Learning to Learn from Corrupted Data for Few-Shot Learning

Yuexuan An[1,2] , **Xingyu Zhao**[1,2] , **Hui Xue**[1,2*]

[1]School of Computer Science and Engineering, Southeast University, Nanjing, 210096, China
[2]MOE Key Laboratory of Computer Network and Information Integration (Southeast University), China
{yx_an, xyzhao, hxue}@seu.edu.cn,

Abstract

Few-shot learning which aims to generalize knowledge learned from annotated base training data to recognize unseen novel classes has attracted considerable attention. Existing few-shot methods rely on completely clean training data. However, in the real world, the training data are always corrupted and accompanied by noise due to the disturbance in data transmission and low-quality annotation, which severely degrades the performance and generalization capability of few-shot models. To address the problem, we propose a unified peer-collaboration learning (PCL) framework to extract valid knowledge from corrupted data for few-shot learning. PCL leverages two modules to mimic the peer collaboration process which cooperatively evaluates the importance of each sample. Specifically, each module first estimates the importance weights of different samples by encoding the information provided by the other module from both global and local perspectives. Then, both modules leverage the obtained importance weights to guide the reevaluation of the loss value of each sample. In this way, the peers can mutually absorb knowledge to improve the robustness of few-shot models. Experiments verify that our framework combined with different few-shot methods can significantly improve the performance and robustness of original models.

1 Introduction

Few-shot Learning (FSL) aims to mimic humans to learn a new concept with limited examples [Chen *et al.*, 2019; Chen *et al.*, 2021]. In order to obtain the human-like capability, few-shot learner first learns profound knowledge from base training dataset and generalizes the knowledge to recognize novel unseen classes with a few examples [An *et al.*, 2021; Jian and Torresani, 2022]. Since few-shot learning was proposed, it has been widely studied and applied in various fields including medical analysis [Wang *et al.*, 2022a], drug discovery [Zeng *et al.*, 2022], remote sensing [Zhang *et al.*, 2023], robotics [Li *et al.*, 2022], etc.

The existing few-shot methods generally assume that base training data are fully clean and intact without considering possible corruption in training data [Wang *et al.*, 2021; Ma *et al.*, 2022a]. However, such an assumption is hard to meet in real application scenarios. In the real world, due to low-quality annotations with ambiguous images [Xue *et al.*, 2019; Tsipras *et al.*, 2020] and accidental channel disturbances in online transmission [Shi *et al.*, 2010], data corruption inevitably exists in the training data. When trained on these data, these methods easily overfit the corrupted data, which degrades the performance of original models [Li *et al.*, 2019]. Therefore, how to identify the valid information from these corrupted data is of great significance for few-shot learning.

Many studies are devoted to reducing the disturbances of these corrupted data. One line is using noise transition matrix that models the relationship between latent clean labels and observed noisy labels by a transition matrix to recover the latent correct label [Bekker and Goldberger, 2016; Zhu *et al.*, 2022; Cheng *et al.*, 2022]. The other line is sample selection that evaluates each sample with noise estimator [Jiang *et al.*, 2018; Han *et al.*, 2018; Wang *et al.*, 2022b; Xia *et al.*, 2022]. However, these methods are limited to noise identification without considering the model generalization, which violates the original intention of few-shot learning. In the meantime, some recent studies are devoted to improving the robustness of few-shot models [Lu *et al.*, 2021; Mazumder *et al.*, 2021; Liang *et al.*, 2022; An *et al.*, 2023]. However, they still rely on a clean base training dataset and artificially introduce known pseudo-noise data during the training process, which is far from mirroring the real world.

To address the above problems, we propose peer-collaboration learning (PCL) to learn from corrupted base class data for few-shot learning. Rather than designing a specific method, in this paper, we propose a model-agnostic PCL framework that can be applied to different few-shot methods to alleviate the impact of data corruption in base training dataset. Our work is inspired by the peer collaboration of human behavior that students in pairs cooperatively scrutinize the quality of each one's work and co-construct solutions to dilemmas which increases the cognition of both peers and makes the evaluation more reliable [Ceci and Peters, 1982; Fawcett and Garton, 2005; Van Meter and Stevens, 2000;

*Corresponding author

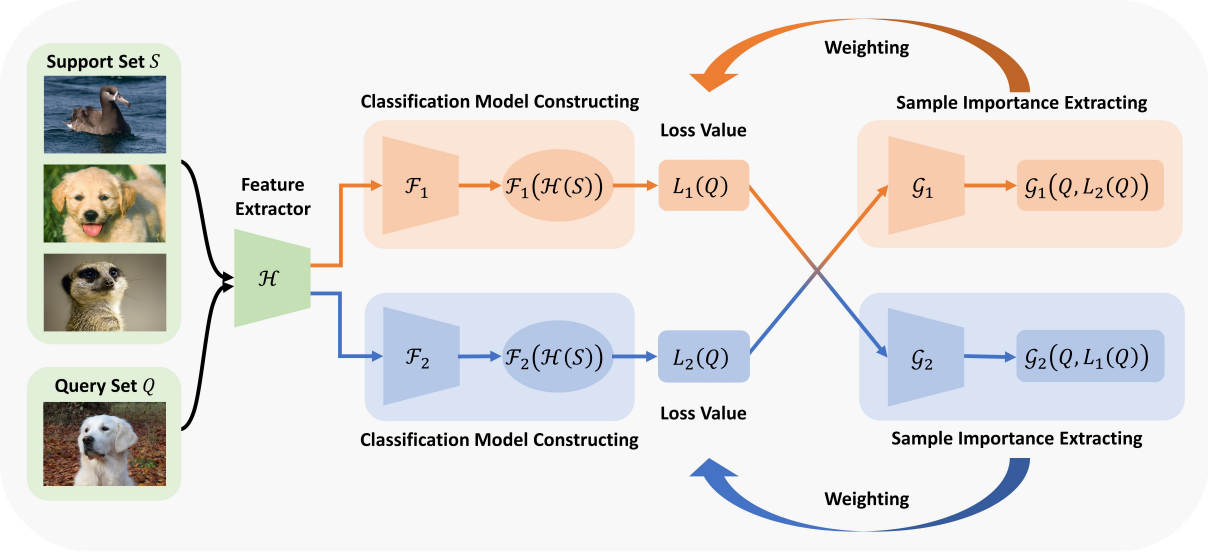

Figure 1: Architecture of the Peer-Collaboration Learning Framework.

Kuhn, 2015]. To mimic the peer collaboration process, PCL leverages two interactive modules to learn from different perspectives and collaboratively evaluate the corrupted data so as to enhance the reliability of few-shot models. The two modules share a common feature extractor, and have individual groups of classification model constructor and sample importance extractor. During the training stage, each module obtains features and loss values of the training samples in its corresponding latent spaces with its classification model contractor, and conveys the training information to the other. Then, each module leverages the sample importance extractor to extract the sample importance weights by encoding the training information provided by the other module from both global and local aspects. Subsequently, the loss values of samples can be reevaluated with the obtained sample importance weights and utilized for the training process of each module. In this way, few-shot models can automatically and efficiently identify the importance of different samples and mitigate the impact of data corruption. In the testing stage, PCL synthesizes the outputs of the two modules and obtains the final prediction.

Our key main contributions can be summarized as follows:

- We present a unified peer-collaboration learning (PCL) framework for few-shot learning with corrupted data. PCL is a model-agnostic framework that can be plugged into various few-shot methods to improve the robustness of these models.
- We design peer-collaboration learning inspired by the peer collaboration of human behavior. PCL leverages two modules to interactively learn from different perspectives and collaboratively evaluate different types of corrupted data.
- We design a novel peer-collaboration mechanism in PCL. This mechanism extracts the sample importance information from both global and local aspects by encoding the feature representations and the corresponding loss values of the training samples.
- We verify the superiority and robustness of PCL through experiments. The experimental results verify that PCL can significantly improve the performance of few-shot methods with various corrupted data.

2 Related Work

Few-Shot Learning. In general, the few-shot method [Ye *et al.*, 2020; Yu *et al.*, 2022] can be divided into two types: optimization-based method and metric-based method. The optimization-based method is to find a single set of model parameters that can be adapted with a few steps of gradient descent to individual tasks [Finn *et al.*, 2017; Oh *et al.*, 2021]. The idea of the metric-based method is to leverage similarity information between images to classify novel classes with few examples [Kang *et al.*, 2021; An *et al.*, 2023; Ma *et al.*, 2022b]. MatchingNet [Vinyals *et al.*, 2016] uses the attention mechanism and memory to acquire the ability to quickly learn new concepts. ProtoNet [Snell *et al.*, 2017] utilizes the idea of K-means to take the average embeddings of samples with the same class as the prototype of the class. DeepEMD [Zhang *et al.*, 2020] leverages the set of local descriptors as the representation for an image adopts the Earth Move's Distance to calculate the distance between representations of two images. However, due to ubiquitous data corruption, these methods might easily overfit the noise, which inevitably causes the severe degradation of performances.

Noise. Noise is an important problem in machine learning. One promising way is the sample selection. Decoupling [Malach and Shalev-Shwartz, 2017] trains two networks concurrently, and then updates models only using the instances that have different predictions from these two networks. Co-teaching [Han *et al.*, 2018] trains two networks, the clean samples selected by each network are used to train the other. Meta-Weight-Net adaptively learns an explicit

weighting function from data [Shu *et al.*, 2019]. CNLCU [Xia *et al.*, 2022] adopts interval estimation instead of point estimation of losses to select samples. Our framework combines the advantages of Co-training in [Jiang *et al.*, 2018; Han *et al.*, 2018], incomplete-dependence of small loss trick in [Xia *et al.*, 2022] and dynamic loss weighting in [Shu *et al.*, 2019] while replacing discarding samples with weighting functions and encoding the local and global information to the improve the evaluation of corrupted data for few-shot models.

Few-Shot Learning with Noise. There are only a few works that focus on few-shot learning with noise. [Lu *et al.*, 2021] proposes a BiLSTM-based attentive module to weigh support samples and form a robust prototype for each class. [Mazumder *et al.*, 2021] combines data augmentation with soft k-means clustering to refine class prototypes. [Liang *et al.*, 2022] develops a transformer-based attentive module to aggregate support samples into class representations. It is worth noting that these methods require an auxiliary clean set, which may not be available in real-world scenarios.

3 Methodology

3.1 Preliminary

Few-Shot Learning. We define the problem to be solved in this paper as follows. The training dataset and testing dataset are represented by $\mathcal{D}_{tr}$ and $\mathcal{D}_{te}$ respectively. In the training stage, the training dataset $\mathcal{D}_{tr} = \{\mathcal{T}_i\}_{i=1}^{I}$ is defined as a set of tasks which are sampled from base classes $\mathcal{C}_b$. Each task $\mathcal{T}_i$ contains a support set S_i and a query set Q_i. The support set S_i is composed of $N \times K$ samples that K samples for each N classes randomly selected from all classes of $\mathcal{D}_{tr}$. The query set Q_i is composed of $N \times M$ samples that have the same classes as S_i but is disjoint from S_i. The model is trained by randomly selecting tasks from the task set $\{\mathcal{T}_i\}_{i=1}^{I}$. In the testing stage, the testing dataset $\mathcal{D}_{te} = \{\mathcal{T}_i^*\}_{i=1}^{I^*}$ is samples from novel class $\mathcal{C}_n$ ($\mathcal{C}_n \bigcap \mathcal{C}_b = \emptyset$), from which a new task $\mathcal{T}_i^* = \{S_i^*, Q_i^*\}$ is sampled. The goal of each few-shot task is to estimate the class of samples in the query set Q_i^* with support set S_i^*.

Data Corruption. In this paper, we mainly consider the following two data corruption: (i) *Sample corruption*. In the training stage, feature noise might exist in $\mathcal{D}_{tr}$. We consider feature noise as an image of random pixels with uniform distribution [Boncelet, 2009] to model sample corruption in channel impairments. (ii) *Label corruption*. In the training stage, label noise might exist in $\mathcal{D}_{tr}$. We consider label noise as a mislabeled image to model label corruption in low-quality annotations.

3.2 Peer-Collaboration Learning Framework

The peer-collaboration learning (PCL) framework aims to alleviate the impact of the noise that may exist in base training data on few-shot models. Inspired by the peer collaboration of human behavior which increases the cognition for both peers and makes the evaluation more reliable [Ceci and Peters, 1982; Fawcett and Garton, 2005; Van Meter and Stevens, 2000; Kuhn, 2015], PCL leverages two modules to collaboratively learn from different perspectives to increase the capability to discriminate the noise. Specifically, the two modules share a common feature extractor, and have individual groups of classification model constructor and sample importance extractor. Through a novel peer-collaboration mechanism, each module extracts the importance information of different samples generated by the other from both global and local aspects. Then, each module adopts the importance information of different samples to guide its own update. In this way, few-shot models can automatically identify the importance of different samples and filter the noise.

As illustrated in Figure 1, PCL trains two different modules simultaneously. The two modules shared a common feature extractor $\mathcal{H}$ and each module has a dividual classification model constructor $\mathcal{F}$ and a sample importance extractor $\mathcal{G}$. We denote the two modules as f_1 and f_2. When processing task $\mathcal{T} = \{S, Q\}$, f_1 and f_2 first construct the classification models $\mathcal{F}_1(\mathcal{H}(S))$ and $\mathcal{F}_2(\mathcal{H}(S))$ according to the support set S by classification model constructor $\mathcal{F}_1$ and $\mathcal{F}_2$, and then calculate the loss values $L_1(Q)$ (corresponding to $\mathcal{F}_1(\mathcal{H}(S))$) and $L_2(Q)$ (corresponding to $\mathcal{F}_2(\mathcal{H}(S))$) relative to the query set Q. Then, the sample information and loss information of both modules are conveyed to each other. Subsequently, the importance values of different samples are calculated by the sample importance extractor $\mathcal{G}_1$ and $\mathcal{G}_2$. Details of the peer-collaboration mechanism are given in the next section. Then the reevaluated loss values can be obtained:

$$\mathcal{L}_1 = -\frac{1}{M}\sum_{m=1}^{M}\left(\mathcal{G}_1(Q, L_2(Q))_m \cdot L_1(q_m)\right), \quad (1)$$

$$\mathcal{L}_2 = -\frac{1}{M}\sum_{m=1}^{M}\left(\mathcal{G}_2(Q, L_1(Q))_m \cdot L_2(q_m)\right), \quad (2)$$

where q_m is the m-th sample in the query set Q.

In PCL, two models are individual equally without bias and simultaneously trained with the same data. In the testing stage, PCL synthesizes the outputs of the two models $\mathcal{F}_1(\mathcal{H}(S))$ and $\mathcal{F}_2(\mathcal{H}(S))$ to obtain the final prediction. Given q^* as the query sample to be predicted, PCL adopts the simple but effective strategy (mean between two models) to yield more robust results inspired by ensemble learning

$$Pred_{q^*} = \frac{1}{2}[\texttt{Softmax}(\mathcal{F}_1(\mathcal{H}(S))(q^*)/\tau) + \texttt{Softmax}(\mathcal{F}_2(\mathcal{H}(S))(q^*)/\tau)], \quad (3)$$

where $\mathcal{F}_1(\mathcal{H}(S))(q^*)$ and $\mathcal{F}_2(\mathcal{H}(S))(q^*)$ are the output logits of classification models, τ is the temperature parameter for better integrating the predicted values of the two modules.

3.3 The Peer-Collaboration Mechanism

PCL uses a novel peer-collaboration mechanism to automatically identify the contributions of different samples for model parameter updating. Generally, different classifiers can generate different decision boundaries. The purpose of PCL is to take advantage of the collaboration of different modules to evaluate the importance of each sample and make the evaluation more reliable.

Classification Model Constructing

The goal of classification model constructor is to map the feature extracted by $\mathcal{H}$ into different spaces and construct classification models in these spaces. Inspired by [Perez *et al.*, 2018] to improve the generalization of FSL models in different spaces, we leverage the mapping function $\hat{h}$ to map the feature:

$$\hat{h}(S) = \mathcal{H}(S) + \delta_1 \left(\texttt{ReLU} \left(\delta_2 \left(h(S) \right) \odot \mathcal{H}(S) + \delta_3 \left(h(S) \right) \right) \right), \tag{4}$$

where $\hat{h}(S)$ is the mapped feature, $\odot$ indicates the hadamard product, $\texttt{ReLU}$ denotes the ReLU function [Glorot *et al.*, 2011], $h(S) = \texttt{BN} \left(\delta_4 \left(\texttt{ReLU} \left(\mathcal{H}(S) \right) \right) \right)$ and $\texttt{BN}$ is the batch normalization operation [Ioffe and Szegedy, 2015], δ_1, δ_2, δ_3 and δ_4 are different linear mappings. After obtaining the mapped features of different spaces, PCL can construct the classification models $\mathcal{F}_1 \left(\mathcal{H}(S) \right)$ and $\mathcal{F}_2 \left(\mathcal{H}(S) \right)$ according to the specific few-shot methods. For query samples Q, the feature is first mapped into the same space as the classification model:

$$\hat{h}(Q) = \mathcal{H}(Q) + \delta_1 \left(\texttt{ReLU} \left(\delta_2 \left(h(Q) \right) \odot \mathcal{H}(Q) + \delta_3 \left(h(Q) \right) \right) \right), \tag{5}$$

where $h(Q) = \texttt{BN} \left(\delta_4 \left(\texttt{ReLU} \left(\mathcal{H}(Q) \right) \right) \right)$. Then the loss value of Q can be obtained according to the loss function of the specific few-shot models.

Sample Importance Extracting

Since the type of data corruption is unknown, we consider two aspects of importance weighting, i.e., global weighting and local weighting. Concretely, global weighting leverages the corresponding loss of all query samples to encode their relative relationship, and then uses the encoding values to weight these samples. By comparing the relative relationship of the query samples from the global perspective, the impact of potential noise could be effectively eliminated. Local weighting encodes the query samples that may contain noise within each class. For a specific class, through obtaining and leveraging the relationship of potential samples with the same class, the impact of confusing samples can be effectively degraded. We take $\mathcal{G}_1$ as an example to introduce the two weighting approaches in detail. The calculation process of $\mathcal{G}_2$ is similar to $\mathcal{G}_1$.

Global Weighting. First, PCL calculates the loss value $L_2(Q)$ of each sample in the query set Q for model $\mathcal{F}_2 \left(\mathcal{H}(S) \right)$, and then sorts them to obtain the ranking of different samples. Then, PCL leverages the ranking information to reform the query samples as a sequence of ordered samples $\hat{Q}$. Subsequently, PCL adopts a linear mapping to transform the samples to the latent embedding values, and then uses BiLSTM parameterized by ϕ_B to encode the embedding values of the query samples to leverage the content-based attention capability of BiLSTM to weight different query samples. Specifically, The forward pass for encoding the sequence of ordered samples $\hat{Q}$ can be given by

$$d_j, e_j = \texttt{BiLSTM} \left(\hat{Q}_j, d_{j-1}, e_{j-1} \,|\phi_B \right), \tag{6}$$

where d and e denote the hidden and cell state vectors inside BiLSTM respectively and $\hat{Q}_j$ is the j-th feature vector in $\hat{Q}$. Similarly, we can attain the backward pass l_j, o_j. We use a multi-layer neural network followed by a softmax layer denoted by ψ_1 to achieve the global weighting:

$$W^{(g)} = \psi_1 \left(\mathcal{C} \left(\mathcal{C} \left(d_1, l_1 \right), \cdots, \mathcal{C} \left(d_{NM}, l_{NM} \right) \right) \right), \tag{7}$$

where $\mathcal{C}$ denotes the concatenation operation and NM are $N \times M$ query samples in $\hat{Q}$.

Local Weighting. The core idea of local weighting is to leverage the intra-class correlation of query samples to identify possible noises. For the N-way task, for each sample, we construct a local correlation vector composed of all correlation scores between a sample and its M same labeled query samples, which represents the feature correlation between the sample and its corresponding class. Then, for each class, the correlation matrix can be obtained. Take class n for example, the correlation matrix $P^{(n)} \in \mathbb{R}^{M \times M}$ reveals the whole correlation between samples labeled with class n and the class n. Specifically, cosine similarity of paired features obtained from the other module is calculated as each corresponding element in P_n:

$$P_{i,j}^{(n)} = \texttt{cos} \left(\hat{h} \left(q_i^{(n)} \right), \hat{h} \left(q_j^{(n)} \right) \right), \tag{8}$$

where $P_{i,j}^{(n)}$ denotes the correlation between the mapped features $\hat{h} \left(q_i^{(n)} \right)$ and $\hat{h} \left(q_j^{(n)} \right)$ corresponding to the i-th and j-th query samples labeled with n.

The self-attention mechanism of transformer is an effective way to leverage the similarities between different samples. Therefore, we use the transformer layer [Vaswani *et al.*, 2017] to encode the correlations among samples. Since sample order in a class is arbitrary, we direct input the correlation matrix $P^{(n)} \in \mathbb{R}^{M \times M}$ into the transformer layer without positional encoding:

$$O_n = \texttt{Transformer} \left(P^{(n)} \,|\phi_T \right). \tag{9}$$

where ϕ_T is the parameters of the transformer layer. Then a multi-layer neural network followed by a M-way softmax layer is used to achieve the local weight vector $W^{(l)}$ for samples labeled with n:

$$W^{(l)} = \psi_2 \left(O_n \right). \tag{10}$$

where ψ_2 is the combination of the multi-layer neural networks and the softmax layer.

After obtaining both global weighting and local weighting for the query set Q, the peer-collaboration weight can be obtained by

$$\mathcal{G}_1(Q, L_2 \left(Q \right)) = W^{(g)} + W^{(l)}. \tag{11}$$

4 Experiments

In this section, the effectiveness of PCL is verified by various experiments. Four typical few-shot methods including MatchingNet [Vinyals *et al.*, 2016], ProtoNet [Snell *et al.*, 2017], S2M2 [Mangla *et al.*, 2020] and DeepEMD [Zhang *et al.*, 2020] are selected to compare with our PCL-based versions. Four methods that deal with noise with two interactive

Algorithm	CIFAR-FS	CUB	*mini*-ImageNet	*tiered*-Imagenet
MatchingNet	68.93±0.74	74.88±0.66	62.75±0.75	60.74±0.81
PCL-MatchingNet (Ours)	70.63±0.76 (**+1.70**)	78.73±0.69 (**+3.85**)	63.78±0.68 (**+1.03**)	62.98±0.77 (**+2.24**)
ProtoNet	68.78±0.77	74.68±0.70	64.69±0.69	61.50±0.74
PCL-ProtoNet (Ours)	74.47±0.73 (**+5.69**)	79.35±0.68 (**+4.67**)	66.83±0.74 (**+2.14**)	63.80±0.77 (**+2.30**)
S2M2	71.01±0.74	78.06±0.68	64.70±0.66	63.53±0.76
PCL-S2M2 (Ours)	75.69±0.74 (**+4.68**)	81.46±0.66 (**+3.40**)	66.82±0.71 (**+2.12**)	66.82±0.74 (**+3.29**)
DeepEMD	73.02±0.74	79.07±0.64	68.88±0.65	66.66±0.78
PCL-DeepEMD (Ours)	77.69±0.67 (**+4.67**)	84.13±0.54 (**+5.06**)	70.31±0.69 (**+1.43**)	69.94±0.81 (**+3.28**)

Table 1: Results on few-shot learning tasks with clean samples (Acc. %). All accuracy results are averaged over 600 test episodes with 95% confidence intervals. The relative accuracy gains with the introduction of PCL are highlighted in bold.

Noise rate	Algorithm	CIFAR-FS	CUB	*mini*-ImageNet	*tiered*-Imagenet
20%	MatchingNet	61.89±0.80	71.51±0.76	58.99±0.71	57.86±0.74
	PCL-MatchingNet (Ours)	66.29±0.83 (**+4.40**)	77.25±0.70 (**+5.74**)	61.47±0.67 (**+2.48**)	59.45±0.72 (**+1.59**)
	ProtoNet	64.21±0.83	70.44±0.70	60.70±0.74	60.07±0.86
	PCL-ProtoNet (Ours)	69.42±0.79 (**+5.21**)	77.69±0.69 (**+7.25**)	64.73±0.68 (**+4.03**)	63.32±0.73 (**+3.25**)
	S2M2	69.10±0.77	77.11±0.69	62.47±0.70	61.90±0.78
	PCL-S2M2 (Ours)	75.54±0.74 (**+6.44**)	78.92±0.64 (**+1.81**)	65.45±0.70 (**+2.98**)	63.75±0.76 (**+1.85**)
	DeepEMD	71.94±0.72	78.87±0.64	67.45±0.68	63.78±0.77
	PCL-DeepEMD (Ours)	75.01±0.74 (**+3.07**)	80.73±0.62 (**+1.86**)	69.25±0.68 (**+1.80**)	68.93±0.82 (**+5.15**)
40%	MatchingNet	58.39±0.77	64.77±0.78	55.82±0.70	53.79±0.74
	PCL-MatchingNet (Ours)	60.66±0.78 (**+2.27**)	69.82±0.74 (**+5.05**)	57.96±0.66 (**+2.14**)	55.59±0.78 (**+1.80**)
	ProtoNet	60.88±0.82	66.26±0.71	56.00±0.71	53.97±0.76
	PCL-ProtoNet (Ours)	62.98±0.85 (**+2.10**)	72.58±0.72 (**+6.32**)	61.32±0.70 (**+5.32**)	58.45±0.76 (**+4.48**)
	S2M2	67.41±0.79	74.41±0.70	61.63±0.71	60.89±0.79
	PCL-S2M2 (Ours)	69.07±0.75 (**+1.66**)	78.67±0.66 (**+4.26**)	62.74±0.69 (**+1.11**)	63.59±0.76 (**+2.70**)
	DeepEMD	71.67±0.71	77.98±0.62	66.37±0.66	62.55±0.76
	PCL-DeepEMD (Ours)	74.96±0.73 (**+3.29**)	80.41±0.62 (**+2.43**)	67.95±0.67 (**+1.58**)	66.52±0.80 (**+3.97**)

Table 2: Results on few-shot learning tasks with 20% & 40% feature noise (Acc. %). All accuracy results are averaged over 600 test episodes with 95% confidence intervals. The relative accuracy gains with the introduction of PCL are highlighted in bold.

modules including Decoupling [Malach and Shalev-Shwartz, 2017], Co-teaching [Han *et al.*, 2018], Co-teaching+ [Yu *et al.*, 2019] and JoCoR [Wei *et al.*, 2020] are selected to verify the performance of PCL in dealing with noisy problems. Four standard few-shot learning datasets, CIFAR-FS [Bertinetto *et al.*, 2019], CUB [Wah *et al.*, 2011], *mini*-ImageNet [Vinyals *et al.*, 2016] and *tiered*-ImageNet [Ren *et al.*, 2018] are selected to test the performance of the compared methods. All the computations are performed on a GPU server with NVIDIA Tesla V100, Intel Xeon Gold 6240 CPU 2.60 GHz processor, and 32 GB GPU memory. Our code is available at `https://github.com/anyuexuan/PCL`.

4.1 Network Structure and Parameters

For fair comparisons, we implement all methods by PyTorch. All algorithms use the four-block-based ConvNet model (C64E) [Chen *et al.*, 2019] as the backbone and the feature embedding dimension is set to 1600. In C64E, each block is comprised of 64-channel 3×3 convolution, batch normalization [Ioffe and Szegedy, 2015], ReLU nonlinearity [Glorot *et al.*, 2011], and 2×2 max-pooling. The feature embedding dimension is set to 1600. The Adam optimizer [Kingma and Ba, 2015] is used by all methods to optimize parameters. The learning rate is set to 10^{-3} for all algorithms. The temperature parameter τ in Eq.(3) is set to 2 for PCL-MatchingNet, 64 for PCL-ProtoNet, 0.05 for PCL-S2M2 and 0.05 for PCL-DeepEMD, respectively. The maximum number of training episodes is set to 40000. Moreover, to prevent the model from incorrectly ignoring the clean samples in early training, we add a small constant which is set to 0.2 to the peer-collaboration weight during the first 20000 episodes.

4.2 Comparison with Clean Data

To verify the versatility of PCL, we compare the performance of different methods on few-shot learning tasks with clean samples. We compare vanilla MatchingNet, ProtoNet, S2M2 and DeepEMD with our corresponding PCL-based methods. Table 1 gives performance comparisons under the 5-way 5-shot setting. For each comparison, the relative accuracy gains with the introduction of PCL are highlighted in bold. We can find that, for all original few-shot methods, the introduction of PCL can ameliorate the model performance obviously. In

Noise rate	Algorithm	CIFAR-FS	CUB	*mini*-ImageNet	*tiered*-Imagenet
20%	MatchingNet	67.02±0.73	73.04±0.76	61.01±0.70	59.98±0.80
	PCL-MatchingNet (Ours)	69.79±0.79 (**+2.77**)	77.68±0.66 (**+4.64**)	62.40±0.73 (**+1.39**)	61.78±0.73 (**+1.80**)
	ProtoNet	65.69±0.79	73.70±0.70	61.39±0.74	58.66±0.76
	PCL-ProtoNet (Ours)	73.03±0.78 (**+7.34**)	79.03±0.63 (**+5.33**)	66.02±0.70 (**+4.63**)	63.22±0.77 (**+4.56**)
	S2M2	69.67±0.76	77.16±0.71	63.86±0.68	61.08±0.77
	PCL-S2M2 (Ours)	74.67±0.73 (**+5.00**)	80.20±0.65 (**+3.04**)	65.63±0.70 (**+1.77**)	65.82±0.74 (**+4.74**)
	DeepEMD	72.55±0.72	78.41±0.63	67.58±0.67	66.00±0.77
	PCL-DeepEMD (Ours)	77.28±0.70 (**+4.73**)	83.66±0.55 (**+5.25**)	68.70±0.72 (**+1.12**)	68.95±0.75 (**+2.95**)
40%	MatchingNet	60.98±0.75	71.31±0.73	56.70±0.72	57.46±0.77
	PCL-MatchingNet (Ours)	64.28±0.80 (**+3.30**)	74.99±0.68 (**+3.68**)	60.74±0.68 (**+4.04**)	58.51±0.71 (**+1.05**)
	ProtoNet	61.44±0.79	70.70±0.72	57.48±0.69	55.10±0.78
	PCL-ProtoNet (Ours)	65.14±0.80 (**+3.70**)	74.70±0.69 (**+4.00**)	62.75±0.69 (**+5.27**)	59.64±0.80 (**+4.54**)
	S2M2	69.03±0.78	75.30±0.68	62.70±0.71	60.59±0.79
	PCL-S2M2 (Ours)	70.99±0.97 (**+1.96**)	79.54±0.68 (**+4.24**)	64.38±0.67 (**+1.68**)	62.44±0.77 (**+1.85**)
	DeepEMD	71.88±0.71	77.56±0.68	66.28±0.69	65.67±0.79
	PCL-DeepEMD (Ours)	74.39±0.74 (**+2.51**)	79.90±0.66 (**+2.34**)	67.64±0.68 (**+1.36**)	66.69±0.77 (**+1.02**)

Table 3: Results on few-shot learning tasks with 20% & 40% label noise (Acc. %). All accuracy results are averaged over 600 test episodes with 95% confidence intervals. The relative accuracy gains with the introduction of PCL are highlighted in bold.

Algorithm	Feature noise		Label noise	
	Noise rate=20%	Noise rate=40%	Noise rate=20%	Noise rate=40%
ProtoNet	60.70±0.74	56.00±0.71	61.39±0.74	57.48±0.69
• Case 1: single module	61.77±0.73	58.44±0.72	62.18±0.68	60.87±0.71
• Case 2: independent training	63.02±0.71	60.20±0.73	63.67±0.73	61.28±0.70
• Case 3: without global weighting	63.44±0.67	57.01±0.73	64.97±0.72	58.59±0.72
• Case 4: without local weighting	63.15±0.75	60.41±0.72	64.26±0.68	61.32±0.72
Ours	**64.73±0.68**	**61.32±0.70**	**66.02±0.70**	**62.75±0.69**

Table 4: Results on few-shot learning tasks under different cases with various noise (Acc. %). All accuracy results are averaged over 600 test episodes with 95% confidence intervals.

most cases, PCL can improve the model performance by a large margin, which illustrates the validity and versatility of PCL in few-shot learning tasks.

4.3 Comparison with Corrupted Data

In this section, we consider the following two settings of data corruption: (i) *Sample corruption*. Each sample has a probability of sample corruption by replacing the sample with noise of random pixels of uniform distribution, which is an extremely noisy situation. (ii) *Label corruption*. Each sample has a probability of label corruption by replacing the sample with label noise that is a random image sampled from other classes In the experiment, for the training stage, we set the value of this probability to 20% and 40%, i.e., 20% and 40% feature noise rate or label noise rate.

Comparisons on Sample Corruption

We compare the performances of different methods in few-shot learning tasks with various rates of feature noise. Table 2 tabulates the results of different methods on few-shot learning tasks with 20% and 40% feature noise. For each comparison, the relative accuracy gains with the introduction of PCL are highlighted in bold. From Table 2, we can find that when feature noise exists in the training set, the performance of four vanilla methods severely decreases, especially in MatchingNet and ProtoNet which are trained on noisy $\mathcal{D}_{tr}$ from scratch (S2M2 and DeepEMD should be pre-trained on clean $\mathcal{D}_{tr}$). Our PCL-based methods can maintain the performance to some extent compared with the results on original clean data. The experimental results show that the introduction of PCL can significantly enhance the robustness and adaptability of few-shot methods faced with feature noise.

Comparisons on Label Corruption

Table 3 tabulates the performance of different methods in few-shot learning tasks with various rates of label noise. For each comparison, the relative accuracy gains with the introduction of PCL are highlighted in bold. In label corruption, our PCL-based methods still display their outstanding performances. From Table 3, we can find that when label noise exists in the training set, the performance of four vanilla methods severely decreases while our PCL-based methods still maintain the performance to some extent compared with the results on original clean data. Besides, the PCL framework

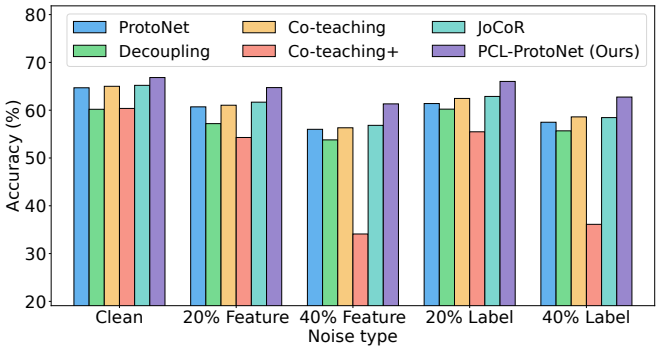

Figure 2: Comparisons among different methods dealing with noise.

can significantly improve the performance of vanilla models and surpass them by a large margin. The experimental results show that the introduction of PCL can significantly enhance the robustness of vanilla few-shot methods faced with label noise. In general, our PCL is a versatile and robust framework that can plug in different few-shot methods to improve the model performances in different scenarios.

4.4 Comparison with Existing Methods Dealing with Noise

To illustrate the advantages of PCL over existing methods that deal with noise in few-shot learning tasks, we select four methods for comparisons including Decoupling [Malach and Shalev-Shwartz, 2017], Co-teaching [Han *et al.*, 2018], Co-teaching+ [Yu *et al.*, 2019] and JoCoR [Wei *et al.*, 2020], which leverage two modules to learn interactively in the noisy environments. All methods are combined with prototype classifier for fair comparisons on *mini*-ImageNet. Moreover, vanilla ProtoNet is also adopted as the baseline method.

Figure 2 illustrates the performances of all methods in few-shot learning tasks with clean data, 20% feature noise, 40% feature noise, 20% label noise and 40% label noise, respectively. From Figure 2, we can find that: 1) when dealing with few-shot learning tasks with clean data, existing methods that deal with noise cannot always improve the performance of vanilla ProtoNet, while our PCL-ProtoNet outperforms all the compared methods by a large margin; 2) when dealing with few-shot learning tasks with corrupted data, the performances of existing methods that deal with noise are even worse than vanilla ProtoNet algorithm in many cases, while our PCL-ProtoNet achieves the best results in all cases. These observations illustrate that existing methods dealing with noise are not effective when handling few-shot learning tasks since they might be limited to noise identification without considering the model generalization. On the contrary, our PCL can effectively combine with existing few-shot methods to improve model performance. Through the peer-collaboration mechanism, both robustness and generalization ability can be promoted effectively.

4.5 Ablation Study

For a comprehensive understanding of our model, we further design four cases to evaluate the effectiveness of global weighting, local weighting and the peer-collaboration mechanism with sample corruption and data corruption on *mini*-ImageNet dataset. For each noise situation, we also add two degrees of noise rate, i.e., 20% and 40% in the base training datasets.

- Case 1: A single module with both global weighting and local weighting.
- Case 2: Each module is trained independently without leveraging losses and features from the other module.
- Case 3: Two modules are trained without global weighting, i.e., remove the global weighting term in Eq.(11).
- Case 4: Two modules are trained without local weighting, i.e., remove the local weighting term in Eq.(11).

Table 4 illustrates the performance of different cases. For each comparison, the best result is highlighted in bold. From Table 4, we can notice that: 1) The performances of the model using a single module with global and local weightings are better than the vanilla ProtoNet in all cases, which demonstrates the effectiveness of the proposed weighting method; 2) Performance drops can be observed when the module interaction is removed, which illustrates the effectiveness of the collaboration; 3) Both global weighting and local weighting is beneficial for improving the performance of the model. When the noise rate is relatively small (20%), local weighting is more important since it can effectively extract and leverage the local correlation among samples to identify noise. When the noise rate is relatively large (40%), the local correlation of samples is easily disturbed, and global weighting is more important because it can weigh the importance of different samples from a global perspective. These experimental results verify that each part of PCL is essential.

5 Conclusion

In this paper, we study few-shot learning with data corruption which is a new and practical topic. To deal with the problem, we propose the versatile and robust peer-collaboration learning (PCL) framework, which can plug in different few-shot methods to extract valid knowledge from corrupted few-shot training data. Through the peer-collaboration mechanism of the two modules, PCL can automatically identify the importance of different samples for reliable model training, thus mitigating the impact of data corruption. Experiments verify that the introduction of PCL can significantly improve the performance and robustness of few-shot methods in various data corruption scenarios. In the future, we will further study PCL in theory and improve the versatility of PCL in practical few-shot learning problems.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62076062) and the Social Development Science and Technology Project of Jiangsu Province (No.BE2022811). Furthermore, the work was also supported by the Collaborative Innovation Center of Wireless Communications Technology and the Big Data Computing Center of Southeast University.

References

[An *et al.*, 2021] Yuexuan An, Hui Xue, Xingyu Zhao, and Lu Zhang. Conditional self-supervised learning for few-shot classification. In *Proc. of IJCAI*, pages 2140–2146, 2021.

[An *et al.*, 2023] Yuexuan An, Hui Xue, Xingyu Zhao, and Jing Wang. From instance to metric calibration: A unified framework for open-world few-shot learning. *IEEE Trans. Pattern Anal. Mach. Intell.*, pages 1–17, 2023.

[Bekker and Goldberger, 2016] Alan Joseph Bekker and Jacob Goldberger. Training deep neural-networks based on unreliable labels. In *Proc. of ICASSP*, pages 2682–2686, 2016.

[Bertinetto *et al.*, 2019] Luca Bertinetto, João F. Henriques, Philip H. S. Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. In *Proc. of ICLR*, 2019.

[Boncelet, 2009] Charles Boncelet. Image noise models. In *The essential guide to image processing*, pages 143–167. 2009.

[Ceci and Peters, 1982] Stephen J. Ceci and Douglas P. Peters. Peer review: A study of reliability. *Change: The Magazine of Higher Learning*, 14(6):44–48, 1982.

[Chen *et al.*, 2019] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. In *Proc. of ICLR*, 2019.

[Chen *et al.*, 2021] Yinbo Chen, Zhuang Liu, Huijuan Xu, Trevor Darrell, and Xiaolong Wang. Meta-baseline: Exploring simple meta-learning for few-shot learning. In *Proc. of ICCV*, pages 9042–9051, 2021.

[Cheng *et al.*, 2022] De Cheng, Tongliang Liu, Yixiong Ning, Nannan Wang, Bo Han, Gang Niu, Xinbo Gao, and Masashi Sugiyama. Instance-dependent label-noise learning with manifold-regularized transition matrix estimation. In *Proc. of CVPR*, pages 16609–16618, 2022.

[Fawcett and Garton, 2005] Lillian M Fawcett and Alison F Garton. The effect of peer collaboration on children's problem-solving ability. *British journal of educational psychology*, 75(2):157–169, 2005.

[Finn *et al.*, 2017] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proc. of ICML*, volume 70, pages 1126–1135, 2017.

[Glorot *et al.*, 2011] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proc. of AISTATS*, pages 315–323, 2011.

[Han *et al.*, 2018] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor W. Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Proc. of NeurIPS*, pages 8536–8546, 2018.

[Ioffe and Szegedy, 2015] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proc. of ICML*, volume 37, pages 448–456, 2015.

[Jian and Torresani, 2022] Yiren Jian and Lorenzo Torresani. Label hallucination for few-shot classification. In *Proc. of AAAI*, pages 7005–7014, 2022.

[Jiang *et al.*, 2018] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *Proc. of ICML*, volume 80, pages 2309–2318, 2018.

[Kang *et al.*, 2021] Dahyun Kang, Heeseung Kwon, Juhong Min, and Minsu Cho. Relational embedding for few-shot classification. In *Proc. of ICCV*, pages 8802–8813, 2021.

[Kingma and Ba, 2015] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. of ICLR*, 2015.

[Kuhn, 2015] Deanna Kuhn. Thinking together and alone. *Educational researcher*, 44(1):46–53, 2015.

[Li *et al.*, 2019] Junnan Li, Yongkang Wong, Qi Zhao, and Mohan S. Kankanhalli. Learning to learn from noisy labeled data. In *Proc. of CVPR*, pages 5051–5059, 2019.

[Li *et al.*, 2022] Yiting Li, Haiyue Zhu, Sichao Tian, Fan Feng, Jun Ma, Chek Sing Teo, Cheng Xiang, Prahlad Vadakkepat, and Tong Heng Lee. Incremental few-shot object detection for robotics. In *Proc. of ICRA*, pages 8447–8453, 2022.

[Liang *et al.*, 2022] Kevin J. Liang, Samrudhdhi B. Rangrej, Vladan Petrovic, and Tal Hassner. Few-shot learning with noisy labels. In *Proc. of CVPR*, pages 9079–9088, 2022.

[Lu *et al.*, 2021] Jiang Lu, Sheng Jin, Jian Liang, and Changshui Zhang. Robust few-shot learning for user-provided data. *IEEE Trans. Neural Networks Learn. Syst.*, 32(4):1433–1447, 2021.

[Ma *et al.*, 2022a] Rongkai Ma, Pengfei Fang, Gil Avraham, Yan Zuo, Tianyu Zhu, Tom Drummond, and Mehrtash Harandi. Learning instance and task-aware dynamic kernels for few-shot learning. In *Proc. of ECCV*, pages 257–274, 2022.

[Ma *et al.*, 2022b] Rongkai Ma, Pengfei Fang, Tom Drummond, and Mehrtash Harandi. Adaptive poincaré point to set distance for few-shot classification. In *Proc. of AAAI*, pages 1926–1934, 2022.

[Malach and Shalev-Shwartz, 2017] Eran Malach and Shai Shalev-Shwartz. Decoupling "when to update" from "how to update". In *Proc. of NeurIPS*, pages 960–970, 2017.

[Mangla *et al.*, 2020] Puneet Mangla, Mayank Singh, Abhishek Sinha, Nupur Kumari, Vineeth N. Balasubramanian, and Balaji Krishnamurthy. Charting the right manifold: Manifold mixup for few-shot learning. In *Proc. of WACV*, pages 2207–2216, 2020.

[Mazumder *et al.*, 2021] Pratik Mazumder, Pravendra Singh, and Vinay P. Namboodiri. RNNP: A robust few-shot learning approach. In *Proc. of WACV*, pages 2663–2672, 2021.

[Oh *et al.*, 2021] Jaehoon Oh, Hyungjun Yoo, ChangHwan Kim, and Se-Young Yun. BOIL: towards representation change for few-shot learning. In *Proc. of ICLR*, 2021.

[Perez *et al.*, 2018] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. Film: Visual reasoning with a general conditioning layer. In *Proc. of AAAI*, pages 3942–3951, 2018.

[Ren *et al.*, 2018] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B. Tenenbaum, Hugo Larochelle, and Richard S. Zemel. Meta-learning for semi-supervised few-shot classification. In *Proc. of ICLR*, 2018.

[Shi *et al.*, 2010] Elaine Shi, Yuan Niu, Markus Jakobsson, and Richard Chow. Implicit authentication through learning user behavior. In *Proc. of ISC*, volume 6531, pages 99–113, 2010.

[Shu *et al.*, 2019] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. In *Proc. of NeurIPS*, pages 1917–1928, 2019.

[Snell *et al.*, 2017] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. In *Proc. of NeurIPS*, pages 4077–4087, 2017.

[Tsipras *et al.*, 2020] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Andrew Ilyas, and Aleksander Madry. From imagenet to image classification: Contextualizing progress on benchmarks. In *Proc. of ICML*, volume 119, pages 9625–9635, 2020.

[Van Meter and Stevens, 2000] Peggy Van Meter and Robert J Stevens. The role of theory in the study of peer collaboration. *The Journal of Experimental Education*, 69(1):113–127, 2000.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proc. of NeurIPS*, pages 5998–6008, 2017.

[Vinyals *et al.*, 2016] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *Proc. of NeurIPS*, pages 3630–3638, 2016.

[Wah *et al.*, 2011] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. Technical Report CNS-TR-2010-001, California Institute of Technology, 2011.

[Wang *et al.*, 2021] Yaqing Wang, Quanming Yao, James T. Kwok, and Lionel M. Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Comput. Surv.*, 53(3):63:1–63:34, 2021.

[Wang *et al.*, 2022a] Xiaoyan Wang, Yiwen Yuan, Dongyan Guo, Xiaojie Huang, Ying Cui, Ming Xia, Zhenhua Wang, Cong Bai, and Shengyong Chen. Ssa-net: Spatial self-attention network for COVID-19 pneumonia infection segmentation with semi-supervised few-shot learning. *Medical Image Anal.*, 79:102459, 2022.

[Wang *et al.*, 2022b] Yikai Wang, Xinwei Sun, and Yanwei Fu. Scalable penalized regression for noise detection in learning with noisy labels. In *Proc. of CVPR*, pages 346–355, 2022.

[Wei *et al.*, 2020] Hongxin Wei, Lei Feng, Xiangyu Chen, and Bo An. Combating noisy labels by agreement: A joint training method with co-regularization. In *Proc. of CVPR*, pages 13723–13732, 2020.

[Xia *et al.*, 2022] Xiaobo Xia, Tongliang Liu, Bo Han, Mingming Gong, Jun Yu, Gang Niu, and Masashi Sugiyama. Sample selection with uncertainty of losses for learning with noisy labels. In *Proc. of ICLR*, 2022.

[Xue *et al.*, 2019] Cheng Xue, Qi Dou, Xueying Shi, Hao Chen, and Pheng-Ann Heng. Robust learning at noisy labeled medical images: Applied to skin lesion classification. In *Proc. of ISBI*, pages 1280–1283, 2019.

[Ye *et al.*, 2020] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Few-shot learning via embedding adaptation with set-to-set functions. In *Proc. of CVPR*, pages 8805–8814, 2020.

[Yu *et al.*, 2019] Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor W. Tsang, and Masashi Sugiyama. How does disagreement help generalization against label corruption? In *Proc. of ICML*, volume 97, pages 7164–7173, 2019.

[Yu *et al.*, 2022] Yunlong Yu, Dingyi Zhang, and Zhong Ji. Masked feature generation network for few-shot learning. In *Proc. of IJCAI*, pages 3695–3701, 2022.

[Zeng *et al.*, 2022] Zheni Zeng, Yuan Yao, Zhiyuan Liu, and Maosong Sun. A deep-learning system bridging molecule structure and biomedical text with comprehension comparable to human professionals. *Nature communications*, 13:862, 2022.

[Zhang *et al.*, 2020] Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. Deepemd: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In *Proc. of CVPR*, pages 12200–12210, 2020.

[Zhang *et al.*, 2023] Jing Zhang, Liqin Liu, Rui Zhao, and Zhenwei Shi. A bayesian meta-learning-based method for few-shot hyperspectral image classification. *IEEE Trans. Geosci. Remote. Sens.*, 61:1–13, 2023.

[Zhu *et al.*, 2022] Zhaowei Zhu, Jialu Wang, and Yang Liu. Beyond images: Label noise transition matrix estimation for tasks with lower-quality features. In *Proc. of ICML*, volume 162, pages 27633–27653, 2022.