Few-Shot Image Classification Along Sparse Graphs

Joseph F Comer Philip L Jacobson
Heiko Hoffmann
HRL Laboratories, LLC
3011 Malibu Canyon Rd, Malibu, CA 90265
joseph.comer@ncf.edu, philip_jacobson@berkeley.edu, drawfind@gmail.com

Abstract

Few-shot learning remains a challenging problem, with unsatisfactory 1-shot accuracies for most real-world data. Here, we present a new perspective for data distributions in the feature space of a deep network and show how to exploit this perspective for few-shot learning. First, we observe that nearest neighbors in the feature space are with high probability members of the same class while generally two random points from one class are not much closer to each other than two points between classes. This observation suggests that classes in feature space form sparse, loosely connected graphs instead of dense clusters. To exploit this property, we propose using label propagation to the nearest unlabeled data and then using a kernel PCA reconstruction error as decision boundary in feature-space for the data distribution of each class. Using this method, which we call "K-Prop," we demonstrate largely improved few-shot learning performances (e.g., 83% accuracy for 1-shot 5-way classification on the RESISC45 satellite-images dataset) for datasets for which a backbone network can be trained to produce high within-class nearest-neighbor probabilities. We demonstrate this relationship using six different datasets.

1. Introduction

Learning from few labeled examples, or "few-shot" learning, is needed for applications where labels are expensive or hard to obtain or where adapting to new data has to be fast. But few-shot learning remains a challenging problem. Particularly, with only 1 to 5 labels per class, classification accuracies are typically low on real-world data [13].

The simplest approach to few-shot learning is to adapt (fine-tune) a pre-trained network to a new target dataset based on a small set of available labeled data [5]. Either the weights of the entire network are adapted or only the final classification layer. In the latter case, the network is typically split into two parts: a backbone network, consisting, e.g., of multiple convolutional and pooling layers and a head consisting of a multi-layer perception or a linear mapping. Here, the backbone maps the input data into a so-called feature space. For few-shot learning, the weights of the backbone are typically frozen while the weights of the head are adapted. With comprehensive pre-training of the backbone (on a meta training set), training a linear classifier in the feature space can provide decent results [22], though the 1-shot results are typically in the best case in the 60-70% accuracy regime for 5-way classification.

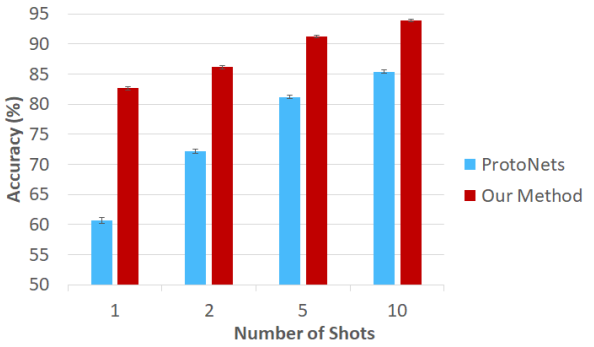

Figure 1: Comparing the accuracy of our method, K-Prop, with ProtoNets on the RESISC45 dataset for 5-way classification (mean ± SE).

As alternatives to a linear classifier baseline, various methods have been suggested, broadly falling into three categories: initialization-based, metric-based, and generative methods. Initialization-based methods, such as MAML [8; 17], search for a good initialization of the model weights for which a good performance on downstream tasks can be achieved with few labeled examples and in few training epochs. Metric-based methods work by replacing the linear classifier with a more sophisticated way of comparing distances between unseen data and the few labeled points. Some examples include ProtoNets [21], which compute distances to prototypical references points for each class, and Adaptive Subspaces [20], which compute distances to subspaces fitted to the data distribution of a class in feature

space (Fig. 2). These methods do not necessarily provide an improvement over a linear classifier, which can, e.g., beat ProtoNets if having a well trained backbone [22]. Lastly, generative methods augment the original data with novel data for training [25]. For example, MetaGAN uses an existing few-shot learning method and boosts its performance by generating additional data, though the reported improvements have been only by a few percent [25].

Here, we provide an alternative to these approaches for classifying data in the feature space. As a backbone network, we consider either a pretrained network or a self-supervised trained network, which is trained on the unlabeled target data. Here, we assume that we have access to a large amount of unlabeled data in the target domain, while the labeled data are scant.

As a background, we observed that the pairwise distances between images in feature space are fairly similar to each other (most are within $\pm 50\%$ of the mean distance). That is, we cannot expect data points of one class be clustered together in Euclidean space. In addition, we observed that for some datasets, the nearest neighbors in feature space belong with high probability ($> 90\%$) to the same class. These two properties suggest a data distribution of a class that resembles a loosely connected sparse graph, and the graphs of different classes are tightly intermingled.

Based on these insights, we construct a few-shot learning method as follows: given a few labeled points in feature space, we first propagate the labels to nearby unlabeled points over nearest-neighbor links. Second, given the resulting data-point distribution, we compute the kernel principal component analysis (kernel PCA) reconstruction error [10] as a distance measure of a test point to each class. The hypersurfaces of equal kernel PCA reconstruction error have been shown to follow the shape of any distribution [10].

Our experiments show that this new method outperforms state-of-the-art few-shot learning methods, like ProtoNets and Adaptive Subspaces, particularly, for 1-shot learning on certain data sets. We found that these datasets in combination with a backbone network share a common property: in feature space, the nearest neighbors are with high probability part of the same class.

Contributions. In summary, we make the following three main contributions:

i Provide a new perspective on how to interpret data in the feature space based on our observation of pairwise data-point distances.

ii Introduce a new method, K-Prop, combining self-supervised learning, label propagation, and kernel PCA for few-shot learning.

iii Show a relationship between high few-shot learning performance with our method and data-point distances in feature space.

The remainder of this article is organized as follows: Section 2 describes related work; Section 3 provides the background for our new method including our observation of pairwise data-point distances and nearest-neighbor relationships; Section 4 describes our proposed method; Section 5 our experiments with the corresponding results in Section 6. Finally, Section 7 concludes with a summary, discussion of limits and potential risks to society, and outlook for future work.

2. Related Work

For few-shot learning, a subset of methods deals with means for mapping from the feature space onto image classes. This mapping is generally much less complex than the mapping from the image space onto the feature space, for which usually deep multi-layer convolutional neural networks are used. This reduced complexity allows for adaptation to a target dataset with only a few labels. Examples of this approach are ProtoNets [21] and Adaptive Subspaces [20].

ProtoNets were introduced as an improvement over MatchingNets [23], which is a popular few-shot learning technique, and Adaptive Subspaces were introduced as an improvement over ProtoNets [20]. From a certain angle, our approach could be viewed as an extension of Adaptive Subspaces to non-linear subspaces, which we describe with kernel PCA. So, there is a natural progression of methods from the reference points in ProtoNets over subspaces to kernel PCA (Fig. 2).

Label propagation has been suggested before. For example, Zhou et al. [26] introduced label propagation through diffusion in a semi-supervised learning setting if the data manifold is sufficiently smooth. More recently, authors studied label propagation in feature space: Iscen et al. [12] construct a k-nearest-neighborhood graph and propagate labels with a diffusion process. Liu et al. [15] construct a neighborhood graph with a Gaussian similarity matrix and learn the parameters for label propagation in a meta-learning setting. Benato et al. [1] map from the feature space onto a t-SNE-generated 2-dimensional plane before propagating labels.

None of these works mentioned the property that we observed: that nearest neighbors can be with high probability within the same class for certain datasets and backbones.

3. Background

In this section, we provide the background for our new method: 1) self-supervised learning of a backbone network, 2) our observation of pairwise data-point distances, and 3) the kernel PCA reconstruction error.

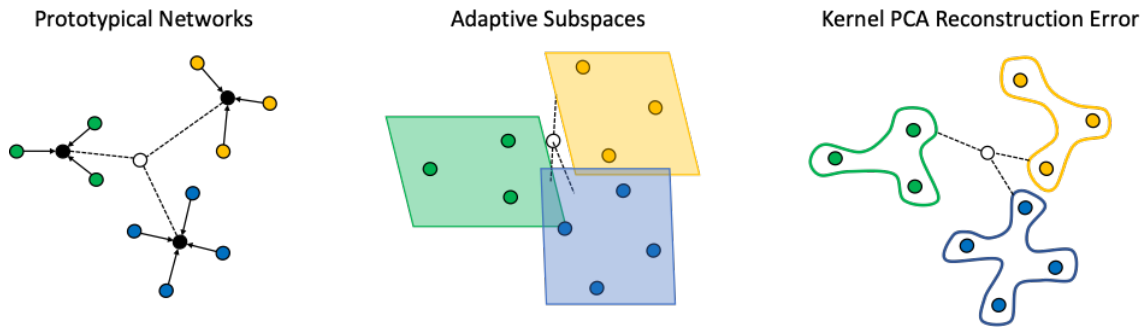

Figure 2: Methods for classification in feature space: ProtoNets [21], Subspaces [20], and kernel PCA.

3.1. Self-supervised learning

As unlabeled data are often less expensive to obtain, one common approach to few-shot learning is to use so-called self-supervision, wherein a proxy task is employed to pretrain a backbone network to produce features which can be leveraged for the downstream few-shot classification task. Using self-supervised learning to train a backbone network has been shown to rival *supervised* training based on linear-classifier accuracy on the trained features for certain datasets [4; 14].

Li et al. [14] trained various architectures in the multistage vision transformers (ViT) family using a self-supervision scheme which builds off of DINO [2]. ViT architectures work by first splitting an image into a regular grid of non-overlapping patches, flattening and (optionally) projecting the patches, and then performing sparse multi-head attention on the collection of patches.

In DINO, an exponential moving average ViT teacher network and a student network of the same architecture are fed different augmentations (views) of the full image, and the view-level features produced by each network are then fed to a shared prediction head which maximizes agreement between the feature representations of the two views. Li et al. [14] used an additional region-level task which similarly enforces similarity of the top-level feature representations of the various patches by first matching each student feature to the most similar output feature of the same layer of the teacher network, and then using the mean similarity of the resulting collection of feature pairs as the loss (see [14] for full details).

3.2. Pairwise data-point distances

We investigated the geometric structure of images, as embedded in the high-dimensional space where each pixel describes one dimension. For a selected subset of Imagenet [7] classes, we computed pairwise distances between images within each class and between classes. As a result, in the original image space, the data points have almost the same distance to each other (Fig. 3).

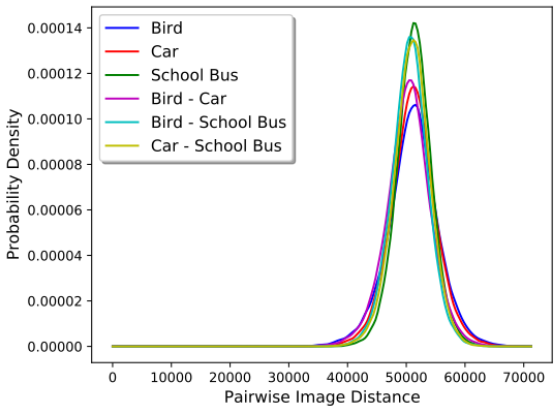

Figure 3: Pairwise distances (within and between classes) in the original image space for selected Imagenet classes.

Next, we mapped the images into the 512-dimensional feature space of a ResNet18 network, which was pre-trained on Imagenet. Here, the pairwise distances show more structure (Fig. 4), but a large part of the data still have similar distances to each other. In addition, for the Imagenette dataset, we also computed the pairwise distances in the 512-dimensional feature space of an Imagenet-pretrained ResNet18 network. For the intra-class pairwise distances, we observed a mean of 23.25 ± 3.48 SD ($n = 4.5 * 10^6$). For the inter-class distances, we observed 29.06 ± 3.15 (mean $\pm$ SD, $n = 4 * 10^7$). That is, the difference between inter and intra-class distances is relatively small, which matches the qualitative observation in Fig. 4.

In addition, we evaluated the probability, p_{NN}, of a nearest neighbor in feature space being in the same class. We computed this probability for the Imagenet-pretrained ResNet18 features (Tab. 1) as well as for the EsViT features (Tab. 2) for six different datasets (RESISC45, CUB, Imagenette, EuroSat, CropDisease, and Fungi - see also Experiments). For RESISC45, Imagenette, EuroSat, and CropDis-

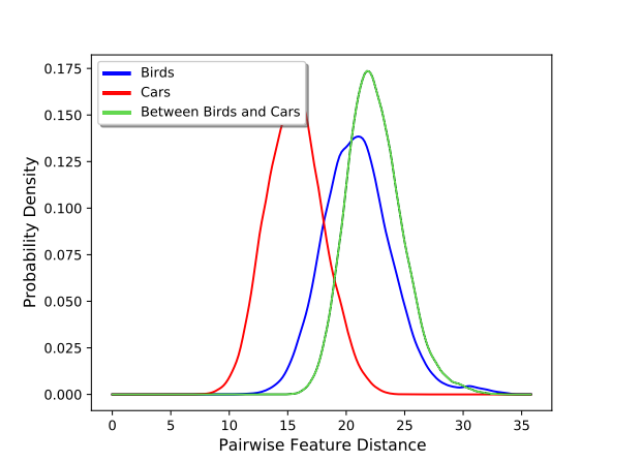

Figure 4: Pairwise distances in the 512-dimensional feature space of an Imagenet-pretrained Resnet18.

ease, we computed 100 trials, in each picking either 5 or 10 classes at random. In each trial, we computed the pairwise distances in feature space between all training data of the selected classes. For CUB and Fungi, we computed $1,000$ trials.

As a result, in most cases, these probabilities, p_{NN}, were in the 90s%. An outlier is CUB, for which p_{NN} was low ($< 60\%$) for the EsViT features.

# of Classes	RESISC	CUB	Imagenette
5	94.5 ± 5.1	90.0 ± 11.2	98.1 ± 1.5
10	89.5 ± 8.2	82.6 ± 14.6	96.7 ± 2.5
# of Classes	EuroSat	Crop	Fungi
5	92.5 ± 6.5	97.7 ± 4.6	80.0 ± 19.0
10	86.4 ± 9.8	95.8 ± 6.5	70.9 ± 20.7

Table 1: Probability that a nearest neighbor is within the same class for the ImageNet-pretrained **Resnet18** features (mean $\pm$ SD).

# of Classes	RESISC	CUB	Imagenette
5	98.7 ± 2.1	59.7 ± 19.0	92.0 ± 4.0
10	97.1 ± 3.3	43.8 ± 20.4	87.8 ± 6.8
# of Classes	EuroSat	Crop	Fungi
5	98.0 ± 1.5	99.3 ± 1.6	88.7 ± 13.3
10	96.3 ± 2.2	98.5 ± 2.6	83.2 ± 15.5

Table 2: Probability that a nearest neighbor is within the same class for the self-supervised trained **EsViT** features (mean $\pm$ SD).

Figure 5a shows a common view of a data-point distribution in feature space. Based on our observations, we found this view to be misleading because it shows point clusters, while actually, points from one class do not cluster: pairwise distances are almost uniform, and the difference between intra-class and inter-class distances is small. So, instead, we hypothesize that the data are distributed along sparse graphs, and the graphs between classes are intermingled, as illustrated in Fig. 5b.

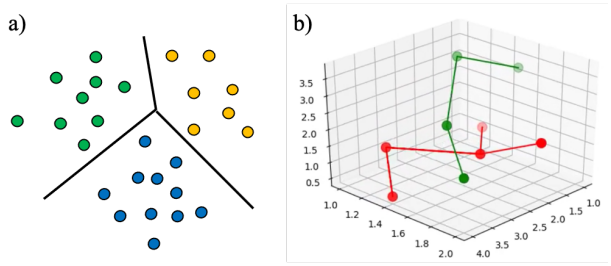

Figure 5: a) Common view of data-point distributions in feature space, b) our interpretation: intermingled sparse graphs. Each color denotes a different class.

3.3. Kernel Principle Component Analysis

Kernel Principal Component Analysis is a kernelized version of the PCA algorithm, essentially, expanding the linear method to non-linear data distributions [19]. Kernel PCA uses the so-called "kernel trick," i.e., the PCA is computed in a high-dimensional (potentially, infinitely dimensional) space, into which all data points, $\{\mathbf{x_i}\}$, are mapped, without actually carrying out the mapping, $\mathbf{\Phi(x_i)}$, into this space because the mapping appears only inside scalar products, and so, the scalar product can be replaced with a kernel function in the original space. A common kernel function is a Gaussian function,

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-||\mathbf{x}_i - \mathbf{x}_j||^2/(2\sigma^2)) \,. \qquad (1)$$

For this function, the corresponding mapping would actually be one into an infinite-dimensional space. Practically, however, the dimensionality is limited by the number of data points.

Computing kernel PCA involves computing the kernel matrix $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, transforming K to account for the non-zero mean of $\{\mathbf{\Phi(x_i)}\}$, and extracting a number, q, of eigenvectors corresponding to the q-largest eigenvalues [19; 10]. Here, for few-shot learning, we are dealing with a low-dimensional kernel matrix, so the computational cost of the eigenvalue extraction is negligible.

When computing kernel PCA for a non-linear data distribution, the corresponding reconstruction error (analogue to the reconstruction error in PCA) was introduced as a novelty-detection measure [10]. For Gaussian kernel functions, it turned out that the equipotential curves/surfaces of the reconstruction error describe well the non-linear shape of a data-point distribution [10]. Thus, we use this reconstruction error to compare between different classes in features space.

Let $\mathbf{z}$ be a vector in feature space and $\{\mathbf{x_i}\}$ be a distribution of vectors in feature space. Then, the reconstruction error of $\mathbf{z}$ is computed as follows [10],

$$L_{RE}(\mathbf{z}) = k(\mathbf{z},\mathbf{z}) - \frac{2}{n}\sum_{i=1}^{n} k(\mathbf{z},\mathbf{x}_i) + \frac{1}{n^2}\sum_{i,j=1}^{n} k(\mathbf{x}_i,\mathbf{x}_j) - \sum_{l=1}^{q} f_l(\mathbf{z})^2 \quad , \quad (2)$$

where f_l are the projections onto the principal components,

$$f_l(\mathbf{z}) = \sum_{i=1}^{n} \alpha_i^l \left[k(\mathbf{z},\mathbf{x}_i) - \frac{1}{n}\sum_{r=1}^{n} k(\mathbf{x}_i,\mathbf{x}_r) - \frac{1}{n}\sum_{r=1}^{n} k(\mathbf{z},\mathbf{x}_r) + \frac{1}{n^2}\sum_{r,s=1}^{n} k(\mathbf{x}_r,\mathbf{x}_s) \right], \quad (3)$$

and α_i^l are the eigenvectors of K. When using a Gaussian kernel, we have two hyperparameters, the width, σ, and the number of principal components, q.

4. Proposed Method

Based on the above background, we propose a new few-shot learning method, *K-Prop*, using self-supervised learning, label propagation, and the kernel PCA reconstruction error (Fig. 6). First, we exploit the fact that for many datasets, a good backbone, and thus a good mapping onto a feature space, can be obtained with self-supervised pre-training. Second, we exploit the nearest-neighbor within-class connections to artificially expand the number of labels with label propagation, and third, we exploit that the reconstruction error for kernel PCA can describe the sparse graph-like distribution of the few labeled points. In the following, we describe those three elements in more detail.

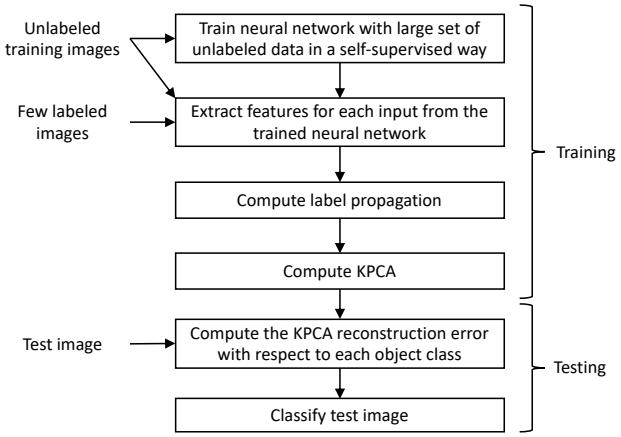

Figure 6: Process flow of our new method, *K-Prop*.

1) For self-supervised learning (SSL), we suggest the Es-ViT method [14]. We use SSL on the target training data (but not on the test data) to avoid cross-domain transfer loss. After training, we freeze the backbone parameters and weights.

2) For label propagation, we iteratively add a fixed number of extra labels. So, we propagate only into the neighborhood of the given labels, instead of diffusing into the entire unlabeled set. In each iteration step, we add only one unlabeled data point: in feature space, we find the point $\mathbf{x}_j$ with the smallest Euclidean distance to any of the points $\mathbf{x}_i$ in the set of labeled points, i.e.,

$$j = \text{argmin}_j \text{min}_i ||\mathbf{x}_i - \mathbf{x}_j|| \, . \quad (4)$$

The resulting $\mathbf{x}_j$ is then added to the set of labeled points, and the iteration continues until a given number of points is added to the number of originally labeled points. Figure 7 illustrates one example of this iterative process. Here, adding a nearest neighbor is shown as a link in a graph. For 1-shot learning, this process results in a sparse graph, and when starting with more labels, we generally end up with a multitude of graphs.

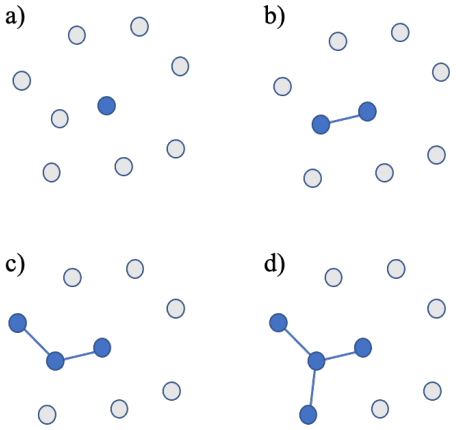

Figure 7: Label propagation: we iteratively add one unlabeled nearest neighbor at a time (a to d) building a graph of labeled points for each class.

3) We compute kernel PCA for each set of labeled feature points $\{\mathbf{x}_i\}$ for each class separately. Here, the labeled data contain the extra labels from the label propagation. When presenting a new test image, we first compute its mapping into the feature space, $\mathbf{z}$, and then compute the reconstruction error, $L_{RE}^c(\mathbf{z})$ for each class, c. We classify the test image based on the smallest reconstruction error, $\text{argmin}_c L_{RE}^c(\mathbf{z})$.

5. Experiments

We evaluate our method on six datasets and compare it against three other state-of-the-art methods and in ablation studies replacing key elements of K-Prop. Moreover,

RESISC

# of Shots	ProtoNets	MatchingNets	Subspaces	Ours (EsViT)	Ours (Resnet18)
1	60.66 ± 0.9	60.21 ± 0.9	60.55 ± 0.8	$\mathbf{83.18 \pm 0.6}$	71.15 ± 0.7
2	72.18 ± 0.7	69.59 ± 0.7	69.17 ± 0.7	$\mathbf{87.06 \pm 0.5}$	76.52 ± 0.6
5	81.21 ± 0.5	75.75 ± 0.5	79.16 ± 0.5	$\mathbf{92.08 \pm 0.4}$	84.84 ± 0.5

Crop

# of Shots	ProtoNets	MatchingNets	Subspaces	Ours (EsViT)	Ours (Resnet18)
1	77.40 ± 0.8	$\mathbf{79.33 \pm 0.8}$	75.87 ± 0.8	78.11 ± 0.8	78.53 ± 0.7
2	$\mathbf{89.08 \pm 0.6}$	85.21 ± 0.7	81.41 ± 0.6	85.20 ± 0.6	83.56 ± 0.6
5	91.38 ± 0.4	90.76 ± 0.4	91.95 ± 0.5	$\mathbf{92.82 \pm 0.4}$	91.20 ± 0.4

EuroSat

# of Shots	ProtoNets	MatchingNets	Subspaces	Ours (EsViT)	Ours (Resnet18)
1	39.93 ± 0.7	37.40 ± 0.7	27.55 ± 0.6	$\mathbf{77.20 \pm 0.5}$	67.90 ± 0.7
2	44.97 ± 0.7	39.57 ± 0.5	36.85 ± 0.6	$\mathbf{83.40 \pm 0.4}$	74.54 ± 0.6
5	55.08 ± 0.5	44.86 ± 0.6	40.42 ± 0.5	$\mathbf{90.44 \pm 0.3}$	82.86 ± 0.4

Imagenette

# of Shots	ProtoNets	MatchingNets	Subspaces	Ours (EsViT)	Ours (Resnet18)
1	31.57 ± 0.6	28.95 ± 0.6	-	$\mathbf{74.73 \pm 0.6}$	91.47 ± 0.4
2	30.99 ± 0.5	30.64 ± 0.5	-	$\mathbf{81.12 \pm 0.4}$	94.77 ± 0.2
5	37.76 ± 0.5	36.98 ± 0.5	-	$\mathbf{84.90 \pm 0.4}$	97.28 ± 0.1

CUB

# of Shots	ProtoNets	MatchingNets	Subspaces	Ours (EsViT)	Ours (Resnet18)
1	71.88 ± 0.9	72.36 ± 0.9	63.30 ± 0.9	40.79 ± 0.7	$\mathbf{81.08 \pm 0.8}$
2	81.44 ± 0.6	80.18 ± 0.7	68.38 ± 0.8	46.44 ± 0.9	$\mathbf{83.77 \pm 0.7}$
5	87.42 ± 0.5	83.64 ± 0.6	78.25 ± 0.6	53.59 ± 1.0	$\mathbf{89.39 \pm 0.6}$

Fungi

# of Shots	ProtoNets	MatchingNets	Subspaces	Ours (EsViT)	Ours (Resnet18)
1	58.87 ± 0.9	$\mathbf{60.79 \pm 0.9}$	35.34 ± 0.8	47.26 ± 1.1	42.87 ± 1.1
2	$\mathbf{74.65 \pm 0.8}$	74.21 ± 0.7	39.12 ± 0.8	68.18 ± 1.3	56.93 ± 1.2
5	$\mathbf{82.13 \pm 0.6}$	80.64 ± 0.8	63.41 ± 0.7	74.29 ± 1.2	64.21 ± 1.2

Table 3: Comparing the performance of our method (with either EsViT or Resnet18 backbone) with ProtoNets, MatchingNets, and Adaptive Subspaces on the RESISC, CropDisease, Eurosat and Imagenette datasets (mean $\pm$ SE). We disregard the Imagenette results with Imagenet-pretrained Resnet18 when comparing with other methods because of unfair advantage. The Adpative Subspaces method failed to converge when training on Imagenette, and so the results are missing.

we consider features produced from backbones trained either by fully-supervised pretraining on a dissimilar source domain or by self-supervision with zero labels on the target domain. In the former case, we used a Resnet18 network with frozen weights, which has been pretrained on Imagenet-1k. In the latter case, we trained a multistage transformer architecture with the self-supervision scheme proposed for EsViT [14].

For EsViT, we used the tiny sliding window architecture (Swin-T [14]). Images were divided into non-overlapping 16×16 pixel patches and two additional (global) random crops of size 224×224, all of which were subject to random transformations (augmentations) as described in [3]. We used a base learning rate of 0.0005 and cosine annealing, with a weight decay scaling linearly from 0.04 to 0.4 over 300 epochs. The network was trained for 300 epochs or until loss convergence.

Labels were propagated inside the training data set. We added 4 extra labels for 1-shot learning, 3 extra labels for 2-shot learning, and 2 extra labels for 5- and 10-shot learning. We chose these small numbers uniformly across datasets for speed and simplicity; although, for some datasets, using more extra labels would help (see Supplemental Material).

For kernel PCA, we used a Gaussian kernel with width $\sigma = 16$ and $q = \lfloor 2k/3 + 1 \rfloor$ principal components, where k is the number of labels per class (before label propagation).

We compare our method against ProtoNets [21], MatchingNets [23], and Adaptive Subspaces [20]. Each of the methods was implemented with a Resnet18 backbone, while otherwise following the training and augmentation routines in [5; 20]. Since these methods require meta learning to adapt parameters, we trained all three models using the labels for half of the classes in the dataset, while the other half was used to evaluate the few-shot learning performance. In contrast, using EsViT, our method did not use any labels (apart from few-shot) but used all classes of the unlabeled training set.

For our target datasets, we used Imagenette [11], RE-

SISC45 [6], CropDisease [16], EuroSat [9], Fungi [18], and CUB [24]. Imagenette is distributed under the Apache license, CropDisease under Creative Commons 1.0 Universal, Fungi under MIT, and CUB under Attribution 4.0 International. The licenses for RESISC45 and EuroSat are unknown. For all datasets, we used the default training/test data split.

For each dataset and for each pre-training scheme, we evaluate 5-way, k-shot performance using $k = 1, 2, 5$. For each k, we randomly generate $1,000$ tasks by sampling, for each task, 5 uniformly random classes from the test set and, then average the classification accuracies over all tasks. The backbone weights were frozen after pretraining. For each task, we evaluate our method using either the EsViT or Resnet18 backbone, compare against other methods and do ablation studies with our method. For these ablation studies, we replaced kernel PCA with a linear classifier and tested the linear classifier either with or without label propagation.

6. Results

Our new method, K-Prop, demonstrated a competitive performance (Tab. 3). For example, for 1-shot learning on RESISC45, K-Prop had a 83% accuracy compared to 61% for ProtoNets (Fig. 1). As another comparison, the best known 1-shot 5-way accuracy reported in the literature for RESISC45 with a Resnet18 backbone is 64.6% [13]. Interestingly, on this dataset, the nearest-neighbors in feature space were with high probability ($> 98\%$) part of the same class if trained with EsViT and higher compared to the Resnet18 backbone. Generally, we found that high p_{NN} for a dataset and backbone corresponded to a high few-shot classification accuracy (Fig. 8).

In addition, our method outperformed the linear classifier on the RESISC45, EuroSat, Imagenette, CUB, and CropDisease datasets for 1-, 2-, and 5-shot learning (Fig. 9). For low p_{NN} ($< 90\%$, Fungi and CUB with EsViT), we found that label propagation hurt the performance and a standard linear classifier was better. Moreover, our ablation studies showed that using the kernel PCA reconstruction error provided a boost over a linear classifier in most settings (Fig. 9, see Supplemental Material for the corresponding numerical values).

7. Conclusions

We provided a new perspective for looking at data distributions in feature space and showed how to exploit it for few-shot learning. Based on our observations, data in feature space tend to be loosely connected through nearest-neighbor connections, and the resulting sparse graphs are intermingled between different classes. Therefore, moderate label propagation followed by classification based on the kernel PCA reconstruction error showed promising results

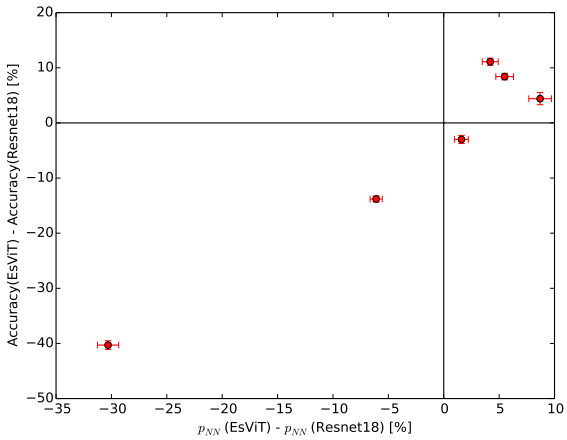

Figure 8: Comparison of 1-shot accuracy boost for EsViT relative to Resnet18 as function of difference in probability, p_{NN}, that nearest-neighbors are in the same class. Each data point corresponds to a different dataset (mean $\pm$ SE).

for few-shot learning. Moreover, we observed that a high probability of nearest neighbors being in the same class was indicative of a high classification accuracy.

Given the requirement of ProtoNets, MatchingNets, and Adaptive Subspaces for meta learning, they had access to additional labels for pretraining; on the other hand, they lacked full access to all classes of the unlabeled training data, which could bias the results in one or the other way (see Supplemental Material for additional comparisons). Moreover, using an Imagenet-pretrained Resnet18 is an unfair advantage for datasets that share similar images (mainly, Imagenette but also CUB to some extent), but we included those results because of the interesting relationship regarding the above nearest-neighbor same-class probability.

Limitations: The above relationship shows that the strategy of using SSL before our label propagation is limited to datasets in which SSL can move same-class data points close together in feature space. This strategy will fail if the inter-class difference is small: e.g., the CUB dataset consists of birds of similar shape, which belong to different classes due to relatively small differences in texture. On the other hand, we found that we can boost 1-shot learning performance if we use label propagation and kernel PCA on an Imagenet-pretrained network. This alternative in turn is limited to datasets that share similarities in features to Imagenet. Finally, despite the large improvements that we have seen for some datasets, the 1-shot accuracies are still too low for applications requiring reliable automated decision making and instead are more suitable for applications with a human in the loop or with multiple redundancies.

Potential negative societal impact: Given the above limitations, the risk to society is low. Mainly, we provide an

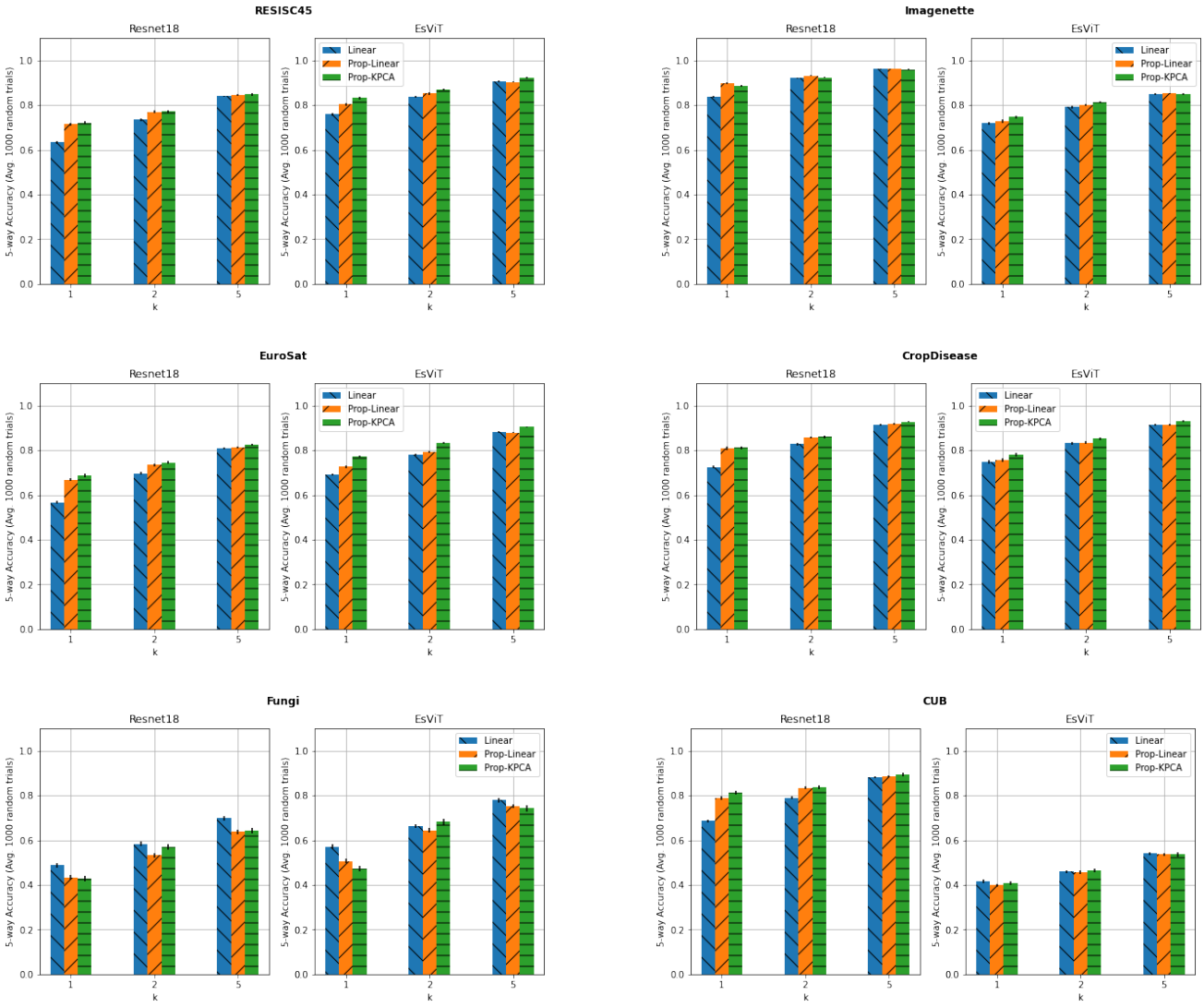

Figure 9: Ablation study with a linear classifier (Linear), label propagation followed by a linear classifier (Prop-Linear), and label propagation followed by kernel PCA (Prop-KPCA) for self-supervised features (EsViT) and features from a Resnet18 backbone pretrained on Imagenette (mean $\pm$ SE).

insight into feature-space distributions. In terms of applications, the most promising would be to update a target tracking system with only a few labels, where the system automatically filters input data (e.g., a video stream) for later analysis by a human operator. Like any tool, this system could potentially be abused by a bad actor, who would gain increased situational awareness. As mitigation, since the accuracies are still low, human verification is likely needed, limiting the potential abuse. In terms of environmental impact, this work has a positive contribution, reducing the required computational time for updating a model to new data because only kernel PCA has to be recomputed with a frozen backbone.

Future work: We hope this work inspires future work in the geometric properties of feature-space distributions leading to a better understanding of such distributions under various learning paradigms (unsupervised, self-supervised, and supervised). As part of that work, it would be interesting to find means to estimate p_{NN} without requiring labeled data. In addition, we plan to explore new ways to further improve the few-shot learning performance.

Acknowledgments

We thank Drs Soheil Kolouri and Navid Naderializadeh for discussions related to this work and Dr Kolouri for his help getting funding for this effort. This material is based upon work supported by the United States Air Force under Contract No. FA8750-19-C-0098. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Air Force and DARPA.

References

[1] Barbara Caroline Benato, Jancarlo Ferreira Gomes, Alexandru Cristian Telea, and Alexandre Xavier Falcão. Semi-supervised deep learning based on label propagation in a 2D embedded space. *CoRR*, abs/2008.00558, 2020. 2

[2] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. 2021. 3

[3] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *CoRR*, abs/2104.14294, 2021. 6

[4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *CoRR*, abs/2002.05709, 2020. 3

[5] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. *CoRR*, abs/1904.04232, 2019. 1, 6

[6] Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883, Oct 2017. 7

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255. Ieee, 2009. 3

[8] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1126–1135. JMLR.org, 2017. 1

[9] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019. 7

[10] Heiko Hoffmann. Kernel PCA for novelty detection. *Pattern Recognition*, 40(3):863–874, 2007. 2, 4, 5

[11] Jeremy Howard. Imagenette. 6

[12] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Label propagation for deep semi-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[13] Ashraful Islam, Chun-Fu Chen, Rameswar Panda, Leonid Karlinsky, Rogério Feris, and Richard J. Radke. Dynamic distillation network for cross-domain few-shot recognition with unlabeled data. *CoRR*, abs/2106.07807, 2021. 1, 7

[14] Chunyuan Li, Jianwei Yang, Pengchuan Zhang, Mei Gao, Bin Xiao, Xiyang Dai, Lu Yuan, and Jianfeng Gao. Efficient self-supervised vision transformers for representation learning, 2021. 3, 5, 6

[15] Yanbin Liu, Juho Lee, Minseop Park, Saehoon Kim, and Yi Yang. Transductive propagation network for few-shot learning. In *International Conference on Learning Representations*, 2019. 2

[16] Sharada P. Mohanty, David P. Hughes, and Marcel Salathé. Using deep learning for image-based plant disease detection. *Frontiers in Plant Science*, 7:1419, 2016. 7

[17] Aravind Rajeswaran, Chelsea Finn, Sham M Kakade, and Sergey Levine. Meta-learning with implicit gradients. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 1

[18] Brigit Schroeder and Yin Cui. FGVCx fungi classification challenge, 2018. 7

[19] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998. 4

[20] Christian Simon, Piotr Koniusz, Richard Nock, and Mehrtash Harandi. Adaptive subspaces for few-shot learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4135–4144, 2020. 1, 2, 3, 6

[21] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning, 2017. 1, 2, 3, 6

[22] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B. Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need? *CoRR*, abs/2003.11539, 2020. 1, 2

[23] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. 2, 6

[24] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 7

[25] Ruixiang Zhang, Tong Che, Zoubin Ghahramani, Yoshua Bengio, and Yangqiu Song. MetaGAN: An adversarial approach to few-shot learning. NIPS'18, page 2371–2380, Red Hook, NY, USA, 2018. Curran Associates Inc. 2

[26] Dengyong Zhou, Olivier Bousquet, Thomas Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16. MIT Press, 2004. 2