Fine-Grained Few-Shot Classification with Feature Map Reconstruction Networks

Davis Wertheimer* Luming Tang* Bharath Hariharan
Cornell University
{dww78,lt453,bh497}@cornell.edu

Abstract

In this paper we reformulate few-shot classification as a reconstruction problem in latent space. The ability of the network to reconstruct a query feature map from support features of a given class predicts membership of the query in that class. We introduce a novel mechanism for few-shot classification by regressing directly from support features to query features in closed form, without introducing any new modules or large-scale learnable parameters. The resulting Feature Map Reconstruction Networks are both more performant and computationally efficient than previous approaches. We demonstrate consistent and significant accuracy gains on four fine-grained benchmarks with varying neural architectures. Our model is also competitive on the non-fine-grained mini-ImageNet benchmark with minimal bells and whistles.

1. Introduction

Convolutional neural classifiers have achieved excellent performance in a wide range of settings and benchmarks, but this performance is achieved through large quantities of labelled images from the relevant classes. In practice, such a large quantity of human-annotated images may not always be available for the categories of interest. Producing a performant classifier in these settings requires a neural network that can rapidly adapt to novel, possibly unseen classes, using a small number of representative images.

This challenge is formalized in the *few-shot classification* problem, where networks are evaluated on individual *episodes* drawn from a task distribution. Each episode has associated classes of interest, with images in each class partitioned into a small *support set* and a larger *query set*. Using the ground truth class labels provided for the support images, the classifier must correctly classify the queries.

A particularly promising approach to few-shot classification is the family of *metric learning* techniques, where the standard parametric linear classifier head is replaced with a class-agnostic distance function. Membership in each class is determined by distance in latent space from a point or points known to belong to that class. Simple distance functions such as cosine distance [7, 4] and Euclidean distance [20] lead to surprisingly powerful classifiers, though more complex [19], non-Euclidean [10], and even learned parametric options [21] are possible, and yield significant gains.

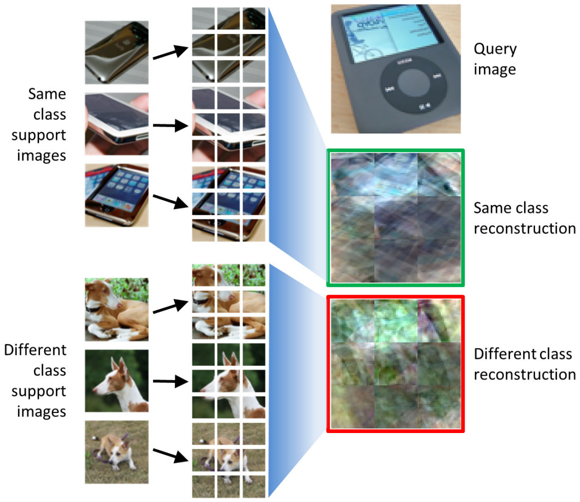
Same class support images
Query image
Same class reconstruction
Different class support images
Different class reconstruction

Figure 1. Visual intuition for FRN: we reconstruct each query image as a weighted sum of components from the support images. Reconstructions from the same class are better than reconstructions from different classes, enabling classification. FRN performs the reconstruction in latent space, as opposed to image space, here.

One overarching problem common to all these techniques is the fact that the convolutional feature extractors used to learn the metric spaces produce *feature maps* characterizing appearance at a *grid of spatial locations*, whereas the chosen distance functions require a *single vectorial representation for the entire image*. The researcher must decide how to convert the feature map into a vector representation. Global average-pooling, the standard solution for parametric softmax classifiers, averages together appearance information from disparate parts of the image, completely dis-

[*] Equal contribution

carding spatial details that might be necessary for fine distinctions. Flattening the feature map tensor into a single long vector preserves the individual feature vectors from each location [20, 21]. However, it also preserves the *location* of each feature vector, which is nuisance information - permuting the locations of the feature map completely alters the flattened embedding, even if the underlying semantic content is unchanged. The only way to remove this nuisance information is to increase both the size and sensitivity of the receptive fields, to the point where the feature vectors at all locations are the same. However, this also destroys granularity, and leads to overfitting to specious cues [5]. Optimally, we would like to preserve spatial detail while disentangling it from location at the same time.

We introduce Feature Map Reconstruction Networks (FRN), which accomplish this by framing class membership as a problem of *reconstructing feature maps*. Given a set of images all belonging to a single class, we produce the associated feature maps and collect the component feature vectors *across locations and images* into a single pool of support features. For each query image, we then attempt to reconstruct *every location* in the feature map as a weighted sum of support features, and the negative average squared reconstruction error is used as the class score. Images from the same class should be easier to reconstruct, since their feature maps contain similar embeddings, while images from different classes will be more difficult and produce larger reconstruction errors. By evaluating the reconstruction of the full feature map, FRN preserves the spatial details of appearance. But by allowing this reconstruction to use feature vectors from *any location* in the support images, FRN explicitly discards nuisance location information.

While prior methods based on feature map reconstruction exist, these methods either rely on constrained iterative procedures [31] or large learned attention modules [5, 9]. Instead, we frame feature map reconstruction as a ridge regression problem, allowing us to rapidly calculate a solution in closed form with only a single learned, soft constraint.

The resulting reconstructions from FRN are high quality and semantically informative, making FRN both simpler and more powerful than prior reconstruction-based approaches. We validate these claims by demonstrating across-the-board superiority on four fine-grained few-shot classification datasets (CUBirds [26], meta-iNat and tiered meta-iNat [29], and FGVC Aircraft [14]) and one general few-shot recognition benchmark (mini-ImageNet [25]). These results hold for both shallow and deep network architectures (Conv-4 [20, 11] and ResNet-12 [8, 23]).

2. Background and Related Work

The few-shot learning setup: Standard few-shot training and evaluation involves sampling task episodes from an overarching task distribution - typically, by repeatedly selecting small subsets from a larger set of classes. The number of classes per episode is referred to as the *way*, while the number of support images per class is the *shot*, so that episodes with five classes and one labelled image per class form a "5-way, 1-shot" classification problem. Few-shot classifiers are trained on a large, disjoint set of classes with many labelled images, typically using this same episodic scheme for each batched iteration of SGD. Optimizing the few-shot classifier over the task distribution teaches it to generalize to new tasks from a similar distribution. The classifier learns to learn new tasks, thus few-shot training is also referred to as "meta-learning" or "meta-training".

Prior work in few-shot learning: Existing approaches to few-shot learning can be loosely organized into the following two main-stream families. Optimization-based methods [6, 18, 16] aim to learn a good parameter initialization for the classifier. These learned weights can then be quickly adapted to novel classes using gradient-based optimization on only a few labeled samples. Metric-based methods, on the other hand, aim to learn a completely task-independent embedding that can generalize to novel categories under a chosen distance metric, such as Euclidean distance [20], cosine distance [7], hyperbolic distance [10], or a distance parameterized by a neural network [21].

As an alternative to the standard meta-learning framework, many recent papers [3, 23, 27] study the performance of standard end-to-end pre-trained classifiers on few-shot tasks. Given minimal modification, these classifiers are actually competitive with or even outperform episodic meta-training methods. Therefore some recent works [31, 30, 4] take advantage of both, and utilize meta-learning after pre-training, further boosting performance.

Few-shot classification through reconstruction: We are not the first to use feature map reconstruction as a proxy for few-shot classification. DeepEMD [31] formulates latent reconstruction as an optimal transport problem, solved using external iterative constrained convex optimization tools. This formulation is sophisticated and powerful, but training and inference come with significant computational cost compared to other methods, due to the reliance on iterative solvers and test-time SGD. CrossTransformer [5] and CrossAttention [9] add attention modules that project query features into the space of support features (or vice versa), and compare the class-conditioned projections to the target to predict class membership. These attention-based approaches introduce many additional learned parameters over and above the network backbone, and place largely arbitrary constraints on the projection matrix (weights are non-negative and rows must sum to 1). In contrast, FRN efficiently calculates least-squares-optimal reconstructions in closed form using only a single learnable constraint.

Closed-form solvers in few-shot learning: The use of closed-form solvers for few-shot classification is also not

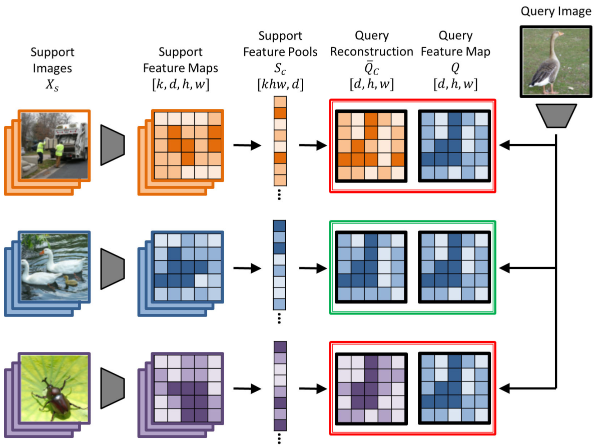
Support Images X_s
Support Feature Maps $[k,d,h,w]$
Support Feature Pools S_c $[khw,d]$
Query Reconstruction $\bar{Q}_c$ $[d,h,w]$
Query Feature Map Q $[d,h,w]$
Query Image

Figure 2. Overview of FRN classification for a k-shot problem. Support images are converted into feature maps (left), which are aggregated into class-conditional pools (middle). The best-fit reconstruction of the query feature map is calculated for each category, and the closest candidate yields the predicted class (right). h, w is feature map resolution and d is the number of channels.

entirely new, though to our knowledge they have not been applied in the explicit context of feature reconstruction. [2] uses ridge regression to map features directly to classification labels, while [21] accomplishes the same mapping with differentiable SVMs. Deep Subspace Networks [19] use the closed-formed projection distance from query embeddings to subspaces spanned by support points as the similarity measure. In contrast, FRN uses closed-form ridge regression to reconstruct entire feature maps, rather than performing direct comparisons between points in any one particular latent space, or regressing directly to class label targets.

3. Method

Feature Map Reconstruction Networks use the quality of query feature map reconstructions from support features as a proxy for class membership. The pool of features associated with each class in the episode is used to calculate a candidate reconstruction, with a better reconstruction indicating higher confidence for the associated class. In this section we describe the reconstruction mechanism of FRN in detail, and derive the closed-form solution used to calculate the reconstruction error and resulting class score. An overview is provided in Fig. 2. We discuss memory-efficient implementations and an optional pre-training scheme, and draw comparisons to prior reconstruction-based approaches.

3.1. Feature Map Ridge Regression

Let X_s denote the set of support images with corresponding class labels in an n-way, k-shot episode. We wish to predict a class label y_q for a single input query image x_q.

The output of the convolutional feature extractor for x_q is a feature map $Q \in \mathbb{R}^{r \times d}$, where r represents the spatial resolution (height times width) of the feature map, and d the number of channels. For each class $c \in C$, we pool all of the features across the k support image feature maps into a single matrix of support features $S_c \in \mathbb{R}^{kr \times d}$. We then attempt to reconstruct Q as a weighted sum of values in S_c by finding the matrix $W \in \mathbb{R}^{r \times kr}$ such that $WS_c \approx Q$. Finding the optimal $\bar{W}$ amounts to solving the linear least-squares problem:

$$\bar{W} = \underset{W}{\text{arg min}} \ ||Q - WS_c||^2 + \lambda ||W||^2 \tag{1}$$

where $||\cdot||$ is the Frobenius norm and λ weights the ridge regression penalty term used to ensure tractability when the linear problem is over- or under-constrained ($kr \neq d$).

The foremost benefit of the ridge regression formulation is that it admits a widely-known closed-form solution for $\bar{W}$ and the optimal reconstruction $\bar{Q}_c$ as follows:

$$\bar{W} = QS_c^T(S_cS_c^T + \lambda I)^{-1} \tag{2}$$

$$\bar{Q}_c = \bar{W}S_c \tag{3}$$

A similarity measure for Q and $\bar{Q}_c$ is given by the mean squared Euclidean distance over all feature map locations:

$$\langle Q, \bar{Q}_c \rangle = \frac{1}{r}||Q - \bar{Q}_c||^2 \tag{4}$$

For a given class c, we use the negative similarity $-\langle Q, \bar{Q}_c \rangle$ as the probability logit. We also incorporate a learnable temperature factor γ, following the findings of [4, 7, 30] that temperature scaling improves few-shot training. The final predicted probability is thus given by:

$$P(y_q = c | x_q) = \frac{e^{(-\gamma \langle Q, \bar{Q}_c \rangle)}}{\sum_{c' \in C} e^{(-\gamma \langle Q, \bar{Q}_{c'} \rangle)}} \tag{5}$$

We optimize our network by sending the predicted class probabilities for the query images in each episode through a cross-entropy loss, as in standard episodic meta-training. An overview of this process can be found in Fig. 2.

3.2. Learning the Degree of Regularization

It is not immediately clear how one should set the regularization parameter λ. Instead of choosing heuristically, we allow the network to *learn* λ through meta-learning. This is significant, as it allows the network to meta-learn the appropriate amount of regularization so that the reconstruction is *discriminative*, rather than strictly least-squares optimal.

Changing λ can have multiple effects. A large λ discourages sparse weights in W, but also reduces the norm of the reconstruction, increasing reconstruction error and limiting its discriminative power. We therefore disentangle the degree of regularization from the magnitude of $\bar{Q}_c$ by introducing a *learned recalibration term* ρ:

$$\bar{Q}_c = \rho \bar{W} S_c \tag{6}$$

By increasing ρ alongside λ, the network gains the ability to penalize large, sparse weights without sending all reconstructions to the origin at the same time.

Parametrizing λ and ρ: Note that in Eq. 1, the objective is the sum of the squared Frobenius norm of two different matrices, a residual error matrix and the weight matrix. These two matrices can have very different sizes: the first is $br \times d$ while the second is $br \times kr$. Thus when kr is much greater or less than d, one term in the objective can easily overwhelm the other. To ensure a balanced objective and stable training, we rescale the regularization term λ by a factor of $\frac{d}{kr}$. λ and ρ are parametrized as e^{α} and e^{β} to ensure non-negativity, with α and β initialized to zero.

Thus, all together, our final prediction is given by:

$$\lambda = \frac{d}{kr} e^{\alpha} \qquad \rho = e^{\beta} \tag{7}$$

$$\bar{Q}_c = \rho \bar{W} S_c = \rho Q S_c^T (S_c S_c^T + \lambda I)^{-1} S_c \tag{8}$$

$$P(y_q = c | x_q) = \frac{e^{(-\gamma \langle Q, \bar{Q}_c \rangle)}}{\sum_{c' \in C} e^{(-\gamma \langle Q, \bar{Q}_{c'} \rangle)}} \tag{9}$$

The model is meta-trained in a similar manner to prior work: sample episodes from a labeled base class dataset and minimize cross entropy on the predicted query labels [20].

Our approach introduces only three learned parameters: α, β and γ. The temperature γ is also used by prior work [7, 4, 30]. Ablations on α and β can be found in Sec. 5.1.

3.3. Parallelization

While we have described our approach as finding reconstructions for a single query image, it is relatively straightforward to find the reconstructions for an entire batch of query images. We are already calculating the optimal reconstruction for each of the r feature vectors in Q independently; all we need to do for a batch of b images is pool the features into a larger matrix $Q' \in \mathbb{R}^{br \times d}$ and run the algorithm as written. Thus for an n-way episode we will only ever need to run the algorithm n times, once for each support matrix S_c, regardless of the quantity or arrangement of queries. These n runs can also be parallelized, given parallel, highly optimized implementations of matrix multiplication and inversion.

3.4. Alternative Formulation

The formula for $\bar{Q}$ in Eq. 8 is efficient to compute when $d \gg kr$, as the most expensive step is inverting a $kr \times kr$ matrix that does not grow with d. Additionally, computing the matrix multiplications from left to right ensures that the network need never store a potentially large $d \times d$ matrix in memory. However, if the feature maps are large or the shot number is particularly high ($kr \gg d$), Eq. 8 may quickly become infeasible to compute. In this case an alternative formulation for $\bar{Q}$ exists, which swaps d for kr in terms of computational requirements. This formulation is owed to the Woodbury Identity as applied in [2]:

$$\bar{Q}_c = \rho \bar{W} S_c = \rho Q (S_c^T S_c + \lambda I)^{-1} S_c^T S_c \tag{10}$$

In this case, the most expensive step is inverting a $d \times d$ matrix, and by computing the matrix multiplications from right to left, we ensure that no large $kr \times kr$ or $br \times kr$ matrices need ever be stored in memory. Since r and d are determined in advance by the network architecture, the researcher is free to employ either formulation depending on the value of k. The network can also decide on the fly at test time. In terms of classifier performance the two formulations are algebraically equivalent and the choice is redundant. We make the arbitrary decision to employ Eq. 10 in our experiments rather than Eq. 8. Pseudo-code for this formulation is provided in the supplementary.

3.5. Auxiliary Loss

In addition to the classification loss, we employ an auxiliary loss that encourages support features from different classes to span the latent space [19]:

$$L_{\text{aux}} = \sum_{i \in C} \sum_{j \in C, j \neq i} ||\hat{S}_i \hat{S}_j^T||^2 \tag{11}$$

where $\hat{S}$ is row-normalized, with features projected to the unit sphere. This loss encourages orthogonality between features from different classes. Similar to [19], we downscale this loss by a factor of 0.03. We use L_{aux} as the auxiliary loss in our subspace network implementation [19], and it replaces the SimCLR episodes in our CrossTransformer implementation [5]. We include it in our own model for consistency, and include an ablation study in Sec. 5.1.

3.6. Pre-Training

Prior work [4, 30] has demonstrated that few-shot classifiers can benefit greatly from non-episodic pre-training. For traditional metric learning-based approaches, the feature extractor is initially trained as a linear classifier with global average-pooling on the full set of training classes. The linear layer is subsequently discarded, and the feature extractor is fine-tuned episodically.

This pre-training does not work out-of-the-box for FRN due to the novel way it performs classification. Because the linear classifier uses average-pooling, the feature extractor does not learn spatially distinct feature maps in the way that FRN requires. Episodic training subsequently diverges.

We therefore devise a new pre-training scheme for FRN. To keep the classifier consistent with FRN meta-training, we continue to use a feature reconstruction error as the predicted class logit. Similar to [31], the classification layer is parametrized as a set of class-specific *dummy features*. Thus in addition to the network backbone, we also have a learnable matrix $M_c \in \mathbb{R}^{r \times d}$ for each category c, which acts as a proxy for S_c. Following Eq. 10, for a sample x_q with feature map $Q \in \mathbb{R}^{r \times d}$, the category prediction is then:

$$\bar{Q}_c = \rho Q(M_c^T M_c + \lambda I)^{-1} M_c^T M_c \tag{12}$$

$$P(y_q = c|x_q) = \frac{e^{(-\gamma \langle Q, \bar{Q}_c \rangle)}}{\sum_{c' \in C} e^{(-\gamma \langle Q, \bar{Q}_{c'} \rangle)}} \tag{13}$$

It should be noted that C in this setting is no longer the sampled subset of episode categories, but rather the entire set of training classes for mini-ImageNet, $|C| = 64$. We then use this output probability distribution to calculate the standard cross-entropy classification loss. During the pre-training stage, we fix $\rho = \lambda = 1$ but keep γ a learnable parameter. After pre-training is finished, all learned matrices $\{M_c|c \in C\}$ are discarded (similar to the pre-trianed MLP classifier in [23, 30, 27, 3, 4]). The pre-trained model size is thus the same as when trained from scratch. We then load the pre-trained backbone parameters and γ into the meta-training pipeline, and train episodically as before.

While pre-training is broadly applicable and generally boosts performance, for the sake of fairness we do not pre-train any of our fine-grained experiments, as baseline methods do not consistently pre-train in these settings.

3.7. Relation to Prior Reconstructive Classifiers

CrossTransformer [5]: While FRN is not the first attempt at building a few-shot classifier based on feature map reconstruction, it is the first to do so explicitly in closed form. Some prior approaches instead approximate $\bar{W}$ using attention and extra learned projection layers. CrossTransformer is one such approach, which we re-implement in our experiments as a baseline (CTX). Using learned linear layers, CrossTransformer reprojects the feature pools S_c and Q into two different "key" and "value" subspaces, yielding S_1, Q_1 and S_2, Q_2. The reconstruction of Q_2 is given by:

$$\bar{Q}_c = \sigma(\frac{1}{\sqrt{d}} Q_1 S_1^T) S_2 \tag{14}$$

where $\sigma(\cdot)$ denotes a row-wise softmax and γ is the same temperature scaling parameter. While Eq. 14 is loosely analogous to Eq. 8, with the $\sqrt{d}$-scaled softmax replacing the inverted matrix term, we find that performance differs in practice. The CrossTransformer layer is also somewhat unwieldy: the two reprojection layers introduce extra parameters into the network, and during training it is necessary to store the $br \times kr$ matrix of attention weights $\sigma(\frac{1}{\sqrt{d}} Q_1 S_1^T)$ for back-propagation. This led to a noticeable memory footprint in our experiments.

DeepEMD [31]: Similar to the above approaches, DeepEMD solves for a $br \times kr$ reconstruction matrix $\bar{W}$ and uses reconstruction quality (measured as transport cost) as a proxy for class membership. This technique is more sophisticated and powerful than ridge regression, but also highly constrained. As a transport matrix, $\bar{W}$ must contain strictly nonnegative values, with rows and columns that sum to 1. More importantly, $\bar{W}$ cannot be calculated in closed form, and instead requires an iterative procedure which can be slow in practice and does not scale well beyond pairs of individual images. DeepEMD also requires finetuning via back-propagation at test time, whereas our approach scales out of the box to a range of values for k, r, d.

Deep Subspace Networks [19]: Subspace networks predict class membership by calculating the distance between the query point and its projections onto the latent subspaces formed by the support images for each class. This is almost exactly analogous to our approach with $r = 1$, with average-pooling performing the necessary spatial reduction. The crucial difference is that subspace networks assume (accurately) that $d \gg k$, whereas in our setting it is not always the case that $d \gg kr$. In fact, for many of our models S spans the latent space, so the projection interpretation falls apart and we instead rely on the ridge regression regularizer to keep the problem well-posed. We re-implement this approach as a baseline in our experiments, and include the original published numbers where available.

4. Experiments

Because Feature Map Reconstruction Networks focus on spatial details without overfitting to pose, we find that they are particularly powerful in the fine-grained few-shot recognition setting, where details are important and pose is not discriminative. We demonstrate clear superiority on four such fine-grained benchmarks. For general few-shot learning, pre-trained FRN achieves highly competitive results without additional whistles and bells.

Implementation details: We conduct experiments on two widely used backbones: 4-layer ConvNet (Conv-4) and ResNet-12. Same as [30, 11], Conv-4 consists of 4 consecutive 64-channel convolution blocks that each downsample by a factor of 2. The shape of the output feature maps for input images of size 84×84 is thus 64×5×5. For ResNet-12, we use the same implementation as [30, 23, 31, 11]. The input image size is the same as Conv-4 and the output feature map shape is 640×5×5. During training, we use the stan-

	Conv-4		**ResNet-12**	
Model	**1-shot**	**5-shot**	**1-shot**	**5-shot**
MatchNet$^\flat$ [25, 30, 31]	67.73	79.00	71.87	85.08
ProtoNet$^\flat$ [20, 30, 31]	63.73	81.50	66.09	82.50
Hyperbolic [10]	64.02	82.53	-	-
FEAT$^\flat$ [30]	68.87	82.90	-	-
DeepEMD$^\flat$ [31]	-	-	75.65	88.69
ICI$^\flat$ [28]	-	-	76.16	90.32
ProtoNet† [20]	63.42	83.01	79.09	90.59
DSN† [19]	65.66	84.54	79.42	90.34
CTX† [5]	69.91	86.83	77.38	89.95
FRN (ours) 1-shot	68.76	-	82.62	-
FRN (ours) 5-shot	**73.44**	**87.87**	**83.16**	**92.59**

Table 1. Performance on CUB using bounding-box cropped images as input, 5-way 1/5-shot. † denotes our own implementations. ♭ denotes the use of non-episodic pre-training.

Model	**Backbone**	**1-shot**	**5-shot**
Baseline++$^\flat$ [3]	ResNet-34	68.00±0.83	84.50±0.51
MatchNet [3, 25]	ResNet-34	73.49±0.89	86.51±0.52
ProtoNet [3, 20]	ResNet-34	72.94±0.91	87.86±0.47
MAML [3, 6]	ResNet-34	67.28±1.08	83.47±0.59
RelationNet [3, 21]	ResNet-34	69.72±0.98	83.18±0.54
LaplacianShot$^\flat$ [32]	ResNet-18	80.96	88.68
S2M2$^\flat$ [15]	WRN-28-10	80.68±0.81	90.85±0.44
Neg-Cosine$^\flat$ [12]	ResNet-18	72.66±0.85	89.40±0.43
Afrasiyabi *et al.*$^\flat$ [1]	ResNet-18	74.22±1.09	88.65±0.55
FRN (ours) 1-shot	ResNet-12	82.02±0.20	-
FRN (ours) 5-shot	ResNet-12	**83.55±0.19**	**92.92±0.10**

Table 2. Performance on CUB using raw images as input, 5-way 1/5-shot. ♭ denotes the use of non-episodic pre-training.

	Conv-4		**ResNet-12**	
Model	**1-shot**	**5-shot**	**1-shot**	**5-shot**
ProtoNet† [20]	47.72	69.42	66.57	82.37
DSN† [19]	47.12	66.36	68.16	81.85
CTX† [5]	50.27	67.30	60.77	76.36
FRN (ours) 1-shot	50.25	-	69.40	-
FRN (ours) 5-shot	**53.20**	**71.17**	**70.17**	**83.81**

Table 3. Performance on Aircraft, 5-way 1/5-shot. † denotes our own implementations.

dard data augmentation as in [30, 31, 27, 3], which includes random crop, right-left flip and color jitter.

For all experiments, we include results for 1-shot and 5-shot settings. Surprisingly, we found that FRN trained with 5-shot episodes consistently outperformed FRN trained with 1-shot episodes, even on 1-shot evaluation. For consistency, we still include the 1-shot model results, which are competitive in their own right.

4.1. Fine-Grained Few-Shot Classification

For our fine-grained experiments, we re-implement three baselines: Prototypical Networks (ProtoNet) [20], CrossTransformer (CTX) [5], and Deep Subspace Networks (DSN) [19]. Evaluation is performed on the standard 5-way, 5-shot and 1-shot settings. We average over 6000 episodes to obtain our accuracy scores and 95% confidence intervals where appropriate. For fair comparison, we do not use pre-training on any of our fine-grained benchmarks, and attempt to keep hyperparameters as close as possible to the standard values for prototypical networks on each dataset. Further details can be found in supplementary.

Caltech-UCSD Birds-200-2011 [26] (CUB) consists of 11,788 images from 200 classes. Following [3], we randomly split categories into 100 classes for training, 50 for validation and 50 for evaluation. Our split is identical to [22]. Prior work on this benchmark pre-processes the data in different ways: [3] uses raw images as input, while [30, 31] crop each image to a human-annotated bounding box. We conduct experiments on both settings for a fair comparison. Results for the cropped setting can be found in Table 1; results for uncropped are in Table 2. FRN is superior across the board, with a notable 3-point jump in accuracy from the nearest baseline in every single 1-shot setting. These results are achieved without any pre-training.

Note that our re-implemented baselines in Table 1 are competitive with (and in some cases beat outright) prior published numbers. This shows that in subsequent experiments without prior published numbers, our implemented baselines still provide fair competition. We do not give FRN an unfair edge - if anything, our baselines are more competitive, not less.

FGVC-Aircraft [14] contains 10,000 images spanning 100 airplane models. Following the same ratio as CUB, we split the classes into 50 train, 25 validation and 25 test. The random split is identical to [22]. The images are pre-cropped to the provided bounding box. Results for this benchmark are provided in Table 3, where FRN once again outperforms baseline methods in all settings.

Meta-iNat and Tiered Meta-iNat [29, 24] are benchmarks of animal species in the wild. These benchmarks are particularly difficult, as class distinctions are fine-grained, and images are not cropped or centered, and may contain multiple instances of the animal in question. We follow the class splits proposed by [29]: of 1135 classes with between 50 and 1000 images, one fifth (227) are randomly assigned to evaluation and the rest are used for training. While [29] originally propose a full 227-way, k-shot evaluation scheme with $10 \leq k \leq 200$, we instead perform standard 5-way, 1-shot and 5-shot evaluation, and leave extension to higher shot for future work. We report mean accuracy only, as per-class accuracy was not meaningfully different.

Tiered meta-iNat represents a more difficult version of meta-iNat where a large domain gap is introduced between the train and test categories. The 354 test classes are populated by insects and arachnids, while the remaining 781 classes (mammals, birds, reptiles, etc.) form the training

	random		tiered	
Model	1-shot	5-shot	1-shot	5-shot
ProtoNet† [20]	55.06	76.31	34.20	57.16
Covar. pool† [29]	56.93	77.08	36.03	57.63
DSN† [19]	58.02	77.27	36.81	60.01
CTX† [5]	59.69	78.60	36.80	61.01
FRN (ours) 1-shot	62.36	-	41.60	-
FRN (ours) 5-shot	**62.89**	**80.60**	**44.20**	**64.26**

Table 4. Performance on meta-iNat and tiered meta-iNat, 5-way 1/5-shot. All methods use Conv-4 as backbone network. † denotes our own implementations.

set. Training and evaluation is otherwise the same as for standard meta-iNat. Results for both benchmarks are provided in Table 4, with FRN again providing the best performance. We conclude that FRN is broadly effective at fine-grained few-shot classification.

4.2. General Few-Shot Classification

Mini-ImageNet [25] is a subset of ImageNet containing 100 classes in total, with 600 examples per class. Following [17], we split categories into 64 classes for training, 16 for validation and 20 for test. Compared to direct episodic meta-training from scratch, recent works [4, 23] gain a large advantage from pre-training on all the training data and labels, followed by episodic fine-tuning. We follow the framework of [30, 31] and pre-train our model on the entire training set as described in Sec. 3.6. Details are provided in supplementary.

Compared with recent state-of-the-art results in Table 5, FRN achieves highly competitive performance. FRN leverages pre-training, but no other extra techniques or tricks. FRN also requires no gradient-based finetuning at inference time, which makes it more efficient than many existing baselines in practice. We analyze the impact of pre-training on few-shot performance in Sec. 5.2.

5. Analysis

We perform an ablation study on our added regularization parameters and auxiliary loss, and analyze the pretraining scheme on mini-ImageNet. Finally, we verify qualitatively that the latent space reconstructions learned by our classifier are superior for images of the same class, and inferior for images of a different class.

5.1. Ablation Study

We perform our ablation study on CUB, using both Conv-4 and ResNet-12. Results are given in Table 6 for the cropped data setting. In this case, 1-shot results come from models trained in a 1-shot manner. We find that the impact of the auxiliary loss is mixed: it helps in the 1-shot setting, but hurts 5-shot performance slightly. We suspect that this is because the pools of support features in the 1-shot setting are information-deficient, and so explicitly encouraging full utilization of the feature space is helpful. The support feature pools in the 5-shot setting are not so deficient, and so do not benefit from the auxiliary loss.

Model	Backbone	1-shot	5-shot
Meta-Baseline$^{\flat}$ [4]	ResNet-12	63.17±0.23	79.26± 0.17
MetaOptNet$^{\sharp}$ [11]	ResNet-12	62.64±0.61	78.63±0.46
DSN‡ [19]	ResNet-12	62.64±0.66	78.83±0.45
CAN$^{\heartsuit\flat}$ [9]	ResNet-12	63.85±0.48	79.44±0.34
MatchNet$^{\heartsuit\flat}$ [25, 30]	ResNet-12	**65.64±0.20**	78.72±0.15
ProtoNet$^{\flat}$ [20, 30]	ResNet-12	62.39±0.21	80.53±0.14
SimpleShot$^{\flat}$ [27]	ResNet-18	62.85±0.20	80.02±0.14
Afrasiyabi *et al.*$^{\heartsuit\diamondsuit\flat}$ [1]	ResNet-18	59.88±0.67	80.35±0.73
Neg-Cosine$^{\diamondsuit\flat}$ [12]	WRN-28-10	61.72±0.81	81.79±0.55
E^{3}BM$^{\heartsuit\diamondsuit\flat}$ [13]	ResNet-25	64.3	81.0
FEAT$^{\heartsuit\flat}$ [30]	ResNet-12	**66.78±0.20**	82.05±0.14
DeepEMD$^{\diamondsuit\flat}$ [31]	ResNet-12	**65.91±0.82**	**82.41±0.56**
RFS-Distill$^{\ddagger\sharp\flat}$ [23]	ResNet-12	64.82±0.60	**82.14±0.43**
FRN (ours) 1-shot$^{\flat}$	ResNet-12	**65.33±0.20**	-
FRN (ours) 5-shot$^{\flat}$	ResNet-12	**66.45±0.19**	**82.83±0.13**

Table 5. Performance of selected competitive few-shot models on mini-ImageNet. ‡ denotes use of data augmentation during evaluation. ♯ denotes use of label smoothing or knowledge distillation. ♡ denotes modules with many additional learnable parameters. ♢ denotes use of SGD during evaluation. ♭ denotes non-episodic pre-training or classifier losses. Bold numbers denote 1-shot accuracy over 65 or 5-shot over 82. FRN numbers are averaged over 10,000 episodes with 95% confidence intervals.

	Conv-4		ResNet-12	
Model	1-shot	5-shot	1-shot	5-shot
no Aux	67.33	**88.36**	82.20	92.65
no λ	66.75	83.92	82.40	93.12
no ρ	67.34	87.52	82.58	92.79
no λ, ρ	66.00	83.41	**82.80**	**93.33**
whole model	**68.76**	87.87	82.62	92.59

Table 6. Ablation study on regularization parameters and auxiliary loss. FRN models are trained under different ablation settings on cropped CUB.

We analyze the contribution of the learned regularization terms λ and ρ by disabling their learnability, setting one or both equal to one over the course of training. The impact of these terms is also mixed. The 4-layer network clearly benefits from learning these terms, but the ResNet-12 architecture benefits from removing them. It seems that a more powerful network is able to overcome regularization problems by massaging the feature space in a more elegant way than the individual λ, ρ terms can provide.

Overall, FRN is not particularly sensitive to these components of the training scheme.

5.2. Pre-Training on mini-ImageNet

We find that pre-training is crucial for competitive mini-ImageNet performance, especially when compared to base-

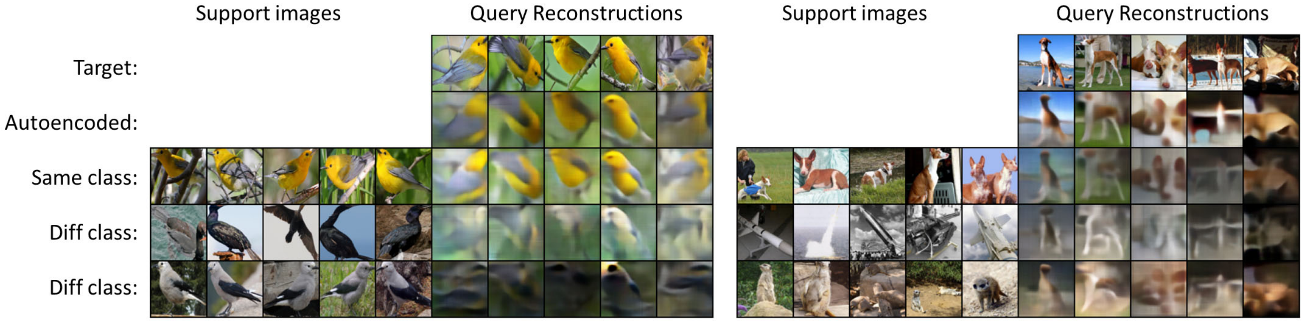

Figure 3. Decoder outputs on CUB (left) and mini-ImageNet (right). Images are regenerated from ground-truth feature maps (row 2), and reconstructions from same-class (row 3) and different-class support images (rows 4, 5). The same-class reconstructions are clearly more faithful to the original. Best viewed digitally.

training setting	1-shot	5-shot
from scratch 1-shot	61.75±0.20	-
from scratch 5-shot	63.03±0.20	78.01±0.15
after pre-train	60.97±0.21	75.11±0.18
finetune 1-shot	65.33±0.20	-
finetune 5-shot	**66.45±0.19**	**82.83±0.13**

Table 7. Impact of pre-training on mini-ImageNet. Both pre-training and episodic finetuning are important.

Input	CUB	mini-IN
ground-truth feature map	.208	.177
same-class reconstruction	.343	.307
diff-class reconstruction	.385	.337

Table 8. L2 pixel error between original images and regenerated ones from different latent inputs on CUB and mini-ImageNet validation sets. Results are averaged over 1,000 trials and 95% confidence intervals are below 1e-3.

lines also utilizing pre-training. However, pre-training alone is not sufficient to produce a competitive classifier. An FRN trained from scratch outperforms a pre-trained FRN evaluated naively (Table 7). The two-round process of pre-training followed by episodic fine-tuning appears to be crucial. This finding is in line with prior work [30, 4].

5.3. Reconstruction Visualization

While our accuracy scores suggest that FRN learns to produce better reconstructions from same-class support images than from those of a different class, we would still like to confirm this intuition visually. In particular, it is not obvious that a better reconstruction in FRN latent space should also be a better reconstruction semantically. To verify this we train an image re-generator for the 5-shot ResNet-12 FRN on CUB and mini-ImageNet. Using an inverted ResNet-12 architecture, this decoder network is trained to take the feature maps of the FRN and map them back to the original corresponding image. Training details for the decoder can be found in supplementary. Results are reported on validation images from each dataset.

If it is the case that same-class feature map reconstructions are more semantically faithful than different-class ones, we should be able to observe a corresponding difference in image quality when we pass each feature map reconstruction through the decoder. Table 8 shows the pixel error of regenerated images from the 5-shot reconstructed feature maps relative to the ground-truth feature map. Our intuition holds: while both reconstructed feature maps produce jumps in pixel error relative to the ground truth, the increase is smaller when the feature map is reconstructed from images of the same class.

Sample outputs from the decoder for CUB and mini-ImageNet can be found in Fig. 3. The reconstructions from ground-truth feature maps are not particularly good, as classifier embeddings are designed to cluster same-class images tightly and discard all other details. Nevertheless, visual quality is high enough that the difference between regenerated same-class reconstructions (row 3) and different-class reconstructions (rows 4, 5) is readily apparent. Additional visualizations are provided in supplementary. We conclude that FRN is doing what we intend: learning reconstructions that are semantically faithful for same-class support images and less faithful otherwise.

6. Conclusion

We introduce Feature Map Reconstruction Networks, a novel approach to few-shot classification based on reconstructing query features in latent space. Solving the reconstruction problem in closed form produces a classifier that is both straightforward and powerful, incorporating fine spatial details without overfitting to position or pose. We demonstrate state-of-the-art performance on four fine-grained few-shot classification benchmarks, and competitive performance in the general setting.

Acknowledgements

This work was funded by the DARPA Learning with Less Labels program (HR001118S0044). The authors would like to thank Cheng Perng Phoo for valuable suggestions and computing cluster assistance.

References

[1] Arman Afrasiyabi, Jean-François Lalonde, and Christian Gagné. Associative alignment for few-shot image classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, August 2020. 6, 7

[2] Luca Bertinetto, Joao F. Henriques, Philip Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. In *International Conference on Learning Representations*, 2019. 3, 4

[3] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Wang, and Jia-Bin Huang. A closer look at few-shot classification. In *International Conference on Learning Representations*, 2019. 2, 5, 6, 12, 13

[4] Yinbo Chen, Xiaolong Wang, Zhuang Liu, Huijuan Xu, and Trevor Darrell. A new meta-baseline for few-shot learning, 2020. 1, 2, 3, 4, 5, 7, 8

[5] Carl Doersch, Ankush Gupta, and Andrew Zisserman. Crosstransformers: spatially-aware few-shot transfer. *Advances in Neural Information Processing Systems*, 33, 2020. 2, 4, 5, 6, 7

[6] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017. 2, 6

[7] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4367–4375, 2018. 1, 2, 3, 4, 11

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

[9] Ruibing Hou, Hong Chang, MA Bingpeng, Shiguang Shan, and Xilin Chen. Cross attention network for few-shot classification. In *Advances in Neural Information Processing Systems*, pages 4003–4014, 2019. 2, 7

[10] Valentin Khrulkov, Leyla Mirvakhabova, Evgeniya Ustinova, Ivan Oseledets, and Victor Lempitsky. Hyperbolic image embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6418–6428, 2020. 1, 2, 6

[11] Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10657–10665, 2019. 2, 5, 7, 11

[12] Bin Liu, Yue Cao, Yutong Lin, Qi Li, Zheng Zhang, Mingsheng Long, and Han Hu. Negative margin matters: Understanding margin in few-shot classification. *arXiv preprint arXiv:2003.12060*, 2020. 6, 7, 12, 13

[13] Yaoyao Liu, Bernt Schiele, and Qianru Sun. An ensemble of epoch-wise empirical bayes for few-shot learning. In *European Conference on Computer Vision (ECCV)*, 2020. 7

[14] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. Technical report, 2013. 2, 6

[15] Puneet Mangla, Nupur Kumari, Abhishek Sinha, Mayank Singh, Balaji Krishnamurthy, and Vineeth N Balasubramanian. Charting the right manifold: Manifold mixup for few-shot learning. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 2218–2227, 2020. 6, 12, 13

[16] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018. 2

[17] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. 2016. 7, 12

[18] Andrei A Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. Meta-learning with latent embedding optimization. *arXiv preprint arXiv:1807.05960*, 2018. 2

[19] Christian Simon, Piotr Koniusz, Richard Nock, and Mehrtash Harandi. Adaptive subspaces for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4136–4145, 2020. 1, 3, 4, 5, 6, 7

[20] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pages 4077–4087, 2017. 1, 2, 4, 6, 7, 13

[21] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1199–1208, 2018. 1, 2, 3, 6

[22] Luming Tang, Davis Wertheimer, and Bharath Hariharan. Revisiting pose-normalization for fine-grained few-shot recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14352–14361, 2020. 6, 12, 13

[23] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need? *arXiv preprint arXiv:2003.11539*, 2020. 2, 5, 7, 11

[24] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8769–8778, 2018. 6

[25] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638, 2016. 2, 6, 7

[26] 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. 2, 6

[27] Yan Wang, Wei-Lun Chao, Kilian Q. Weinberger, and Laurens van der Maaten. Simpleshot: Revisiting nearest-neighbor classification for few-shot learning. *arXiv preprint arXiv:1911.04623*, 2019. 2, 5, 6, 7

[28] Yikai Wang, Chengming Xu, Chen Liu, Li Zhang, and Yanwei Fu. Instance credibility inference for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12836–12845, 2020. 6

[29] Davis Wertheimer and Bharath Hariharan. Few-shot learning with localization in realistic settings. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6558–6567, 2019. 2, 6, 7, 12

[30] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Few-shot learning via embedding adaptation with set-to-set functions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8808–8817, 2020. 2, 3, 4, 5, 6, 7, 8, 11

[31] Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. Deepemd: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2, 5, 6, 7, 11

[32] Imtiaz Masud Ziko, Jose Dolz, Eric Granger, and Ismail Ben Ayed. Laplacian regularized few-shot learning. *arXiv preprint arXiv:2006.15486*, 2020. 6, 12, 13