algorithms

MDPI

Article

Squeezing Backbone Feature Distributions to the Max for Efficient Few-Shot Learning

Yuqing Hu [1,2,*], Stéphane Pateux [1] and Vincent Gripon [2]

[1] Orange S.A., F-84100 Paris, France; stephane.pateux@orange.com
[2] IMT Atlantique, Lab-STICC, UMR CNRS 6285, F-29238 Brest, France; vincent.gripon@imt-atlantique.fr
* Correspondence: yuqing.hu@imt-atlantique.fr

Abstract: In many real-life problems, it is difficult to acquire or label large amounts of data, resulting in so-called few-shot learning problems. However, few-shot classification is a challenging problem due to the uncertainty caused by using few labeled samples. In the past few years, many methods have been proposed with the common aim of transferring knowledge acquired on a previously solved task, which is often achieved by using a pretrained feature extractor. As such, if the initial task contains many labeled samples, it is possible to circumvent the limitations of few-shot learning. A shortcoming of existing methods is that they often require priors about the data distribution, such as the balance between considered classes. In this paper, we propose a novel transfer-based method with a double aim: providing state-of-the-art performance, as reported on standardized datasets in the field of few-shot learning, while not requiring such restrictive priors. Our methodology is able to cope with both inductive cases, where prediction is performed on test samples independently from each other, and transductive cases, where a joint (batch) prediction is performed.

Keywords: few-shot learning; inductive and transductive learning; transfer learning; optimal transport

Citation: Hu, Y.; Pateux, S.; Gripon, V. Squeezing Backbone Feature Distributions to the Max for Efficient Few-Shot Learning. *Algorithms* **2022**, *15*, 147. https://doi.org/10.3390/a15050147

Academic Editors: Mounim A. El Yacoubi, Mehdi Ammi and Hui Yu

Received: 8 April 2022
Accepted: 21 April 2022
Published: 26 April 2022

1. Introduction

Thanks to their outstanding performance, deep learning methods have been widely considered for vision tasks such as image classification and object detection. In order to reach top performance, these systems are typically trained using very large labeled datasets that are representative enough of the inputs to be processed afterward.

However, in many applications, it is costly to acquire or annotate data, resulting in the impossibility of creating such large labeled datasets. Under this condition, it is challenging to optimize deep learning architectures considering the fact they typically are made of way more parameters than the dataset can efficiently tune. This is the reason why in the past few years, few-shot learning (i.e., the problem of learning with few labeled examples) has become a trending research subject in the field. In more detail, there are two settings that authors often consider: (a) "inductive few-shot", where only a few labeled samples are available during training, and prediction is performed on each test input independently, and (b) "transductive few-shot", where prediction is performed on a batch of (non-labeled) test inputs, allowing to take into account their joint distribution.

Few-shot learning is critical to many applications. To name a few, it has been considered for vision [1–3], audio [4–6], language [7–9], and medical imaging [10–12]. More generally, few-shot learning can be used to provide proofs-of-concept while limiting the costs of data labeling or to help in pseudo-annotation of datasets. This importance of the problem of few-shot learning explains the abundant literature across the recent years.

Many works in the domain are built based on a "learning to learn" guidance, where the pipeline is to train an optimizer [13–15] with different tasks of limited data so that the model is able to learn generic experience for novel tasks. Namely, the model learns a set of initialization parameters that are in an advantageous position for the model to adapt to a

new (small) dataset. Recently, the trend evolved towards using well-thought-out feature extractors, called backbones [1,2,16–19], that are trained one time on a large generic dataset in order to produce easily classified feature vectors.

A main problem of the existing methods is that they typically require priors about the data balance between considered classes to perform at their best [1,20]. These methods could be patched to work efficiently under other regimes but would still require the knowledge of data distribution between classes. In this work, we introduce a new methodology with a double aim: 1—providing state-of-the-art performance, as reported using standardized benchmarks in the field of few-shot learning, and 2—not requiring any priors about data distribution among classes.

To achieve this goal, we introduce a novel methodology, summarized in Figure 1, that combines feature preprocessing, self-distillation and an optimal transport-based framework. The utility of these ingredients is demonstrated using ablation tests.

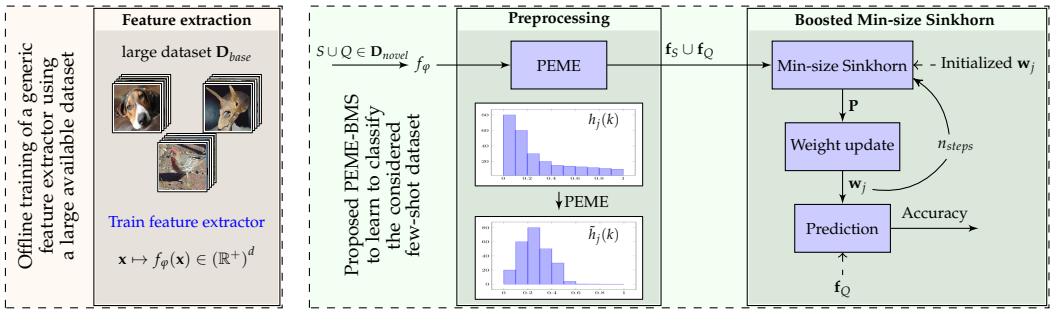

Figure 1. Illustration of the proposed method. A feature extractor is trained using a generic dataset. Obtained features on the few-shot dataset are then preprocessed using PEME (Power, Euclidian normalization, Mean subtraction, Euclidean normalization) to better align with a Gaussian distribution. They are then either directly fed to a classifier (inductive case), or processed through an optimal transport inspired algorithm using self-distillation and Boosted Min-Size Sinkhorn (transductive case).

The outline of the paper is as follows. In Section 2, we introduce related work and discuss the novelty of the proposed approach. In Section 3, we introduce the proposed methodology. Section 4 contains several experiments and benchmark results, along with corresponding discussions. Finally, Section 5 presents the conclusion.

2. Related Work

A large volume of works in few-shot classification is based on meta learning [15] methods, where the training data are transformed into few-shot learning episodes to better fit in the context of a few examples. In this branch, optimization-based methods [13–15,21–23] train a well-initialized optimizer so that it quickly adapts to unseen classes with a few epochs of training. Other works [24,25] apply data augmentation techniques to artificially increase the size of the training data in order for the model to generalize better to unseen data.

In the past few years, there has been a growing interest in transfer-based methods. The main idea consists of training feature extractors able to efficiently segregate novel classes it has never seen. For example, in [2,18], the authors train the backbone with a distance-based classifier [26] that takes into account the inter-class distance. In [2], the authors utilize self-supervised learning techniques [27] to co-train an extra rotation classifier for the output features, improving the accuracy in few-shot settings. More recent works adopt a two-stage training procedure [28–30] where the authors first batch-train a model, then use episodic training to better adjust class prototypes. There are also methods that train a model with a combination of different ingredients [31,32], e.g., distillation [33,34] under a teacher-student framework to better find the nuances between samples. Aside from approaches focused on training a more robust model, other approaches are built on top of a pre-trained feature extractor (backbone). For instance, in [35], the authors

implement a nearest class mean classifier to associate an input with a class whose centroid is the closest in terms of the ℓ_2 distance. In [20], an iterative approach is used to adjust the class prototypes. In [19], the authors build a graph neural network to gather the feature information from similar samples. Generally, transfer-based techniques often reach the best performance on standardized benchmarks.

Although many works involve feature extraction, few have explored the features in terms of their distribution [2,36,37]. Often, assumptions are made that the features in a class align to a certain distribution, even though these assumptions are seldom experimentally discussed. In our work, we analyze the impact of the feature distributions and how they can be transformed for better processing and accuracy. We also introduce a new algorithm to improve the quality of the association between input features and corresponding classes in typical few-shot settings.

Let us highlight the main contributions of this work. (1) We propose a novel preprocessing method to be applied to raw extracted features in order to make them more aligned with Gaussian assumptions. (2) We introduce a Wasserstein-based method to better align the distribution of features with that of the considered classes and combine it with self-distillation. (3) We show that the proposed method can bring a large increase in accuracy with a variety of feature extractors and datasets, leading to state-of-the-art results in the considered benchmarks. This work is an extended version of [1], with the main difference that here we consider the broader case where we do not know the proportion of samples belonging to each considered class in the case of a transductive few-shot, leading to a new algorithm called the Boosted Min-size Sinkhorn. We also propose more efficient preprocessing steps, leading to overall better performance in both inductive and transductive settings. Finally, we introduce the use of logistic regression with self-distillation in our methodology instead of a simple nearest class mean classifier.

3. Materials and Methods

In this section, we introduce the problem statement. We also discuss the various steps of the proposed method, including training the feature extractors, preprocessing the feature representations, and classifying them. Note that we made the code of our method available at https://github.com/yhu01/BMS (accessed on 1 February 2022).

3.1. Problem Statement

We consider a typical few-shot learning problem. Namely, we are given a *base* dataset $\mathbf{D}_{base}$ and a *novel* dataset $\mathbf{D}_{novel}$ such that $\mathbf{D}_{base} \cap \mathbf{D}_{novel} = \emptyset$. $\mathbf{D}_{base}$ contains a large number of labeled examples from K different classes and can be used to train a generic feature extractor. $\mathbf{D}_{novel}$, also referred to as a task or episode in other works, contains a small number of labeled examples (support set $\mathbf{S}$), along with some unlabeled ones (query set $\mathbf{Q}$), all from n *new* classes that are distinct from the K classes in $\mathbf{D}_{base}$. Our goal is to predict the classes of unlabeled examples in the query set. The following parameters are of particular importance to define such a few-shot problem: the number of classes in the novel dataset n (called n-way), the number of labeled samples per class s (called s-shot) and the number of unlabeled samples per class q. Therefore, the novel dataset contains a total of $l + u$ samples, where $l = ns$ are labeled, and $u = nq$ are unlabeled. In the case of an inductive few-shot, the prediction is performed independently on each one of the query samples. In the case of a transductive few-shot [20,38], the prediction is performed considering all unlabeled samples together. Contrary to our previous work [1], we do not consider knowing the proportion of samples in each class in the case of a transductive few-shot.

3.2. Feature Extraction

The first step is to train a neural network backbone model using only the base dataset. In this work, we consider multiple backbones with various training procedures. Once the considered backbone is trained, we obtain robust embeddings that should generalize well to novel classes. We denote by f_φ the backbone function, obtained by extracting the

output of the penultimate layer from the considered architecture, with φ being the trained architecture parameters. Thus, considering an input vector $\mathbf{x}$, $f_\varphi(\mathbf{x})$ is a feature vector with d dimensions that can be thought of as a simpler-to-manipulate representation of $\mathbf{x}$. Note that, importantly, in all backbone architectures used in the experiments of this work, the penultimate layers are obtained by applying a ReLU function so that all feature components coming out of f_φ are nonnegative.

3.3. Feature Preprocessing

As mentioned in Section 2, many works hypothesize, explicitly or not, that the features from the same class are aligned with a specific distribution (often Gaussian-like). However, this aspect is rarely experimentally verified. In fact, it is very likely that features obtained using the backbone architecture are not Gaussian. Indeed, usually, the features are obtained after applying a ReLU function [39] and exhibit a positive and yet skewed distribution mostly concentrated around 0 (more details can be found in the next section).

Multiple works in the domain [20,35] discuss the different statistical methods (e.g., batch normalization) to better fit the features into a model. Although these methods may have provable assets for some distributions, they could worsen the process if applied to an unexpected input distribution. This is why we propose to preprocess the obtained raw feature vectors so that they better align with typical distribution assumptions in the field. Denote $f_\varphi(\mathbf{x}) = [f_\varphi^1(\mathbf{x}), \ldots, f_\varphi^h(\mathbf{x}), \ldots, f_\varphi^d(\mathbf{x})] \in (\mathbb{R}^+)^d, \mathbf{x} \in \mathbf{D}_{novel}$ as the obtained features on $\mathbf{D}_{novel}$, and let $f_\varphi^h(\mathbf{x}), 1 \leq h \leq d$ denote its value in the hth position. The preprocessing methods applied in our proposed algorithms are as follows:

(E) Euclidean normalization. Also known as L2-normalization, which is widely used in many related works [19,35,37], this step scales the features to the same area so that large variance feature vectors do not predominate the others. Euclidean normalization can be given by:

$$f_\varphi(\mathbf{x}) \leftarrow \frac{f_\varphi(\mathbf{x})}{\|f_\varphi(\mathbf{x})\|_2} \tag{1}$$

(P) Power transform. The power transform method [1,40] simply consists of taking the power of each feature vector coordinate. The formula is given by:

$$f_\varphi^h(\mathbf{x}) \leftarrow (f_\varphi^h(\mathbf{x}) + \epsilon)^\beta, \quad \beta \neq 0 \tag{2}$$

where $\epsilon = 1 \times 10^{-6}$ is used to make sure that $f_\varphi(\mathbf{x}) + \epsilon$ is strictly positive in every position, and β is a hyper-parameter. The rationale of the preprocessing above is that power transform, often used in combination with euclidean normalization, has the functionality of reducing the skew of the distribution and mapping it to a close-to-Gaussian distribution, adjusted by β. After experiments, we found that $\beta = 0.5$ gives the most consistent results for our considered experiments, which corresponds to a square-root function that has a wide range of usage on features [41]. We will analyze this ability and the effect of power transform in more detail in Section 4. Note that power transform can only be applied if considered feature vectors contain nonnegative entries, which will always be the case in the remainder of this work.

(M) Mean subtraction. With mean subtraction, each sample is translated using $\mathbf{m} \in (\mathbb{R}^+)^d$, the projection center. This is often used in combination with euclidean normalization in order to reduce the task bias and better align the feature distributions [20]. The formula is given by:

$$f_\varphi(\mathbf{x}) \leftarrow f_\varphi(\mathbf{x}) - \mathbf{m} \tag{3}$$

The projection center is often computed as the mean values of feature vectors related to the problem [20,35]. In this paper, we compute it either as the mean feature vector of the base dataset (denoted as $\mathrm{M_b}$) or the mean vector of the novel dataset (denoted as $\mathrm{M_n}$), depending on the few-shot settings. Of course, in both of these cases, the rationale

is to consider a proxy to what would be the exact mean value of feature vectors on the considered task.

In our proposed method, we deploy these preprocessing steps in the following order: Power transform (P) on the raw features, followed by a Euclidean normalization (E). Then, we perform mean subtraction (M) followed by another Euclidean normalization at the end. The resulting abbreviation is PEME, in which M can be either M_b or M_n, as mentioned above. In our experiments, we found that using M_b in the case of inductive few-shot learning and M_n in the case of transductive few-shot learning consistently led to the most competitive results. More details on why we used this methodology are available in the experiment section.

When facing an inductive problem, a simple classifier such as a Nearest-Class-Mean classifier (NCM) can be used directly after this preprocessing step. The resulting methodology is denoted PEM_bE-NCM. However, in the case of transductive settings, we also introduce an iterative procedure, denoted BMS for Boosted Min-size Sinkhorn, meant to leverage the joint distribution of unlabeled samples. The resulting methodology is denoted PEM_nE-BMS. The details of the BMS procedure are presented thereafter.

3.4. Boosted Min-Size Sinkhorn

In the case of transductive few-shot, we introduce a method that consists of iteratively refining estimates for the probability each unlabeled sample belongs to any of the considered classes. This method is largely based on the one we introduced in [1], except it does not require priors about sample distributions in each of the considered classes. Denoting $i \in [1, \ldots, l+u]$ as the sample index in $\mathbf{D}_{novel}$ and $j \in [1, \ldots, n]$ as the class index, the goal is to maximize the following log post-posterior function:

$$\begin{aligned} L(\theta) &= \sum_i \log P(l(\mathbf{x}_i) = j | \mathbf{x}_i; \theta) \\ &= \sum_i \log \frac{P(\mathbf{x}_i, l(\mathbf{x}_i) = j; \theta)}{P(\mathbf{x}_i; \theta)} \\ &\propto \sum_i \log \frac{P(\mathbf{x}_i | l(\mathbf{x}_i) = j; \theta)}{P(\mathbf{x}_i; \theta)}, \end{aligned} \tag{4}$$

Here, $l(\mathbf{x}_i)$ denotes the class label for sample $\mathbf{x}_i \in \mathbf{Q} \cup \mathbf{S}$, $P(\mathbf{x}_i; \theta)$ denotes the marginal probability, and θ represents the model parameters to estimate. Assuming a Gaussian distribution on the input features for each class, here we define $\theta = \mathbf{w}_j, \forall j$ where $\mathbf{w}_j \in \mathbb{R}^d$ stand for the weight parameters for class j. We observe that Equation (4) can be related to the cost function utilized in optimal transport [42], which is often considered to solve classification problems, with constraints on the sample distribution over classes. To that end, a well-known Sinkhorn [43] mapping method is proposed. The algorithm aims at computing a class allocation matrix among novel class data for a minimum Wasserstein distance. Namely, an allocation matrix $\mathbf{P} \in \mathbb{R}_+^{(l+u)\times n}$ is defined where $\mathbf{P}[i, j]$ denotes the assigned portion for sample i to class j, and it is computed as follows:

$$\begin{aligned} \mathbf{P} &= \text{Sinkhorn}(\mathbf{C}, \mathbf{p}, \mathbf{q}, \lambda) \\ &= \underset{\tilde{\mathbf{P}} \in \mathbb{U}(\mathbf{p}, \mathbf{q})}{\text{argmin}} \sum_{ij} \tilde{\mathbf{P}}[i, j]\mathbf{C}[i, j] + \lambda H(\tilde{\mathbf{P}}), \end{aligned} \tag{5}$$

where $\mathbb{U}(\mathbf{p}, \mathbf{q}) \in \mathbb{R}_+^{(l+u)\times n}$ is a set of positive matrices for which the rows sum to $\mathbf{p}$ and the columns sum to $\mathbf{q}$, $\mathbf{p}$ denotes the distribution of the amount that each sample uses for class allocation, and $\mathbf{q}$ denotes the distribution of the amount of samples allocated to each class. Therefore, $\mathbb{U}(\mathbf{p}, \mathbf{q})$ contains all the possible ways of allocation. In the same equation, $\mathbf{C}$ can be viewed as a cost matrix that is of the same size as $\mathbf{P}$, each element in $\mathbf{C}$ indicates the cost of its corresponding position in $\mathbf{P}$. We will define the particular formula of the cost function for each position $\mathbf{C}[i, j], \forall i, j$ in details later on in the section. As for the second term on

the right of (5), it stands for the entropy of $\tilde{\mathbf{P}}$: $H(\tilde{\mathbf{P}}) = -\sum_{ij} \tilde{\mathbf{P}}[i,j] \log \tilde{\mathbf{P}}[i,j]$, regularized by a hyper-parameter λ. Increasing λ would force the entropy to become smaller, so that the mapping is less diluted. This term also makes the objective function strictly convex [43,44] and thus a practical and effective computation. From lemma 2 in [43], the result of the Sinkhorn allocation has the typical form $\mathbf{P} = \text{diag}(\mathbf{u}) \cdot \exp(-\mathbf{C}/\lambda) \cdot \text{diag}(\mathbf{v})$. It is worth noting that here we assume a soft class allocation, meaning that each sample can be "sliced" into different classes. We will present our proposed method in detail in the following paragraphs.

Given all that is presented above, in this paper, we propose an Expectation–Maximization (*EM*) [45] based method, which alternates between updating the allocation matrix $\mathbf{P}$ and estimating the parameter θ of the designed model, in order to minimize Equation (5) and maximize Equation (4). For a starter, we define a weight matrix $\mathbf{W}$ with n columns (i.e., one per class) and d rows (i.e., one per dimension of feature vectors), and for column j in $\mathbf{W}$, we denote it as the weight parameters $\mathbf{w}_j \in \mathbb{R}^d$ for class j in correspondence with Equation (4). It is initialized as follows:

$$\mathbf{w}_j = \mathbf{W}[:,j] = \mathbf{c}_j / \|\mathbf{c}_j\|_2, \tag{6}$$

where

$$\mathbf{c}_j = \frac{1}{s} \sum_{\mathbf{x} \in \mathbf{S}, \ell(\mathbf{x}) = j} f_\varphi(\mathbf{x}). \tag{7}$$

We can see that $\mathbf{W}$ contains the average of feature vectors in the support set for each class, followed by a L2-normalization on each column so that $\|\mathbf{w}_j\|_2 = 1, \forall j$.

Then, we iterate multiple steps that we describe thereafter.

a Computing costs

As previously stated, the proposed algorithm is an *EM*-like one that iterately updates model parameters for optimal estimates. Therefore, this step, along with Min-size Sinkhorn presented in the next step, is considered as the *E*-step of our proposed method. The goal is to find membership probabilities for the input samples; namely, we compute $\mathbf{P}$ that minimizes Equation (5).

Here, we assume Gaussian distributions, and features in each class have the same variance and are independent from one another (covariance matrix $\mathbf{\Sigma} = \mathbf{I}\sigma^2$). We observe that, ignoring the marginal probability, Equation (4) can be boiled down to negative L2 distances between extracted samples $f_\varphi(\mathbf{x}_i), \forall i$ and $\mathbf{w}_j, \forall j$, which is initialized in Equation (6) in our proposed method. Therefore, based on the fact that $\mathbf{w}_j$ and $f_\varphi(\mathbf{x}_i)$ are both normalized to be unit length vectors ($f_\varphi(\mathbf{x}_i)$ being preprocessed using PEME introduced in the previous section), here we define the cost between sample i and class j to be the following equation:

$$\begin{aligned} \mathbf{C}[i,j] &\propto (f_\varphi(\mathbf{x}_i) - \mathbf{w}_j)^2 \\ &= 1 - \mathbf{w}_j^T f_\varphi(\mathbf{x}_i), \end{aligned} \tag{8}$$

which corresponds to the cosine distance.

b Min-size Sinkhorn

In [1], we proposed a Wasserstein distance-based method in which the Sinkhorn algorithm is applied at each iteration so that the class prototypes are updated iteratively in order to find their best estimates. Although the method showed promising results, it is established on the condition that the distribution of the query set is known, e.g., a uniform distribution among classes on the query set. This is not ideal, given the fact that any priors about $\mathbf{Q}$ should be supposedly kept unknown when applying a method. The methodology introduced in this paper can be seen as a generalization of that introduced in [1] that does not require priors about $\mathbf{Q}$.

In the classical settings, the Sinkhorn algorithm aims at finding the optimal matrix $\mathbf{P}$, given the cost matrix $\mathbf{C}$ and regulation parameter λ presented in Equation (4)). Typically, it initiates $\mathbf{P}$ from a softmax operation over the rows in $\mathbf{C}$, then it iterates between normalizing

columns and rows of $\mathbf{P}$, until the resulting matrix becomes close to doubly stochastic according to $\mathbf{p}$ and $\mathbf{q}$. However, in our case, we do not know the distribution of samples over classes. To address this, we firstly introduce the parameter k, initialized so that $k \leftarrow s$, meant to track an estimate of the cardinal of the class containing the least number of samples in the considered task. Then, we propose the following modification to be applied to the matrix $\mathbf{P}$ once initialized: we normalize each row as in the classical case but only normalize the columns of $\mathbf{P}$ for which the sum is less than the previously computed min-size k [20]. This ensures at least k elements are allocated for each class, but not exactly k samples as in the balanced case.

The principle of this modified Sinkhorn solution is presented in Algorithm 1.

Algorithm 1 Min-size Sinkhorn

Inputs: $\mathbf{C}, \mathbf{p} = \mathbf{1}_{l+u}, \mathbf{q} = k\mathbf{1}_n, \lambda$
Initializations: $\mathbf{P} = Softmax(-\lambda \mathbf{C})$
for $iter = 1$ **to** 50 **do**
  $\mathbf{P}[i,:] \leftarrow \mathbf{p}[i] \cdot \frac{\mathbf{P}[i,:]}{\sum_j \mathbf{P}[i,j]}, \forall i$
  $\mathbf{P}[:,j] \leftarrow \mathbf{q}[j] \cdot \frac{\mathbf{P}[:,j]}{\sum_i \mathbf{P}[i,j]}$ if $\sum_i \mathbf{P}[i,j] < \mathbf{q}[j], \forall j$
end for
return P

c Updating weights

This step is considered as the M-step of the proposed algorithm, in which we use a variant of the logistic regression algorithm in order to find the model parameter θ in the form of weight parameters $\mathbf{w}_j$ for each class. Note that $\mathbf{w}_j$, if normalized, is equivalent to the prototype for class j in this case. Given the fact that in Equation (4), we also take into account the marginal probability, it can be further broken down as:

$$P(\mathbf{x}_i; \theta) = \sum_j P(\mathbf{x}_i | l(\mathbf{x}_i) = j; \theta) P(l(\mathbf{x}_i) = j), \tag{9}$$

We observe that Equation (4) corresponds to applying a softmax function on the negative logits computed through an L2-distance function between samples and class prototypes (normalized). This fits the formulation of a linear hypothesis between $f_\varphi(\mathbf{x}_i)$ and $\mathbf{w}_j$ for logit calculations, hence the rationale for utilizing logistic regression in our proposed method. Note that contrary to classical logistical regression, we implement here a form of self-distillation. Indeed, we use soft labels contained in $\mathbf{P}$ instead of one-hot class indicator targets, and these targets are refined iteratively.

The procedure of this step is as follows: now that we have a polished allocation matrix $\mathbf{P}$, we firstly initialize the weights $\mathbf{w}_j$ as follows:

$$\mathbf{w}_j \leftarrow \mathbf{u}_j / \|\mathbf{u}_j\|_2, \tag{10}$$

where

$$\mathbf{u}_j \leftarrow \sum_i \mathbf{P}[i,j] f_\varphi(\mathbf{x}_i) / \sum_i \mathbf{P}[i,j]. \tag{11}$$

We can see that elements in $\mathbf{P}$ are used as coefficients for feature vectors to linearly adjust the class prototypes [1]. Similar to Equation (6), here $\mathbf{w}_j$ is the normalized newly-computed class prototype that is a vector of length 1.

Next, we further adjust weights by applying a logistic regression, and the optimization is performed by minimizing the following loss:

$$\frac{1}{l+u} \cdot \sum_i \sum_j \; - \log\left(\frac{\exp(\mathbf{S}[i,j])}{\sum_{\gamma=1}^{n} \exp(\mathbf{S}[i,\gamma])}\right) \cdot \mathbf{P}[i,j], \tag{12}$$

where $\mathbf{S} \in \mathbb{R}^{(l+u)\times n}$ contains the logits, and each element is computed as:

$$\mathbf{S}[i,j] = \kappa \cdot \frac{\mathbf{w}_j^T f_\varphi(\mathbf{x}_i)}{\|\mathbf{w}_j\|_2}. \tag{13}$$

Note that κ is a scaling parameter, it can also be seen as a temperature parameter that adjusts the confidence metric to be associated with each sample. It is learnt jointly with $\mathbf{W}$.

The deployed logistic regression comes with hyperparameters on its own. In our experiments, we use an SGD optimizer with a gradient step of 0.1 and 0.8 as the momentum parameter, and we train over e epochs. Here, we point out that $e \geq 0$ is considered an influential hyperparameter in our proposed algorithm, $e = 0$ indicates a simple update of $\mathbf{W}$ as the normalized adjusted class prototypes (Equation (10)) computed from $\mathbf{P}$ in Equation (11), without further adjustment of logistic regression. In addition, note that when $e > 0$, we project columns of $\mathbf{W}$ to the unit hypersphere at the end of each epoch.

d Estimating the class minimum size

We can now refine our estimate for the min-size k for the next iteration. To this end, we firstly compute the predicted label of each sample as follows:

$$\hat{\ell}(\mathbf{x}_i) = \arg\max_j(\mathbf{P}[i,j]), \tag{14}$$

which can be seen as the current (temporary) class prediction.

Then, we compute:

$$k = \min_j \{k_j\}, \tag{15}$$

where $k_j = \#\{i, \hat{\ell}(\mathbf{x}_i) = j\}$, $\#\{\cdot\}$ representing the cardinal of a set.

Summary of the proposed method: all steps of the proposed method are summarized in Algorithm 2. In our experiments, we also report the results obtained when using a prior about $\mathbf{Q}$ as in [1]. In this case, k does not have to be estimated throughout the iterations and can be replaced with the actual exact targets for the Sinkhorn. We denote this prior-dependent version PEM_nE-BMS* (with an added $*$).

Algorithm 2 Boosted Min-size Sinkhorn (BMS)

Parameters: λ, e
Inputs: Preprocessed $f_\varphi(\mathbf{x}), \forall \mathbf{x} \in \mathbf{D}_{novel} = \mathbf{Q} \cup \mathbf{S}$
Initializations: $\mathbf{W}$ as normalized mean vectors over the support set for each class (Equation (6)); Min-size $k \leftarrow s$.
for $iter = 1$ **to** 20 **do**
  Compute cost matrix $\mathbf{C}$ using $\mathbf{W}$ (Equation (8)). # E-step
  Apply Min-size Sinkhorn to compute $\mathbf{P}$ (Algorithm 1). # E-step
  Update weights $\mathbf{W}$ using $\mathbf{P}$ with logistic regression (Equations (10)–(13)). # M-step
  Estimate class predictions $\hat{\ell}$ and min-size k using $\mathbf{P}$ (Equations (14) and (15)).
end for
return $\hat{\ell}$

3.5. Implementation Details

In order to stress the genericity of our proposed method with regards to the chosen backbone architecture and training strategy, we perform experiments using WRN [46], ResNet18 and ResNet12 [47], along with some other pretrained backbones (e.g., DenseNet [35,48]). For each dataset, we train the feature extractor with base classes and test the performance using novel classes. Therefore, for each test run, n classes are drawn uniformly at random among novel classes. Among these n classes, s labeled examples and q unlabeled examples per class are uniformly drawn at random to form $\mathbf{D}_{novel}$. The WRN and ResNet are trained following [2]. In the inductive setting, we use our proposed preprocessing steps PEM_bE

followed by a basic Nearest Class Mean (NCM) classifier. In the transductive setting, the preprocessing steps are denoted as PEM_nE in that we use the mean vector of a novel dataset for mean subtraction, followed by BMS or BMS* depending on whether we have prior knowledge onf the distribution of query set **Q** among classes. Note that we perform a QR decomposition on preprocessed features in order to speed up the computation for the classifier that follows. All our experiments are performed using $n = 5, q = 15, s = 1$ or 5. In our experiments, we perform 10,000 random runs to obtain the mean accuracy score and indicate confidence scores (95%) when relevant. For our proposed PEM_nE-BMS, we train $e = 0$ epoch in the case of 1-shot and $e = 40$ epochs in the case of 5-shot. As for PEM_nE-BMS*, we set $e = 20$ for 1-shot and $e = 40$ for 5-shot. As for the regularization parameter λ in Equation (5), it is fixed to 8.5 for all settings. The impact of these hyperparameters is detailed in the next sections.

4. Results and Discussions

4.1. Comparison with State-of-the-Art Methods

Performance on standardized benchmarks: in the first experiment, we conduct our proposed method on different benchmarks and compare the performance with other state-of-the-art solutions. The results are presented in Tables 1 and 2, and we observe that our method reaches the state-of-the-art performance in both inductive and transductive settings on all the few-shot classification benchmarks. Particularly, the proposed PEM_nE-BMS* brings important gains in both 1-shot and 5-shot settings, and the prior-independent PEM_nE-BMS also obtains competitive results on 5-shot. Note that for tieredImageNet we implement our method based on a pre-trained DenseNet121 backbone following the procedure described in [35]. From these experiments, we conclude that the proposed method can bring an increase in accuracy with a variety of backbones and datasets, leading to a state-of-the-art performance. In terms of execution time, we measured an average of 0.004 s per run. These results confirm the ability of the proposed methodology to reach state-of-the-art performance using the standardized benchmarks of the field of few-shot learning.

Table 1. The 1-shot and 5-shot accuracy of state-of-the-art methods in the literature on miniImageNet and tieredImageNet, compared with the proposed solution. Best results are in bold.

			miniImageNet	
Setting	**Method**	**Backbone**	**1-Shot**	**5-Shot**
Inductive	Matching Networks [49]	WRN	$64.03 \pm 0.20\%$	$76.32 \pm 0.16\%$
	SimpleShot [35]	DenseNet121	$64.29 \pm 0.20\%$	$81.50 \pm 0.14\%$
	S2M2_R [2]	WRN	$64.93 \pm 0.18\%$	$83.18 \pm 0.11\%$
	PT + NCM [1]	WRN	$65.35 \pm 0.20\%$	$83.87 \pm 0.13\%$
	DeepEMD[29]	ResNet12	$65.91 \pm 0.82\%$	$82.41 \pm 0.56\%$
	FEAT[28]	ResNet12	$66.78 \pm 0.20\%$	$82.05 \pm 0.14\%$
	PEM_bE-NCM (ours)	WRN	$\mathbf{68.43 \pm 0.20\%}$	$\mathbf{84.67 \pm 0.13\%}$
Transductive	BD-CSPN [50]	WRN	$70.31 \pm 0.93\%$	$81.89 \pm 0.60\%$
	LaplacianShot [51]	DenseNet121	$75.57 \pm 0.19\%$	$87.72 \pm 0.13\%$
	Transfer + SGC [19]	WRN	$76.47 \pm 0.23\%$	$85.23 \pm 0.13\%$
	TAFSSL [20]	DenseNet121	$77.06 \pm 0.26\%$	$84.99 \pm 0.14\%$
	TIM-GD [52]	WRN	77.80%	87.40%
	MCT [53]	ResNet12	$78.55 \pm 0.86\%$	$86.03 \pm 0.42\%$
	EPNet [54]	WRN	$79.22 \pm 0.92\%$	$88.05 \pm 0.51\%$
	PT + MAP [1]	WRN	$82.92 \pm 0.26\%$	$88.82 \pm 0.13\%$
	PEM_nE-BMS (ours)	WRN	$82.07 \pm 0.25\%$	$89.51 \pm 0.13\%$
	PEM_nE-BMS* (ours)	WRN	$\mathbf{83.35 \pm 0.25\%}$	$\mathbf{89.53 \pm 0.13\%}$

Table 1. *Cont.*

			tieredImageNet	
Setting	**Method**	**Backbone**	**1-Shot**	**5-Shot**
Inductive	ProtoNet [55]	ConvNet4	53.31 ± 0.89%	72.69 ± 0.74%
	LEO [56]	WRN	66.33 ± 0.05%	81.44 ± 0.09%
	SimpleShot [35]	DenseNet121	71.32 ± 0.22%	86.66 ± 0.15%
	PT + NCM [1]	DenseNet121	69.96 ± 0.22%	86.45 ± 0.15%
	FEAT[28]	ResNet12	70.80 ± 0.23%	84.79 ± 0.16%
	DeepEMD[29]	ResNet12	71.16 ± 0.87%	86.03 ± 0.58%
	RENet[30]	ResNet12	71.61 ± 0.51%	85.28 ± 0.35%
	PEM_bE-NCM (ours)	DenseNet121	**71.86** ± **0.21**%	**87.09** ± **0.15**%
Transductive	BD-CSPN [50]	WRN	78.74 ± 0.95%	86.92 ± 0.63%
	LaplacianShot [51]	DenseNet121	80.30 ± 0.22%	87.93 ± 0.15%
	MCT [53]	ResNet12	82.32 ± 0.81%	87.36 ± 0.50%
	TIM-GD [52]	WRN	82.10%	89.80%
	TAFSSL [20]	DenseNet121	84.29 ± 0.25%	89.31 ± 0.15%
	PT + MAP [1]	DenseNet121	85.75 ± 0.26%	90.43 ± 0.14%
	PEM_nE-BMS (ours)	DenseNet121	85.08 ± 0.25%	91.08 ± 0.14%
	PEM_nE-BMS* (ours)	DenseNet121	**86.07** ± **0.25**%	**91.09** ± **0.14**%

Table 2. The 1-shot and 5-shot accuracy of state-of-the-art methods on CUB and CIFAR-FS. Best results are in bold.

			CUB	
Setting	**Method**	**Backbone**	**1-Shot**	**5-Shot**
Inductive	Baseline++ [18]	ResNet10	69.55 ± 0.89%	85.17 ± 0.50%
	MAML [13]	ResNet10	70.32 ± 0.99%	80.93 ± 0.71%
	ProtoNet [55]	ResNet18	72.99 ± 0.88%	86.64 ± 0.51%
	Matching Networks [49]	ResNet18	73.49 ± 0.89%	84.45 ± 0.58%
	FEAT[28]	ResNet12	73.27 ± 0.22%	85.77 ± 0.14%
	DeepEMD[29]	ResNet12	75.65 ± 0.83%	88.69 ± 0.50%
	RENet[30]	ResNet12	79.49 ± 0.44%	91.11 ± 0.24%
	S2M2_R [2]	WRN	80.68 ± 0.81%	90.85 ± 0.44%
	PT + NCM [1]	WRN	80.57 ± 0.20%	91.15 ± 0.10%
	PEM_bE-NCM (ours)	WRN	**80.82** ± **0.19**%	**91.46** ± **0.10**%
Transductive	LaplacianShot [51]	ResNet18	80.96%	88.68%
	TIM-GD [52]	ResNet18	82.20%	90.80%
	BD-CSPN [50]	WRN	87.45%	91.74%
	Transfer + SGC [19]	WRN	88.35 ± 0.19%	92.14 ± 0.10%
	PT + MAP [1]	WRN	91.55 ± 0.19%	93.99 ± 0.10%
	LST + MAP [57]	WRN	91.68 ± 0.19%	94.09 ± 0.10%
	PEM_nE-BMS (ours)	WRN	91.01 ± 0.19%	94.60 ± 0.09%
	PEM_nE-BMS* (ours)	WRN	**91.91** ± **0.18**%	**94.62** ± **0.09**%
			CIFAR-FS	
Setting	**Method**	**Backbone**	**1-Shot**	**5-Shot**
Inductive	ProtoNet [55]	ConvNet64	55.50 ± 0.70%	72.00 ± 0.60%
	MAML [13]	ConvNet32	58.90 ± 1.90%	71.50 ± 1.00%
	RENet[30]	ResNet12	74.51 ± 0.46%	86.60 ± 0.32%
	BD-CSPN [50]	WRN	72.13 ± 1.01%	82.28 ± 0.69%
	S2M2_R [2]	WRN	74.81 ± 0.19%	87.47 ± 0.13%
	PT + NCM [1]	WRN	74.64 ± 0.21%	87.64 ± 0.15%
	PEM_bE-NCM (ours)	WRN	**74.84** ± **0.21**%	**87.73** ± **0.15**%

Table 2. *Cont.*

Setting	Method	Backbone	CIFAR-FS 1-Shot	CIFAR-FS 5-Shot
Transductive	DSN-MR [58]	ResNet12	78.00 ± 0.90%	87.30 ± 0.60%
	Transfer + SGC [19]	WRN	83.90 ± 0.22%	88.76 ± 0.15%
	MCT [53]	ResNet12	87.28 ± 0.70%	90.50 ± 0.43%
	PT + MAP [1]	WRN	87.69 ± 0.23%	90.68 ± 0.15%
	LST + MAP [57]	WRN	87.79 ± 0.23%	90.73 ± 0.15%
	PEM_nE-BMS (ours)	WRN	86.93 ± 0.23%	91.18 ± 0.15%
	PEM_nE-BMS* (ours)	WRN	**87.83** ± **0.22**%	**91.20** ± **0.15**%

Performance on cross-domain settings: in this experiment, we test our method in a cross-domain setting, where the backbone is trained with the base classes in miniImageNet but tested with the novel classes in the CUB dataset. As shown in Table 3, the proposed method gives the best accuracy both in the case of 1-shot and 5-shot, for both inductive and transductive settings. The ability of the proposed methodology to leverage feature vectors trained on a different dataset points out that its efficacy is not restricted to constrained settings where data distribution between the base and novel have to be identical.

Table 3. The 1-shot and 5-shot accuracy of state-of-the-art methods when performing cross-domain classification (backbone: WRN). Best results are in bold.

Setting	Method	1-Shot	5-Shot
Inductive	Baseline++ [18]	40.44 ± 0.75%	56.64 ± 0.72%
	Manifold Mixup [59]	46.21 ± 0.77%	66.03 ± 0.71%
	S2M2_R [2]	48.24 ± 0.84%	70.44 ± 0.75%
	PT + NCM [1]	48.37 ± 0.19%	70.22 ± 0.17%
	PEM_bE-NCM (ours)	**50.71** ± **0.19**%	**73.15** ± **0.16**%
Transductive	LaplacianShot [51]	55.46%	66.33%
	Transfer + SGC [19]	58.63 ± 0.25%	73.46 ± 0.17%
	PT + MAP [1]	63.17 ± 0.31%	76.43 ± 0.19%
	PEM_nE-BMS (ours)	62.93 ± 0.28%	79.10 ± 0.18%
	PEM_nE-BMS* (ours)	**63.90** ± **0.31**%	**79.15** ± **0.18**%

4.2. Ablation Studies

Ablation study on the proposed method: in this section, we have a closer look at the impact of our proposed methodology steps. The idea is to better understand the contribution of each step to the final performance. Namely, we conduct an ablation study on the prediction accuracy with or without (1) PEME, which is the proposed preprocessing steps on extracted raw features, and (2) proposed Boosted Min-sized Sinkhorn algorithm that integrates self-distillation for refined prototypes. Note that in the case of BMS*, the algorithm is equivalent to MAP presented in [1] without the newly proposed self-distillation method. In Table 4, we can see that both PEME and self-distillation play an important role in improving the prediction performance. As such, this experiment supports the interest of both steps to reach the best possible accuracy.

Table 4. Ablation study on our proposed PEME and BMS* with self-distillation on miniImageNet (backbone: WRN). Best results are in bold.

w/PEME	BMS* w/Self-Distillation	Accuracy 1-Shot	Accuracy 5-Shot
		$75.60 \pm 0.29\%$	$84.13 \pm 0.16\%$
✓		$82.92 \pm 0.26\%$	$88.82 \pm 0.13\%$
	✓	$80.19 \pm 0.27\%$	$87.40 \pm 0.13\%$
✓	✓	$\mathbf{83.35 \pm 0.25}\%$	$\mathbf{89.53 \pm 0.13}\%$

Generalization to backbone architectures. To further stress the interest of the ingredients inn the proposed method reaching top performance, in Table 5 we investigate the impact of our proposed method on different backbone architectures and benchmarks in the transductive setting. For comparison purposes, we also replace our proposed BMS algorithm with a standard K-Means algorithm where class prototypes are initialized with the available labeled samples for each class. We can observe that: (1) the proposed method consistently achieves the best results for any fixed backbone architecture, (2) the feature extractor trained on WRN outperforms the others with our proposed method on different benchmarks, (3) there are significant drops in accuracy with k-means, which stresses the interest of BMS, and (4) the prior on $\mathbf{Q}$ (BMS vs. BMS*) is of major interest for 1-shot, boosting the performance by an approximation of 1% on all tested feature extractors. Overall, these experiments demonstrate the interest of the proposed methodology with respect to existing alternatives.

Table 5. The 1-shot and 5-shot accuracy of the proposed method on different backbones and benchmarks. Comparison with the k-means algorithm. Best results are in bold.

Method	Backbone	miniImageNet 1-Shot	miniImageNet 5-Shot	CUB 1-Shot	CUB 5-Shot	CIFAR-FS 1-Shot	CIFAR-FS 5-Shot
K-MEANS	ResNet12	$72.73 \pm 0.23\%$	$84.05 \pm 0.14\%$	$87.35 \pm 0.19\%$	$92.31 \pm 0.10\%$	$78.39 \pm 0.24\%$	$85.73 \pm 0.16\%$
	ResNet18	$73.08 \pm 0.22\%$	$84.67 \pm 0.14\%$	$87.16 \pm 0.19\%$	$91.97 \pm 0.09\%$	$79.95 \pm 0.23\%$	$86.74 \pm 0.16\%$
	WRN	$76.67 \pm 0.22\%$	$86.73 \pm 0.13\%$	$88.28 \pm 0.19\%$	$92.37 \pm 0.10\%$	$83.69 \pm 0.22\%$	$89.19 \pm 0.15\%$
BMS (ours)	ResNet12	$77.62 \pm 0.28\%$	$86.95 \pm 0.15\%$	$90.14 \pm 0.19\%$	$94.30 \pm 0.10\%$	$81.65 \pm 0.25\%$	$88.38 \pm 0.16\%$
	ResNet18	$79.30 \pm 0.27\%$	$87.94 \pm 0.14\%$	$90.50 \pm 0.19\%$	$94.29 \pm 0.09\%$	$84.16 \pm 0.24\%$	$89.39 \pm 0.15\%$
	WRN	$82.07 \pm 0.25\%$	$89.51 \pm 0.13\%$	$91.01 \pm 0.18\%$	$94.60 \pm 0.09\%$	$86.93 \pm 0.23\%$	$91.18 \pm 0.15\%$
BMS* (ours)	ResNet12	$79.03 \pm 0.28\%$	$87.01 \pm 0.15\%$	$91.34 \pm 0.19\%$	$94.32 \pm 0.09\%$	$82.87 \pm 0.27\%$	$88.43 \pm 0.16\%$
	ResNet18	$80.56 \pm 0.27\%$	$87.98 \pm 0.14\%$	$91.39 \pm 0.19\%$	$94.31 \pm 0.09\%$	$85.17 \pm 0.25\%$	$89.42 \pm 0.16\%$
	WRN	$\mathbf{83.35 \pm 0.25}\%$	$\mathbf{89.53 \pm 0.13}\%$	$\mathbf{91.91 \pm 0.18}\%$	$\mathbf{94.62 \pm 0.09}\%$	$\mathbf{87.83 \pm 0.22}\%$	$\mathbf{91.20 \pm 0.15}\%$

Preprocessing impact: in Table 6, we compare our proposed feature preprocessing PEME with other preprocessing techniques such as batch normalization, which standardizes extracted feature values into $[0, 1]$ for a considered task, along with other ones being used in [35]. The experiment is conducted on miniImageNet (backbone: WRN). For all that is put into comparison, we run either an NCM classifier or BMS after preprocessing, depending on the settings. The obtained results clearly show the interest of PEME compared with existing alternatives, and we also observe that the power transform helps increase the accuracy on both inductive and transductive settings.

Table 6. Comparison of 1-shot and 5-shot accuracy on miniImageNet (backbone: WRN) when using various preprocessing steps on the extracted features. Best results are in bold.

	Inductive (NCM)		Transductive (BMS)	
Preprocessing	**1-Shot**	**5-Shot**	**1-Shot**	**5-Shot**
None	$55.30 \pm 0.21\%$	$78.34 \pm 0.15\%$	$77.62 \pm 0.26\%$	$87.96 \pm 0.13\%$
Batch Norm [60]	$66.81 \pm 0.20\%$	$83.57 \pm 0.13\%$	$73.74 \pm 0.21\%$	$88.07 \pm 0.13\%$
L2N [35]	$65.37 \pm 0.20\%$	$83.46 \pm 0.13\%$	$73.84 \pm 0.21\%$	$88.15 \pm 0.13\%$
CL2N [35]	$63.88 \pm 0.20\%$	$80.85 \pm 0.14\%$	$73.12 \pm 0.28\%$	$86.47 \pm 0.15\%$
EM_bE	$68.05 \pm 0.20\%$	$83.76 \pm 0.13\%$	$80.28 \pm 0.26\%$	$88.36 \pm 0.13\%$
PEM_bE	$\mathbf{68.43 \pm 0.20\%}$	$\mathbf{84.67 \pm 0.13\%}$	$82.01 \pm 0.26\%$	$89.50 \pm 0.13\%$
EM_nE	\	\	$80.14 \pm 0.27\%$	$88.39 \pm 0.13\%$
PEM_nE	\	\	$\mathbf{82.07 \pm 0.25\%}$	$\mathbf{89.51 \pm 0.13\%}$

Effect of power transform: we firstly conduct a Gaussian hypothesis test on each of the 640 coordinates of raw extracted features (backbone: WRN) for each of the 20 novel classes (dataset: miniImageNet). Following D'Agostino and Pearson's methodology [61,62] and $p = 1e-3$, only one of the $640 \times 20 = 12800$ tests return positive, suggesting a very low pass rate for raw features. However, after applying the power transform, we record a pass rate that surpasses 50%, suggesting a considerably increased number of positive results for Gaussian tests. This experiment shows the effect of power transform being able to adjust feature distributions into more Gaussian-like ones.

To better show the effect of this proposed technique on feature distributions, we depict in Figure 2 the distributions of an arbitrarily selected feature for three randomly selected novel classes of miniImageNet when using WRN, before and after applying the power transform. In addition, we also added to the figure the feature distributions after applying batch normalization for comparison purposes. We observe quite clearly that (1) raw features exhibit a positive distribution mostly concentrated around 0, a similar behavior is also observed for batch norm, and (2) power transform is able to reshape the feature distributions to close-to-Gaussian distributions. We observe similar behaviors with other datasets as well. Moreover, in order to visualize the impact of this technique with respect to the position of feature points, in Figure 3, we plot the feature vectors of three randomly selected classes from $\mathbf{D}_{novel}$. Note that all feature vectors in this experiment are reduced to 3-dimensional ones corresponding to their largest eigenvalues. From Figure 3, we can observe that the power transform, often followed by an L2-normalization, can help shape the class distributions to become more gathered and Gaussian-like [57].

Influence of the number of unlabeled samples: in order to better understand the gain in accuracy due to having access to more unlabeled samples, we depict in Figure 4 the evolution of accuracy as a function of q, when the number of classes $n = 5$ is fixed. Interestingly, the accuracy quickly reaches a close-to-asymptotical plateau, emphasizing the ability of the method to quickly exploit available information in the task.

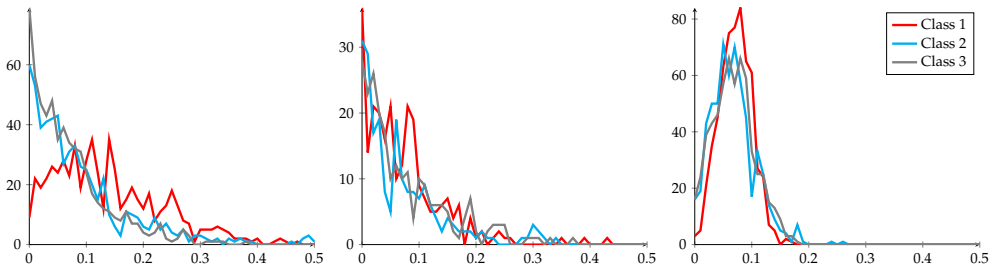

Figure 2. Distributions of an arbitrarily chosen feature for 3 novel classes with different preprocessing techniques: raw (**left**), batch norm (**middle**) and power transform (**right**).

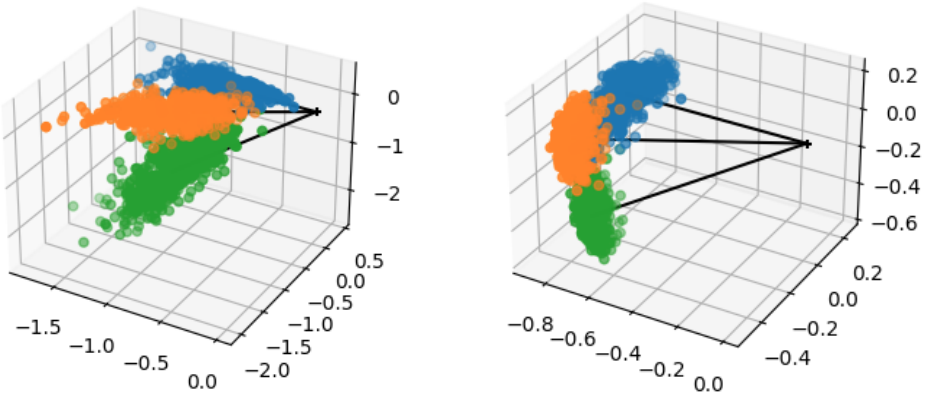

Figure 3. Plot of feature vectors (extracted from WRN) from 3 randomly selected classes (each with its own color). (**left**) Naive features. (**right**) Preprocessed features using power transform.

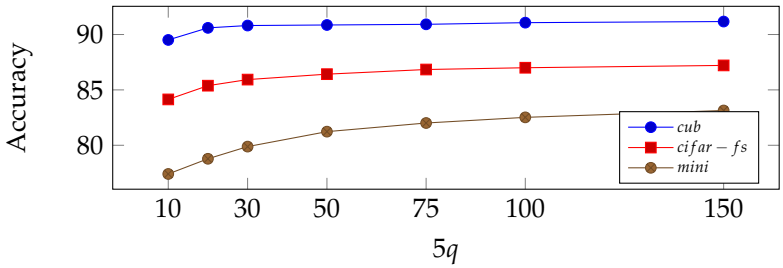

Figure 4. Accuracy of 5-way, 1-shot classification setting on miniImageNet, CUB and CIFAR-FS as a function of q.

Influence of hyperparameters: in order to test how much impact the hyperparameters could have on our proposed method in terms of prediction accuracy, here we select two important hyperparameters that are used in BMS and observe their impact. Namely, the number of training epochs e in logistic regression and the regulation parameter λ used for computing the prediction matrix $\mathbf{P}$. In Figure 5, we show the accuracy of our proposed method as a function of e (top) and λ (bottom). Results are reported for BMS* in 1-shot settings, and for BMS in 5-shot settings. From the figure, we can see a slight uptick of accuracy as e or λ increase, followed by a downhill when they become larger, implying an overfitting of the classifier. We chose our optimal parameters from these experiments. We note that, interestingly, the performance of the method appears quite robust to a non-optimal choice of these parameters.

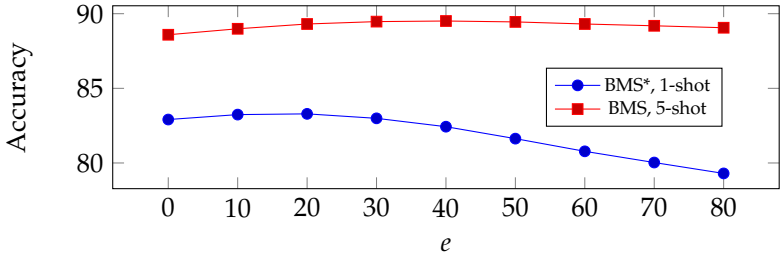

Figure 5. *Cont.*

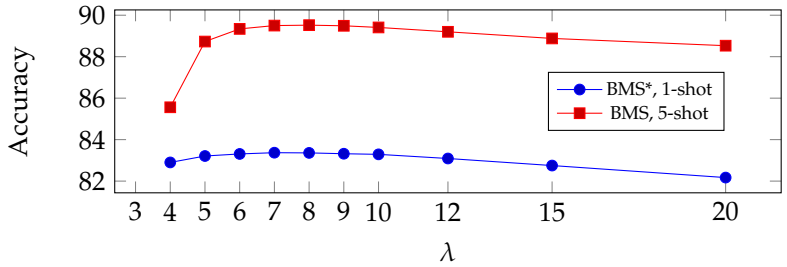

Figure 5. Accuracy of the proposed method on miniImageNet (backbone: WRN) as a function of training epoch e (**top**) and regulation parameter λ (**bottom**).

Convergence analysis: in this section, we discuss the convergence of the proposed method in Algorithm 2, namely the convergence of $\mathbf{P}$ as a function of the number of iteration step noted n_{step}. We conduct this experiment in a 5-way, 1-shot setting on miniImageNet (backbone: WRN). In Figure 6 (left), we depict $\|\Delta\mathbf{P}\|_2$ as a function of n_{step}, with $\|\Delta\mathbf{P}\|_2$ being defined as $\|\mathbf{P}(t+1)-\mathbf{P}(t)\|^2, 1 \leq t \leq n_{step}$, namely the Euclidean difference between the current $\mathbf{P}$ and the one computed in the previous step. Furthermore, we remind the reader that the goal of the proposed algorithm is to minimize the energy computed in Equation (5). Therefore, in Figure 6 (right), we depict the energy (value of Equation (5)) as a function of n_{step}. We can see that both $\|\Delta\mathbf{P}\|_2$ and energy tend to stabilize with more iteration steps.

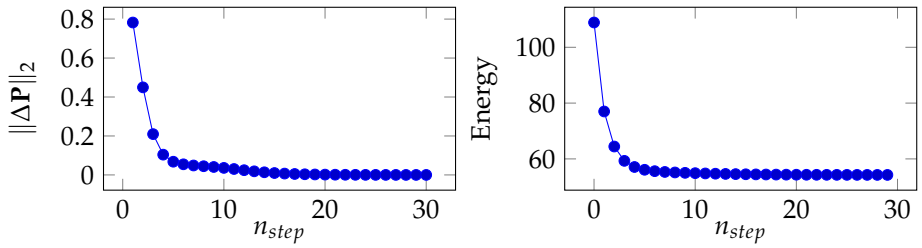

Figure 6. Convergence of BMS (1-shot on miniImagenet). (**left**) $\|\Delta\mathbf{P}\|_2$ as a function of n_{step}. (**right**) Energy (1-shot on miniImagenet) as a function of n_{step}.

Proposed method on backbones pre-trained with external data: in this experiment, we compare our proposed method BMS* with the work in [63] that pre-trains the backbone with the help of external illumination data for augmentation, followed by PT + MAP in [1] for class center estimation. Here, we use the same backbones as [63] and replace PT + MAP with our proposed BMS* under the same conditions. Results are presented in Table 7. Note that we also show the re-implemented results of [63], and our method reaches superior performance on all tested benchmarks using external data in [63].

Table 7. The proposed method on backbones pre-trained with external data. Note that *-re* denotes the re-implementation of an existing method. Best results are in bold.

Benchmark	Method	1-Shot	5-Shot
miniImageNet	Illu-Aug [63]	$82.99 \pm 0.23\%$	$89.14 \pm 0.12\%$
	Illu-Aug-*re*	$83.53 \pm 0.25\%$	$89.38 \pm 0.12\%$
	PEM_nE-BMS* (ours)	$\mathbf{83.85 \pm 0.25}\%$	$\mathbf{90.07 \pm 0.12}\%$
CUB	Illu-Aug [63]	$94.73 \pm 0.14\%$	$96.28 \pm 0.08\%$
	Illu-Aug-*re*	$94.63 \pm 0.15\%$	$96.06 \pm 0.08\%$
	PEM_nE-BMS* (ours)	$\mathbf{94.78 \pm 0.15}\%$	$\mathbf{96.43 \pm 0.07}\%$
CIFAR-FS	Illu-Aug [63]	$87.73 \pm 0.22\%$	$91.09 \pm 0.15\%$
	Illu-Aug-*re*	$87.76 \pm 0.23\%$	$91.04 \pm 0.15\%$
	PEM_nE-BMS* (ours)	$\mathbf{87.83 \pm 0.23}\%$	$\mathbf{91.49 \pm 0.15}\%$

Proposed method on Few-Shot Open-Set Recognition: Few-Shot Open-Set Recognition (FSOR) as a new trending topic deals with the fact that there are open data mixed in query set **Q** that do not belong to any of the supposed classes used for label predictions. Therefore, this often requires a robust classifier that is able to correctly classify the non-open data as well as rejecting the open ones. In Table 8, we apply our proposed PEME for feature preprocessing, followed by an NCM classifier and compare the results with other state-of-the-art alternatives. We observe that our proposed method is able to surpass the others in terms of accuracy and AUROC.

Table 8. Accuracy and AUROC of the proposed method for Few-Shot Open-Set Recognition. Best results are in bold.

	miniImageNet				**tieredImageNet**			
	1-Shot		**5-Shot**		**1-Shot**		**5-Shot**	
Method	**Acc**	**AUROC**	**Acc**	**AUROC**	**Acc**	**AUROC**	**Acc**	**AUROC**
ProtoNet [55]	64.01%	51.81%	80.09%	60.39%	68.26%	60.73%	83.40%	64.96%
FEAT [28]	67.02%	57.01%	82.02%	63.18%	70.52%	63.54%	84.74%	70.74%
NN [64]	63.82%	56.96%	80.12%	63.43%	67.73%	62.70%	83.43%	69.77%
OpenMax [65]	63.69%	62.64%	80.56%	62.27%	68.28%	60.13%	83.48%	65.51%
PEELER [66]	65.86%	60.57%	80.61%	67.35%	69.51%	65.20%	84.10%	73.27%
SnaTCHer [67]	67.60%	70.17%	82.36%	77.42%	70.85%	74.95%	85.23%	82.03%
PEM_bE-NCM (ours)	**68.43**%	**72.10**%	**84.67**%	**80.04**%	**71.87**%	**75.44**%	**87.09**%	**83.85**%

4.3. Proposed Method on Merged Features

In this section, we investigate the effect of our proposed method on merged features. Namely, we perform a direct concatenation of raw feature vectors extracted from multiple backbones at the beginning, followed by BMS. In Table 9, we chose the feature vectors from three backbones (WRN, ResNet18 and ResNet12) and evaluated the performance with different combinations. We observe that (1) a direct concatenation, depending on the backbones, can bring about 1% gain in both 1-shot and 5-shot settings compared with the results in Table 5 with feature vectors extracted from one single feature extractor. (2) BMS* reached new state-of-the-art results on few-shot learning benchmarks with feature vectors concatenated from WRN, ResNet18 and ResNet12, given that no external data are used.

Table 9. The 1-shot and 5-shot accuracy on miniImageNet, CUB and CIFAR-FS on our proposed PEM_nE-BMS with multi-backbones (backbone training procedure follows [2], '+' denotes a concatenation of backbone features).

	miniImageNet		**CUB**		**CIFAR-FS**	
Backbone	**1-Shot**	**5-Shot**	**1-Shot**	**5-Shot**	**1-Shot**	**5-Shot**
RN18 + RN12	80.32%	89.07%	92.31%	95.62%	85.44%	90.58%
WRN + RN12	82.63%	90.43%	92.69%	95.96%	87.11%	91.50%
WRN + RN18	83.05%	90.57%	92.66%	95.79%	87.53%	91.70%
WRN + RN18 + RN12	82.90%	90.64%	93.32%	96.31%	87.62%	91.84%
WRN + RN18 + RN12 *	84.37%	90.69%	**94.26**%	**96.32**%	**88.44**%	**91.86**%
6×WRN *	**85.54**%	**91.53**%	\	\	\	\

: BMS.

To further study the impact of the number of backbones on prediction accuracy, in Figure 7 we depict the performance of our proposed method as a function of the number of backbones. Note that, here, we operate on feature vectors of 6 WRN backbones (dataset: miniImageNet) concatenated one after another, which makes a total of 6 slots corresponding to a $640 \times 6 = 3840$ feature size. Each of them is trained the same way as in [2], and we randomly select the multiples of 640 coordinates within the slots to denote the number

of concatenated backbones used. The performance result is the average of 100 random selections, and we test with both BMS and BMS* for 1-shot, and BMS* for 5-shot. From Figure 7, we observe that, as the number of backbones increases, there is a relatively steady growth in terms of accuracy in multiple settings of our proposed method, indicating the interest of BMS in merged features.

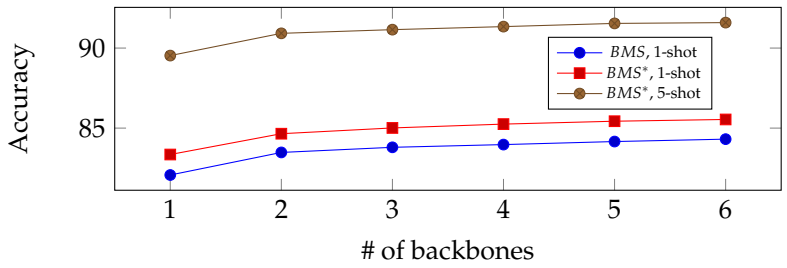

Figure 7. Accuracy of the proposed method in different settings as a function of the number of backbones (dataset: miniImageNet).

5. Conclusions

In this paper, we introduced a new pipeline to solve the few-shot classification problem. It comes with the two following assets: first, it is able to reach state-of-the-art accuracy on standardized benchmarks of the field, and second, it does not require any explicit priors about data distribution between classes, as opposed to many previous works in the domain. Using extensive experiments, we demonstrated that the proposed methodology can be used in a variety of settings, including cross-domain, multiple backbones, open-set recognition . . .. Using ablation tests, we showed the importance of the introduced steps in the methodology. The proposed methodology comes with only a few extra hyperparameters, on which our experiments suggest that a fine tuning is not necessarily required. Thus we believe that the proposed method is applicable to many practical engineering problems. In future work, we would like to better understand the fundamental reasons why the proposed preprocessing is able to boost performance. We would also like to find automatic ways to tune the hyperparameters.

Author Contributions: Conceptualization, Y.H., S.P., and V.G.; methodology, Y.H. and S.P.; software, S.P. and V.G.; validation, Y.H., S.P., and V.G.; formal analysis, Y.H., S.P., and V.G.; investigation, Y.H.; resources, S.P. and V.G.; data curation, Y.H.; writing—original draft preparation, Y.H.; writing—review and editing, Y.H., S.P., and V.G.; visualization, Y.H.; supervision, S.P. and V.G.; project administration, S.P. and V.G; funding acquisition, S.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Orange, France.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data supporting reported results can be found at https://github.com/yhu01/BMS (accessed on 1 February 2022).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Hu, Y.; Gripon, V.; Pateux, S. Leveraging the feature distribution in transfer-based few-shot learning. In *Artificial Neural Networks and Machine Learning—ICANN 2021, Proceedings of the 30th International Conference on Artificial Neural Networks, Bratislava, Slovakia, 14–17 September 2021*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 487–499.
2. Mangla, P.; Kumari, N.; Sinha, A.; Singh, M.; Krishnamurthy, B.; Balasubramanian, V.N. Charting the right manifold: Manifold mixup for few-shot learning. In Proceedings of the IEEE Winter Conference on Applications of Computer Vision, Snowmass Village, CO, USA, 1–5 March 2020; pp. 2218–2227.
3. Wang, X.; Huang, T.E.; Gonzalez, J.; Darrell, T.; Yu, F. Frustratingly Simple Few-Shot Object Detection. In Proceedings of the 37th International Conference on Machine Learning, ICML 2020, Virtual Event, 13–18 July 2020; Volume 119, pp. 9919–9928.

4. Wang, Y.; Bryan, N.J.; Cartwright, M.; Bello, J.P.; Salamon, J. Few-Shot Continual Learning for Audio Classification. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, 6–11 June 2021; pp. 321–325.
5. Wolters, P.; Careaga, C.; Hutchinson, B.; Phillips, L. A Study of Few-Shot Audio Classification. *arXiv* **2020**, arXiv:2012.01573.
6. Zhang, S.; Qin, Y.; Sun, K.; Lin, Y. Few-Shot Audio Classification with Attentional Graph Neural Networks. In *Interspeech 2019, Proceedings of the 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15–19 September 2019*; Kubin, G., Kacic, Z., Eds.; ISCA: Yuma, AZ, USA, 2019; pp. 3649–3653.
7. Bansal, T.; Jha, R.; McCallum, A. Learning to Few-Shot Learn Across Diverse Natural Language Classification Tasks. In Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain, 8–13 December 2020; Scott, D., Bel, N., Zong, C., Eds.; International Committee on Computational Linguistics: Barcelona, Spain, 2020; pp. 5108–5123.
8. Bansal, T.; Jha, R.; Munkhdalai, T.; McCallum, A. Self-Supervised Meta-Learning for Few-Shot Natural Language Classification Tasks. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, 16–20 November 2020; Webber, B., Cohn, T., He, Y., Liu, Y., Eds.; Association for Computational Linguistics: Seattle, USA 2020; pp. 522–534.
9. Bao, Y.; Wu, M.; Chang, S.; Barzilay, R. Few-shot Text Classification with Distributional Signatures. In Proceedings of the 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, 26–30 April 2020.
10. Sun, L.; Li, C.; Ding, X.; Huang, Y.; Chen, Z.; Wang, G.; Yu, Y.; Paisley, J.W. Few-shot medical image segmentation using a global correlation network with discriminative embedding. *Comput. Biol. Med.* **2022**, *140*, 105067. [CrossRef] [PubMed]
11. Tang, H.; Liu, X.; Sun, S.; Yan, X.; Xie, X. Recurrent mask refinement for few-shot medical image segmentation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 11–17 October 2021; pp. 3918–3928.
12. Ouyang, C.; Biffi, C.; Chen, C.; Kart, T.; Qiu, H.; Rueckert, D. Self-supervision with superpixels: Training few-shot medical image segmentation without annotation. In *Computer Vision—ECCV 2020, Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23– 28 August 2020* ; Springer: Berlin/Heidelberg, Germany, 2020; pp. 762–780.
13. Finn, C.; Abbeel, P.; Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; Volume 70, pp. 1126–1135.
14. Ravi, S.; Larochelle, H. Optimization as a Model for Few-Shot Learning. In Proceedings of the 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, 24–26 April 2017.
15. Thrun, S.; Pratt, L. *Learning to Learn*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2012.
16. Torrey, L.; Shavlik, J. Transfer learning. In *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*; IGI Global: Hershey, PA, USA, 2010; pp. 242–264.
17. Das, D.; Lee, C.S.G. A Two-Stage Approach to Few-Shot Learning for Image Recognition. *IEEE Trans. Image Process.* **2020**, *29*, 3336–3350. [CrossRef] [PubMed]
18. Chen, W.; Liu, Y.; Kira, Z.; Wang, Y.F.; Huang, J. A Closer Look at Few-shot Classification. In Proceedings of the 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, 6–9 May 2019.
19. Hu, Y.; Gripon, V.; Pateux, S. Graph-based interpolation of feature vectors for accurate few-shot classification. In Proceedings of the 2020 25th International Conference on Pattern Recognition (ICPR), Milan, Italy, 10–15 January 2021; pp. 8164–8171.
20. Lichtenstein, M.; Sattigeri, P.; Feris, R.; Giryes, R.; Karlinsky, L. Tafssl: Task-adaptive feature sub-space learning for few-shot classification. In *Computer Vision—ECCV 2020, Proceedings of the 16th European Conference, Glasgow, UK, 23–28 August 2020*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 522–539.
21. Li, Z.; Zhou, F.; Chen, F.; Li, H. Meta-SGD: Learning to Learn Quickly for Few Shot Learning. *arXiv* **2017**, arXiv:1707.09835.
22. Antoniou, A.; Edwards, H.; Storkey, A.J. How to train your MAML. In Proceedings of the 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, 6–9 May 2019.
23. Bertinetto, L.; Henriques, J.F.; Torr, P.H.S.; Vedaldi, A. Meta-learning with differentiable closed-form solvers. In Proceedings of the 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, 6–9 May 2019.
24. Zhang, H.; Zhang, J.; Koniusz, P. Few-shot Learning via Saliency-guided Hallucination of Samples. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 2770–2779.
25. Chen, Z.; Fu, Y.; Wang, Y.X.; Ma, L.; Liu, W.; Hebert, M. Image deformation meta-networks for one-shot learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 8680–8689.
26. Mensink, T.; Verbeek, J.; Perronnin, F.; Csurka, G. Metric learning for large scale image classification: Generalizing to new classes at near-zero cost. In *Computer Vision—ECCV 2012, Proceedings of the European Conference on Computer Vision, Florence, Italy, 7–13 October 2012* ; Springer: Berlin/Heidelberg, Germany, 2012; pp. 488–501.
27. Chapelle, O.; Scholkopf, B.; Zien, A. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Trans. Neural Netw.* **2009**, *20*, 542–542. [CrossRef]
28. Ye, H.J.; Hu, H.; Zhan, D.C.; Sha, F. Few-shot learning via embedding adaptation with set-to-set functions. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 8808–8817.
29. Zhang, C.; Cai, Y.; Lin, G.; Shen, C. DeepEMD: Few-Shot Image Classification With Differentiable Earth Mover's Distance and Structured Classifiers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 12203–12213.
30. Kang, D.; Kwon, H.; Min, J.; Cho, M. Relational Embedding for Few-Shot Classification. *arXiv* **2021**, arXiv:2108.09666.

31. Rizve, M.N.; Khan, S.H.; Khan, F.S.; Shah, M. Exploring Complementary Strengths of Invariant and Equivariant Representations for Few-Shot Learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, Virtual, 19–25 June 2021; pp. 10836–10846.
32. Rajasegaran, J.; Khan, S.H.; Hayat, M.; Khan, F.S.; Shah, M. Self-supervised Knowledge Distillation for Few-shot Learning. *arXiv* **2020**, arXiv:2006.09785.
33. Zhang, Z.; Sabuncu, M.R. Self-Distillation as Instance-Specific Label Smoothing. In *Advances in Neural Information Processing Systems 33, Proceedings of the Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, Virtual, 6–12 December 2020*; Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H., Eds.; MIT Press: Cambridge, MA, USA, 2020.
34. Tian, Y.; Wang, Y.; Krishnan, D.; Tenenbaum, J.B.; Isola, P. Rethinking Few-Shot Image Classification: A Good Embedding is All You Need? In *Computer Vision—ECCV 2020, Proceedings of the 16th European Conference, Glasgow, UK, 23–28 August 2020*; Vedaldi, A., Bischof, H., Brox, T., Frahm, J., Eds.; Springer: Berlin/Heidelberg, Germany, 2020; Volume 12359, pp. 266–282. [CrossRef]
35. Wang, Y.; Chao, W.; Weinberger, K.Q.; van der Maaten, L. SimpleShot: Revisiting Nearest-Neighbor Classification for Few-Shot Learning. *arXiv* **2019**, arXiv:1911.04623.
36. Gripon, V.; Hacene, G.B.; Löwe, M.; Vermet, F. Improving Accuracy of Nonparametric Transfer Learning via Vector Segmentation. In Proceedings of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, AB, Canada, 15–20 April 2018; pp. 2966–2970.
37. Yang, S.; Liu, L.; Xu, M. Free Lunch for Few-shot Learning: Distribution Calibration. In Proceedings of the 9th International Conference on Learning Representations, ICLR 2021, Virtual, 3–7 May 2021.
38. Liu, Y.; Lee, J.; Park, M.; Kim, S.; Yang, E.; Hwang, S.J.; Yang, Y. Learning to Propagate Labels: Transductive Propagation Network for Few-Shot Learning. In Proceedings of the 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, 6–9 May 2019.
39. Agarap, A.F. Deep Learning using Rectified Linear Units (ReLU). *arXiv* **2018**, arXiv:1803.08375.
40. Tukey, J.W. *Exploratory Data Analysis*; Springer: Reading, MA, USA 1977; Volume 2.
41. Cinbis, R.G.; Verbeek, J.; Schmid, C. Approximate fisher kernels of non-iid image models for image categorization. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *38*, 1084–1098. [CrossRef] [PubMed]
42. Villani, C. *Optimal Transport: Old and New*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2008; Volume 338.
43. Cuturi, M. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2013; pp. 2292–2300.
44. Solomon, J.; De Goes, F.; Peyré, G.; Cuturi, M.; Butscher, A.; Nguyen, A.; Du, T.; Guibas, L. Convolutional wasserstein distances: Efficient optimal transportation on geometric domains. *ACM Trans. Graph. (TOG)* **2015**, *34*, 1–11. [CrossRef]
45. Dempster, A.P.; Laird, N.M.; Rubin, D.B. Maximum likelihood from incomplete data via the EM algorithm. *J. R. Stat. Soc. Ser. B* **1977**, *39*, 1–22.
46. Zagoruyko, S.; Komodakis, N. Wide Residual Networks. In Proceedings of the British Machine Vision Conference 2016, BMVC 2016, York, UK, 19–22 September 2016; Wilson, R.C., Hancock, E.R., Smith, W.A.P., Eds.; BMVA Press: York, UK, 2016.
47. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778.
48. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
49. Vinyals, O.; Blundell, C.; Lillicrap, T.; Wierstra, D. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2016; pp. 3630–3638.
50. Liu, J.; Song, L.; Qin, Y. Prototype rectification for few-shot learning. In *Computer Vision—ECCV 2020, Proceedings of the 16th European Conference, Glasgow, UK, 23–28 August 2020*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 741–756.
51. Ziko, I.; Dolz, J.; Granger, E.; Ayed, I.B. Laplacian regularized few-shot learning. In Proceedings of the International Conference on Machine Learning, Vienna, Austria, 12–18 July 2020; pp. 11660–11670.
52. Boudiaf, M.; Ziko, I.M.; Rony, J.; Dolz, J.; Piantanida, P.; Ayed, I.B. Transductive Information Maximization For Few-Shot Learning. *arXiv* **2020**, arXiv:2008.11297.
53. Kye, S.M.; Lee, H.; Kim, H.; Hwang, S.J. Transductive Few-shot Learning with Meta-Learned Confidence. *arXiv* **2020**, arXiv:2002.12017.
54. Rodríguez, P.; Laradji, I.; Drouin, A.; Lacoste, A. Embedding propagation: Smoother manifold for few-shot classification. In *Computer Vision—ECCV 2020, Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020* ; Springer: Berlin/Heidelberg, Germany, 2020; pp. 121–138.
55. Snell, J.; Swersky, K.; Zemel, R. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2017; pp. 4077–4087.
56. Rusu, A.A.; Rao, D.; Sygnowski, J.; Vinyals, O.; Pascanu, R.; Osindero, S.; Hadsell, R. Meta-Learning with Latent Embedding Optimization. In Proceedings of the 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, 6–9 May 2019.
57. Chobola, T.; Vasata, D.; Kordík, P. Transfer learning based few-shot classification using optimal transport mapping from preprocessed latent space of backbone neural network. *arXiv* **2021**, arXiv:2102.05176.

58. Simon, C.; Koniusz, P.; Nock, R.; Harandi, M. Adaptive Subspaces for Few-Shot Learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 4136–4145.
59. Verma, V.; Lamb, A.; Beckham, C.; Najafi, A.; Mitliagkas, I.; Lopez-Paz, D.; Bengio, Y. Manifold mixup: Better representations by interpolating hidden states. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 6438–6447.
60. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 448–456.
61. DIAgostino, R. An omnibus test of normality for moderate and large sample sizes. *Biometrika* **1971**, *58*, 341–348. [CrossRef]
62. D'AGOSTINO, R.; Pearson, E.S. Tests for departure from normality. Empirical results for the distributions of b 2 and $\sqrt{b}$. *Biometrika* **1973**, *60*, 613–622. [CrossRef]
63. Zhang, H.; Cao, Z.; Yan, Z.; Zhang, C. Sill-Net: Feature Augmentation with Separated Illumination Representation. *arXiv* **2021**, arXiv:2102.03539.
64. Júnior, P.R.M.; De Souza, R.M.; Werneck, R.d.O.; Stein, B.V.; Pazinato, D.V.; de Almeida, W.R.; Penatti, O.A.; Torres, R.d.S.; Rocha, A. Nearest neighbors distance ratio open-set classifier. *Mach. Learn.* **2017**, *106*, 359–386. [CrossRef]
65. Bendale, A.; Boult, T.E. Towards open set deep networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 1563–1572.
66. Liu, B.; Kang, H.; Li, H.; Hua, G.; Vasconcelos, N. Few-shot open-set recognition using meta-learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 8798–8807.
67. Jeong, M.; Choi, S.; Kim, C. Few-shot Open-set Recognition by Transformation Consistency. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 12566–12575.