, we repeat the process until $Q$ is empty.

At termination, all data is labeled in $S$. The predicted labels over the original queries are the output in the case of transductive inference. In semi-supervised classification, we use $S$ to learn a new classifier and make predictions on new queries, as in inductive inference.

4. Experiments

4.1. Setup

Datasets We use four common few-shot classification benchmark datasets, $\text{mini}$$\text{ImageNet}$ [61, 49], $\text{tiered}$$\text{ImageNet}$ [9], CUB [10, 15] and CIFAR-FS [10, 27]. More details are provided in the supplementary material.
Table 1. Ablation study of algorithmic components of our method iLPC on miniImageNet. Inductive: baseline using only support examples. Transductive: iterative label cleaning, without which we just output predictions (11). iProb: iterative selection of top examples per class directly as column-wise maxima of $P$ (5) instead of iLC. Class: linear classifier used for prediction instead of LP, as in [63], with balancing still applied on output probabilities. †: default setting of iLPC.

<table>
<thead>
<tr>
<th>METHOD</th>
<th>NETWORK</th>
<th>mini/IMAGENET</th>
<th>tiered/IMAGENET</th>
<th>CIFAR-FS</th>
<th>CUB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1-shot</td>
<td>5-shot</td>
<td>1-shot</td>
<td>5-shot</td>
</tr>
<tr>
<td>LR+ICI [63]</td>
<td>ResNet-12A</td>
<td>66.80</td>
<td>79.26</td>
<td>80.79</td>
<td>87.92</td>
</tr>
<tr>
<td>LR+ICI [63]*</td>
<td>ResNet-12A</td>
<td>66.85±0.92</td>
<td>78.89±0.55</td>
<td>82.40±0.84</td>
<td>88.80±0.50</td>
</tr>
<tr>
<td>iLPC (ours)</td>
<td>ResNet-12A</td>
<td>69.79±0.99</td>
<td>79.82±0.55</td>
<td>83.49±0.84</td>
<td>89.48±0.67</td>
</tr>
<tr>
<td>PT+MAP [19]</td>
<td>WRN-28-10</td>
<td>82.92±0.26</td>
<td>88.82±0.13</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PT+MAP [19]*</td>
<td>WRN-28-10</td>
<td>82.88±0.73</td>
<td>88.78±0.40</td>
<td>88.15±0.71</td>
<td>92.32±0.40</td>
</tr>
<tr>
<td>LR+ICI [63]*</td>
<td>WRN-28-10</td>
<td>80.61±0.80</td>
<td>87.93±0.44</td>
<td>86.79±0.76</td>
<td>91.73±0.40</td>
</tr>
<tr>
<td>iLPC (ours)</td>
<td>WRN-28-10</td>
<td>83.05±0.79</td>
<td>88.82±0.42</td>
<td>88.50±0.73</td>
<td>92.46±0.42</td>
</tr>
</tbody>
</table>

Table 2. Transductive inference, comparison with LR+ICI [63] and PT+MAP [19]. *: our reproduction with official code on our datasets.

Tasks We consider $N$-way, $K$-shot classification tasks with $N = 5$ randomly sampled novel classes and $K \in \{1, 5\}$ randomly selected examples per class for support set $S$, that is, $L = 5K$ examples in total. For the query set $Q$, we randomly sample 15 additional examples per class, that is, $M = 75$ examples in total, which is the most common choice in the literature [36, 32, 66].

In the semi-supervised setting, the unlabeled set $U$ contains an additional number of randomly sampled examples per novel class. This number depends on $K$. We use two settings, namely 30/50 and 100/100, where the first number (30 or 100) refers to 1-shot and the second (50 or 100) to 5-shot. Again, these are the two most common choices in semi-supervised few shot learning [32, 63, 29, 50, 66].

Unless otherwise stated, we use 1000 tasks and report mean accuracy and 95% confidence interval on the test set.

Competitors As discussed in the supplementary material, there are several flaws in experimental evaluation in the liter-
with such reproduced results of three state-of-the-art methods: LR+ICI [63], PT+MAP [19] and MCT [29]. Only [63] is published, while the other two are pre-prints.

### Networks

We use publicly available pre-trained backbone convolutional neural networks that are trained on the base-class training set. We experiment with two popular networks, namely, the residual network ResNet-12 [42] and the wide residual network WRN-28-10 [52].

In particular, to compare with [63], we use pre-trained weights of the ResNet-12 provided by [63], which we call ResNet-12A, as well as official public code\(^1\) for testing. To compare with [19], we use pre-trained weights of a WRN-28-10 provided by [38]\(^2\), which are the same used by [19], as well as official public code\(^3\) for testing. To compare with [29], we use official public code\(^4\) to train from scratch another version of ResNet-12 used by [29], which we call ResNet-12B, as well as the same code for testing.

### Feature pre-processing

Each method uses its own feature pre-processing. LR+ICI [63] uses \(\ell_2\)-normalization and PCA to reduce ResNet-12A to 5 dimensions. PT+MAP [19] uses element-wise power transform, \(\ell_2\)-normalization and centering of WRN-28-10 features. MCT [29] uses flattening of the output tensor of ResNet-12B rather than spatial pooling. By default, we use the same choices as [19, 29] for WRN-28-10 and ResNet-12B. For ResNet-12A however, we use \(\ell_2\)-normalization only on transductive inference and we do not use any dimensionality reduction.

### Implementation details

We use PyTorch [43] and scikit-learn [44]. Label cleaning is based on a linear classifier on top of \(f\), initialized by imprinting the average of support features per class and then trained using (12). We use SGD with momentum 0.9 and weight decay 0.0005. We use a learning rate of \(\eta\) for 1000 iterations. For inductive (resp. semi-supervised) learning, we use logistic regression on support (resp. also pseudo-labeled) examples, learned using scikit-learn [63]. The row-wise sum \(p\) (9) is fixed to 1. The supplementary material includes more choices. It also includes inference time comparisons.

### 4.2. Ablation study

#### Hyperparameters

Our hyperparameters include \(\gamma\) and \(k\) used in the nearest neighbor graph (1), \(\alpha\) in label propagation (4), \(\tau\) in balancing (7) and the learning rate \(\eta\) of label cleaning. We optimize them on the validation set of every dataset. Common choices for \(k\) and \(\alpha\) are in \([15, 20]\) and in \([0.5, 0.8]\), respectively. We set \(\gamma = 3\), \(\tau = 3\) and \(\eta = 0.1\). More details and precise choices per dataset are given in the supplementary material.

\(^1\)https://github.com/Yikai-Wang/ICI-FSL
\(^2\)https://github.com/nupurkmr9/S2M2_fewshot
\(^3\)https://github.com/yhu01/PT-MAP
\(^4\)https://github.com/seongmin-kye/MCT
Algorithmic components  Table 1 ablates our method in the presence or not of individual components, as well as using alternative components. The use of queries with label propagation gives a gain of transductive over inductive inference, up to 6% in 1-shot, while being on par with the linear classifier in 5-shot. In 1-shot, balancing and iterative label cleaning bring another gain of 4-5% each independently, while the combination of the two brings 8-9%. The performance of iterative label cleaning is further justified by its superior of performance when compared to selecting examples based on $P$ instead.

4.3. Label cleaning: loss distribution

To illustrate our label cleaning, we conduct two experiments, showing the distribution of the loss value (12). In the first, shown in Figure 2(a,b), we inject label noise uniformly at random to the 20% (a) and 40% (b) of 500 labeled examples. The correctly and incorrectly labeled examples have very different loss distributions. Importantly, while previous work on noisy labels [3, 21, 57] attempts to detect clean examples by an optimal threshold on the loss value, we only need few clean examples per iteration. Examples with minimal loss value are clean.

The second experiment is on two novel 1-shot transductive tasks, shown in Figure 2(c,d). We use 50 unlabeled queries per class and we predict pseudo-labels according to (11). Label cleaning is more challenging here because the two distributions are more overlapping. This is natural because predictions are more informed than uniform, even if incorrect. Still, a large proportion of clean examples have a smaller loss value than the minimal value of noisy ones.

4.4. Effectiveness of class balancing

To show the effectiveness of our class balancing module, we carry out experiments in a novel setting for unbalanced few-shot transductive inference. In this setting, the number of queries per class for every few-shot task is drawn uniformly at random from \{10 \ldots 20\}. We use no balancing, or we use balancing with uniform class distribution (7), or, assuming the prior class distribution $u \in \mathbb{R}^N$ is known, we replace $\frac{1}{N}$ in (7) by $u$. As shown in Table 7, balancing improves accuracy by a large margin, but only if the prior class distribution is known, otherwise it is harmful.

4.5. Transductive inference

Table 2 compares our iLPC with LR+ICI [63] and PT+MAP [19] under the standard setting of 15 unlabelled queries per class. The truly fair comparison is with our reproductions, indicated by *. Apart from the default networks, we also use WRN-28-10 with LR+ICI [63], since it is more powerful. Our iLPC is on par with PT+MAP [19] under this setting and superior to LR+ICI [63] by up to 3% on miniImageNet 1-shot.

We also experiment with 50 unlabeled queries per class, or $M = 250$ in total. As shown in Table 3, the gain over PT+MAP [19] increases significantly, up to 2% on miniImageNet 1-shot. This can be attributed to the fact that PT+MAP [19] operates on Euclidean space, while we capture the manifold structure, which manifests itself in the presence of more data. A 10-shot experiment, is shown in Table 4. The gain is around 0.5%.

Table 5 shows that PT+MAP [19] is very sensitive to feature pre-processing, losing up to 40% without it, while our iLPC more robust, losing only up to 5%. Similarly, Figure 3 shows that LR+ICI [63] is sensitive to dimension reduction, working best at only 5 dimensions. By contrast, our iLPC is very stable and only fails at 2 dimensions.

Table 6 compares our iLPC with MCT [29]. We reproduce MCT results by training from scratch ResNet-12B using the official code and we test both methods without data augmentation (horizontal flipping) and without meta-learned scaling function. The objective is to compare the two transductive methods under the same backbone network and the same training process, which is clearly superior to ResNet-12A. Under these settings, MCT is slightly better in 5-shot but iLPC outperforms it by a large margin in 1-shot.

4.6. Semi-supervised learning

As shown in Table 8, iLPC is superior to LR+ICI [63] in all settings by an even larger margin than in transductive inference, e.g. nearly 3.5% in miniImageNet 1-shot. This can be attributed to capturing the manifold structure of the data, since there is more unlabeled data in this case. Because PT+MAP [19] does not experiment with semi-supervised learning, we adapt it in the same way as ours, using the default WRN-28-10, outperforming it in most experiments.
Table 8. Semi-supervised few-shot learning, comparison with [63, 19]. ∗: our reproduction with official code on our datasets. †: our reproduction with official code on our datasets.

<table>
<thead>
<tr>
<th>METHOD</th>
<th>NETWORK</th>
<th>SPLIT</th>
<th>miniI mageNET 1-shot</th>
<th>miniI mageNET 5-shot</th>
<th>tieredI mageNET 1-shot</th>
<th>tieredI mageNET 5-shot</th>
<th>Cifar-FS 1-shot</th>
<th>Cifar-FS 5-shot</th>
<th>CUB 1-shot</th>
<th>CUB 5-shot</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR+ICI [63]</td>
<td>ResNet-12A</td>
<td>30/50</td>
<td>69.66</td>
<td>80.11</td>
<td>84.01</td>
<td>89.00</td>
<td>76.51</td>
<td>84.32</td>
<td>89.58</td>
<td>92.48</td>
</tr>
<tr>
<td>LR+ICI [63]</td>
<td>ResNet-12A</td>
<td>30/50</td>
<td>67.57±0.07</td>
<td>79.07±0.48</td>
<td>83.32±0.61</td>
<td>89.06±0.51</td>
<td>75.99±0.08</td>
<td>84.01±0.62</td>
<td>88.50±0.71</td>
<td>-</td>
</tr>
<tr>
<td>iLPC (ours)</td>
<td>ResNet-12A</td>
<td>30/50</td>
<td>70.99±0.11</td>
<td>81.06±0.49</td>
<td>85.04±0.76</td>
<td>89.63±0.47</td>
<td>78.57±0.80</td>
<td>85.84±0.56</td>
<td>90.11±0.64</td>
<td>-</td>
</tr>
<tr>
<td>LR+ICI [63]</td>
<td>WRN-28-10</td>
<td>30/50</td>
<td>81.31±0.44</td>
<td>88.53±0.45</td>
<td>88.48±0.67</td>
<td>92.03±0.43</td>
<td>86.03±0.77</td>
<td>89.57±0.51</td>
<td>90.82±0.69</td>
<td>-</td>
</tr>
<tr>
<td>PT+MAP [19]†</td>
<td>WRN-28-10</td>
<td>30/50</td>
<td>83.14±0.32</td>
<td>88.95±0.38</td>
<td>89.16±0.64</td>
<td>92.30±0.39</td>
<td>87.05±0.48</td>
<td>89.98±0.49</td>
<td>91.52±0.83</td>
<td>-</td>
</tr>
<tr>
<td>iLPC (ours)</td>
<td>WRN-28-10</td>
<td>30/50</td>
<td>83.58±0.19</td>
<td>89.68±0.37</td>
<td>89.35±0.66</td>
<td>92.61±0.39</td>
<td>87.03±0.72</td>
<td>90.34±0.50</td>
<td>91.69±0.85</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 9. Transductive inference state of the art. ∗: our reproduction with official code on our datasets. †: tieredImageNet as reported by [37].

<table>
<thead>
<tr>
<th>METHOD</th>
<th>NETWORK</th>
<th>SPLIT</th>
<th>miniI mageNET 1-shot</th>
<th>miniI mageNET 5-shot</th>
<th>tieredI mageNET 1-shot</th>
<th>tieredI mageNET 5-shot</th>
<th>Cifar-FS 1-shot</th>
<th>Cifar-FS 5-shot</th>
<th>CUB 1-shot</th>
<th>CUB 5-shot</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR+ICI [63]</td>
<td>ResNet-12A</td>
<td>30/50</td>
<td>66.85±0.02</td>
<td>78.89±0.55</td>
<td>82.40±0.84</td>
<td>88.80±0.50</td>
<td>75.36±0.76</td>
<td>84.57±0.77</td>
<td>86.53±0.79</td>
<td>92.11±0.35</td>
</tr>
<tr>
<td>CAN+Top-k [16]</td>
<td>ResNet-12</td>
<td>30/50</td>
<td>67.19±0.55</td>
<td>80.64±0.35</td>
<td>73.21±0.58</td>
<td>84.93±0.38</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DPGN [65]</td>
<td>ResNet-12</td>
<td>30/50</td>
<td>67.77±0.32</td>
<td>84.60±0.43</td>
<td>72.45±0.51</td>
<td>87.24±0.39</td>
<td>77.90±0.50</td>
<td>90.20±0.40</td>
<td>75.71±0.47</td>
<td>91.48±0.33</td>
</tr>
<tr>
<td>MCT (instance)</td>
<td>ResNet-12B</td>
<td>30/50</td>
<td>78.55±0.06</td>
<td>86.03±0.42</td>
<td>82.32±0.81</td>
<td>87.36±0.90</td>
<td>85.61±0.69</td>
<td>90.03±0.46</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>EP [51]</td>
<td>WRN-28-10</td>
<td>30/50</td>
<td>70.74±0.05</td>
<td>84.34±0.55</td>
<td>78.50±0.91</td>
<td>88.36±0.57</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SIB [17]†</td>
<td>WRN-28-10</td>
<td>30/50</td>
<td>70.00±0.00</td>
<td>79.20±0.40</td>
<td>72.90</td>
<td>82.80</td>
<td>80.00±0.60</td>
<td>85.3±0.40</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SIB+E³BM [37]</td>
<td>WRN-28-10</td>
<td>30/50</td>
<td>71.40±0.50</td>
<td>81.20±0.40</td>
<td>75.60±0.6</td>
<td>84.30±0.4</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>LaplacianShot [69]</td>
<td>WRN-28-10</td>
<td>30/50</td>
<td>74.86±0.19</td>
<td>84.13±0.14</td>
<td>80.18±0.21</td>
<td>87.56±0.15</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PT+MAP [19]†</td>
<td>WRN-28-10</td>
<td>30/50</td>
<td>82.88±0.73</td>
<td>88.78±0.40</td>
<td>88.15±0.71</td>
<td>92.32±0.40</td>
<td>86.91±0.72</td>
<td>90.50±0.49</td>
<td>91.37±0.61</td>
<td>93.93±0.32</td>
</tr>
<tr>
<td>iLPC (ours)</td>
<td>WRN-28-10</td>
<td>30/50</td>
<td>83.05±0.19</td>
<td>88.82±0.42</td>
<td>88.50±0.75</td>
<td>92.46±0.42</td>
<td>86.51±0.75</td>
<td>90.60±0.48</td>
<td>91.03±0.63</td>
<td>94.11±0.30</td>
</tr>
</tbody>
</table>

Table 10. Semi-supervised few-shot learning state of the art. †: as reported by [66].

4.7. Comparison with the state of the art

Table 9 and Table 10 compare our iLPC with a larger collection of recent methods on the transductive and semi-supervised settings, respectively. Even when the network and data split appears to be the same, we acknowledge that our results are not directly comparable with any method other than our reproductions. As discussed in the supplementary material, this is due to the very diverse choices made in the bibliography, e.g. versions of network, training settings, versions of datasets, or pre-processing. For instance, ResNet-12 is different than either ResNet-12A or ResNet-12B.

For this reason, we focus on the best result by each method, including ours. Necessarily, methods experimenting with WRN-28-10 have an advantage. Still, at least among those, iLPC performs best by a large margin in both settings, with the closest second best being PT+MAP [19].

5. Conclusion

Our solution is conceptually simple and combines in a unique way ideas that have been successful in problems related to our task at hand. Label propagation exploits the manifold structure of the data, which becomes important in the presence of more data, while still being competitive otherwise. Class balancing provides a strong hint in correcting predictions when certain classes dominate. Label cleaning, originally introduced for learning with noisy labels, is also very successful in learning predicted pseudo-labels. Iterative reuse of few pseudo-labels as true labels bypasses the difficulty of single-shot detection of clean examples.

Importantly, reasonable baselines, like predicting pseudo-labels by a classifier or iteratively re-using pseudo-labels without cleaning, fail completely. When compared under fair settings, our iLPC outperforms or is on par with state-of-the-art methods. It is also significantly more robust against feature pre-processing on which other methods rely.
References


