
Robust Meta-learning with Sampling Noise and Label Noise via Eigen-Reptile

Dong Chen¹ Lingfei Wu² Siliang Tang^{*1} Xiao Yun² Bo Long² Yueting Zhuang¹

Abstract

Recent years have seen a surge of interest in meta-learning techniques for tackling the few-shot learning (FSL) problem. However, the meta-learner is prone to overfitting since there are only a few available samples, which can be identified as sampling noise on a clean dataset. Besides, when handling the data with noisy labels, the meta-learner could be extremely sensitive to label noise on a corrupted dataset. To address these two challenges, we present Eigen-Reptile (ER) that updates the meta-parameters with the main direction of historical task-specific parameters. Specifically, the main direction is computed in a fast way, where the scale of the calculated matrix is related to the number of gradient steps for the specific task instead of the number of parameters. Furthermore, to obtain a more accurate main direction for Eigen-Reptile in the presence of many noisy labels, we further propose Introspective Self-paced Learning (ISPL). We have theoretically and experimentally demonstrated the soundness and effectiveness of the proposed Eigen-Reptile and ISPL. Particularly, our experiments on different tasks show that the proposed method is able to outperform or achieve highly competitive performance compared with other gradient-based methods with or without noisy labels. The code and data for the proposed method are provided for research purposes <https://github.com/Anfeather/Eigen-Reptile>.

1. Introduction

Meta-learning, also known as learning to learn, is one of the keys to few-shot learning (FSL) (Vinyals et al., 2016; Chi

¹College of Computer Science and Technology, Zhe Jiang University, Hang Zhou, China ²JD.COM Silicon Valley Research Center, 675 E Middlefield Rd, Mountain View, CA 94043 USA. *Siliang Tang is the corresponding author. Correspondence to: Siliang Tang <siliang@zju.edu.cn>.

et al., 2021), which usually trains meta-parameters as initialization that can fast adapt to new tasks with few samples. One line of the meta-learning is gradient-based methods that optimize meta-parameters by bi-level loop, i.e., inner loop and outer loop, which update task-specific parameters and meta-parameters, respectively. However, fewer samples often lead to a higher risk of overfitting (Zintgraf et al., 2019), due to the ubiquitous sampling noise and label noise. Particularly, a popular first-order method, Reptile (Nichol et al., 2018), updates the meta-parameters towards the inner loop direction, which is from the current initialization to the last task-specific parameters.

Sampling noise is a trend for models to overfit the randomly selected samples. As shown in Figure 1, the model tends to fit the selected sample in every gradient update and results in a high risk of overfitting (Zintgraf et al., 2019). Specifically, in (a) of Figure 1, if we sample the cat 1, the meta-learner will have a bias to the left (i.e. overfit cat 1 and the left, bold line is the corresponding update direction) and vice versa, which is an update direction disturbance caused by overfitting sampling noise. Many prior works have proposed different solutions for the aforementioned meta-overfitting problem, such as using dropout (Bertinetto et al., 2018; Lee et al., 2020), and modifying the loss function (Jamal & Qi, 2019) etc. As the disturbance is a random deviation deviated from the unbiased direction according to the selected sample, we regard disturbance as gradient noise (Wu et al., 2019). In other words, we cast the meta-overfitting problem as a disturbance of gradient noise. (Neelakantan et al., 2015) and other works have proved that introducing additional gradient noise can improve the generalization of neural networks with a large number of samples. However, for FSL, there are only a few samples of each task; thus, the model will overfit the noise (Zhang et al., 2016) and degrade the model.

Meta-learner is also inevitably affected by label noise because of the required large number of tasks. Specifically, high-quality manual labeling data is often time-consuming and expensive. And low-cost approaches to collect low-quality annotated data, such as from search engines, will introduce noisy labels. Conceptually, the initialization learned by existing meta-learning algorithms can severely degrade in the presence of noisy labels. As shown in (b) of Figure 1, noisy labels cause a significant disturbance in the update direction, which is overfitting on corrupted samples (Han

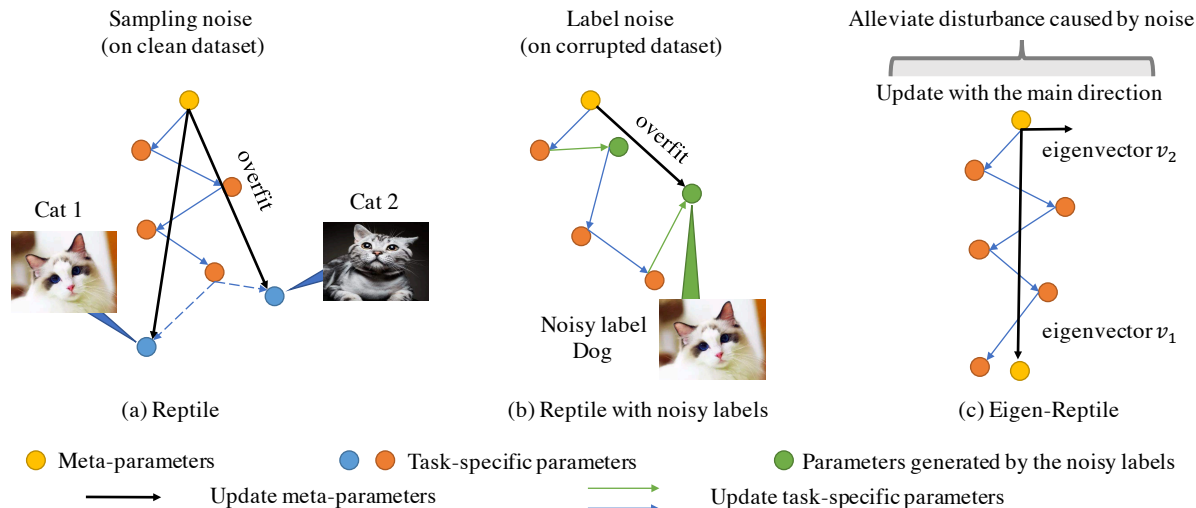


Figure 1. Inner loop steps of Reptile, Eigen-Reptile. Reptile updates meta-parameters towards the last task-specific parameters, which is biased. Eigen-Reptile considers all samples more fair with the main direction of historical task-specific parameters. Note that the main direction is the eigenvector corresponding to the largest eigenvalue.

et al., 2020; Yu et al., 2019). Furthermore, conventional algorithms for noisy labels require much more data for each class (Yao et al., 2018; Patrini et al., 2017). Therefore, new methods are needed to alleviate the gradient direction disturbance caused by overfitting label noise.

For the direction disturbance caused by overfitting sampling and label noise, we propose a simple yet effective method, coined Eigen-Reptile (ER), built upon Reptile with a different meta-update direction. In particular, as shown in (c) of Figure 1, Eigen-Reptile updates the meta-parameters with the main direction of task-specific parameters, which is more unbiased by considering all samples. However, it is unrealistic to compute parameters’ main direction due to the large scale of neural network parameters. Therefore, we introduce the process of fast computing the main direction into FSL, which computes the eigenvectors of the inner loop step scale matrix instead of the parameter scale matrix. Intuitively, noisy labels will degrade the main direction, which in turn degrades the Eigen-Reptile. To get a more accurate main direction for Eigen-Reptile, especially when the proportion of noisy labels is high, we propose Introspective Self-paced Learning (ISPL). ISPL combines Self-paced Learning (Kumar et al., 2010) with the ensemble idea to discard samples that may degrade the main direction from the dataset. We have theoretically and experimentally demonstrated the soundness and effectiveness of the proposed methods.

Experimental results show that Eigen-Reptile significantly outperforms the baseline, Reptile, by 22.93% and 5.85% on the corrupted and clean dataset, respectively. The main contributions of this paper can be summarized as follows:

- We cast the meta-overfitting issue as gradient update direction disturbance and discuss the reason for overfitting from the new perspective.
- We propose Eigen-Reptile that effectively alleviates disturbance caused by sampling and label noise. Besides, we propose ISPL, which improves the computed main direction in the presence of many noisy labels.
- We theoretically verify the effectiveness of the proposed methods for the aforementioned challenges.
- The proposed methods outperform or achieve highly competitive performance compared with recent methods on various benchmark datasets.

2. Related Work

Meta-Learning with overfitting. There are three main types of meta-learning approaches: metric-based meta-learning approaches (Ravi & Larochelle, 2016; Andrychowicz et al., 2016; Santoro et al., 2016), model-based meta-learning approaches (Vinyals et al., 2016; Koch et al., 2015; Mordatch, 2018; Snell et al., 2017; Oreshkin et al., 2018) and gradient-based meta-learning approaches (Finn et al., 2017; Jamal & Qi, 2019; Li et al., 2020a; Zintgraf et al., 2018; Li et al., 2017; Rajeswaran et al., 2019). In this paper, we focus on gradient-based meta-learning approaches which can be viewed as the bi-level loop. The goal of the outer loop is to update the meta-parameters on a variety of tasks, while task-specific parameters are learned through only a small amount of data in the inner loop. In addition, some algorithms achieve state-of-the-art results by additionally training a model with all classes on meta-training set (Yang

et al., 2020; Hu et al., 2020) like transfer learning (Zuo et al., 2018; Liu, 2020); thus, we do not discuss these algorithms in this paper.

Due to too few samples, meta-learner inevitably tends to overfit in FSL (Mishra et al., 2017). (Bertinetto et al., 2018) find that regularization such as dropout can alleviate meta-overfitting and (Yin et al., 2019) propose meta-regularization on weights; (Rajendran et al., 2020) introduce an information-theoretic framework of meta-augmentation to make meta-learner generalize to new tasks; (Ni et al., 2021; Yang et al., 2021) improve the performance of meta-learners by data augmentation, which can be combined with various meta-learning algorithms, especially for works (Lee et al., 2019; Cao et al., 2019) need more few samples.

Learning with noisy labels. Learning with noisy labels has been a long-standing problem (Fréney & Verleysen, 2013; Han et al., 2018b;a). There are many approaches to solve it, such as studying the denoise loss function (Hendrycks et al., 2018; Patrini et al., 2017; Arazo et al., 2019), relabeling (Lin et al., 2014), and so on. Nevertheless, most of these methods require much data for each class. For noisy FSL, a gradient-based meta-learner is trained to optimize an initialization on various tasks with noisy labels. As there are few samples of each class, the traditional algorithms for noisy labels cannot be applied. When the existing gradient-based meta-learning algorithms, such as Reptile, update meta-parameters, they focus on the samples that generate the last gradient step. And these samples may be corrupted, which makes the parameters learned by meta-learner susceptible to noisy labels. To better solve the problem of noisy FSL, we further proposed ISPL based on the idea of Self-paced Learning (SPL) (Kumar et al., 2010; Khan et al., 2011; Basu & Christensen, 2013; Tang et al., 2012) to learn more accurate main direction for Eigen-Reptile. ISPL constructs prior models to decide which sample should be discarded when train task-specific models. In contrast, the model with SPL learns the samples gradually from easy to complex, and the model itself decides the order, which can improve the robustness like adversarial training (Neelakantan et al., 2015; Zhang et al., 2020; Gao et al., 2020; Du et al., 2021).

3. Preliminaries

Gradient-based meta-learning aims to learn meta-parameters ϕ as initialization that can adapt to new tasks after a few iterations. The dataset D is usually divided into the meta-training set $D_{meta-train}$ and meta-testing set $D_{meta-test}$, which are used to optimize meta-parameters and evaluate its generalization, respectively. For meta-training, we have tasks $\{\mathcal{T}_i\}_{i=1}^B$ drawn from task distribution $p(\mathcal{T})$, each task has its own train set D_{train} and test set D_{test} , and the tasks in $D_{meta-test}$ are defined in the same way. Note that there are only a small number of samples for each task in FSL.

Specifically, the N-way K-shot classification task refers to K examples for each of the N classes. Generally, the number of shots in meta-training should match the one at meta-test time to obtain the best performance (Cao et al., 2019). In this paper, we follow (Lee et al., 2019; Cao et al., 2019) to increase the training shots appropriately to get the main direction of individual tasks during meta-training. To minimize the test loss of individual tasks, meta-parameters ϕ need to be updated n times to get suitable task-specific parameters $\tilde{\phi}$. That is minimizing loss of $N \times K$ samples

$$\min_{\phi} \mathbb{E}_{p(\mathcal{T})} \left[\frac{1}{N \times K} \sum_{(x,y) \in \mathcal{D}_{test}} -\log q(y | x, \tilde{\phi}) \right] \quad (1)$$

where $\tilde{\phi} = U^n(D_{train}, \phi)$, U^n represents n inner loop steps through gradient descent or Adam (Kingma & Ba, 2014) on batches from D_{train} to update the meta-parameters ϕ , $q(\cdot)$ is the predictive distribution.

When considering updating the meta-parameters in the outer loop, different algorithms have different rules. In the case of Reptile, after n inner loop steps, the meta-parameters will be updated towards the inner loop direction, which is from the current initialization to the last task-specific parameters:

$$\phi \leftarrow \phi + \beta(\tilde{\phi} - \phi) \quad (2)$$

where β is a scalar stepsize hyperparameter that controls the update rate of meta-parameters.

4. Method

The proposed Eigen-Reptile alleviates the meta-learner overfitting sampling noise (on a clean dataset) and label noise (on a corrupted dataset) by eliminating the disturbance during gradient update. Furthermore, ISPL improves the performance of Eigen-Reptile by computing a more accurate main direction when there are noisy labels in $D_{meta-train}$.

4.1. Eigen-Reptile for Clean and Corrupted Dataset

To alleviate overfitting sampling and label noise and improve the generalizability of meta-learner, we propose Eigen-Reptile, which updates d -dimensional meta-parameters with the main direction of historical task-specific parameters. We train the task-specific model with n inner loop steps (i.e., n gradient updates) that start from the meta-parameters ϕ . Let i -th column $\mathbf{W}_{:,i} \in \mathbb{R}^{d \times 1}$ of historical task-specific parameter matrix $\mathbf{W} \in \mathbb{R}^{d \times n}$ be the parameters after i -th gradient update, i.e., $\mathbf{W}_{:,i} = U^i(D_{train}, \phi)$. And treat $\mathbf{W}_{:,i}$ as a d -dimensional parameter point w_i in the parameter space. $e \in \mathbb{R}^{d \times 1}$ is a unit vector that represents the main direction of n parameter points in \mathbf{W} . Intuitively, projecting all parameter points onto e should retain the most information.

We represent the parameter points by a straight line of the form $\mathbf{w} = \bar{\mathbf{w}} + l\mathbf{e}$, where $\bar{\mathbf{w}}$ is the mean point, and l is the signed distance of a point \mathbf{w} to $\bar{\mathbf{w}}$. Then the loss function $J(l_1, l_2, \dots, l_n, \mathbf{e}) = \sum_{i=1}^n \|\bar{\mathbf{w}} + l_i\mathbf{e} - \mathbf{w}_i\|^2$. And partially differentiating J with respect to l_i , we get $l_i = \mathbf{e}^\top(\mathbf{w}_i - \bar{\mathbf{w}})$. Plugging in this expression for l_i in J ,

$$J(\mathbf{e}) = -\mathbf{e}^\top \mathbf{S} \mathbf{e} + \sum_{i=1}^n \|\mathbf{w}_i - \bar{\mathbf{w}}\|^2 \quad (3)$$

where $\mathbf{S} = \sum_{i=1}^n (\mathbf{w}_i - \bar{\mathbf{w}})(\mathbf{w}_i - \bar{\mathbf{w}})^\top$ is a scatter matrix. According to Equation (3), minimizing J is equivalent to maximizing $\mathbf{e}^\top \mathbf{S} \mathbf{e}$. Note that \mathbf{e} needs to be roughly consistent with the gradient update direction $\bar{\mathbf{V}}$ in the inner loop, as the computed \mathbf{e} without \pm . Therefore, we add a linear constraint $\bar{\mathbf{V}}^\top \mathbf{e} > 0$ and use Lagrange multiplier method as

$$\max \mathbf{e}^\top \mathbf{S} \mathbf{e} \quad \text{s.t.} \begin{cases} \bar{\mathbf{V}}^\top \mathbf{e} > 0 \\ \mathbf{e}^\top \mathbf{e} = 1 \end{cases}, \text{ where } \bar{\mathbf{V}} = \frac{1}{\lfloor n/2 \rfloor} \sum_{i=1}^{\lfloor n/2 \rfloor} \mathbf{w}_{n-i+1} - \mathbf{w}_i \quad (4)$$

We get the objective function

$$g(\mu, \mathbf{e}, \lambda, \eta) = \mathbf{e}^\top \mathbf{S} \mathbf{e} - \lambda(\mathbf{e}^\top \mathbf{e} - 1) + \mu(-\bar{\mathbf{V}}^\top \mathbf{e} + \eta^2), \quad (5)$$

where $\lambda \neq 0, \mu \geq 0$

then partially differentiating g in Equation (5) with respect to $\mu, \mathbf{e}, \lambda, \eta$,

$$\begin{cases} -\bar{\mathbf{V}}^\top \mathbf{e} + \eta^2 = 0 \\ 2\mathbf{S} \mathbf{e} - 2\lambda \mathbf{e} - \mu \bar{\mathbf{V}} = 0 \\ \mathbf{e}^\top \mathbf{e} - 1 = 0 \\ 2\mu\eta = 0 \end{cases} \quad (6)$$

According to Equation (6), if $\eta = 0$, then $\bar{\mathbf{V}}$ and \mathbf{e} are orthogonal, which obviously does not meet our expectations. So we get $\eta \neq 0$, and $\mu = 0$, then $\mathbf{S} \mathbf{e} = \lambda \mathbf{e}$. We can see \mathbf{e} is the eigenvector of \mathbf{S} corresponding to the largest eigenvalue λ , which is the required main direction. It should be noted that even if $\bar{\mathbf{V}}$ is not directly related to \mathbf{e} , in Eigen-Reptile, the linear constraint $\bar{\mathbf{V}}^\top \mathbf{e} > 0$ in Equation (4) and line 23 of Algorithm 1 must be retained as it determines the update direction of the outer-loop. Otherwise, the algorithm will not converge, which has been proven empirically.

A concerned question about $\mathbf{S} \mathbf{e} = \lambda \mathbf{e}$ is that the scatter matrix $\mathbf{S} \in R^{d \times d}$ grows quadratically with the number of parameters d . As the large number of parameters typically used in neural networks, computing eigenvalues and eigenvectors of \mathbf{S} could come at a prohibitive cost (the worst-case complexity is $\mathcal{O}(d^3)$). To avoid calculating the eigenvectors of \mathbf{S} directly, we focus on $\mathbf{W}^\top \mathbf{W}$ (centralize \mathbf{W} by subtracting the mean $\bar{\mathbf{w}}$, and the scatter matrix $\mathbf{S} = \mathbf{W} \mathbf{W}^\top$). As $\mathbf{W}^\top \mathbf{W} \hat{\mathbf{e}} = \hat{\lambda} \hat{\mathbf{e}}$, multiply both sides of the equation with \mathbf{W} ,

$$\mathbf{W} \mathbf{W}^\top \underbrace{\mathbf{W} \hat{\mathbf{e}}}_{\mathbf{e}} = \underbrace{\hat{\lambda}}_{\lambda} \underbrace{\mathbf{W} \hat{\mathbf{e}}}_{\mathbf{e}} \quad (7)$$

It can be found from Equation (7) that $\mathbf{W}^\top \mathbf{W} \in R^{n \times n}$ and $\mathbf{W} \mathbf{W}^\top \in R^{d \times d}$ have the same eigenvalue, $\lambda = \hat{\lambda}$. Furthermore, we get the eigenvector of $\mathbf{W} \mathbf{W}^\top$ as $\mathbf{e} = \mathbf{W} \hat{\mathbf{e}}$. The main advantage of Equation (7) is that the intermediate matrix $\mathbf{W}^\top \mathbf{W}$ now grows quadratically with the inner loop steps. As we are interested in FSL, n is very small. It will be much easier to compute the eigenvector $\hat{\mathbf{e}}$ of $\mathbf{W}^\top \mathbf{W}$. Then we get the eigenvector \mathbf{e} of $\mathbf{W} \mathbf{W}^\top$ based on $\hat{\mathbf{e}}$. Moreover, we project parameter update vectors $\mathbf{w}_{i+1} - \mathbf{w}_i, i = 1, 2, \dots, n-1$ on \mathbf{e} to get the corresponding update stepsize ν , so meta-parameters ϕ can be updated as

$$\phi \leftarrow \phi + \beta \nu \zeta \mathbf{e}, \text{ where } \zeta = \frac{\lambda}{\sum_{m=1}^n \lambda_m}, \quad \nu = \sum_{i=1}^{n-1} (\mathbf{w}_{i+1} - \mathbf{w}_i) \mathbf{e} \quad (8)$$

where β is a scalar stepsize hyperparameter that controls the update rate of meta-parameters, ζ is the proportion of the largest eigenvalue to the sum of all eigenvalues. The larger the value of ζ , the more accurate the meta-parameter update direction. The Eigen-Reptile algorithm is summarized in Algorithm 1.

4.2. Analysis of Eigen-Reptile

To illustrate the validity of Eigen-Reptile for alleviating overfitting sampling noise, we present Theorem 4.1 (the gradient disturbances generated by overfitting sampling noise are slightly random deviations according to the selected samples, which can be regarded as gradient noise (Wu et al., 2019)).

Theorem 4.1. *Assume that the gradient noise variable x follows Gaussian distribution (Hu et al., 2017; Jastrzbski et al., 2017; Mandt et al., 2016), i.e., $x \sim \mathcal{N}(0, \sigma^2)$. Moreover, x and neural network parameter variable are assumed to be uncorrelated. The observed covariance matrix \mathbf{C} equals noiseless covariance matrix \mathbf{C}_t plus gradient noise covariance matrix \mathbf{C}_x . Then, we get*

$$\begin{aligned} \mathbf{C} &= \frac{1}{n-1} \mathbf{S} = \mathbf{C}_t + \mathbf{C}_x = \mathbf{P}_t (\mathbf{\Lambda}_t + \mathbf{\Lambda}_x) \mathbf{P}_t^\top \\ &= \mathbf{P}_t (\mathbf{\Lambda}_t + \sigma^2 \mathbf{I}) \mathbf{P}_t^\top = \mathbf{P}_t \mathbf{\Lambda} \mathbf{P}_t^\top = \mathbf{P} \mathbf{\Lambda} \mathbf{P}^\top \end{aligned} \quad (9)$$

where \mathbf{P}_t and \mathbf{P} are the orthonormal eigenvector matrices of \mathbf{C}_t and \mathbf{C} respectively, $\mathbf{\Lambda}_t$ and $\mathbf{\Lambda}$ are the corresponding diagonal eigenvalue matrices, and \mathbf{I} is an identity matrix. It can be seen from Equation (9) that \mathbf{C} and \mathbf{C}_t has the same eigenvectors. We defer the proof to the Appendix B.

Theorem 4.1 shows that eigenvectors are not affected by gradient noise. Therefore, Eigen-Reptile can find a more generalizable starting point for new tasks without overfitting sampling noise (on the clean dataset). As for label noise (on the corrupted dataset), the analysis is shown in Appendix D. We also show the complexity analysis in Appendix C, which illustrates that Reptile and Eigen-Reptile are the same in spatial complexity and time complexity.

Algorithm 1 Eigen-Reptile

Input: Distribution over tasks $P(\mathcal{T})$, outer step size β .

```

1: Initialize meta-parameters  $\phi$ 
2: while not converged do
3:    $\mathbf{W} = [], \nu = 0$ 
4:   Sample batch of tasks  $\{\mathcal{T}_i\}_{i=1}^B \sim P(\mathcal{T})$ 
5:   for each task  $\mathcal{T}_i$  do
6:      $\phi_i = \phi$ 
7:     Sample train set  $D_{train}$  of  $\mathcal{T}_i$ 
8:     for  $j = 1, 2, 3, \dots, n$  do
9:        $\phi_i^j = U^j(D_{train}, \phi_i)$ 
10:       $\mathbf{W}$  appends  $\mathbf{W}_{:,j} = \text{flatten}(\phi_i^j)$ 
11:    end for
12:    Mean centering,  $\mathbf{W} = \mathbf{W} - \bar{\mathbf{w}}, \bar{\mathbf{w}} \in R^{d \times 1}$ 
13:    Compute matrix  $\hat{\mathbf{A}}$  and eigenvector matrix  $\hat{\mathbf{P}}$  of scatter matrix  $\mathbf{W}^\top \mathbf{W}$ 
14:    Eigenvalues  $\lambda_1 > \lambda_2 > \dots > \lambda_n$  in  $\hat{\mathbf{A}}$ 
15:    Compute matrix of  $\mathbf{W} \mathbf{W}^\top$ ,  $\mathbf{P} = \mathbf{W} \hat{\mathbf{P}}$ 
16:    Let the eigenvector corresponding to  $\lambda_1$  be a unit vector,  $\|e_i^1\|_2 = 1$ 
17:    for  $j = 1, 2, 3, \dots, n-1$  do
18:       $\nu = \nu + (\mathbf{W}_{:,j+1} - \mathbf{W}_{:,j})e_i^1$ 
19:    end for
20:     $e_i^1 = \frac{\lambda_1}{\sum_{m=1}^n \lambda_m} \times e_i^1$ 
21:    Calculate the approximate direction of task-specific gradient update  $\bar{\mathbf{V}}$ :
22:     $\bar{\mathbf{V}} = \frac{1}{\lfloor n/2 \rfloor} \sum_{i=1}^{\lfloor n/2 \rfloor} \mathbf{W}_{:,n-i+1} - \mathbf{W}_{:,i}$ 
23:    if  $e_i^1 \cdot \bar{\mathbf{V}} < 0$  then
24:       $e_i^1 = -e_i^1$ 
25:    end if
26:  end for
27:  Average the main directions to get
28:   $\tilde{\mathbf{e}} = (1/B) \sum_{i=1}^B e_i^1$ 
29:  Update meta-parameters  $\phi \leftarrow \phi + \beta \times \nu/B \times \tilde{\mathbf{e}}$ 
end while
    
```

4.3. The Introspective Self-paced Learning for More Accurate Main Direction

As shown in [Appendix D](#), Eigen-Reptile addresses the noisy FSL problem by separating noisy information. However, with the increase of noise ratio, the eigenvector will gradually become invalid. To get a more accurate eigenvector, we propose Introspective Self-paced Learning (ISPL).

Self-paced learning (SPL) learns the samples from low losses to high losses, which is proven beneficial in achieving a better generalization result ([Khan et al., 2011](#); [Basu & Christensen, 2013](#); [Tang et al., 2012](#)). Besides, some previous work ([Zhu et al., 2019](#)) solve the problem of traditional noisy labels by SPL. Nevertheless, in a meta-learning setting, a meta-learner is trained on various tasks; the initial

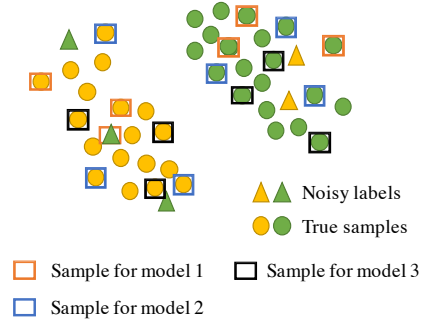


Figure 2. Randomly sample examples to build prior models.

model may have lower losses for trained classes and higher losses for unseen classes or noisy samples. For this reason, we cannot train the task-specific model in the same way as SPL to solve the noisy FSL problem. To this end, we propose an improved SPL algorithm to help Eigen-Reptile achieve better performance for the problem of the noisy labels. As shown in Figure 2, even though the two categories of the yellow and green show an excellent distribution that can be well separated, some samples are marked wrong. Then, we build three prior models. Specifically, we randomly sample three times, and model 1 is trained with a corrupted label. Due to different samples learned by prior models, building multiple models to vote on the data will obtain more accurate losses, which is a kind of ensemble learning. Moreover, samples with losses above a certain threshold will be discarded. Furthermore, we imitate SPL to add the hidden variable $v = 0$ or 1 that is decided by Q prior models before the loss of each sample to control whether the sample should be abandoned. And we get the task-specific loss as

$$L_{ISPL}(\phi, \mathbf{v}) = \sum_{i=1}^h v_i L(x_i, y_i, \phi), \quad (10)$$

$$\text{where } v_i = \arg \min_{v_i} \frac{v_i}{Q} \sum_{j=1}^Q L_j(x_i, y_i, \phi_j) - \gamma v_i$$

where h is the number of samples x from dataset D_{train} , y is label, γ is the sample selection parameter, which gradually decreases, parameter of model j is $\phi_j = U^n(D_j, \phi)$, $D_j \in D_{train}$. Note that we update the meta-parameters with the model trained on h samples from D_{train} . The objective of Equation (10) is choosing samples whose summary loss is lower than γ , as the hidden variable $v = 1$, and vice versa. ISPL is summarized in [Appendix A](#).

4.4. Analysis of ISPL

Intuitively, it is difficult to say whether discarding high-loss samples containing correct and corrupted samples will improve the accuracy of eigenvector, so we prove the effectiveness of ISPL by Theorem 4.2.

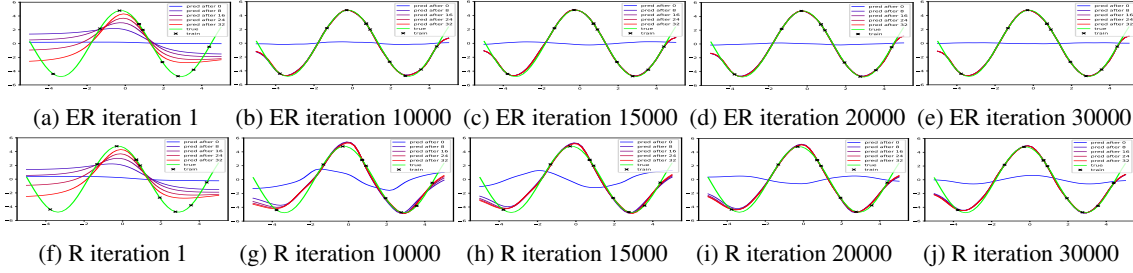


Figure 3. Eigen-Reptile (ER) and Reptile (R) training process on the regression toy test. (a), (b), (c), (d), (e) and (f), (g), (h), (i), (j) show that after the gradient update 0, 8, 16, 24, 32 times based on initialization learned by Eigen-Reptile and Reptile respectively.

Theorem 4.2. Let W_o be the parameter matrix only generated by the corrupted samples. Compute the eigenvalues and eigenvectors of the observed expected parameter matrix

$$\begin{aligned} & \frac{1}{\lambda} \mathbb{E}(\mathbf{C}_{tr}) \mathbf{e} \\ &= \mathbf{P}_o (\mathbf{I} - \frac{\Lambda_o}{\lambda}) \mathbf{P}_o^\top \mathbf{e} \\ &\approx \mathbf{P}_o (\mathbf{I} - \frac{\lambda_o}{\lambda} \mathbf{I}) \mathbf{P}_o^\top \mathbf{e} > \mathbf{P}_o (\mathbf{I} - \frac{\lambda_o - \xi}{\lambda - \xi} \mathbf{I}) \mathbf{P}_o^\top \mathbf{e} \end{aligned} \quad (11)$$

where \mathbf{C}_{tr} is the covariance matrix generated by clean samples, λ is the observed largest eigenvalue, λ_o is the largest eigenvalue in the corrupted diagonal eigenvalue matrix Λ_o , \mathbf{P}_o is the orthonormal eigenvector matrix of corrupted covariance matrix. According to Equation (11), if λ_o/λ is smaller, the observed eigenvector \mathbf{e} is more accurate. Assume that the discarded high loss samples have the same contributions ξ to λ and λ_o , representing the observed and corrupted main directional variance, respectively. Note that these two kinds of data have the same effect on the gradient updating of the model, as they all generate high loss (neither new clean samples from new tasks nor corrupted samples are not familiar to the model). Furthermore, it is easy to find that $(\lambda_o - \xi)/(\lambda - \xi)$ is smaller than λ_o/λ . We defer the proof to the [Appendix E](#).

Theorem 4.2 shows that discard high loss samples can improve the accuracy of the observed eigenvector learned with corrupted samples. Therefore, ISPL can improve the performance of Eigen-Reptile.

5. Experimental Results and Discussion

In our experiments, we aim to (1) compare different update directions of meta-parameters, (2) evaluate the effectiveness of Eigen-Reptile to alleviate overfitting sampling and label noise, (3) test the robustness of Eigen-Reptile to some hyperparameters, (4) evaluate the improvement of ISPL to Eigen-Reptile in the presence of noisy labels. All experiments run on a 2080 Ti.

5.1. Meta-learning with the Best Update Direction on Clean Dataset

In this experiment, we try to compare main direction with other update directions and evaluate the effectiveness of Eigen-Reptile to alleviate overfitting sampling noise by the 1D sine wave K -shot regression problem (Nichol et al., 2018). Each task is defined by a sine curve $y(x) = A \sin(x + b)$, where the amplitude $A \sim U([0.1, 5.0])$ and phase $b \sim U([0, 2\pi])$. The amplitude A and phase b are varied between tasks. The goal of each task is to fit a sine curve with the data points sampled from the corresponding $y(x)$. We calculate the loss in ℓ_2 using 50 equally-spaced points from the whole interval $[-5.0, 5.0]$ for each task. The loss is

$$\int_{-5.0}^{5.0} \|y(x) - \hat{y}(x)\|^2 dx \quad (12)$$

where $\hat{y}(x)$ is the predicted function that start from the initialization learned by meta-learner.

The K -shot regression task fits a selected sine curve through K points, here $K = 10$. For the regressor, we use a small neural network, which is the same as (Nichol et al., 2018), except that the activation functions are Tanh. Specifically, the small network includes an input layer of size 1, followed by two hidden layers of size 64, and then an output layer of size 1. All meta-learners use the same regressor that is trained for 30000 iterations with inner loop steps 5, batch size 10, and a fixed inner loop learning rate of 0.02.

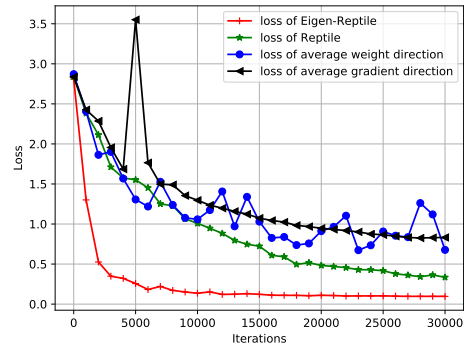


Figure 4. Loss of different update direction.

Table 1. Accuracy of FSL on Mini-Imagenet N-way K-shot. The \pm shows 95% confidence interval over tasks. The number in (\cdot) denotes the number of filters.

Algorithm	5-way 1-shot	5-way 5-shot
MAML (Finn et al., 2017)	48.70 \pm 1.84%	63.11 \pm 0.92%
FOML (Finn et al., 2017)	48.07 \pm 1.75%	63.15 \pm 0.91%
GNN (Gidaris & Komodakis, 2018)	50.30%	66.40%
TAML (Jamal & Qi, 2019)	51.77 \pm 1.86%	65.60 \pm 0.93%
Meta-dropout (Lee et al., 2019)	51.93 \pm 0.67%	67.42 \pm 0.52%
Warp-MAML (Flennerhag et al., 2020)	52.30 \pm 0.80%	68.4 \pm 0.60%
MC (128) (Park & Oliva, 2020)	54.08 \pm 0.93%	67.99 \pm 0.73%
sparse-MAML (Von Oswald et al., 2021)	51.04 \pm 0.59%	68.05 \pm 0.84%
MeTAL (Baik et al., 2021)	52.63 \pm 0.37%	70.52 \pm 0.29%
MixtFSL (Afrasiyabi et al., 2021)	52.82 \pm 0.63%	70.67 \pm 0.57%
Reptile (32) (Nichol et al., 2018)	49.97 \pm 0.32%	65.99 \pm 0.58%
Eigen-Reptile (32)	51.80 \pm 0.90%	68.10 \pm 0.50%
Eigen-Reptile (64)	53.25 \pm 0.45%	69.85 \pm 0.85%

Table 2. Few Shot Classification on CIFAR-FS N-way K-shot accuracy. The \pm shows 95% confidence interval over tasks.

Algorithm	5-way 1-shot	5-way 5-shot
MAML	58.90 \pm 1.90%	71.50 \pm 1.00%
PROTO NET	55.50 \pm 0.70%	72.00 \pm 0.60%
GNN	61.90%	75.30%
ECM	55.14 \pm 0.48%	71.66 \pm 0.39%
Reptile	58.30 \pm 1.20%	75.45 \pm 0.55%
Eigen-Reptile	61.90 \pm 1.40%	78.30 \pm 0.50%

We report the results of Reptile and Eigen-Reptile in Figure 3. It can be seen that the curve fitted by Eigen-Reptile is closer to the true green curve, which shows that Eigen-Reptile performs better. According to (Jamal & Qi, 2019), the initial model with a larger entropy before adapting to new tasks would better alleviate meta-overfitting. As shown in Figure 3, from 1 to 30000 iterations, Eigen-Reptile is more generalizable than Reptile as the initial blue line of Eigen-Reptile is closer to a straight line (larger entropy), which shows that the initialization learned by Eigen-Reptile is less affected by overfitting. Furthermore, Figure 4 shows that update meta-parameters by the main direction converges faster and gets a lower loss than other update directions, such as the average of gradient direction and average of task-specific weights direction.

5.2. Alleviate Overfitting Sampling Noise on Clean Dataset

We verify the effectiveness of Eigen-Reptile alleviate overfitting sampling noise on two clean few-shot classification datasets Mini-Imagenet (Vinyals et al., 2016) and CIFAR-FS (Bertinetto et al., 2018).

The Mini-Imagenet dataset contains 100 classes, each with 600 images. We follow (Ravi & Larochelle, 2016) to divide the dataset into three disjoint subsets: meta-training set, meta-validation set, and meta-testing set with 64 classes, 16 classes, and 20 classes, respectively. We follow the few-shot learning protocols from prior work (Vinyals et al., 2016), except that the number of the meta-training shot is 15 as (Lee et al., 2019; Cao et al., 2019), which is still much smaller than the number of samples required by traditional tasks. Moreover, we run our algorithm on the dataset for the different number of test shots and compare our results to other meta-learning algorithms. What needs to be reminded is that approaches that use deeper, residual networks or pretrained all-way classifier on meta-training set can achieve higher accuracies (Gidaris & Komodakis, 2018; Yang et al., 2020; Hu et al., 2020). So for a fair comparison, we only compare algorithms that use convolutional networks without a pretrained model as Reptile does. Specifically, our model follows (Nichol et al., 2018), which has 4 modules with a 3×3 convolutions and 32 filters, 2×2 max-pooling etc. The images are downsampled to 84×84 , and the loss function is the cross-entropy error. We use Adam with $\beta_1 = 0$ in the inner loop. Our model is trained for 100000 iterations with a fixed inner loop learning rate of 0.0005, 7 inner-loop steps and batch size 10.

The results of Eigen-Reptile and other meta-learning approaches are summarized in Table 1. The proposed Eigen-Reptile (64 filters) achieves highly competitive performance compared with other algorithms for 5-shot and 1-shot classification problems. More specifically, for 1-shot, the results of MC are similar to that of Eigen-Reptile. However, as a second-order optimization algorithm, the computational cost of MC is much higher than that of Eigen-Reptile (as shown in Appendix C, Eigen-Reptile is a first-order algorithm like

Table 3. Average test accuracy of 5-way 1-shot on the Mini-Imagenet with label noise. S and AS denotes symmetric and asymmetric noise, respectively. All methods are trained with early stopping to against noisy labels (Li et al., 2020b), especially when $p = 0.5$, the results of 20000 or more iterations for Reptile are only 20% that equivalent to random guessing. Besides, for a fair comparison, we force all methods to get similar results when $p = 0$ to compare the robustness when p is higher. Therefore, the reported results are lower than that of Table 1.

Algorithm	$p = 0.0$	$p = 0.1$		$p = 0.2$		$p = 0.5$	
		S	AS	S	AS	S	AS
MeTAL (Baik et al., 2021)	47.57%	44.64%	46.44%	40.67%	45.02%	27.05%	41.53%
Reptile (Nichol et al., 2018)	47.64%	46.08%	47.30%	43.49%	45.51%	23.33%	42.03%
Reptile+ISPL	47.23%	46.50%	47.00%	43.70%	45.42%	21.83%	41.09%
Eigen-Reptile	47.87%	47.18%	47.42%	45.01%	46.50%	27.23%	42.29%
Eigen-Reptile+ISPL	47.26%	47.20%	47.24%	45.49%	46.83%	28.68%	43.71%

Reptile). As for 5-shot, the results of MeTAL and MixtFSL are slightly higher than that of Eigen-Reptile, while Eigen-Reptile performs better on the 1-shot task. Furthermore, the results of Eigen-Reptile with 32 filters are much better than that of Reptile with 32 filters for each task. Compared with Reptile, Eigen-Reptile uses the main direction to update the meta-parameters to alleviate the meta-overfitting caused by sampling noise. More importantly, Eigen-Reptile outperforms the state-of-the-art meta-overfitting preventing method Meta-dropout (Lee et al., 2019), which is based on regularization. This result shows the effectiveness of addressing the meta-overfitting problem from the perspective of alleviating gradient noise.

The results of CIFAR-FS with new baseline ECM (Ravichandran et al., 2019) and PROTO Nets (Snell et al., 2017) are shown in Table 2. The settings of Eigen-Reptile in this experiment are the same as that of Mini-Imagenet experiments. Moreover, we do not compare algorithms with additional tricks, such as higher way. It can be seen from Table 2, the performance of Eigen-Reptile is still far better than Reptile without any hyperparameter adjustment.

We follow (Lee et al., 2019; Cao et al., 2019) to vary the number of inner-loops and the number of corresponding training shots to show the robustness of Eigen-Reptile on the 5-way 5-shot problem in Appendix F, and results show that after the number of inner-loops reaches 7 with 15 train shots, the test accuracy tends to be stable.

5.3. Alleviate Overfitting Label Noise on Corrupted Dataset

We conduct the 5-way 1-shot experiment with noisy labels generated by corrupting the original labels of Mini-Imagenet. There are symmetric label noise and asymmetric label noise in this experiment. For symmetric label noise, correct labels are flipped to other labels with equal probability, i.e., in the case of symmetric noise of ratio p , a sample retains the correct label with probability $1 - p$, and it be-

comes some other label with probability $p/(N - 1)$. On the other hand, for asymmetric label noise, we randomly flip the labels of one class to the labels of another fixed class with probability p .

All meta-learners with 32 filters are trained with early stopping to against noisy labels (Li et al., 2020b) and get similar results when $p = 0$. This experiment examines the robustness of different methods when facing different p (i.e., when $p \neq 0$, the closer the results are to that of $p = 0$, the better the method’s robustness). The sample selection parameter γ of ISPL is 10, which decreases by 0.6 every 1000 iterations, and the number of prior models is 2. Moreover, we only introduce noise during meta-training, where the train shot is 30.

As shown in Table 3, for symmetric label noise, with the increase of the ratio p , the performance of Reptile decreases rapidly. When $p = 0.5$, the initialization learned by Reptile can hardly meet the requirements of quickly adapting to new tasks with few samples. In contrast, MeTAL with an adaptive loss function performs relatively better when $p = 0.5$. On the other hand, Eigen-Reptile is less affected by symmetric label noise in all cases. As for asymmetric label noise, meta-learners are trained on tasks with the same noise transition matrix, which allows meta-learners to learn more useful information, so the results are higher than that with symmetric noise. Like the results of symmetric label noise, Eigen-Reptile outperforms Reptile in all tasks.

Eigen-Reptile+ISPL achieves better results than that of Eigen-Reptile when p is low. Specifically, ISPL plays a more significant role when p is higher. However, when $p = 0$, ISPL harms Eigen-Reptile, as ISPL only discards correct samples. These results correspond to the conclusion in Appendix D, with the increase of noise ratio, the eigenvector will gradually become invalid, which demonstrates the effectiveness of ISPL and verify the idea that ISPL collaborates with Eigen-Reptile by providing improved main direction. In addition, ISPL does not significantly improve or even degrades the results of Reptile. This is because too

many high-loss samples are removed, causing Reptile to fail to converge quickly with the same number of iterations.

These experimental results show that Eigen-Reptile and ISPL can effectively separate noisy information, thereby alleviating overfitting label noise.

6. Conclusion

This paper proposes a gradient-based meta-learning algorithm Eigen-Reptile. It updates the meta-parameters with the main direction, proven by theory and experiments to effectively alleviate the overfitting on sampling and label noise. Furthermore, to get closer to real-world situations, we introduce noisy labels into the meta-training dataset. The proposed ISPL constructs prior models to select samples for Eigen-Reptile to get a more accurate main direction in the presence of noisy labels.

Acknowledgements

This work has been supported in part by National Key Research and Development Program of China (2018AAA0101900), Zhejiang NSF (LR21F020004), Alibaba-Zhejiang University Joint Research Institute of Frontier Technologies, Key Research and Development Program of Zhejiang Province, China (No. 2021C01013), Chinese Knowledge Center of Engineering Science and Technology (CKCEST).

References

- Afrasiyabi, A., Lalonde, J.-F., and Gagné, C. Mixture-based feature space learning for few-shot image classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9041–9051, 2021.
- Andrychowicz, M., Denil, M., Gomez, S., Hoffman, M. W., Pfau, D., Schaul, T., Shillingford, B., and De Freitas, N. Learning to learn by gradient descent by gradient descent. In *Advances in neural information processing systems*, pp. 3981–3989, 2016.
- Arazo, E., Ortego, D., Albert, P., O’Connor, N. E., and McGuinness, K. Unsupervised label noise modeling and loss correction. *arXiv preprint arXiv:1904.11238*, 2019.
- Baik, S., Choi, J., Kim, H., Cho, D., Min, J., and Lee, K. M. Meta-learning with task-adaptive loss function for few-shot learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9465–9474, 2021.
- Basu, S. and Christensen, J. Teaching classification boundaries to humans. In *Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013.
- Bertinetto, L., Henriques, J. F., Torr, P. H., and Vedaldi, A. Meta-learning with differentiable closed-form solvers. *arXiv preprint arXiv:1805.08136*, 2018.
- Cao, T., Law, M., and Fidler, S. A theoretical analysis of the number of shots in few-shot learning. *arXiv preprint arXiv:1909.11722*, 2019.
- Chi, H., Liu, F., Yang, W., Lan, L., Liu, T., Niu, G., and Han, B. Meta discovery: Learning to discover novel classes given very limited data. *arXiv preprint arXiv:2102.04002*, 2021.
- Du, X., Zhang, J., Han, B., Liu, T., Rong, Y., Niu, G., Huang, J., and Sugiyama, M. Learning diverse-structured networks for adversarial robustness. *arXiv preprint arXiv:2102.01886*, 2021.
- Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1126–1135. JMLR. org, 2017.
- Flennerhag, S., Rusu, A., Pascanu, R., Visin, F., Yin, H., and Hadsell, R. Meta-learning with warped gradient descent. In *International Conference on Learning Representations 2020*, 2020.
- Frénay, B. and Verleysen, M. Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems*, 25(5):845–869, 2013.
- Gao, R., Liu, F., Zhang, J., Han, B., Liu, T., Niu, G., and Sugiyama, M. Maximum mean discrepancy is aware of adversarial attacks. *arXiv preprint arXiv:2010.11415*, 2020.
- Gidaris, S. and Komodakis, N. Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4367–4375, 2018.
- Han, B., Yao, J., Niu, G., Zhou, M., Tsang, I., Zhang, Y., and Sugiyama, M. Masking: A new perspective of noisy supervision. *arXiv preprint arXiv:1805.08193*, 2018a.
- Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I., and Sugiyama, M. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *arXiv preprint arXiv:1804.06872*, 2018b.
- Han, B., Niu, G., Yu, X., Yao, Q., Xu, M., Tsang, I., and Sugiyama, M. Sigua: Forgetting may make learning with noisy labels more robust. In *International Conference on Machine Learning*, pp. 4006–4016. PMLR, 2020.
- Hendrycks, D., Mazeika, M., Wilson, D., and Gimpel, K. Using trusted data to train deep networks on labels corrupted by severe noise. In *Advances in neural information processing systems*, pp. 10456–10465, 2018.

- Hu, S. X., Moreno, P. G., Xiao, Y., Shen, X., Obozinski, G., Lawrence, N. D., and Damianou, A. Empirical bayes transductive meta-learning with synthetic gradients. *arXiv preprint arXiv:2004.12696*, 2020.
- Hu, W., Li, C. J., Li, L., and Liu, J.-G. On the diffusion approximation of nonconvex stochastic gradient descent. *arXiv preprint arXiv:1705.07562*, 2017.
- Jamal, M. A. and Qi, G.-J. Task agnostic meta-learning for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 11719–11727, 2019.
- Jastrzbski, S., Kenton, Z., Arpit, D., Ballas, N., Fischer, A., Bengio, Y., and Storkey, A. Three factors influencing minima in sgd. *arXiv preprint arXiv:1711.04623*, 2017.
- Khan, F., Mutlu, B., and Zhu, J. How do humans teach: On curriculum learning and teaching dimension. In *Advances in neural information processing systems*, pp. 1449–1457, 2011.
- Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- Koch, G., Zemel, R., and Salakhutdinov, R. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2. Lille, 2015.
- Kumar, M. P., Packer, B., and Koller, D. Self-paced learning for latent variable models. In *Advances in Neural Information Processing Systems*, pp. 1189–1197, 2010.
- Lee, H., Nam, T., Yang, E., and Hwang, S. Meta dropout: Learning to perturb latent features for generalization. In *International Conference on Learning Representations*, 2020.
- Lee, H. B., Nam, T., Yang, E., and Hwang, S. J. Meta dropout: Learning to perturb latent features for generalization. In *International Conference on Learning Representations*, 2019.
- Li, J., Wang, X., Tang, S., Shi, H., Wu, F., Zhuang, Y., and Wang, W. Y. Unsupervised reinforcement learning of transferable meta-skills for embodied navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12123–12132, 2020a.
- Li, M., Soltanolkotabi, M., and Oymak, S. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In *International conference on artificial intelligence and statistics*, pp. 4313–4324. PMLR, 2020b.
- Li, Z., Zhou, F., Chen, F., and Li, H. Meta-sgd: Learning to learn quickly for few-shot learning. *arXiv preprint arXiv:1707.09835*, 2017.
- Lin, C. H., Weld, D. S., et al. To re (label), or not to re (label). In *Second AAAI conference on human computation and crowdsourcing*, 2014.
- Liu, F. *Towards Realistic Transfer Learning Methods: Theory and Algorithms*. PhD thesis, 2020.
- Mandt, S., Hoffman, M., and Blei, D. A variational analysis of stochastic gradient algorithms. In *International conference on machine learning*, pp. 354–363, 2016.
- Mishra, N., Rohaninejad, M., Chen, X., and Abbeel, P. A simple neural attentive meta-learner. *arXiv preprint arXiv:1707.03141*, 2017.
- Mordatch, I. Concept learning with energy-based models. *arXiv preprint arXiv:1811.02486*, 2018.
- Neelakantan, A., Vilnis, L., Le, Q. V., Sutskever, I., Kaiser, L., Kurach, K., and Martens, J. Adding gradient noise improves learning for very deep networks. *arXiv preprint arXiv:1511.06807*, 2015.
- Ni, R., Goldblum, M., Sharaf, A., Kong, K., and Goldstein, T. Data augmentation for meta-learning. In *International Conference on Machine Learning*, pp. 8152–8161. PMLR, 2021.
- Nichol, A., Achiam, J., and Schulman, J. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.
- Oreshkin, B., López, P. R., and Lacoste, A. Tadam: Task dependent adaptive metric for improved few-shot learning. In *Advances in Neural Information Processing Systems*, pp. 721–731, 2018.
- Park, E. and Oliva, J. B. Meta-curvature. 2020.
- Patrini, G., Rozza, A., Krishna Menon, A., Nock, R., and Qu, L. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1944–1952, 2017.
- Rajendran, J., Irpan, A., and Jang, E. Meta-learning requires meta-augmentation. *arXiv preprint arXiv:2007.05549*, 2020.
- Rajeswaran, A., Finn, C., Kakade, S. M., and Levine, S. Meta-learning with implicit gradients. In *Advances in Neural Information Processing Systems*, pp. 113–124, 2019.
- Ravi, S. and Larochelle, H. Optimization as a model for few-shot learning. 2016.

- Ravichandran, A., Bhotika, R., and Soatto, S. Few-shot learning with embedded class models and shot-free meta training. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 331–339, 2019.
- Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., and Lillicrap, T. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pp. 1842–1850, 2016.
- Snell, J., Swersky, K., and Zemel, R. Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pp. 4077–4087, 2017.
- Tang, Y., Yang, Y.-B., and Gao, Y. Self-paced dictionary learning for image classification. In *Proceedings of the 20th ACM international conference on Multimedia*, pp. 833–836, 2012.
- Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. Matching networks for one shot learning. In *Advances in neural information processing systems*, pp. 3630–3638, 2016.
- Von Oswald, J., Zhao, D., Kobayashi, S., Schug, S., Caccia, M., Zucchet, N., and Sacramento, J. Learning where to learn: Gradient sparsity in meta and continual learning. *Advances in Neural Information Processing Systems*, 34, 2021.
- Wu, J., Hu, W., Xiong, H., Huan, J., Braverman, V., and Zhu, Z. On the noisy gradient descent that generalizes as SGD. 2019.
- Yang, L., Li, L., Zhang, Z., Zhou, X., Zhou, E., and Liu, Y. Dpgn: Distribution propagation graph network for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13390–13399, 2020.
- Yang, S., Liu, L., and Xu, M. Free lunch for few-shot learning: Distribution calibration. *arXiv preprint arXiv:2101.06395*, 2021.
- Yao, J., Wang, J., Tsang, I. W., Zhang, Y., Sun, J., Zhang, C., and Zhang, R. Deep learning from noisy image labels with quality embedding. *IEEE Transactions on Image Processing*, 28(4):1909–1922, 2018.
- Yin, M., Tucker, G., Zhou, M., Levine, S., and Finn, C. Meta-learning without memorization. *arXiv preprint arXiv:1912.03820*, 2019.
- Yu, X., Han, B., Yao, J., Niu, G., Tsang, I., and Sugiyama, M. How does disagreement help generalization against label corruption? In *International Conference on Machine Learning*, pp. 7164–7173. PMLR, 2019.
- Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.
- Zhang, J., Xu, X., Han, B., Niu, G., Cui, L., Sugiyama, M., and Kankanhalli, M. Attacks which do not kill training make adversarial learning stronger. In *International Conference on Machine Learning*, pp. 11278–11287. PMLR, 2020.
- Zhu, P., Ma, W., and Hu, Q. Self-paced robust deep face recognition with label noise. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 425–435. Springer, 2019.
- Zintgraf, L., Shiarli, K., Kurin, V., Hofmann, K., and Whiteson, S. Fast context adaptation via meta-learning. In *International Conference on Machine Learning*, pp. 7693–7702. PMLR, 2019.
- Zintgraf, L. M., Shiarlis, K., Kurin, V., Hofmann, K., and Whiteson, S. Caml: Fast context adaptation via meta-learning. 2018.
- Zuo, H., Lu, J., Zhang, G., and Liu, F. Fuzzy transfer learning using an infinite gaussian mixture model and active learning. *IEEE Transactions on Fuzzy Systems*, 27(2):291–303, 2018.

A. PSEUDO-CODE

Algorithm 2 Introspective Self-paced Learning

Input: Dataset D_{train} , initialization ϕ , batch size b , selection parameter γ , attenuation coefficient μ , the number of prior models Q .

- 1: Initialize network parameters $\phi^* = \phi$ for a sampled task
 - 2: **for** $j = 1, 2, 3, \dots, Q$ **do**
 - 3: Sample examples D_j from D_{train} for training $model_j$, $\phi_j = U^m(D_j, \phi^*)$
 - 4: **end for**
 - 5: Train task-specific parameters:
 - 6: **for** $i = 1, 2, 3, \dots, n$ **do**
 - 7: Compute hidden variable vector v :
 - 8: $v = \arg \min_v v_q \sum_{q=1}^b L_q - \gamma \sum_{q=1}^b v_q$,
 where $L_q = \frac{1}{Q} \sum_{j=1}^Q L_j(x_q, y_q, \phi_j)$
 - 9: Update task-specific parameters ϕ^* :
 - 10: $\phi^* = \arg \min_{\phi^*} L_{ISPL}(\phi^*, v)$
 - 11: $\gamma = \gamma - \mu$
 - 12: **end for**
-

B. Proof of Theorem 1

Gradient update always with gradient noise inserted at every iteration due to sampling noise, which caused Reptile, MAML, etc. tend to overfit the trained samples. In this section, we will prove that Eigen-Reptile can alleviate meta-overfitting by alleviating gradient noise.

Theorem 1 Assume that the gradient noise variable x follows Gaussian distribution (Hu et al., 2017; Jastrzbski et al., 2017; Mandt et al., 2016), $x \sim \mathcal{N}(0, \sigma^2)$. Furthermore, x and neural network parameter variable are assumed to be uncorrelated. The observed covariance matrix C equals noiseless covariance matrix C_t plus gradient noise covariance matrix C_x . Then, we get

$$\begin{aligned} C &= \frac{1}{n-1} S = C_t + C_x = P_t(\Lambda_t + \Lambda_x)P_t^\top \\ &= P_t(\Lambda_t + \sigma^2 I)P_t^\top = P_t \Lambda P_t^\top = P \Lambda P^\top \end{aligned} \quad (13)$$

where P_t and P are the orthonormal eigenvector matrices of C_t and C respectively, Λ_t and Λ are the corresponding diagonal eigenvalue matrices, and I is an identity matrix. It can be seen from Equation (13) that C and C_t has the same eigenvectors.

Proof. In the following proof, we assume that the probability density function of gradient noise variable x follows Gaussian distribution, $x \sim \mathcal{N}(0, \sigma^2)$. Treat the parameters in the neural network as variables, and the parameters obtained by each gradient update as samples. Furthermore, gradient noise and neural network parameters are assumed to be uncorrelated.

For observed parameter matrix $W \in R^{d \times n}$, there are n samples, let $W_{i,:} \in R^{1 \times n}$ be the observed values of the i -th variable W_i , and $W = [W_{1,:}^\top, \dots, W_{i,:}^\top, \dots, W_{d,:}^\top]^\top$. Similarly, we denote the noiseless parameter matrix by $W^t = [(W_{1,:}^t)^\top, \dots, (W_{i,:}^t)^\top, \dots, (W_{d,:}^t)^\top]^\top$, and

$$W = W^t + X \quad (14)$$

Where $X = [X_{1,:}^\top, \dots, X_{i,:}^\top, \dots, X_{d,:}^\top]^\top$ is the dataset of noise variables. Then, centralize each variable by

$$\bar{W}_k = W_k - \frac{1}{n} \sum_{i=1}^n W_{k,:}(i) \quad (15)$$

So we get $\bar{W} = [\bar{W}_1^\top, \dots, \bar{W}_d^\top]^\top$. Suppose W^t is also centralized by the same way and get $\bar{W}^t = [\bar{W}_1^t, \dots, \bar{W}_d^t]^\top$.

Then, we have:

$$\overline{\mathbf{W}} = \overline{\mathbf{W}^t} + \mathbf{X} \quad (16)$$

Computing the covariance matrix of $\overline{\mathbf{W}}$:

$$\begin{aligned} \mathbf{C} &= \frac{1}{n} \overline{\mathbf{W}} \overline{\mathbf{W}}^\top \\ &= \frac{1}{n} (\overline{\mathbf{W}^t} + \mathbf{X})(\overline{\mathbf{W}^t}^\top + \mathbf{X}^\top) \\ &= \frac{1}{n} (\overline{\mathbf{W}^t} \overline{\mathbf{W}^t}^\top + \overline{\mathbf{W}^t} \mathbf{X}^\top + \mathbf{X} \overline{\mathbf{W}^t}^\top + \mathbf{X} \mathbf{X}^\top) \end{aligned} \quad (17)$$

Since $\overline{\mathbf{W}^t}$ and \mathbf{X} are uncorrelated, $\overline{\mathbf{W}^t} \mathbf{X}^\top$ and $\mathbf{X} \overline{\mathbf{W}^t}^\top$ are approximately zero matrices. Thus:

$$\mathbf{C} \approx \frac{1}{n} (\overline{\mathbf{W}^t} \overline{\mathbf{W}^t}^\top + \mathbf{X} \mathbf{X}^\top) = \mathbf{C}_t + \mathbf{C}_x \quad (18)$$

The component $C_x(i, j)$ is the correlation between \mathbf{X}_i and \mathbf{X}_j which corresponds to the i -th and j -th rows of \mathbf{X} . As the two noise variables are not related to each other, if $i \neq j$, then $C_x(i, j) = 0$. So $\mathbf{C}_x \in R^{d \times d}$ is a diagonal matrix with diagonal elements σ^2 . Decompose \mathbf{C}_t as:

$$\mathbf{C}_t = \mathbf{P}_t \mathbf{\Lambda}_t \mathbf{P}_t^\top \quad (19)$$

where \mathbf{P}_t is the noiseless orthonormal eigenvector matrix and $\mathbf{\Lambda}_t$ is the noiseless diagonal eigenvalue matrix, then

$$\mathbf{C}_x = \mathbf{\Lambda}_x \mathbf{P}_t \mathbf{P}_t^\top = \mathbf{P}_t \mathbf{\Lambda}_x \mathbf{P}_t^\top = \mathbf{P}_t \mathbf{C}_x \mathbf{P}_t^\top \quad (20)$$

where $\mathbf{\Lambda}_x = \sigma^2 \mathbf{I}$, and \mathbf{I} is the identity matrix. Thus,

$$\begin{aligned} \mathbf{C} &= \mathbf{C}_t + \mathbf{C}_x \\ &= \mathbf{P}_t \mathbf{\Lambda}_t \mathbf{P}_t^\top + \mathbf{P}_t \mathbf{\Lambda}_x \mathbf{P}_t^\top \\ &= \mathbf{P}_t (\mathbf{\Lambda}_t + \mathbf{\Lambda}_x) \mathbf{P}_t^\top \\ &= \mathbf{P}_t \mathbf{\Lambda} \mathbf{P}_t^\top \end{aligned} \quad (21)$$

where $\mathbf{\Lambda} = \mathbf{\Lambda}_t + \mathbf{\Lambda}_x$. It can be seen from Equation (21) that \mathbf{C} and \mathbf{C}_t has the same eigenvector matrix. In other words, **eigenvector is not affected by gradient noise.** \square

C. Algorithm Complexity Analysis of Eigen-Reptile

As for the time complexity of Eigen-Reptile, the cost of single gradient descent in the inner-loop is $\mathcal{O}(d)$, where d is the number of network parameters. The cost of the scatter/covariance matrix computations is $\mathcal{O}(n^2d)$, where n is the number of inner-loop. Moreover, the worst-case complexity of computing eigenvalue decomposition is $\mathcal{O}(n^3)$. Finally, the computational complexity of restoring eigenvector is $\mathcal{O}(nd)$. We set the maximal number of outer-loop to T . Hence the time complexity of Eigen-Reptile is

$$\mathcal{O}(T(n^3 + n^2d + nd + nd)) = \mathcal{O}(Td) \quad (22)$$

In FSL, n is small (in this paper $n = 7$), so the overall time complexity is $\mathcal{O}(Td)$.

As for Reptile, the time complexity is also $\mathcal{O}(Td)$, which means that the time complexity of both Reptile and Eigen-Reptile is much lower than the second-order optimization algorithms.

As for spatial complexity, Eigen-Reptile needs to store a $d \times n$ matrix and a $n \times n$ matrix. The overall spatial complexity is $\mathcal{O}(d)$, while the spatial complexity of Reptile is $\mathcal{O}(d)$, too.

It can be seen that, compared to Reptile, Eigen-Reptile is the same in spatial complexity and time complexity. Still, the accuracy of Eigen-Reptile is much higher than that of Reptile.

D. Effectiveness of Eigen-Reptile for Label Noise

Eigen-Reptile alleviates the overfitting on label noise by separating noisy information. More specifically, from the perspective of signal to noise ratio (SNR) :

$$SNR = \frac{\delta_{clean}^2}{\delta_{noise}^2} \quad (23)$$

where δ_{clean}^2 is the variance of weight points introduced by clean data, and δ_{noise}^2 is the variance of weight points introduced by corrupted data.

Normally, $SNR \gg 1$, which shows that the effective information is far more than noisy information. Specifically, in a task-specific training, weight matrix \mathbf{W} is composed of all weight points, and we decompose it as:

$$\mathbf{W} = \mathbf{U}\mathbf{S}\mathbf{V}^T = \sum_{k=1}^r \sigma_k \mathbf{u}_k \mathbf{v}_k^T \quad (24)$$

where r is the rank of \mathbf{W} , \mathbf{u}_k is the eigenvector of the covariance matrix $\mathbf{W}\mathbf{W}^T$, \mathbf{v}_k is the eigenvector of the covariance matrix $\mathbf{W}^T\mathbf{W}$.

The singular values are ordered in decreasing magnitude as $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_r$ which are the positive square roots of the eigenvalues of $\mathbf{W}\mathbf{W}^T$. In Eigen-Reptile, we only remain the eigenvector corresponding to the largest eigenvalue, e.g., σ_1^2 , which can be viewed as the δ_{clean}^2 in SNR. In contrast, the noisy information δ_{noise}^2 is removed by omitting the low singular values. However, with the increase of noise ratio, Eigen-Reptile is more and more likely to suffer from the problem that the influence of corrupted samples gradually dominates the main update direction.

E. Proof of Theorem 2

In this section, we will prove that discarding high loss samples will result in a more accurate main direction when there are noisy labels.

Theorem 2 *Let \mathbf{W}_o be the parameter matrix only generated by the corrupted samples. Compute the eigenvalues and eigenvectors of the observed expected parameter matrix*

$$\begin{aligned} \frac{1}{\lambda} \mathbb{E}(\mathbf{C}_{tr})\mathbf{e} &= \mathbf{P}_o(\mathbf{I} - \frac{\mathbf{\Lambda}_o}{\lambda})\mathbf{P}_o^\top \mathbf{e} \\ &\approx \mathbf{P}_o(\mathbf{I} - \frac{\lambda_o}{\lambda}\mathbf{I})\mathbf{P}_o^\top \mathbf{e} > \mathbf{P}_o(\mathbf{I} - \frac{\lambda_o - \xi}{\lambda - \xi}\mathbf{I})\mathbf{P}_o^\top \mathbf{e} \end{aligned} \quad (25)$$

where \mathbf{C}_{tr} is the covariance matrix generated by clean samples, λ is the observed largest eigenvalue, λ_o is the largest eigenvalue in the corrupted diagonal eigenvalue matrix $\mathbf{\Lambda}_o$, \mathbf{P}_o is the orthonormal eigenvector matrix of corrupted covariance matrix. According to Equation (25), if λ_o/λ is smaller, the observed eigenvector \mathbf{e} is more accurate. Assume that the discarded high loss samples have the same contributions ξ to λ and λ_o , representing the observed and corrupted main directional variance, respectively. Note that these two kinds of data have the same effect on the gradient updating of the model, as they all generate high loss (neither new clean samples from new tasks nor corrupted samples are not familiar to the model). Furthermore, it is easy to find that $(\lambda_o - \xi)/(\lambda - \xi)$ is smaller than λ_o/λ .

Proof. Here, we use \mathbf{w} to represent the parameter point obtained after a gradient update. For convenience, let \mathbf{w} be generated by a single sample, $\mathbf{w} \in R^{d \times 1}$. Then the parameter matrix can be obtained,

$$\mathbf{W} = [\mathbf{w}_1^{tr} \quad \mathbf{w}_2^{tr} \quad \dots \quad \mathbf{w}_1^o \quad \dots \quad \mathbf{w}_m^o \quad \dots \quad \mathbf{w}_n^{tr}] \quad (26)$$

where \mathbf{w}^o represents the parameters generated by the corrupted sample, and \mathbf{w}^{tr} represents the parameters generated by the true sample. Furthermore, there are n parameter points generated by n samples. Moreover, there are m corrupted parameter

points generated by m corrupted samples. Mean centering \mathbf{W} , and show the observed covariance matrix \mathbf{C} as

$$\begin{aligned}
 \mathbf{C} &= \frac{1}{n} \mathbf{W} \mathbf{W}^\top \\
 &= \frac{1}{n} [\mathbf{w}_1^{tr} \quad \mathbf{w}_2^{tr} \quad \cdots \quad \mathbf{w}_n^{tr}] \begin{bmatrix} (\mathbf{w}_1^{tr})^\top \\ (\mathbf{w}_2^{tr})^\top \\ \vdots \\ (\mathbf{w}_n^{tr})^\top \end{bmatrix} \\
 &= \frac{1}{n} (\mathbf{w}_1^{tr} (\mathbf{w}_1^{tr})^\top + \cdots + \mathbf{w}_1^o (\mathbf{w}_1^o)^\top + \cdots + \mathbf{w}_m^o (\mathbf{w}_m^o)^\top \\
 &\quad + \cdots + \mathbf{w}_n^{tr} (\mathbf{w}_n^{tr})^\top)
 \end{aligned} \tag{27}$$

It can be seen from the decomposition of \mathbf{C} that the required eigenvector is related to the parameters obtained from the true samples and the parameters obtained from the noisy samples. For a single parameter point

$$\begin{aligned}
 \mathbf{w} \mathbf{w}^\top &= \begin{bmatrix} a_1 \\ \vdots \\ a_i \\ \vdots \\ a_d \end{bmatrix} [a_1 \quad \cdots \quad a_i \quad \cdots \quad a_d] \\
 &= \begin{bmatrix} a_1^2 & a_1 a_2 & \cdots & a_1 a_d \\ a_2 a_1 & a_2^2 & \cdots & a_2 a_d \\ \vdots & \vdots & \ddots & \vdots \\ a_d a_1 & a_d a_2 & \cdots & a_d^2 \end{bmatrix}
 \end{aligned} \tag{28}$$

As we discard all high loss samples that make the model parameters change significantly, and the randomly generated noisy labels may cause the gradient to move in any direction, we assume that the variance of corrupted parameter point variables is δ . Compute the expectations of all variables in the corrupted parameter point

$$\begin{aligned}
 \mathbb{E}(\mathbf{w} \mathbf{w}^\top) &= \\
 &\begin{bmatrix} \delta_1^2 + \mathbb{E}(a_1)^2 & \mathbb{E}(a_1 a_2) & \cdots & \mathbb{E}(a_1 a_d) \\ \mathbb{E}(a_2 a_1) & \delta_2^2 + \mathbb{E}(a_2)^2 & \cdots & \mathbb{E}(a_2 a_d) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbb{E}(a_d a_1) & \mathbb{E}(a_d a_2) & \cdots & \delta_d^2 + \mathbb{E}(a_d)^2 \end{bmatrix} = \Omega
 \end{aligned} \tag{29}$$

Let the sum of all corrupted $\frac{1}{n} \mathbb{E}(\mathbf{w} \mathbf{w}^\top)$ be Ω_o , then

$$\begin{aligned}
 \Omega_o &= \\
 \frac{1}{n} &\begin{bmatrix} m\delta^2 + \sum_{j=1}^m \mathbb{E}(a_{j1})^2 & \cdots & \sum_{j=1}^m \mathbb{E}(a_{j1} a_{jd}) \\ \sum_{j=1}^m \mathbb{E}(a_{j2} a_{j1}) & \cdots & \sum_{j=1}^m \mathbb{E}(a_{j2} a_{jd}) \\ \vdots & \ddots & \vdots \\ \sum_{j=1}^m \mathbb{E}(a_{jd} a_{j1}) & \cdots & m\delta^2 + \sum_{j=1}^m \mathbb{E}(a_{jd})^2 \end{bmatrix}
 \end{aligned} \tag{30}$$

And let the sum of all true $\frac{1}{n} \mathbf{w} \mathbf{w}^\top$ be \mathbf{C}_{tr} . So the expectation of \mathbf{C} can be written as,

$$\mathbb{E}(\mathbf{C}) = \mathbb{E}(\mathbf{C}_{tr}) + \Omega_o \tag{31}$$

Treat eigenvector and eigenvalue as definite values, we get

$$(\Omega_o + \mathbb{E}(\mathbf{C}_{tr})) \mathbf{e} = \lambda \mathbf{e} \tag{32}$$

where e is the observed eigenvector, λ is the corresponding eigenvalue. Divide both sides of the equation by λ .

$$\begin{aligned} \frac{1}{\lambda} \mathbb{E}(C_{tr})e &= (\mathbf{I} - \frac{1}{\lambda} \mathbf{\Omega}_o)e \\ &= \mathbf{P}_o(\mathbf{I} - \frac{1}{\lambda} \mathbf{\Lambda}_o)\mathbf{P}_o^\top e \\ &\approx \mathbf{P}_o(\mathbf{I} - \frac{\lambda_o}{\lambda} \mathbf{I})\mathbf{P}_o^\top e \end{aligned} \quad (33)$$

where λ_o is the largest eigenvalue in the corrupted diagonal eigenvalue matrix $\mathbf{\Lambda}_o$, \mathbf{P}_o is the orthonormal eigenvector matrix of $\mathbf{\Omega}_o$. According to Equation (33), if λ_o/λ is smaller, e is more accurate. Discard some samples with the largest losses, which may contain true samples and noisy samples. Assume that the discarded high loss samples have the same contributions ξ to λ and λ_o , as these two kinds of data have the same effect on the gradient updating of the model. Compare the ratio of eigenvalues before and after discarding, get

$$\underbrace{\frac{\lambda_o}{\lambda}}_{\text{before}} - \underbrace{\frac{\lambda_o - \xi}{\lambda - \xi}}_{\text{after}} = \frac{\xi(\lambda - \lambda_o)}{\lambda(\lambda - \xi)} > 0 \quad (34)$$

Obviously, $\lambda > \lambda_o$, and if we don't discard all samples, then $\lambda > \xi$. So Equation (34) > 0 , which means discarding high loss samples could reduce λ_o/λ . **Therefore, discarding high loss samples can improve the accuracy of eigenvector in the presence of noisy labels.**

For further analysis, we assume that any two variables are independently and identically distributed, the expectation of variable a , $\mathbb{E}(a) = \epsilon$. Thus,

$$\frac{1}{\lambda} \mathbf{\Omega}_o = \frac{p}{\lambda} \begin{bmatrix} \delta^2 + \epsilon^2 & \dots & \epsilon^2 \\ \epsilon^2 & \dots & \epsilon^2 \\ \vdots & \ddots & \vdots \\ \epsilon^2 & \dots & \delta^2 + \epsilon^2 \end{bmatrix} \quad (35)$$

where p is the proportion of noisy labels, $np = m$. As can be seen from Equation (35), if $p\epsilon^2/\lambda \approx 0$, then $\mathbf{\Omega}_o/\lambda$ is a diagonal matrix. According to proof. B, the observed eigenvector e is unaffected by noisy labels with the corresponding eigenvalue $\frac{p(\delta^2 + \epsilon^2)}{\lambda}$. \square

F. Hyperparameter Analysis

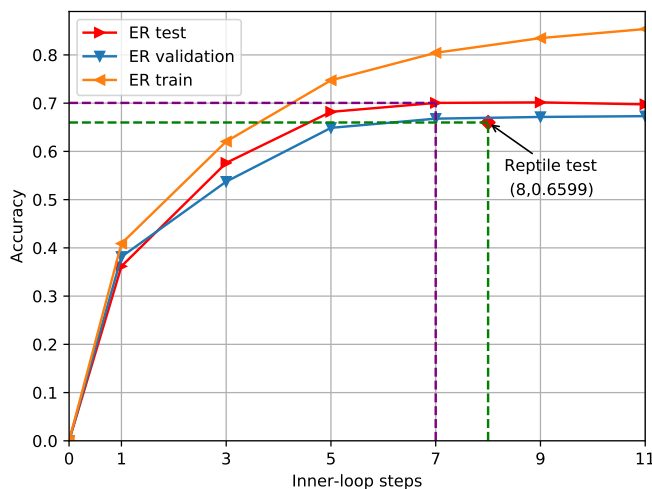


Figure 5. The number of inner-loop and accuracy of 5-way 5-shot task on Mini-Imagenet.

As shown in Figure 5, after the number of inner-loops i reaches 7, the test accuracy tends to be stable, which shows that changing the number of inner-loops within a specific range has little effect on Eigen-Reptile. That is, Eigen-Reptile is robust to this hyperparameter. As for train shot, to make the trained task-specific parameters as unbiased as possible, we specify train shot roughly satisfies $\lceil \frac{i \times \text{batch_size}}{N} \rceil + 1$, where N is the number of classes. So when $i = 7$, the number of train shots is 15. It is important to note that in our experiments, Reptile with the hyperparameters set by its authors, the number of inner-loops is 8, the number of train shots is 15, and the corresponding accuracy is 65.99%, which is much lower than the result of Eigen-Reptile.

G. Relationship between Eigen-Reptile and ISPL

Eigen-Reptile and ISPL are strongly correlated because of the following three aspects: (1) intuitively, with the increase of noise ratio, Eigen-Reptile is more likely to suffer from the problem that the influence of corrupted samples gradually dominates the main update direction. Therefore, ISPL is proposed to improve the accuracy of the main direction for Eigen-Reptile by introspective priors. (2) theoretically, Theorem 4.2 shows that ISPL discards high loss samples to reduce λ_o / λ in Equation 11, which helps improve the accuracy of the observed eigenvector learned with corrupted samples. (3) empirically, in Table 3, stable performance improvement of Eigen-Reptile + ISPL over Eigen-Reptile can be observed, and the performance gap becomes more significant when the noise ratio p increases (which corresponds to the conclusion in *Appendix D* - the high noise ratio is more likely to degrade the largest singular value and the corresponding eigenvector). The experimental results demonstrate the effectiveness of ISPL and verify the idea that ISPL collaborates with Eigen-Reptile by providing improved main direction (which would be more helpful as the noise ratio becomes larger).