Pattern Recognition 161 (2025) 111304

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Exploiting unlabeled data in few-shot learning with manifold similarity and label cleaning

Michalis Lazarou [a],*, Tania Stathaki [a], Yannis Avrithis [b]

[a] Imperial College London, United Kingdom
[b] Institute of Advanced Research on Artificial Intelligence (IARAI), Austria

ARTICLE INFO

Keywords:
Computer vision
Few-shot learning
Semi-supervised learning
Transductive learning

ABSTRACT

Few-shot learning investigates how to solve novel tasks given limited labeled data. Exploiting unlabeled data along with the limited labeled has shown substantial improvement in performance. In this work we propose a novel algorithm that exploits unlabeled data in order to improve the performance of few-shot learning. We focus on transductive few-shot inference, where the entire test set is available at inference time, and semi-supervised few-shot learning where unlabeled data are available and can be exploited. Our algorithm starts by leveraging the manifold structure of the labeled and unlabeled data in order to assign accurate pseudo-labels to the unlabeled data. Iteratively, it selects the most confident pseudo-labels and treats them as labeled improving the quality of pseudo-labels at every iteration. Our method surpasses or matches the state of the art results on four benchmark datasets, namely *mini*ImageNet, *tiered*ImageNet, CUB and CIFAR-FS, while being robust over feature pre-processing and the quantity of available unlabeled data. Furthermore, we investigate the setting where the unlabeled data contains data from distractor classes and propose ideas to adapt our algorithm achieving new state of the art performance in the process. Specifically, we utilize the unnormalized manifold class similarities obtained from label propagation for pseudo-label cleaning and exploit the uneven pseudo-label distribution between classes to remove noisy data. The publicly available source code can be found at https://github.com/MichalisLazarou/iLPC.

1. Introduction

Deep learning has changed the field of pattern recognition with many breakthroughs over the last decade. However, a fundamental limitation of deep learning models is their reliance on large labeled datasets which can be very costly to obtain due to the requirement of expensive human labor. Moreover, in certain applications, such as rare species classification the data itself is scarce, which makes it very difficult to train deep learning models.

Few-shot learning attempts to address the aforementioned limitations by investigating how to make deep learning models learn from limited labeled data. Some of the most popular research directions in few-shot learning include: *meta-learning* [1], *transfer learning* [2], synthetic data generation [3] and exploiting unlabeled data [4]. In this work we focus on the research direction of exploiting unlabeled data in order to improve the performance of few-shot learning. This direction is very attractive because unlabeled data is much easier to obtain than labeled, even in the few-shot regime. *Transductive* [5] and *semi-supervised few-shot learning* methods [6] utilize labeled and unlabeled data at the same time and have shown substantial superiority over their traditional inductive counterparts. This is because labeled and unlabeled data can be exploited at the same time to make more accurate predictions.

Even though exploiting unlabeled data brings impressive performance improvements, there are still limitations. First, it is challenging to train neural networks from scratch using limited labeled and unlabeled data. For this reason, we use strong pre-trained networks since it has been shown to be very effective in few-shot learning [7]. Second, assigning accurate pseudo-labels to the unlabeled data can be challenging. Graph-based methods such as *label propagation* [8] have been successful in this respect by relying on the underlying manifold structure of the labeled and unlabeled data to assign accurate pseudo-labels to the unlabeled data. However, the pseudo-labels obtained by label propagation are still not accurate enough and identifying the ones that are accurate is another limitation.

In the research direction of *learning with noisy labels*, several methods have been introduced to detect noisy labels in order to remove the corresponding data from the training set [9]. For example, a way to "clean" noisy labels is by using a small capacity classifier and filtering input data and target labels according to their loss value

* Corresponding author.
E-mail addresses: michalis.lazarou14@imperial.ac.uk (M. Lazarou), t.stathaki@imperial.ac.uk (T. Stathaki), yannis@avrithis.net (Y. Avrithis).

https://doi.org/10.1016/j.patcog.2024.111304
Received 30 March 2023; Received in revised form 18 December 2024; Accepted 19 December 2024
Available online 28 December 2024

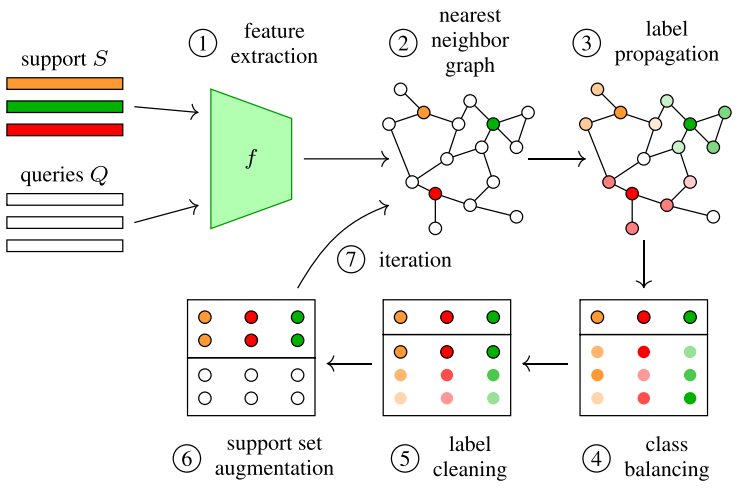

Fig. 1. Visualization of the proposed method. See text for details.

statistics [10]. However, all these methods focus on the scenario of noisy labels obtained by weak human annotation, or, for the sake of controllable experiments, obtained at random. To our knowledge, despite the plethora of relevant label cleaning solutions, none has been used on pseudo-labels assigned to unlabeled data.

In this work, inspired from ideas in *semi-supervised learning* and *learning from noisy labels*, we introduce a novel method that iteratively assigns accurate pseudo-labels to the unlabeled data and selects the most confident pseudo-labels to augment the labeled dataset. Focusing on transductive inference, Fig. 1 provides a visualization of our method. Specifically, a set of labeled support examples S and unlabeled queries Q are provided and are embedded in the feature space by using network f. Label propagation [11] is carried out in order to obtain the manifold class-similarity matrix that associates examples to the support classes. The submatrix corresponding to unlabeled examples, P, is normalized over examples and classes using the Sinkhorn–Knopp algorithm [12], assuming a uniform distribution over classes. Pseudo-labels are extracted from P, and by using the label cleaning method O2U-Net [10] only the most confident examples are retained per class. Finally, we move these examples from Q to S and iterate this procedure until Q is empty.

Furthermore, we investigate the more challenging distractive semi-supervised few-shot setting, where the unlabeled data contains data from *distractor* classes which are classes that are not represented in the support set. This is a more realistic setting that reflects real-world scenarios, where unlabeled data may not necessarily belong to the classes of interest. We discover that utilizing the *manifold class similarity matrix* resulting from label propagation for pseudo-label cleaning significantly improves the performance in the distractive setting. Specifically, the cleaning process assumes most pseudo-labels are correct, but distractors introduce more label noise. To reduce this noise, the absolute manifold similarity for each example can help distinguish distractor examples, since their similarity to every support class is expected to be lower than unlabeled examples of every support class. In addition, we hypothesize that removing class balancing will be beneficial in this setting because each distractor class will be semantically more similar to one or a few of the support classes, thus most of the distractor examples will be pseudo-labeled in those support classes, leaving the remaining classes less affected. That is, in the presence of distractors, we expect the distribution of pseudo-labels over classes to be imbalanced. Using a simple truncation mechanism, we can remove the distractor examples to further improve the performance. This works because we leverage O2U-Net, which iteratively selects the most confident examples for each class, ensuring that correctly pseudo-labeled examples are selected first leaving the distractor examples to be selected last.

The contributions of this work are the following:

1. We are the first to propose the synergistic approach of predicting pseudo-labels by leveraging the data manifold and interpreting confident pseudo-label prediction as a form of label cleaning as a way to improve the performance of transductive, semi-supervised and distractive semi-supervised few-shot learning.
2. We investigate the more realistic setting of *distractive semi-supervised learning* and propose three ideas to address it effectively.
3. Our algorithm achieves new state of the art performance or is on par with other state of the art methods in transductive, semi-supervised and distractive semi-supervised few-shot learning.

Extensions A preliminary version of our work was published as a conference paper [13]. Here we extend this work by making a number of contributions as follows:

(2.1) We investigate the distractive semi-supervised few-shot learning setting for our method as well as for LR+ICI [6], PT+MAP [5] and PLCM [14].
(2.2) We propose three ideas to improve the performance in this setting, namely: removing class balancing, using a *truncation* method to remove distractor examples and using unnormalized *manifold class similarities* obtained from label propagation in the pseudo-label cleaning process.
(2.3) We analyze and explain why each idea improves the performance in the distractive setting.

2. Related work

2.1. Few-shot learning

Meta-learning This is a popular paradigm, where the training set is partitioned in episodes resembling in structure the novel tasks. There are three main meta-learning research directions, model-based, optimization-based and metric-based. *Model-based* methods rely on the properties of specific model architectures, such as recurrent Neural Networks (RNN) [15] and memory-augmented networks [16]. *Optimization-based* methods focus on learning a robust model initialization through bi-level optimization. Some prominent ideas include gradient-based solutions [1], using another neural network such as an LSTM [17] and interpreting the inner level optimization as the denoising stage of diffusion models [18]. *Metric-based* methods attempt to compare examples in the embedding space by comparing an individual query to

all examples of a class [19], to the class prototype [20] or by comparing the similarity between two examples using Siamese networks [21].

Predicting weights, data augmentation Another line of research consists of generating new parameters or even data. For instance, it is common to learn to predict data-dependent network parameters in the last layer (classifier) [22] or even in intermediate convolutional layers [23]. Alternatively, one can learn to generate novel-task data by using state of the art generative models such as GANs [24], diffusion models [25] or other specialized few-shot generative models [3].

Transfer learning More recently, it has been shown that learning a powerful representation on the entire training set is more effective than sampling few-shot training episodes that resemble novel tasks [2]. In doing so, one may use standard loss functions [2], knowledge distillation [26], other common self-supervision and regularization methods [7] or use multi-level feature training [27]. A complementary research direction is exploit the relationship between the available embeddings and obtain more discriminative embeddings by fusing the original ones [28]. We follow this *transfer learning* approach, which allows us to decouple representation learning from the core few-shot learning idea and provide fairer comparisons with the competition.

2.2. Few-shot learning using unlabeled data

Leveraging unlabeled data is of great interest in the few-shot learning research community due to the ease of obtaining them. Two common settings that investigate how to exploit unlabeled data are transductive few-shot learning and semi-supervised few-shot learning.

Transductive few-shot learning In this setting, all novel-class unlabeled query examples are assumed available at inference time. 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* [29] and *embedding propagation* [30], that smooths the decision boundaries and improves the robustness to noise. *Meta-confidence transduction* (MCT) [31] meta-learns a data-dependent scaling function that normalizes every example and iteratively updates class centers. *Laplacian shot* [32] proposed to use Laplacian regularization in order to encourage nearby queries to have the same label. PT+MAP [5] uses a soft-k means iterative approach to update class centers as well as balance over classes. *Cross-attention* [33], 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. *Relation fusion propagation network* [34] simultaneously models both feature and label relationships through the use of a graph neural network. *A2LP* [35] proposes a novel label propagation algorithm with superior performance while *TIM* [36] proposes a novel function that adapts the weights of a classifier by maximizing the mutual information between query features and their label predictions. *EASE* [37] learns a discriminant subspace by maximizing the inter-class distance and minimizing the intra-class distance. *Adaptive manifold* [38] and *ProtoLP* [39] explore how to utilize label propagation with iterative centroid refinement. *Transductive CLIP* [40] proposes to combine visual and text information through vision-language model.

Semi-supervised few-shot learning In this case, labeled novel-class support examples and additional unlabeled data is given. A classifier may be learned on both to make predictions on novel-class queries.

One of the first works that addresses this setting uses unlabeled examples to adapt prototypical networks [20], while discriminating from distractor classes [4]. Common semi-supervised solutions are also adapted to few-shot classification, for instance *learning to self-train* [41], which adapts *pseudo-label* [42] and *TransMatch* [43], which is an adaptation of MixMatch [44]. *Instance credibility inference* [6] predicts pseudo-labels iteratively, using a linear classifier to select the most confident pseudo-labels to augment the support set. *PTN* [45] proposes to improve the capacity of graph-based semi-supervised methods in message passing by adapting the Poisson learning method [46]. *Cluster-FLS* [47] proposes a clustering method that uses both labeled and unlabeled examples to produce accurate pseudo-labels.

Distractive semi-supervised few-shot learning A more realistic semi-supervised setting is *distractive semi-supervised few-shot learning*, where the unlabeled data contains data from *distractor* classes that are not represented in the support set. The majority of works in the few-shot learning literature do not consider this setting. We discuss here the few exceptions that we are aware of.

The first work to address this setting [4] extended the prototypical networks [20] by using an additional prototype for the distractor classes. It also proposed a soft-masking mechanism with the intuition that the unlabeled examples that are closer to the class prototypes are more important than the ones that are further away. Our work differs from [4] in that we do not use prototypes and we use manifold similarity instead of Euclidean distance.

Some of the works discussed in Section 2.2 also provide experimental results on the distractive setting without addressing it directly. In contrast to these works, we explicitly address this setting by proposing ways of adapting our method to improve its performance.

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 train a backbone network $f : \mathcal{X} \to \mathbb{R}^d$ which maps an image from 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* [19], a novel task is defined by sampling a *support set* S from D_{novel}, consisting of N classes, denoted as C_{supp} 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_{novel}. Queries are treated independently of each other. This is referred to as *inductive inference*.

In *transductive inference* [29], a *query set* Q consisting of M unlabeled examples is also sampled from D_{novel}. Given f, S and Q, the problem is to make predictions on Q, without necessarily learning a classifier. In doing so, one may exploit the distribution of examples in Q, which is important because M is assumed greater than L. We denote by $T := L + M$ the total number of labeled support examples and unlabeled queries.

In the *semi-supervised* setting [41], an unlabeled set U of M unlabeled examples is also sampled from D_{novel}. Given f, S and U, the problem is to learn to make predictions on new queries from D_{novel}, as in the inductive case. Again, $M > L$ and we may use the distribution of U.

In the *distractive semi-supervised* setting, we additionally sample an unlabeled set U_{dist} from D_{novel} of M_{dist} unlabeled examples from a set C_{dist} of distractor classes. The support and distractor classes are distinct: $C_{\text{supp}} \cap C_{\text{dist}} = \emptyset$. We denote by $U' := U \cup U_{\text{dist}}$ the set of unlabeled and distractor examples and by $M' := M + M_{\text{dist}}$ their total number.

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 construction

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 $y_i \in [N] := \{1, \dots, N\}$. We embed all examples from S and Q into $V = \{\mathbf{v}_1, \dots, \mathbf{v}_T\} \subset \mathbb{R}^d$ and ℓ_1-normalize them, where $\mathbf{v}_i := f(x_i)$ for $i \in [T]$. Following [8], we construct a k-nearest neighbor graph of the features in V, represented by a sparse $T \times T$ non-negative *affinity matrix* A, with

$$A_{ij} := \begin{cases} [\mathbf{v}_i^\top \mathbf{v}_j]_+^\gamma, & \text{if } i \neq j \wedge \mathbf{v}_i \in \mathrm{NN}_k(\mathbf{v}_j) \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

for $i \in [T]$, $j \in [T]$, where $\mathrm{NN}_k(\mathbf{v})$ are the k-nearest neighbors of $\mathbf{v}$ 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

$$\mathcal{W} := D^{-1/2} W D^{-1/2}, \tag{2}$$

where $D = \mathrm{diag}(W\mathbf{1}_T)$ is the $T \times T$ degree matrix of W, $\mathrm{diag}(W\mathbf{1}_T)$ is the square matrix where the values in the diagonal are the values of vector $W\mathbf{1}_T$ while the rest of the matrix is set to zero and $\mathbf{1}_T$ is the all-ones vector of size T.

3.3. Label propagation

Following [11], we define the $T \times N$ *label matrix* Y as

$$Y_{ij} := \begin{cases} 1, & \text{if } i \leq L \wedge y_i = j \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

for $i \in [T]$, $j \in [T]$. 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

$$Z := (I - \alpha\mathcal{W})^{-1} Y, \tag{4}$$

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 [11]. We call Z the *manifold class similarity* or simply *class similarity* matrix, because every row i expresses how similar example x_i is to each of the N support classes. However, before making predictions, we balance over classes.

3.4. Class balancing

We focus on the $M \times N$ submatrix

$$P := Z_{L+1:T,:} \tag{5}$$

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

$$P_{ij} \leftarrow P_{ij}^{\tau} \tag{6}$$

for $i \in [M]$, $j \in [N]$, where $\tau > 1$, encouraging hard predictions. Parameter τ is analogous to the *scale* (or *inverse temperature*) of logits in softmax-based classifiers [22].

Inspired by [5], we normalize P to a given row-wise sum $\mathbf{p} \in \mathbb{R}^M$ and column-wise sum $\mathbf{q} \in \mathbb{R}^N$. Each element $p_i \in [0, 1]$ of $\mathbf{p}$ represents a confidence of example x_{L+i} for $i \in [M]$; it can be a function of the ith row of P or set to 1. Each element $q_j \geq 0$ of $\mathbf{q}$ represents a weight of class j for $j \in [N]$. In the absence of such information, we set

$$\mathbf{q} := \frac{1}{N}(\mathbf{p}^\top \mathbf{1}_M)\mathbf{1}_N, \tag{7}$$

assuming a uniform distribution of queries over classes.

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

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

We use the *Sinkhorn–Knopp* algorithm [12] for this projection, which alternates between rescaling the rows of P to sum to $\mathbf{p}$ and its columns to sum to $\mathbf{q}$,

$$P \leftarrow \mathrm{diag}(\mathbf{p})\,\mathrm{diag}(P\mathbf{1}_N)^{-1} P \tag{9}$$

$$P \leftarrow P\,\mathrm{diag}(P^\top \mathbf{1}_M)^{-1}\,\mathrm{diag}(\mathbf{q}), \tag{10}$$

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

$$\hat{y}_{L+i} := \arg\max_{j \in [N]} P_{ij} \tag{11}$$

that corresponds to the maximum element of the ith 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 [42], such that the quality of pseudo-labels improves with training. Since data is limited here, we would like to select pseudo-labeled examples in Q that are most likely to be correct, treat 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 [10]. 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 correct labels are more likely to have a lower loss value than examples with noisy labels. We can thus use the loss value as a *confidence score* for selection.

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

$$\ell := -\sum_{i=1}^{L} \log g(x_i)_{y_i} - \sum_{i=1}^{M} \log g(x_{L+i})_{\hat{y}_{L+i}}. \tag{12}$$

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.

For $i \in [M]$, the loss term

$$\ell_i := -\log g(x_{L+i})_{\hat{y}_{L+i}}, \tag{13}$$

corresponding to the pseudo-labeled query x_{L+i}, is used as confidence score. Following O2U-Net [10], we use large learning rate and collect the average loss $\bar{\ell}_i$ over all epochs, for $i \in [M]$. We leverage these loss statistics and select the queries having the least average loss [10].

The extreme case is to select one pseudo-labeled example per class. In particular, let

$$\mathcal{L}_j := \{i \in [M] : \hat{y}_{L+i} = j\} \tag{14}$$

be the index set of examples that are pseudo-labeled in class j for $j \in [N]$. Then, the index set of selected examples is

$$\mathcal{I} := \left\{ \arg\min_{i \in \mathcal{L}_j} \bar{\ell}_i : j \in [N] \right\}. \tag{15}$$

That is, for each class $j \in [N]$, we select the example x_{L+i} with the minimum value $\bar{\ell}_i$ from the subset of examples in $\mathcal{L}_j$. 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.

$$S \leftarrow S \cup \{(x_{L+i}, \hat{y}_{L+i})\}_{i \in \mathcal{I}} \tag{16}$$

$$Q \leftarrow Q \setminus \{x_{L+i}\}_{i \in \mathcal{I}} \tag{17}$$

3.6. Iterative inference

Although label propagation and class balancing make predictions on the entire unlabeled query set Q, we apply cleaning to select ν pseudo-labeled examples per class, which we move from Q to the support set S. We iterate the entire process, selecting ν pseudo-labeled examples 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 predicted in the next iteration.

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 [10]. We select one query per class with the least average loss over all epochs (15), which we move from Q to S as labeled (16), (17). 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.

3.7. Distractor classes

In the more realistic setting of distractive semi-supervised learning, part of the unlabeled data does not belong to the support classes. We experiment with three ideas to address this challenge, discussed below. We will use a simple running example to explain our ideas intuitively: two in-distribution classes, "dog" and "car", and a distractor class, "wolf", which is semantically more similar to "dog".

3.7.1. Removing class balancing

While we could still use class balancing, we rather choose to remove it completely, motivated by the hypothesis discussed in Section 1: removing class-balancing will lead to most of the distractor examples being classified in the most semantically similar support classes, leaving the remaining classes less affected. Referring to our running example, "wolf" examples will be mostly classified as "dog" among support classes.

In particular, in the absence of balancing, we can no longer select one pseudo-labeled example per class at every iteration as described in Section 3.5. Instead, if there are no pseudo-labels remaining in a given class, we continue iterating without selecting any pseudo-labeled example from that class.

3.7.2. Truncation

Following our intuition that each distractor class will be semantically more similar to one or a few support classes, we expect that some support classes will contain much more pseudo-labeled examples than others. However, we still expect the support examples to be classified with more confidence than the distractor examples. In our running example, we expect the "dog" examples to be selected before the "wolf" examples in augmenting the support set S.

Using this intuition, we only keep the first R examples of every class, effectively removing the majority of distractor examples in the process. We set R to the least number of pseudo-labeled examples over all classes. In particular, we simply terminate the iteration when the pseudo-labeled examples of any class are exhausted.

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 $|S| = L + M$

repeat
- $\mathcal{W} \leftarrow \text{GRAPH}(f, S, Q; \gamma, k)$ ▷ adjacency matrix (1),(2)
- $Y \leftarrow \text{LABEL}(S)$ ▷ label matrix (3)
- $Z \leftarrow \text{LP}(\mathcal{W}, Y; \alpha)$ ▷ label propagation (4)
- $P \leftarrow Z_{L+1:L+M,:}$ ▷ unlabeled submatrix (5)
- $P \leftarrow \text{POWER}(P; \tau)$ ▷ power transform (6)
- $(\mathbf{p}, \mathbf{q}) \leftarrow \text{BALANCE}(P)$ ▷ class balance (7)
- $P \leftarrow \text{SINKHORN}(P; \mathbf{p}, \mathbf{q})$ ▷ Sinkhorn-Knopp (9),(10)
- $\hat{Y} \leftarrow \text{PREDICT}(P)$ ▷ pseudo-labels (11)
- $\mathcal{I} \leftarrow \text{CLEAN}(f, S, Q, \hat{Y}, \mathbf{p})$ ▷ label cleaning (12),(15)
- $(S, Q) \leftarrow \text{AUGMENT}(S, Q, \mathcal{I})$ ▷ augment support (16),(17)

until $Q = \emptyset$ ▷ all queries are predicted

3.7.3. Unnormalized class similarities

The cleaning process of Section 3.5 assumes that most of the pseudo-labels obtained by label propagation and class balancing are correct. In the presence of distractors however, the label noise will be much higher. A way to suppress label noise is to use the unnormalized class similarity matrix Z instead of the normalized probability matrix P.

In particular, in matrix Z obtained by label propagation (4), every row i expresses how similar example x_i is to the N support classes. For $i \in [M']$, let

$$z_i := \max_{j \in [N]} Z_{L+i,j} \tag{18}$$

be the maximum similarity over classes for example x_{L+i}. We expect this similarity to be low for examples of distractor classes, which can be used to distinguish them from support examples. By contrast, in the probability matrix P after normalization, this information is lost.

We investigate two variants of the idea of using Z for cleaning, as follows.

Class similarities only Rather than using the average loss $\bar{\ell}_i$ of every example as confidence score (Section 3.5), we use the unnormalized class similarity z_i (18) directly. In the extreme case of selecting one example per class, the index set of selected examples is

$$\mathcal{I} := \left\{ \arg\max_{i \in \mathcal{L}'_j} z_i \,:\, j \in [N] \right\}. \tag{19}$$

That is, for each class $j \in [N]$, we select the example x_{L+i} with the minimum value z_i from $\mathcal{L}'_j$, where in this case

$$\mathcal{L}'_j := \{ i \in [M'] \,:\, \hat{y}_{L+i} = j \} \tag{20}$$

is the index set of examples that are pseudo-labeled in class j for $j \in [N]$. We will refer to this variant as iLPC$_z$ in the following, including truncation and no class balancing.

Class similarities for loss weighting As a second variant, we use both the unnormalized class similarities and the loss statistics to define a confidence score for cleaning. In particular, we use label cleaning as described in Section 3.5, calculating the average loss $\bar{\ell}_i$ for every pseudo-labeled example x_{L+i}, for $i \in [M']$. We then weigh $\bar{\ell}_i$ by $z_i^{-\beta}$, where z_i is the corresponding class similarity and $\beta > 0$:

$$\hat{\ell}_i := z_i^{-\beta} \bar{\ell}_i. \tag{21}$$

The selection process is exactly the same as in (15) or (19), now using the *weighted average loss* $\hat{\ell}_i$ instead of $\bar{\ell}_i$ or z_i. We will refer to this variant as iLPC$_{zl}$ in the following, including truncation and no class balancing.

4. Experiments

4.1. Setup

Datasets We use four common few-shot classification benchmark datasets, *mini*ImageNet [19], *tiered*ImageNet [4], CUB [2] and CIFAR-FS [2]. We follow the experimental protocol of [13] for every dataset and backbone accordingly.

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 [41].

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. Therefore the total number of unlabeled examples in U, is $M = 150$ and $M = 250$ in the 1-shot and 5-shot setting respectively in the 30/50 setting and $M = 500$ in the 100/100 setting. Again, these are the two most common choices in semi-supervised few shot learning as shown in [6,41].

In the *distractive* semi-supervised setting, the unlabeled set $U' := U \cup U_{\text{dist}}$ additionally contains unlabeled data from distractor classes C_{dist}, which are mutually exclusive with the support classes C_{supp}. We follow the same experimental setting as [41], where $|C_{\text{dist}}| = 3$ and the number of examples per distractor class in U_{dist} is the same as the number of examples per support class in U, that is, 30 in 1-shot and 50 in 5-shot tasks in the 30/50 setting. This means that the total number of unlabeled examples, $M' = M + M_{\text{dist}}$ is 240 in 1-shot and 400 in 5-shot tasks for the 30/50 setting.

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

Competitors There are several flaws in experimental evaluation in the literature, like the use of different networks, training, versions of datasets and feature pre-processing. Fair comparison is impossible unless one uses public code to reproduce results under exactly the same setup.

In this work, we provide completely fair comparisons with four state-of-the-art methods by using their publicly available source code, reproducing their results. This means that all methods are compared fairly under exactly the same experimental evaluation. The state-of-the-art methods that we provide fair comparisons are: LR+ICI [6], PT+MAP [5], MCT [31] and PLCM [14].

PLCM utilizes training episodes from D_{base} to train a GMM model that learns the loss distribution of the pseudo-labels and provides a credibility score based on that distribution. The training episodes consist of the same settings as described in Section 4.1, that is 30 and 50 unlabeled examples per class in the 1-shot and 5-shot settings respectively. Regarding the 1-shot CUB experiment in the distractive setting since not all classes from D_{base} have enough examples to satisfy the 30 unlabeled examples per class, we utilized training episodes of 28 unlabeled examples per class.

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 [15] and the wide residual network WRN-28-10 [7].

In particular, to compare with [6], we use pre-trained weights of the ResNet-12 provided by [6], which we call ResNet-12A, as well as official public code[1] for testing. To compare with [5], we use pre-trained weights of a WRN-28-10 provided by [7],[2] which are the same used by [5], as well as official public code[3] for testing. To compare with [31], we use the official public code[4] to train from scratch another version of ResNet-12 used by [31], which we call ResNet-12B, as well as the same code for testing. To compare with PLCM [14], we use the official public code.[5]

Table 1
Selected hyperparameters. *m*IN: *mini*ImageNet. *t*IN: *tiered*ImageNet. CFS: CIFAR-FS.

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

Feature pre-processing Each method uses its own feature pre-processing. LR+ICI [6] uses ℓ_2-normalization and PCA to reduce ResNet-12A to 5 dimensions. PT+MAP [5] uses element-wise power transform, ℓ_2-normalization and centering of WRN-28-10 features. MCT [31] uses flattening of the output tensor of ResNet-12B rather than spatial pooling. By default, we use the same choices as [5] for WRN-28-10 and [31] for ResNet-12B. For ResNet-12A however, we use ℓ_2-normalization only on transductive inference and we do not use any dimensionality reduction.

Implementation details Our implementations are based on PyTorch [48] and scikit-learn [49]. 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 stochastic gradient descent with momentum 0.9 and weight decay 0.0005. We use a learning rate of η for 1000 iterations. For inductive (resp. semi-supervised) learning, we use logistic regression on support (resp. also pseudo-labeled) examples, learned using scikit-learn [6]. The row-wise sum $\mathbf{p}$ (9) is set to 1 and the column-wise sum $\mathbf{q}$ (10) is set to $\frac{M}{N}$ unless otherwise stated.

Hyperparameters Our hyperparameters include γ and k used in the nearest neighbor graph (1), α in label propagation (4), τ in balancing (7) and the learning rate η of label cleaning. Common choices for k and α are in $[15, 20]$ and in $[0.5, 0.8]$, respectively. We set $\gamma = 3$, $\tau = 3$ and $\eta = 0.1$. Following [6], we set $\nu = 3$ and select the 3 examples per class having the least average loss at every iteration in the transductive setting. We set $\nu = 1$ and $\nu = 5$ in the 1-shot and 5-shot settings respectively in the semi-supervised and distractive semi-supervised setting. Regarding the hyper-parameters for the distractive setting, we set $\beta = 5$. Apart from α and K we optimize our hyperparameters using the *mini*ImageNet validation set. Regarding the values of α and K we optimize them using the validation set of every dataset. The hyperparameters chosen for every dataset can be seen in Table 1

4.2. Ablation study

4.2.1. Transductive inference

Algorithmic components Table 2 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

[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.
[5] https://github.com/freeEntropy/PLCM.

Table 2
Ablation study of algorithmic components of our method iLPC on *mini*ImageNet. Inductive: baseline using only support examples. LP: label propagation. Balance: class balancing (7). iLC: 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 [6], with balancing still applied on output probabilities.

Inference	Components					ResNet-12A		WRN-28-10	
	LP	Balance	iLC	iProb	Class	1-shot	5-shot	1-shot	5-shot
Inductive					✓	56.30±0.62	75.59±0.47	68.17±0.60	84.33±0.43
Transductive	✓					61.09±0.70	75.32±0.50	74.24±0.68	84.09±0.42
Transductive		✓			✓	58.39±0.63	76.62±0.49	70.92±0.61	86.14±0.40
Transductive	✓	✓				65.04±0.75	76.82±0.50	79.42±0.69	85.34±0.43
Transductive			✓		✓	65.54±0.87	78.76±0.53	80.29±0.76	88.00±0.40
Transductive	✓		✓			65.57±0.89	78.03±0.54	78.58±0.76	88.02±0.41
Transductive		✓	✓		✓	68.79±0.96	**79.93**±0.56	82.04±0.78	**88.89**±0.41
Transductive[a]	✓	✓	✓			**69.79**±0.99	79.82±0.55	**83.05**±0.79	88.82±0.42
Transductive	✓	✓		✓		58.27±0.91	74.11±0.56	80.75±0.76	87.62±0.44

[a] Default setting of iLPC.

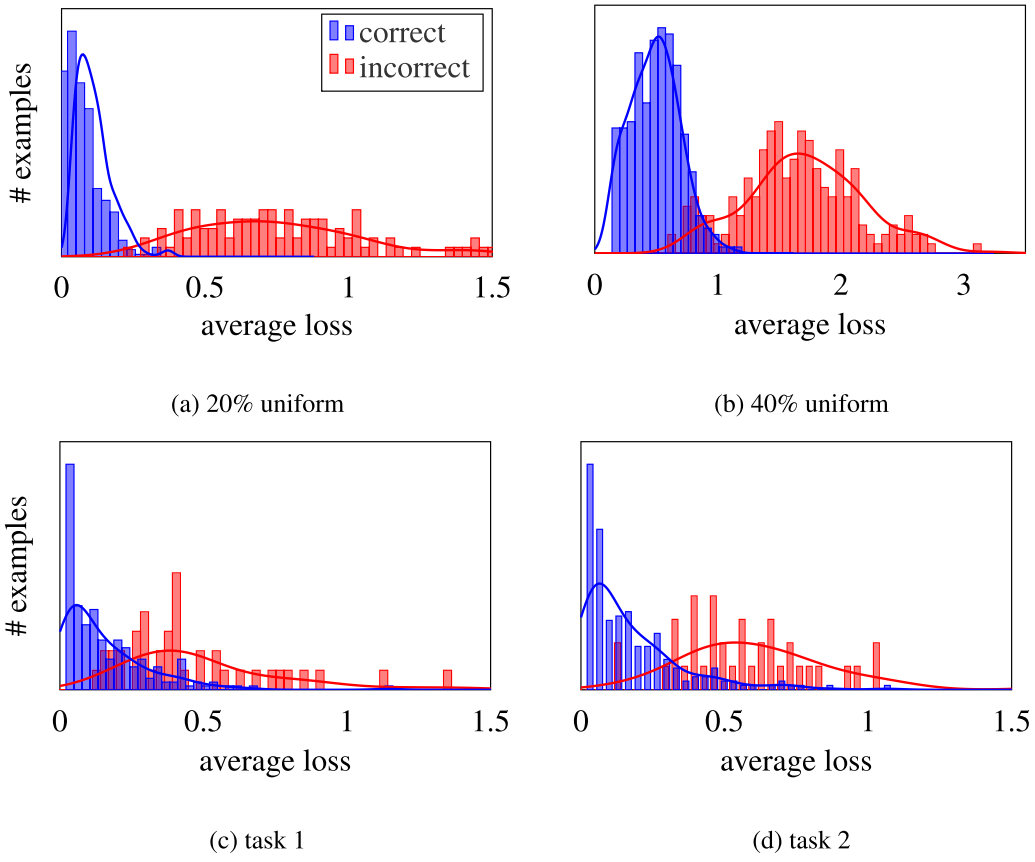

Fig. 2. (a,b) Distributions of loss values (12) for correctly and incorrectly labeled examples, normalized independently. Uniform label noise: (a) 20%, (b) 40%. Pseudo-labels predicted by (11) for two different 1-shot transductive *mini*ImageNet tasks (c,d).

Table 3
Imbalanced transductive inference with our iLPC. Number of queries per class drawn uniformly at random from $10, \ldots, 20$. None: no balancing. Uni: Uniform distribution. True: True distribution.

Balancing	Network	*mini*ImageNet		*tiered*ImageNet		Cifar-FS		CUB	
		1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
None	WRN-28-10	78.06±0.82	87.80±0.42	86.04±0.73	90.74±0.46	85.32±0.76	89.64±0.48	89.67±0.64	92.98±0.31
Uni	WRN-28-10	77.50±0.78	83.68±0.39	83.02±0.67	86.17±0.40	81.47±0.69	84.83±0.45	85.22±0.57	87.99±0.28
True	WRN-28-10	**82.68**±0.82	**89.07**±0.41	**89.17**±0.70	**92.67**±0.44	**87.32**±0.74	**90.92**±0.48	**91.24**±0.60	**94.14**±0.30

Table 4
Ablation study of algorithmic components of our iLPC$_z$ and iLPC$_{zl}$ in distractive semi-supervised learning. CUB 5-shot omitted as no class has the required 70 examples. B: class balancing; Z: cleaning by unnormalized *class similarities only* (iLPC$_z$); Z-norm: same, but matrix Z (4) is row-wise normalized first; W: cleaning by *class similarities for loss weighting* (iLPC$_{zl}$). T: truncation.

Components					Split	*mini*ImageNet		*tiered*ImageNet		CIFAR-FS		CUB	
B	T	Z-norm	Z	W		1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
ResNet-12A													
✓					30/50	58.22±0.87	70.13±0.62	71.09±0.85	77.10±0.66	65.14±0.88	74.83±0.63	73.47±0.80	–
					30/50	60.27±0.94	73.72±0.63	74.29±0.93	80.93±0.70	66.27±0.96	77.55±0.66	77.05±0.87	–
	✓				30/50	60.70±0.95	74.09±0.62	74.58±0.93	82.78±0.63	66.46±0.94	77.69±0.65	77.34±0.85	–
		✓			30/50	60.38±0.92	72.99±0.65	74.19±0.93	84.19±0.61	68.51±0.93	78.24±0.62	78.18±0.86	–
			✓		30/50	61.99±0.93	77.44±0.55	77.78±0.88	86.50±0.52	70.93±0.89	82.66±0.55	82.09±0.84	–
	✓		✓		30/50	62.72±0.93	**78.24**±0.54	78.72±0.88	**87.92**±0.50	**71.87**±0.89	83.47±0.55	83.15±0.84	–
				✓	30/50	62.45±0.93	77.36±0.54	78.23±0.87	86.44±0.52	71.00±0.89	82.61±0.55	82.58±0.84	–
	✓			✓	30/50	**63.19**±0.93	78.17±0.54	**79.13**±0.87	87.78±0.50	71.84±0.88	**83.49**±0.55	**83.66**±0.82	–
WRN-28-10													
✓					30/50	65.84±0.82	75.41±0.58	73.46±0.76	79.51±0.62	72.88±0.75	78.23±0.60	75.67±0.76	–
					30/50	71.65±0.87	82.44±0.56	78.28±0.86	85.64±0.62	76.47±0.83	84.05±0.62	81.03±0.79	–
	✓				30/50	72.57±0.88	83.09±0.54	79.53±0.85	87.27±0.58	76.70±0.82	84.44±0.60	81.88±0.77	–
		✓			30/50	72.10±0.88	81.64±0.57	79.72±0.85	86.55±0.61	74.99±0.80	83.37±0.60	82.53±0.77	–
			✓		30/50	74.56±0.83	85.97±0.43	83.45±0.77	90.21±0.44	78.53±0.85	88.27±0.50	85.92±0.71	–
	✓		✓		30/50	76.06±0.85	**87.15**±0.43	84.86±0.77	**91.35**±0.43	79.17±0.85	89.02±0.49	**87.32**±0.68	–
				✓	30/50	74.74±0.84	86.06±0.44	83.56±0.77	90.19±0.44	79.46±0.83	88.30±0.50	85.95±0.71	–
	✓			✓	30/50	**76.22**±0.86	87.13±0.42	**84.92**±0.76	91.26±0.43	**80.07**±0.83	**89.05**±0.50	**87.32**±0.70	–

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 performance when compared to selecting examples based on P instead.

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 Fig. 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 [10] 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 Fig. 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.

Class balancing To show the effectiveness of class balancing, we carry out experiments in a novel setting for imbalanced few-shot transductive inference. In this setting, the number of queries per class for every few-shot task is drawn uniformly at random from $\{10 \dots 20\}$. We use no balancing, or we use balancing with uniform class distribution (7), or, assuming the prior class distribution $\mathbf{u} \in \mathbb{R}^N$ is known, we replace $\frac{\mathbf{1}_N}{N}$ in (7) by $\mathbf{u}$. As shown in Table 3, balancing improves accuracy by a large margin, but only if the prior class distribution is known, otherwise it is harmful.

4.2.2. Distractive semi-supervised few-shot learning

We investigate the effect of each of our proposed ideas in Section 3.7 and their main hyperparameters.

Removing class balancing As shown in Table 4, removing class balancing from our method provides a significant performance boost, specifically ∼5% in 1-shot and ∼7% in 5-shot using WRN-28-10 and ∼2% in 1-shot and ∼4% in 5-shot using ResNet-12A. This confirms our hypothesis that the distribution of pseudo-labels of distractors is imbalanced, hence class balancing is harmful.

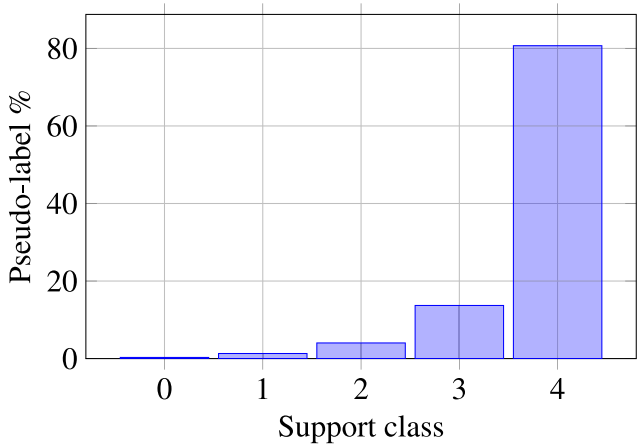

Fig. 3. Average pseudo-label distribution of a single distractor class over the support classes for 1000 tasks in 5-way 5-shot *mini*ImageNet, with 50 unlabeled examples per class using WRN-28-10. Here, ± denotes the standard deviation.

Table 5
Ablation study of β of our iLPC$_{zl}$ (21) in the distractive setting on *mini*ImageNet.

β	ResNet-12A		WRN-28-10	
	1-shot	5-shot	1-shot	5-shot
	1-shot	5-shot	1-shot	5-shot
0.5	62.29±0.91	76.23±0.57	74.46±0.85	85.34±0.46
1	**62.73**±0.91	77.04±0.55	74.87±0.85	85.76±0.44
2	62.71±0.93	77.26±0.55	**74.95**±0.83	86.04±0.44
5	62.45±0.93	77.36±0.54	74.74±0.84	**86.06**±0.42
10	62.20±0.93	**77.37**±0.55	74.66±0.84	**86.06**±0.43

In order to investigate our hypothesis further, we carry out the following experiment. Using only one distractor class, we obtain the distribution of distractor pseudo-labels over classes for each novel task. We rank classes by ascending order of the percentage of distractor examples assigned to that class. The average over 1000 tasks is shown in Fig. 3 As it can be seen in Fig. 3, more than 80% on average of the distractor examples are classified to the class with the most distractor examples. This provides a further confirmation of our hypothesis.

Truncation From Table 4 it is clear that truncation provides a performance boost in 1-shot and 5-shot settings for both WRN-28-10 and

ResNet-12A in all datasets. There are two underlying hypotheses for this improvement. The first follows the intuition behind the positive effect of removing class balancing in that the significant majority of distractor examples will be classified to a specific support class. The second is that most of these examples will be classified at the end of the iterative process. This is because we expect examples of each support class to have a higher similarity to that class than the distractor examples.

Unnormalized class similarities We first validate our hypothesis that the unnormalized class similarities Z (4) are appropriate for cleaning. For this, we compare to a baseline of *normalized class similarities*, where we row-wise ℓ_1-normalize matrix Z before applying (19). As shown in Table 4, for both 1-shot and 5-shot and for both WRN-28-10 and ResNet-12A, the unnormalized similarities provide a performance boost of as much as ∼5%. This is because the unnormalized similarities of all examples to a given class (in a column of Z) are directly comparable, such that the example with the greatest similarity is the most similar to that class. In contrast, normalization discards this information because it takes place per example (row-wise in Z).

We then investigate the effect of scalar β of iLPC$_{zl}$ (*class similarities for loss weighing*) under no class balancing and no truncation. It can be seen in Table 5 that the optimal value of β is ∼2 for 1-shot and ∼10 for 5-shot. We observe that the behavior with respect to β is consistent across both 1-shot and 5-shot settings and both backbones. We set $\beta = 5$ in all remaining experiments.

Finally, in Table 4, we can see that both variants of using Z for cleaning provide a significant performance boost. Combining with truncation and no class balancing, we improve our method by as much as ∼10% using both ResNet-12A and WRN-28-10. Variant iLPC$_{zl}$ works better in the majority of settings, which shows that the unnormalized class similarity (18) and the average loss (13) per example provide complementary information.

4.3. Comparisons

4.3.1. Transductive few-shot inference

Table 6 compares our iLPC with LR+ICI [6] and PT+MAP [5] under the standard setting of 15 unlabeled queries per class. The truly fair comparison is with our reproductions, indicated by 'a'. Apart from the default networks, we also use WRN-28-10 with LR+ICI [6], since it is more powerful. Our iLPC is on par with PT+MAP [5] under this setting and superior to LR+ICI [6] by up to 3% on *mini*ImageNet 1-shot.

We also experiment with 50 unlabeled queries per class, or $M = 250$ in total. As shown in Table 7, iLPC outperforms both LR+ICI [6] and PT+MAP [5]. Moreover, the gain over PT+MAP [5] increases significantly, up to 2% on *mini*ImageNet 1-shot. This can be attributed to the fact that PT+MAP [5] 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 8. The gain is around 0.5%.

Table 9 shows that PT+MAP [5] is very sensitive to feature preprocessing, losing up to 40% without it, while our iLPC is more robust, losing only up to 5%.

Table 10 compares our iLPC with MCT [31]. 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.

4.3.2. Semi-supervised few-shot learning

As shown in Table 11, iLPC is superior to LR+ICI [6] in all settings by an even larger margin than in transductive inference, *e.g.*by nearly 3.5% in *mini*ImageNet 1-shot. This can be attributed to capturing the manifold structure of the data more accurately, since there are more unlabeled data in this case. Because PT+MAP [5] 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.

4.3.3. Distractive semi-supervised learning

We compare both variants of our method, iLPC$_z$ (*class similarities only*) and iLPC$_{zl}$ (*class similarities for loss weighting*), with LR+ICI, PT+MAP and PLCM in the distractive setting. Both variants of our method include truncation and no class balancing. For fair comparisons, we extend all three competitors by adding truncation to LR+ICI and PLCM and removing class balancing from PT+MAP. LR+ICI and PLCM do not use balancing, while for PT+MAP is unclear how we could add truncation.

It can seen from Table 12 that both iLPC$_z$ and iLPC$_{zl}$ outperform LR+ICI and PT+MAP in both 1-shot and 5-shot settings using both backbones on all datasets, with and without the extensions by large margins. Especially noteworthy is the gain of our method over LR+ICI and PT+MAP using ResNet-12A, since it is greater than 2% in all settings/datasets and reaches as high as 8%. Furthermore, it can be seen that both iLPC$_z$ and iLPC$_{zl}$ outperform PLCM in every setting apart from the 1-shot *mini*ImageNet, where PLCM is only slightly better.

We also investigated the behavior of iLPC$_z$ and iLPC$_z$ and the best performing baselines as the number of unlabeled data from distractor classes increases in Table 13. We follow the setup described in Section 4.1 and sample 30 and 50 unlabeled examples per distractor class to be included in the unlabeled set, U', for the 1-shot and 5-shot settings respectively while $|C_{\text{dist}}|$ is increased. As an example in the case where $|C_{\text{dist}}| = 5$, $M_{\text{dist}} = 5 \times 30 = 150$ and $M_{\text{dist}} = 5 \times 50 = 250$ in the 1-shot and 5-shot settings respectively. It can be seen that iLPC$_z$ and iLPC$_{zl}$ outperform all baselines as $|C_{\text{dist}}|$ increases and outperform all baselines except in the 1-shot when $|C_{\text{dist}}| = 3$. Interestingly, as the number of distractor examples increases the performance gap between our method and the other baselines increases, highlighting the robustness of our method.

4.4. Comparison with other state of the art methods

Tables 14–16 compare our solutions with a larger collection of recent methods on the transductive, semi-supervised and distractive 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. This is due to the very diverse choices made in the bibliography, *e.g.*versions of network, training settings, versions of datasets, or preprocessing. 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. Usually, methods experimenting with WRN-28-10 have an advantage. At the time of publication of our preliminary version [13], iLPC achieved the state of the art performance in transductive and semi-supervised few-shot learning setting. EASE [37] has since surpassed our performances in some but not all experiments in the transductive setting as it can be seen on Table 14. Nevertheless, as it can be seen from Tables 15 and 16, iLPC, iLPC$_z$ and iLPC$_{zl}$ still outperform all other methods in the semi-supervised and distractive semi-supervised few-shot learning settings.

Table 6
Transductive inference, comparison with LR+ICI [6] and PT+MAP [5].

Method	Network	*mini*ImageNet		*tiered*ImageNet		CIFAR-FS		CUB	
		1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
LR+ICI [6]	ResNet-12A	66.80	79.26	80.79	87.92	73.97	84.13	88.06	92.53
LR+ICI [6][a]	ResNet-12A	66.85±0.92	78.89±0.55	82.40±0.84	88.80±0.50	75.36±0.97	84.57±0.57	86.53±0.79	92.11±0.35
iLPC (ours)	ResNet-12A	**69.79**±0.99	**79.82**±0.55	**83.49**±0.88	**89.48**±0.47	**77.14**±0.95	**85.23**±0.55	**89.00**±0.70	**92.74**±0.35
PT+MAP [5]	WRN-28-10	82.92±0.26	88.82±0.13	–	–	87.69±0.23	90.68±0.15	91.55±0.19	93.99±0.10
PT+MAP [5][a]	WRN-28-10	82.88±0.73	88.78±0.40	88.15±0.71	92.32±0.40	**86.91**±0.72	90.50±0.49	**91.37**±0.61	93.93±0.32
LR+ICI [6][a]	WRN-28-10	80.61±0.80	87.93±0.44	86.79±0.76	91.73±0.40	84.88±0.79	89.75±0.48	90.18±0.65	93.35±0.30
iLPC (ours)	WRN-28-10	**83.05**±0.79	**88.82**±0.42	**88.50**±0.75	**92.46**±0.42	86.51±0.75	**90.60**±0.48	91.03±0.63	**94.11**±0.30

[a] Our reproduction with official code on our datasets.

Table 7
Transductive inference, 50 queries per class.

Method	Network	*mini*ImageNet		*tiered*ImageNet		CIFAR-FS	
		1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
LR+ICI [6][a]	WRN-28-10	82.38±0.86	88.78±0.39	88.59±0.74	92.11±0.39	86.39±0.79	90.02±0.49
PT+MAP [5][a]	WRN-28-10	83.79±0.71	88.94±0.33	88.87±0.64	92.01±0.36	87.63±0.66	90.15±0.46
iLPC (ours)	WRN-28-10	**85.98**±0.74	**90.54**±0.31	**90.02**±0.70	**92.94**±0.37	**88.54**±0.68	**90.92**±0.46

[a] Our reproduction with official code on our datasets.

Table 8
Transductive 10-shot inference using WRN-28-10. *m*IN: *mini*ImageNet. *t*IN: *tiered*ImageNet.

Method	*m*IN	*t*IN	CIFAR-FS	CUB
LR+ICI [6][a]	88.69±0.38	91.88±0.41	90.23±0.45	93.66±0.28
PT+MAP [5][a]	89.97±0.34	93.33±0.34	91.30±0.45	94.24±0.28
iLPC (ours)	**90.51**±0.35	**93.61**±0.38	**91.59**±0.44	**94.75**±0.26

[a] Our reproduction with official code on our datasets.

Table 9
Transductive inference, ablation over PT+MAP [5] pre-processing. PRE: power transform, normalization, centering.

Method	Pre	*mini*ImageNet		*tiered*ImageNet	
		1-shot	5-shot	1-shot	5-shot
PT+MAP [5][a]		48.57±0.81	75.67±0.82	49.67±0.77	88.32±0.50
iLPC (ours)		78.89±0.90	86.80±0.46	86.52±0.47	91.07±0.47
PT+MAP [5][a]	✓	82.88±0.73	88.78±0.40	88.15±0.71	92.32±0.40
iLPC (ours)	✓	**83.05**±0.79	**88.82**±0.42	**88.50**±0.75	**92.46**±0.42

[a] Our reproduction with official code on our datasets.

Table 10
Transductive inference, comparison with MCT [31] using ResNet-12B.

Method	Network	*mini*ImageNet	
		1-shot	5-shot
MCT (instance, flip) [31]	ResNet-12B	78.55±0.86	86.03±0.42
MCT (no scale) [31]	ResNet-12B	67.26±0.60	**81.90**±0.43
iLPC (ours)	ResNet-12B	**75.58**±1.16	81.58±0.50
iLPC (ours)	ResNet-12A	69.79±0.99	79.82±0.55

[a] Our reproduction with official code on our datasets, without augmentation and without scaling.

4.5. Limitations

Even though our method shows impressive performance in a wide range of experiments, there are still certain limitations. Because of the iterative pseudo-label selection procedure, the inference time increases when the number of unlabeled data increases. In addition, the label cleaning module consists of a linear classifier that is trained on top of the labeled and pseudo-labeled features, increasing the inference time even more. Finally, our method uses several hyperparameters: α, k, γ, τ, β, ν, $\mathbf{q}$, $\mathbf{p}$. Although α and k are calibrated separately per dataset, we use the same values of the remaining hyperparameters for all datasets. A non-uniform distribution over classes, $\mathbf{q}$, can only be used if we have prior knowledge of the unlabeled data.

Some potential research avenues for addressing the limitations of our work include exploring meta-learning as a way to initialize all the hyper-parameters effectively and eliminate hyperparameter tuning on the validation set.

5. Conclusion

In this work, we studied few-shot learning in the presence of labeled and unlabeled data, specifically in the transductive, semi-supervised and distractive semi-supervised few-shot settings. We successfully addressed the limitations of assigning accurate pseudo-labels and selecting the most confident pseudo-labels. Our algorithm is conceptually simple and provides a unique combination of ideas from research directions related to our task at hand.

Our method achieves state of the art performance on four benchmark datasets using different backbones. Label propagation exploits the manifold structure of the data, which becomes important in the presence of more data, while still being competitive otherwise. In the transductive and semi-supervised settings, class balancing exploits prior knowledge about the class distribution that improves the overall performance. Label cleaning, originally introduced for learning with noisy labels, is also found very successful in detecting the most confident pseudo-labels. Iterative reuse of the most confident pseudo-labels as true labels further improves the performance.

Importantly, reasonable baselines, like predicting pseudo-labels by a classifier or iteratively re-using pseudo-labels without cleaning, fail completely. Under fair comparisons, our iLPC outperforms or is on par with state-of-the art methods. Moreover, our algorithm shows superior performance when more unlabeled data is used and is more robust than other methods with respect to feature pre-processing.

Additionally, we investigated the distractive semi-supervised few-shot setting and proposed ideas to make our algorithm robust in this setting, achieving state of the art performance. We verified our hypothesis that pseudo-labels are not balanced over classes in this setting by improving the performance when class balancing is removed. Additionally, we proposed a simple truncation mechanism and the use of unnormalized manifold class similarities. We have found that all three ideas improve the performance significantly in the distractive setting, both individually and when combined, thus confirming that their effect is complementary.

Table 11
Semi-supervised few-shot learning, comparison with [5,6]. CUB 5-shot omitted: no class has the required 70 examples.

Method	Network	Split	*mini*ImageNet		*tiered*ImageNet		CIFAR-FS		CUB	
			1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
LR+ICI [6]	ResNet-12A	30/50	69.66	80.11	84.01	89.00	76.51	84.32	89.58	92.48
LR+ICI [6][a]	ResNet-12A	30/50	67.57±0.97	79.07±0.56	83.32±0.87	89.06±0.51	75.99±0.98	84.01±0.62	88.50±0.71	–
iLPC (ours)	ResNet-12A	30/50	**70.99**±0.91	**81.06**±0.49	**85.04**±0.79	**89.63**±0.47	**78.57**±0.80	**85.84**±0.56	**90.11**±0.64	–
LR+ICI [6][a]	WRN-28-10	30/50	81.31±0.84	88.53±0.43	88.48±0.67	92.03±0.43	86.03±0.77	89.57±0.53	90.82±0.59	–
PT+MAP [5][b]	WRN-28-10	30/50	83.14±0.72	88.95±0.38	89.16±0.61	92.30±0.39	**87.05**±0.69	89.98±0.49	91.52±0.53	–
iLPC (ours)	WRN-28-10	30/50	**83.58**±0.79	**89.68**±0.37	**89.35**±0.68	**92.61**±0.39	87.03±0.72	**90.34**±0.50	**91.69**±0.55	–

[a] Our reproduction with official code on our datasets.
[b] Our adaptation to semi-supervised, based on official code.

Table 12
Distractive semi-supervised learning, comparison with [5,6]. CUB 5-shot omitted as no class has the required 70 examples. T: truncation; nb: no class balancing; iLPC$_z$: cleaning by class similarities only; iLPC$_{zl}$: cleaning by class similarities for loss weighing; both include truncation and no class balancing.

Method	Network	Split	*mini*ImageNet		*tiered*ImageNet		CIFAR-FS		CUB	
			1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
LR+ICI [6][a]	ResNet-12A	30/50	58.21±0.90	70.67±0.66	72.23±0.95	80.21±0.68	64.99±0.92	75.07±0.67	74.85±0.87	–
LR+ICI$_T$ [6][a]	ResNet-12A	30/50	58.52±0.90	70.78±0.66	72.46±0.95	80.71±0.68	65.22±0.91	75.13±0.66	75.02±0.86	–
PT+MAP [5][a]	ResNet-12A	30/50	60.93±0.76	72.35±0.61	72.28±0.79	81.49±0.62	67.72±0.76	77.59±0.59	75.33±0.77	–
PT+MAP$_{nb}$ [5][a]	ResNet-12A	30/50	60.65±0.75	71.96±0.61	72.00±0.79	80.70±0.65	67.90±0.77	77.44±0.60	75.31±0.78	–
PLCM [14][b]	ResNet-12A	30/50	**63.55**±0.86	76.91±0.55	78.72±0.85	87.16±0.51	69.93±0.89	81.41±0.57	82.70±0.75	–
PLCM$_T$ [14][b]	ResNet-12A	30/50	63.52±0.87	76.72±0.55	78.70±0.85	86.82±0.51	69.79±0.88	81.18±0.57	82.77±0.75	–
iLPC$_z$ (ours)	ResNet-12A	30/50	62.72±0.93	**78.24**±0.54	78.72±0.88	**87.92**±0.50	**71.87**±0.89	83.47±0.55	83.15±0.84	–
iLPC$_{zl}$ (ours)	ResNet-12A	30/50	63.19±0.93	78.17±0.54	**79.13**±0.87	87.78±0.50	71.84±0.88	**83.49**±0.55	**83.66**±0.82	–
LR+ICI [6][a]	WRN-28-10	30/50	68.65±0.91	79.18±0.60	76.27±0.85	83.39±0.64	74.40±0.81	81.17±0.63	77.64±0.82	–
LR+ICI$_T$ [6][a]	WRN-28-10	30/50	68.77±0.91	79.31±0.60	76.70±0.85	84.05±0.62	74.58±0.80	81.55±0.62	78.09±0.81	–
PT+MAP [5][a]	WRN-28-10	30/50	68.06±0.83	75.84±0.66	76.69±0.77	82.51±0.64	74.69±0.76	80.06±0.62	79.19±0.76	–
PT+MAP$_{nb}$ [5][a]	WRN-28-10	30/50	72.80±0.83	82.10±0.56	80.03±0.78	85.84±0.60	78.21±0.78	84.04±0.60	82.77±0.72	–
PLCM [14][b]	WRN-28-10	30/50	76.14±0.79	85.42±0.44	83.29±0.75	89.60±0.47	81.24±0.75	87.40±0.51	85.73±0.67	–
PLCM$_T$ [14][b]	WRN-28-10	30/50	**76.29**±0.80	85.47±0.44	83.27±0.75	89.22±0.48	81.28±0.75	86.88±0.50	86.05±0.67	–
iLPC$_z$ (ours)	WRN-28-10	30/50	76.06±0.85	**87.15**±0.43	84.86±0.77	**91.35**±0.43	79.17±0.85	89.02±0.49	**87.32**±0.68	–
iLPC$_{zl}$ (ours)	WRN-28-10	30/50	76.22±0.86	87.13±0.42	**84.92**±0.76	91.26±0.43	**80.07**±0.83	**89.05**±0.50	**87.32**±0.70	–

[a] Our adaptation to the distractive setting, based on official code.
[b] Implementation using official code.

Table 13
Distractive semi-supervised learning, comparison for different number of distractor classes using WRN-28-10 backbone.

Method	Number of distractor classes				
	3	4	5	6	7
5-way 1-shot *mini*ImageNet					
LR+ICI$_T$ [6][a]	68.77±0.91	66.47±0.88	65.15±0.90	63.59±0.88	61.75±0.89
PT+MAP$_{nb}$ [5][a]	72.80±0.83	71.37±0.84	70.40±0.84	69.75±0.82	68.01±0.83
PLCM [14][b]	76.14±0.79	74.31±0.80	73.60±0.80	72.43±0.78	71.29±0.77
PLCM$_T$ [14][b]	**76.29**±0.80	74.03±0.80	73.26±0.79	72.43±0.79	71.01±0.77
iLPC$_z$ (ours)	76.06±0.85	75.06±0.85	74.50±0.84	73.43±0.86	71.80±0.86
iLPC$_{zl}$ (ours)	76.22±0.86	**75.09**±0.85	**74.58**±0.84	**73.56**±0.84	**71.91**±0.86
5-way 5-shot *mini*ImageNet					
LR+ICI$_T$ [6][a]	79.31±0.60	76.54±0.64	75.21±0.64	73.44±0.66	71.25±0.67
PT+MAP$_{nb}$ [5][a]	82.10±0.56	80.38±0.62	79.96±0.61	78.83±0.63	77.54±0.64
PLCM [14][b]	85.42±0.44	83.78±0.48	83.38±0.48	82.34±0.50	81.14±0.50
PLCM$_T$ [14][b]	85.47±0.44	83.54±0.48	83.02±0.48	82.06±0.50	80.82±0.50
iLPC$_z$ (ours)	**87.15**±0.43	85.81±0.43	**85.59**±0.42	**84.35**±0.44	**83.70**±0.42
iLPC$_{zl}$ (ours)	87.13±0.42	**85.84**±0.42	85.39±0.41	84.25±0.45	83.64±0.42

[a] Our adaptation to the distractive setting, based on official code.
[b] Implementation using official code.

In future work, it would be interesting to investigate the use of our method in more challenging settings, such as large scale semi-supervised representation learning and real-world open set recognition.

CRediT authorship contribution statement

Michalis Lazarou: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Tania Stathaki:** Writing – review & editing, Supervision, Resources, Project administration. **Yannis Avrithis:** Writing – review & editing, Writing – original draft, Supervision, Methodology, Investigation, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Table 14
Transductive inference state of the art.

Method	Network	*mini*ImageNet		*tiered*ImageNet		CIFAR-FS		CUB	
		1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
LR+ICI [6][a]	ResNet-12A	66.85±0.92	78.89±0.55	82.40±0.84	88.80±0.50	75.36±0.97	84.57±0.57	86.53±0.79	92.11±0.35
CAN+Top-*k* [33]	ResNet-12	67.19±0.55	80.64±0.35	73.21±0.58	84.93 ±0.38	–	–	–	–
MCT (instance) [31]	ResNet-12B	78.55±0.86	86.03±0.42	82.32±0.81	87.36±0.50	85.61±0.69	90.03±0.46	–	–
EP [30]	WRN-28-10	70.74±0.85	84.34±0.53	78.50±0.91	88.36±0.57	–	–	–	–
LaplacianShot [32]	WRN-28-10	74.86±0.19	84.13±0.14	80.18±0.21	87.56±0.15	–	–	–	–
PT+MAP [5][a]	WRN-28-10	82.88±0.73	88.78±0.40	88.15±0.71	92.32±0.40	**86.91**±0.72	90.50±0.49	**91.37**±0.61	93.93±0.32
EASE+SIAMESE [37]	WRN-28-10	83.00±0.21	**88.92**±0.13	**88.96**±0.23	**92.63**±0.13	**87.60**±0.23	**90.60**±0.16	91.68±0.19	**94.12**±0.09
iLPC (ours)	WRN-28-10	**83.05**±0.79	88.82±0.42	88.50±0.75	92.46±0.42	86.51±0.75	**90.60**±0.48	91.03±0.63	94.11±0.30

[a] Our reproduction with official code on our datasets.

Table 15
Semi-supervised few-shot learning state of the art.

Method	Network	Split	*mini*ImageNet	
			1-shot	5-shot
LST [41]	ResNet-12	30/50	70.10±1.90	78.70±0.80
LR+ICI [6]	ResNet-12A	30/50	69.66	80.11
MCT (instance) [31]	ResNet-12B	30/50	73.80±0.70	84.40±0.50
PLCM [14]	ResNet-12	30/50	72.00±1.0	83.71±0.63
k-means [4][a]	WRN-28-10	100/100	52.35±0.89	67.67±0.65
TransMatch [43]	WRN-28-10	100/100	63.02±1.07	81.06±0.59
PTN [45]	WRN-28-10	100/100	81.57±0.94	87.17±0.58
Cluster-FSL [47]	WRN-28-10	100/100	82.63±0.79	89.16±0.35
iLPC (ours)	WRN-28-10	100/100	**87.62**±0.67	**90.51**±0.36

[a] As reported by [43].

Table 16
Distractive semi-supervised few-shot learning state of the art. Results stated as reported in [31,41].

Method	Network	Split	*mini*ImageNet		*tiered*ImageNet	
			1-shot	5-shot	1-shot	5-shot
Masked soft *k*-means [4]	ResNet-12	30/50	61.00	72.00	66.90	80.20
TPN [29]	ResNet-12	30/50	61.30	72.40	71.50	82.70
LST [41]	ResNet-12	30/50	64.10	77.40	73.50	83.40
MCT (instance) [31]	ResNet-12B	30/50	69.60±0.70	81.30±0.50	74.50±0.70	84.00±0.50
$iLPC_z$	WRN-28-10	30/50	76.06±0.85	**87.15**±0.43	84.86±0.77	**91.35**±0.43
$iLPC_{zl}$	WRN-28-10	30/50	**76.22**±0.86	87.13±0.42	**84.92**±0.76	91.26±0.43

Data availability

The publicly available link to our source code and datasets used in this work can be found in the abstract of our manuscript.

References

[1] Chelsea Finn, Pieter Abbeel, Sergey Levine, Model-agnostic meta-learning for fast adaptation of deep networks, in: Int. Conf. Mach. Learn., 2017.

[2] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Wang, Jia-Bin Huang, A closer look at few-shot classification, in: Int. Conf. Learn. Represent., 2019.

[3] Michalis Lazarou, Tania Stathaki, Yannis Avrithis, Tensor feature hallucination for few-shot learning, in: WACV, 2022, pp. 3500–3510.

[4] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B Tenenbaum, Hugo Larochelle, Richard S Zemel, Meta-learning for semi-supervised few-shot classification, 2018, arXiv preprint arXiv:1803.00676.

[5] Yuqing Hu, Vincent Gripon, Stéphane Pateux, Leveraging the feature distribution in transfer-based few-shot learning, in: ICANN, Springer, 2021, pp. 487–499.

[6] Yikai Wang, C. Xu, Chen Liu, Liyong Zhang, Yanwei Fu, Instance credibility inference for few-shot learning, IEEE Conf. Comput. Vis. Pattern Recognit. (2020).

[7] Puneet Mangla, Nupur Kumari, Abhishek Sinha, Mayank Singh, Balaji Krishnamurthy, Vineeth N Balasubramanian, Charting the right manifold: Manifold mixup for few-shot learning, in: WACV, 2020.

[8] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, Ondrej Chum, Label propagation for deep semi-supervised learning, in: IEEE Conf. Comput. Vis. Pattern Recog., 2019.

[9] Jiaming Song, Lunjia Hu, Yann Dauphin, M. Auli, Tengyu Ma, Robust and on-the-fly dataset denoising for image classification, 2020, arXiv preprint arXiv:2003.10647.

[10] Jinchi Huang, Lie Qu, Rongfei Jia, Binqiang Zhao, O2U-Net: A simple noisy label detection approach for deep neural networks, Int. Conf. Comput. Vis. (2019).

[11] Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, Bernhard Schölkopf, Learning with local and global consistency, in: Adv. Neural Inform. Process. Syst., 2003.

[12] Philip A. Knight, The Sinkhorn-Knopp algorithm: convergence and applications, SIAM J. Matrix Anal. Appl. (2008).

[13] Michalis Lazarou, Tania Stathaki, Yannis Avrithis, Iterative label cleaning for transductive and semi-supervised few-shot learning, in: Int. Conf. Comput. Vis., 2021, pp. 8751–8760.

[14] Kai Huang, Jie Geng, Wen Jiang, Xinyang Deng, Zhe Xu, Pseudo-loss confidence metric for semi-supervised few-shot learning, in: Int. Conf. Comput. Vis., 2021.

[15] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, Pieter Abbeel, A simple neural attentive meta-learner, 2017, arXiv preprint arXiv:1707.03141.

[16] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, Timothy Lillicrap, Meta-learning with memory-augmented neural networks, in: Int. Conf. Mach. Learn., 2016.

[17] Sachin Ravi, Hugo Larochelle, Optimization as a model for few-shot learning, 2016.

[18] Baoquan Zhang, Chuyao Luo, Demin Yu, Xutao Li, Huiwei Lin, Yunming Ye, Bowen Zhang, Metadiff: Meta-learning with conditional diffusion for few-shot learning, in: AAAI, 2024.

[19] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al., Matching networks for one shot learning, in: Adv. Neural Inform. Process. Syst., 2016.

[20] Jake Snell, Kevin Swersky, Richard Zemel, Prototypical networks for few-shot learning, in: Adv. Neural Inform. Process. Syst., 2017.

[21] Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, Siamese neural networks for one-shot image recognition, in: Int. Conf. Mach. Learn. Worksh., 2015.

[22] Spyros Gidaris, Nikos Komodakis, Dynamic few-shot visual learning without forgetting, in: IEEE Conf. Comput. Vis. Pattern Recog., 2018.

[23] Luca Bertinetto, João F Henriques, Jack Valmadre, Philip Torr, Andrea Vedaldi, Learning feed-forward one-shot learners, in: Adv. Neural Inform. Process. Syst., 2016.
[24] Kai Li, Yulun Zhang, Kunpeng Li, Yun Fu, Adversarial feature hallucination networks for few-shot learning, in: IEEE Conf. Comput. Vis. Pattern Recog., 2020.
[25] Wentao Hu, Jiarun Liu, Jiawei Wang, Hui Tian, Meta-DM: Applications of diffusion models on few-shot learning, in: IEEE Int. Conf. Image Process., IEEE, 2024.
[26] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, Phillip Isola, Rethinking few-shot image classification: a good embedding is all you need? 2020, arXiv preprint arXiv:2003.11539.
[27] Mingjiang Liang, Shaoli Huang, Shirui Pan, Mingming Gong, Wei Liu, Learning multi-level weight-centric features for few-shot learning, Pattern Recognit. 128 (2022) 108662.
[28] Xingye Chen, Wenxiao Wu, Li Ma, Xinge You, Changxin Gao, Nong Sang, Yuanjie Shao, Exploring sample relationship for few-shot classification, Pattern Recognit. 159 (2025) 111089.
[29] Yanbin Liu, Juho Lee, Minseop Park, Saehoon Kim, Eunho Yang, Sung Ju Hwang, Yi Yang, Learning to propagate labels: Transductive propagation network for few-shot learning, 2018, arXiv preprint arXiv:1805.10002.
[30] Pau Rodríguez, Issam Laradji, Alexandre Drouin, Alexandre Lacoste, Embedding propagation: Smoother manifold for few-shot classification, Eur. Conf. Comput. Vis. (2020).
[31] Seong Min Kye, Hae Beom Lee, Hoirin Kim, Sung Ju Hwang, Meta-learned confidence for few-shot learning, 2020, arXiv preprint arXiv:2002.12017.
[32] Imtiaz Ziko, Jose Dolz, Eric Granger, Ismail Ben Ayed, Laplacian regularized few-shot learning, in: Int. Conf. Mach. Learn., PMLR, 2020, pp. 11660–11670.
[33] Ruibing Hou, Hong Chang, MA Bingpeng, Shiguang Shan, Xilin Chen, Cross attention network for few-shot classification, in: Adv. Neural Inform. Process. Syst., 2019.
[34] Yixiang Huang, Hongyu Hao, Weichao Ge, Yang Cao, Ming Wu, Chuang Zhang, Jun Guo, Relation fusion propagation network for transductive few-shot learning, Pattern Recognit. 151 (2024) 110367.
[35] Michalis Lazarou, Yannis Avrithis, Guangyu Ren, Tania Stathaki, Adaptive anchor label propagation for transductive few-shot learning, in: IEEE Int. Conf. Image Process., 2023.
[36] Malik Boudiaf, Imtiaz Ziko, Jérôme Rony, Jose Dolz, Pablo Piantanida, Ismail Ben Ayed, Information maximization for few-shot learning, in: H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, H. Lin (Eds.), Adv. Neural Inform. Process. Syst., 2020.
[37] Hao Zhu, Piotr Koniusz, EASE: Unsupervised discriminant subspace learning for transductive few-shot learning, in: IEEE Conf. Comput. Vis. Pattern Recog., 2022.
[38] Michalis Lazarou, Yannis Avrithis, Tania Stathaki, Adaptive manifold for imbalanced transductive few-shot learning, in: WACV, 2024.
[39] Hao Zhu, Piotr Koniusz, Transductive few-shot learning with prototype-based label propagation by iterative graph refinement, in: IEEE Conf. Comput. Vis. Pattern Recog., 2023.
[40] Ségolène Martin, Yunshi Huang, Fereshteh Shakeri, Jean-Christophe Pesquet, Ismail Ben Ayed, Transductive zero-shot and few-shot CLIP, in: IEEE Conf. Comput. Vis. Pattern Recog., 2024.
[41] Xinzhe Li, Qianru Sun, Yaoyao Liu, Qin Zhou, Shibao Zheng, Tat-Seng Chua, Bernt Schiele, Learning to self-train for semi-supervised few-shot classification, in: Adv. Neural Inform. Process. Syst., 2019.
[42] Dong-Hyun Lee, Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks, in: Int. Conf. Mach. Learn. Worksh., 2013.
[43] Zhongjie Yu, L. Chen, Zhongwei Cheng, Jiebo Luo, TransMatch: A transfer-learning scheme for semi-supervised few-shot learning, IEEE Conf. Comput. Vis. Pattern Recognit. (2020).
[44] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, Colin A Raffel, Mixmatch: A holistic approach to semi-supervised learning, in: Adv. Neural Inform. Process. Syst., 2019.
[45] Huaxi Huang, Junjie Zhang, Jian Zhang, Qiang Wu, Chang Xu, PTN: A Poisson transfer network for semi-supervised few-shot learning, in: AAAI, 2021.
[46] Jeff Calder, Brendan Cook, Matthew Thorpe, Dejan Slepcev, Poisson learning: graph based semi-supervised learning at very low label rates, in: Int. Conf. Mach. Learn., PMLR, 2020, pp. 1306–1316.
[47] Jie Ling, Lei Liao, Meng Yang, Jia Shuai, Semi-supervised few-shot learning via multi-factor clustering, in: IEEE Conf. Comput. Vis. Pattern Recog., 2022.
[48] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, Adam Lerer, Automatic differentiation in pytorch, 2017.
[49] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in python, JMLR (2011).

Michalis Lazarou received his Ph.D. in 2024 from the Department of Electrical and Electronic Engineering at Imperial College London. He conducted his Ph.D. research under the supervision of Dr. Tania Stathaki in the Signal Processing and Communications group. He received his Master of Engineering at Electrical and Electronic Engineering from Imperial College London in 2018. His current research interests include machine learning algorithms and computer vision applications.

Tania Stathaki was born in Athens, Greece. She received the Masters degree in Electronics and Computer Engineering from the Department of Electrical and Computer Engineering, National Technical University of Athens, Athens, Greece, and the Ph.D. degree in Signal Processing from Imperial College, London, U.K. She was a Lecturer with the Department of Information Systems and Computing, Brunel University, London, U.K., and an Assistant Professor with the Department of Technology Education and Digital Systems, University of Piraeus, Piraeus, Greece. She is currently a Reader (Associate Professor) with the Department of Electrical and Electronic Engineering of Imperial College.

Yannis Avrithis is a Research Director in the Information Management Systems Institute (IMSI) of Athena Research Center, Greece, and a Principal Investigator at the Institute of Advanced Research on Artificial Intelligence (IARAI), Austria, carrying out research on computer vision and machine learning. Before that he was at the LinkMedia team of Inria Rennes-Bretagne Atlantique, France, at the National and Kapodistrian University of Athens (NKUA) and at the National Technical University of Athens (NTUA), Greece, where he lead the Image and Video Analysis (IVA) team. He holds the HDR qualification from University of Rennes 1,Ph.D. and Diploma degrees from NTUA and M.Sc. degree from Imperial College, University of London, UK.