Appendix

Organization of the appendix. In Appendix A, we provide the major notations used in the main paper in Tab. 6 and the pseudo code of the proposed algorithm in Algorithm 1. In Appendix B, we present the illustrations for C-CL, E-CL, and the PAC-Bayesian hierarchical framework of DCML in task, subject and curriculum levels. In Appendix C, we provide the convergence analysis. In Appendix D, we provide detailed proofs for all the major theoretical results. Additional experimental settings, results and ablation studies are provided in Appendix E.

A Notations and Algorithm

Table 6 summarizes the major notations used in the paper. Algorithm 1 provides the detailed DCML training process.

Table 6: Notations and Descriptions

Notation	**Description**
θ	parameter of meta-model
θ_i	parameter of task-specific model for task $\mathcal{T}_i$
α	learning rate of meta-model
β	learning rate of task-specific model
$\mathcal{Z}$	Data domain space
$z_{ij} = (x_{ji}, y_{ij})$	a data sample $z_{ij} \sim \mathcal{Z}$
$\mathcal{D}$	data distribution defined over domain $\mathcal{Z}$
$\mathcal{T}$	a task
τ	full task distribution constituted by all the meta-train classes
τ_i	a subject, i.e., a task distribution constituted by a subset the meta-train classes
$\mathcal{D}^m \sim \tau_i$	a data distribution sampled from the task distribution τ_i
$S_i \sim \mathcal{D}^m$	an instantiation of data distribution $\mathcal{D}^m$ containing the support set and query set samples of task $\mathcal{T}_i$ with m data examples
$\mathcal{D}_i^{sup}, \mathcal{D}_i^{que}$	support and query set for task $\mathcal{T}_i$
n_i	number of task sampled in subject τ_i
$p_t(\mathbf{x})$	CP-statistics the output logit vector of image $\mathbf{x}$ of with each element $p_t^{cp_{jk}}$ denoting a class pair j and k
μ, σ	mean/variance of historic CP-statistics $p_t^{cp_{jk}}$ over $t = 0, ..., T$, T is the number of checkpoints in total
μ_{th}, σ_{th}	predefined threshold of μ, σ for C-CL
$C_{easy}, C_{similar}, C_{noisy}$	group of easy/similar/noisy class pairs
N_j	total number of sampled instances in class j
λ	predefined threshold for E-CL for each task
$\mathcal{H}$	hypothesis space
$\mathcal{M}(\mathcal{H})$	all possible measures over space $\mathcal{H}$
P	prior distribution over hypotheses $P \in \mathcal{M}(\mathcal{H})$
$Q : \mathcal{Z}^m \times \mathcal{M} \to \mathcal{M}$	posterior distribution $Q \in \mathcal{M}(\mathcal{H})$, which is a mapping from data domain with m samples $\mathcal{Z}^m$ and some measure space $\mathcal{M}$ the some measure space $\mathcal{M}$
$\mathcal{P}$	hyper-prior over P, $\mathcal{P} \in \mathcal{M}(\mathcal{M}(\mathcal{H}))$
$\mathcal{Q}$	hyper-posterior over P, $\mathcal{Q} \in \mathcal{M}(\mathcal{M}(\mathcal{H}))$
$D(Q\|\|P)$	KL divergence between distributions Q and P

B Illustrations

In Fig. 4 and Fig. 5, we provide the illustration for class-level sampling and example-level sampling, respectively. In Fig. 4, the classes in the data distribution are colored with different shades of blue, the harder the classes, the bluer. For general meta-learning, the tasks are formed by randomly sampled from N classes from the data distribution. In this process, classes with different hardness are included and sometimes extremely noisy classes are also included, making the tasks less valuable and highly noisy. In our C-CL, we arrange the classes into different subjects according to their hardness in an easy to complex order to form a curriculum. We also exclude the noisy classes from the curriculum. In this way, when forming tasks from the curriculum, we forms more meaningful tasks with classes sampled from similar difficulty levels and not sampling from the extremely noisy and less valuable classes.

Algorithm 1: Dual-level Curriculum Meta-Learning (DCML)

```
1:  **INPUT** meta-model $\theta^0$, task distributions $\tau$, class and example thresholds $\sigma_{th}^0$, $\mu_{th}^0$, $\lambda$
2:  // *Line 4-10, warm-up and train target model on full dataset*
3:  Initialize meta-model $\theta^{t-1} = \theta^0$
4:  **while** not done **do**
5:      Sample a batch of tasks $\{\mathcal{T}_j\}$ from $\tau$
6:      **for** each task $\mathcal{T}_j$ **do**
7:          Update task-specific model: $\theta_j = \theta^{t-1} - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_j}(\theta^0, \mathcal{D}_j^{\text{sup}})$
8:      **end for**
9:      Update meta-model: $\theta^t \leftarrow \theta^{t-1} - \beta \nabla_{\theta^{t-1}} \sum_{\mathcal{T}_j \sim \tau} \mathcal{L}_{\mathcal{T}_j}(\theta_j, \mathcal{D}_j^{\text{que}})$
10:     **if** time step condition meet **then**
11:         Save the proxy model: $\theta_p = \theta^t$
12:     **end if**
13: **end while**
14: Update class groups $C_{easy}, C_{similar}, C_{noisy}$ using the proxy models $\theta_p$s according to $\mu_{th}^i$, $\sigma_{th}^i$
15: Update task distribution subset $\tau_i = C_{easy}^i \bigcup C_{similar}^i$ using Eqn. (4)
16: // *Line 15-30, Dual-level Curriculum training*
17: **while** not done **do**
18:     Sample a batch of tasks $\{\mathcal{T}_j\}$ from $\tau_i$
19:     **for** each task $\mathcal{T}_j$ **do**
20:         Update support set weight vector $\mathbf{v}_j^{\text{sup}}$ for $\mathcal{D}_j^{\text{sup}}$ using proxy model $\theta_p$ as in Eqn. (6)
21:         Update task-specific model: $\theta_j = \theta^t - \alpha \nabla_{\theta^t} \mathbf{v}_j^{\text{sup}} \mathcal{L}_{\mathcal{T}_j}(\theta^t, \mathcal{D}_j^{\text{sup}})$
22:         Update query set weight vector $\mathbf{v}_j^{\text{que}}$ for $\mathcal{D}_j^{\text{que}}$ and using task-specific model $\theta_j$
23:     **end for**
24:     Update meta-model with fixed query set weight vector: $\theta^{t+1} \leftarrow \theta^t - \beta \nabla_{\theta^t} \sum_{\mathcal{T}_j \sim \tau_i} \mathbf{v}_j^{\text{que}} \mathcal{L}_{\mathcal{T}_j}(\theta_j, \mathcal{D}_j^{\text{que}})$
25:     **if** time step condition meet **then**
26:         Save the proxy model: $\theta_p^{t+1} = \theta^{t+1}$
27:     **end if**
28:     **if** time curriculum update condition meet **then**
29:         Increase $\mu_{th}^i$, $\sigma_{th}^i$ by 10% to enlarge the class curriculum
30:         Update class groups $C_{easy}, C_{similar}, C_{noisy}$ using $\theta_p$s according to $\mu_{th}^i$, $\sigma_{th}^i$
31:         Update task distribution subset $\tau_i = C_{easy}^i \bigcup C_{similar}^i$ using Eqn. (4)
32:     **end if**
33: **end while**
```

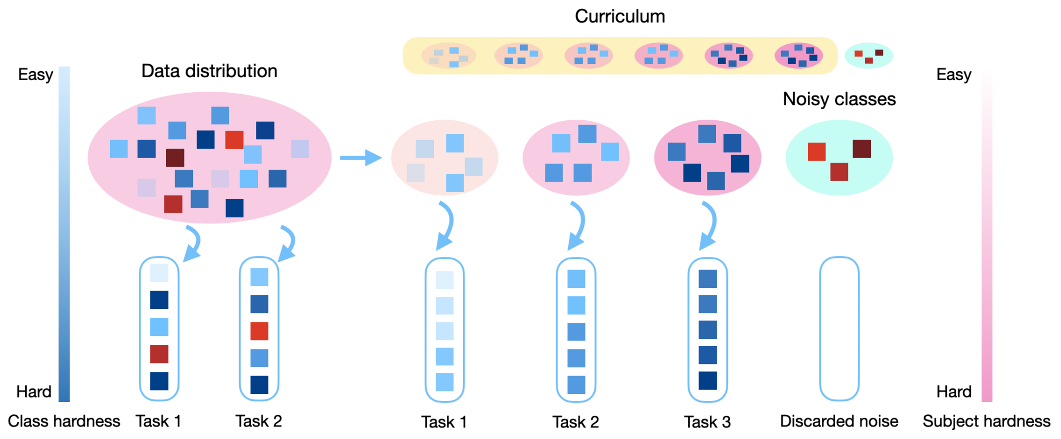


Figure 4: Class-level sampling: The data distribution is formed by different classes, which may contain extremely noisy ones. C-CL ranks the hardness of the classes from easy to complex, and group classes with similar hardness into a subjects, a series of subject forms a curriculum. Besides, the noisy classes are excluded from the curriculum.

In Fig. 5, we show the process of E-CL with an example of a 3-way 1-shot task. We first randomly sample three classes from the chosen subject, then we randomly oversample a batch of four examples from each classes, which are ordered by their losses, the task is finally formed with the examples with least loss from the batch, which are considered to be clean.

Fig. 6 demonstrates the Hierarchical PAC-Bayesian framework for DCML, which includes three-levels: task, subject, and curriculum. Specifically, for a single task, a prior model (*i.e.,* meta-model) $P(h)$ is generated from the hyper-posterior $\mathcal{Q}(P(h))$, and adapted to new posterior $Q(h)$ with observed dataset S. Within the subject, the hyper-posterior $\mathcal{Q}(P(h))$ is learned with an hyper-prior $\mathcal{P}(P(h))$ which is sampled from the measure space $\mathcal{M}(\mathcal{M}(\mathcal{H}))$, and updated to the intermediate hyper-posterior $\mathcal{Q}(P(h))$ with a batch of tasks sampled from the i-th subject, which is defined by data distribution $\mathcal{D}^i$ and task environment τ_i. A curriculum is formed by a series of subjects from easy to complex. The hyper-posterior $\mathcal{Q}(P(h))$ is then trained tasks sampled from the curriculum in an easy to complex order.

C Convergence Analysis

Given the complex dual-level design, one may concern whether the model converges effectively. In this section, we provide theoretical analysis to demonstrate the nice convergence behavior in the following theorem. Our analysis leverages a self-paced regularizer (Jiang et al. 2014a), which is shown by Definition 1. We further assume the loss function $\ell_{\mathcal{T}_i,j}$ is twice continuously differentiable and L-smooth.

The self-paced function for meta-learning is defined as:

Definition 1. Denote v as a weight variable, ℓ as the loss, and λ as the model age parameter. A function $\mathcal{J}(\lambda,\ell)$ is a self-paced function if

$$\mathcal{J}(\lambda,\ell) = \arg\min_{v\in[0,1]} v\ell + R(v;\lambda)$$

where

- $R(v;\lambda)$ is convex with respect to $v \in [0,1]$.
- $\mathcal{J}(\lambda,\ell)$ is monotonically decreasing with respect to ℓ and $\lim_{\ell\to 0}\mathcal{J}(\lambda,\ell)=1$, $\lim_{\ell\to\infty}\mathcal{J}(\lambda,\ell)=0$.
- $\mathcal{J}(\lambda,\ell)$ is monotonically decreasing with respect to λ and $\lim_{\lambda\to 0}\mathcal{J}(\lambda,\ell)=1$, $\lim_{\lambda\to\infty}\mathcal{J}(\lambda,\ell)=0$.

Theorem 4. *For any self-paced function* $\mathrm{F}(\theta,\mathbf{v})$ *with the hard-weighting regularizer* $R^h(\mathbf{v}^t_{\mathcal{T}_i})$ *and the update of model parameter* θ *and weight parameter* $\mathbf{v}$ *as:*

$$\theta^{t+1} = \theta^t - \beta_t \nabla_\theta \mathrm{F}(\theta^t,\mathbf{v}^t)\mid_{\mathcal{D}^{que}} ,$$
$$\mathbf{v}^{t+1} = \mathbf{v}^t - \beta_t \nabla_{\mathbf{v}} \mathrm{F}(\theta^{t+1},\mathbf{v}^t)\mid_{\mathcal{D}^{que}} ,$$

the gradient descent method finds stationary points for $\mathrm{F}(\theta^{t+1},\mathbf{v}^{t+1})$ *as*

$$\lim_{t\to\infty} \mathbb{E}[\mathrm{F}(\theta^{t+1},\mathbf{v}^{t+1})] - \mathbb{E}[\mathrm{F}(\theta^t,\mathbf{v}^t)] = 0.$$

The theorem makes an analysis on the expectation because the tasks and examples are randomly sampled in meta-learning settings. The stationary points indicate the convergence.

Next, we present the complete proofs of the Theorem 4.

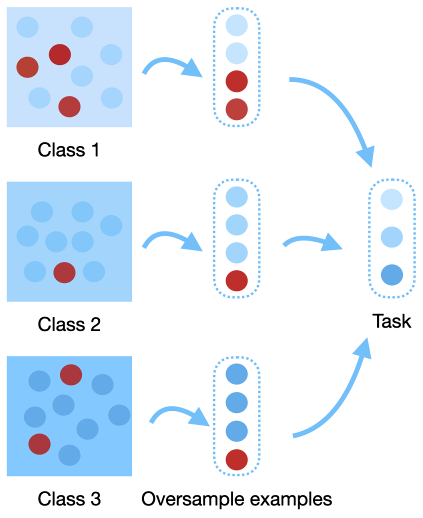


Figure 5: Example-level sampling: take a 3-way 1-shot task as an example, we first randomly sample 3 classes from current subject; then for each class, we over sample 4 examples, which may contain some noisy ones; finally, we use exampling sampling to remove the examples with higher loss and only keep one clean example for each class.

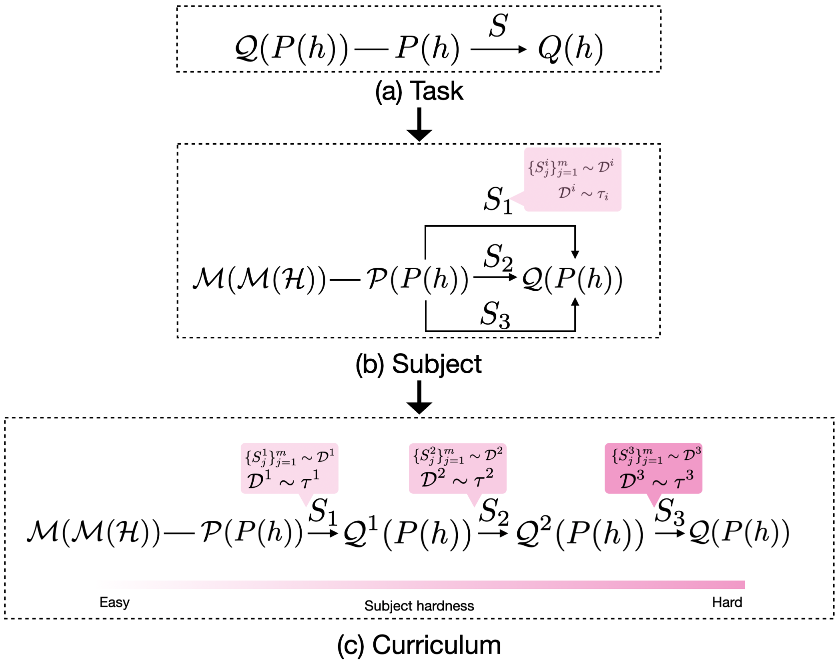

$\mathcal{Q}(P(h))$ — $P(h) \xrightarrow{S} Q(h)$
(a) Task
$\{S_j^i\}_{j=1}^m \sim \mathcal{D}^i$
$\mathcal{D}^i \sim \tau_i$
S_1
S_2
S_3
$\mathcal{M}(\mathcal{M}(\mathcal{H}))$ — $\mathcal{P}(P(h)) \rightarrow \mathcal{Q}(P(h))$
(b) Subject
$\{S_j^1\}_{j=1}^m \sim \mathcal{D}^1$
$\mathcal{D}^1 \sim \tau^1$
$\{S_j^2\}_{j=1}^m \sim \mathcal{D}^2$
$\mathcal{D}^2 \sim \tau^2$
$\{S_j^3\}_{j=1}^m \sim \mathcal{D}^3$
$\mathcal{D}^3 \sim \tau^3$
$\mathcal{M}(\mathcal{M}(\mathcal{H}))$ — $\mathcal{P}(P(h)) \xrightarrow{S_1} \mathcal{Q}^1(P(h)) \xrightarrow{S_2} \mathcal{Q}^2(P(h)) \xrightarrow{S_3} \mathcal{Q}(P(h))$
Easy
Subject hardness
Hard
(c) Curriculum

Figure 6: DCML with Hierarchical PAC-Bayesian framework. (a) For a single task, a prior model (*i.e.,* meta-model) $P(h)$ is generated from the hyper-posterior $\mathcal{Q}(P(h))$, and adapted to new posterior $Q(h)$ with observed dataset S. (b) Within the subject, the hyper-posterior $\mathcal{Q}(P(h))$ is learned with an hyper-prior $\mathcal{P}(P(h))$ which is sampled from the measure space $\mathcal{M}(\mathcal{M}(\mathcal{H}))$, and updated to the intermediate hyper-posterior $\mathcal{Q}(P(h))$ with a batch of tasks sampled from the i-th subject, which is defined by data distribution $\mathcal{D}^i$ and task environment τ_i. (c) A curriculum is formed by a series of subjects from easy to complex. The hyper-posterior $\mathcal{Q}(P(h))$ is then trained tasks sampled from the curriculum in an easy to complex order.

Proof. For DCML, it should be noted that after a sufficiently large finite number of iterations t_0, the curriculum parameter λ is no longer adjusted. We now consider the update of model parameter θ and weight parameter $\mathbf{v}$:

$$\theta^{t+1} = \theta^t - \beta_t \nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t) \mid_{\mathcal{D}^{\text{que}}}$$
$$\mathbf{v}^{t+1} = \mathbf{v}^t - \beta_t \nabla_{\mathbf{v}} \mathrm{F}(\theta^{t+1}, \mathbf{v}^t) \mid_{\mathcal{D}^{\text{que}}}$$

where $\mathcal{D}^{\text{que}}$ is the query set of a mini-batch of task sampled uniformly at random from easy class distribution $p^t(\mathcal{T})$, and $\mathrm{F}(\theta^t, \mathbf{v}^t)$ denotes the loss function in equation 1 for simplicity. Therefore, we can rewrite the update as follows

$$\theta^{t+1} = \theta^t - \beta_t \left[\nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t) + \xi^t\right]$$
$$\mathbf{v}^{t+1} = \mathbf{v}^t - \beta_t \left[\nabla_{\mathbf{v}} \mathrm{F}(\theta^{t+1}, \mathbf{v}^t) + \psi^t\right]$$

where $\xi^t = \nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t) \mid_{\mathcal{D}^{\text{que}}} - \nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t)$ and $\psi^t = \nabla_{\mathbf{v}} \mathrm{F}(\theta^{t+1}, \mathbf{v}^t) \mid_{\mathcal{D}^{\text{que}}} - \nabla_{\mathbf{v}} \mathrm{F}(\theta^{t+1}, \mathbf{v}^t)$. And we have $\mathbb{E}\left[\xi^t\right] = 0$, $\mathbb{E}\left[\psi^t\right] = 0$, and $\mathbb{E}\left[\|\xi^t\|\right] \leq \sigma^2$, $\mathbb{E}\left[\|\psi^t\|\right] \leq \rho^2$.

We can show that

$$\begin{array}{rl} & \mathrm{F}(\theta^{t+1}, \mathbf{v}^{t+1}) - \mathrm{F}(\theta^t, \mathbf{v}^t) = \mathrm{F}(\theta^{t+1}, \mathbf{v}^{t+1}) - \mathrm{F}(\theta^{t+1}, \mathbf{v}^t) + \mathrm{F}(\theta^{t+1}, \mathbf{v}^t) - \mathrm{F}(\theta^t, \mathbf{v}^t) \\ \leq & -(\beta_t - \frac{L\beta_t^2}{2}) \left\|\nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t)\right\|_2^2 - (\beta_t - L\beta_t^2)\langle \nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t), \xi^t \rangle \\ & + \frac{L\beta_t^2}{2} \left\|\xi^t\right\|_2^2 - (\beta_t - \frac{L\beta_t^2}{2}) \left\|\nabla_{\mathbf{v}} \mathrm{F}(\theta^{t+1}, \mathbf{v}^t)\right\|_2^2 \\ & -(\beta_t - L\beta_t^2)\langle \nabla_{\mathbf{v}} \mathrm{F}(\theta^{t+1}, \mathbf{v}^t), \psi^t \rangle + \frac{L\beta_t^2}{2} \left\|\psi^t\right\|_2^2 \end{array} \tag{10}$$

Taking expectation of both side, we have,

$$\begin{array}{rl} & \mathbb{E}[\mathrm{F}(\theta^{t+1}, \mathbf{v}^{t+1})] - \mathbb{E}[\mathrm{F}(\theta^t, \mathbf{v}^t)] \\ \leq & -(\beta_t - \frac{L\beta_t^2}{2})\mathbb{E}[\|\nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t)\|_2^2] + \frac{L\beta_t^2\sigma^2}{2} - (\beta_t - \frac{L\beta_t^2}{2})\mathbb{E}[\left\|\nabla_{\mathbf{v}} \mathrm{F}(\theta^{t+1}, \mathbf{v}^t)\right\|_2^2] + \frac{L\beta_t^2\rho^2}{2} \end{array} \tag{11}$$

where we have made use of $\mathbb{E}[\langle \nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t), \xi^t \rangle] = 0$ and $\mathbb{E}[\langle \nabla_{\mathbf{v}} \mathrm{F}(\theta^{t+1}, \mathbf{v}^t), \psi^t \rangle] = 0$.

Summing up Eq.(11) from $t = t_0$ to T for both sides, we have

$$\begin{array}{rl} & \sum_{t=t_0}^T \{\mathbb{E}[\mathrm{F}(\theta^{t+1}, \mathbf{v}^{t+1})] - \mathbb{E}[\mathrm{F}(\theta^t, \mathbf{v}^t)]\} = \mathbb{E}[\mathrm{F}(\theta^{T+1}, \mathbf{v}^{T+1})] - \mathrm{F}(\theta^{t_0}, \mathbf{v}^{t_0}) \\ \leq & -\sum_{t=t_0}^T (\beta_t - \frac{L\beta_t^2}{2})\mathbb{E}[\|\nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t)\|_2^2] - \sum_{t=t_0}^T (\beta_t - \frac{L\beta_t^2}{2})\mathbb{E}[\left\|\nabla_{\mathbf{v}} \mathrm{F}(\theta^{t+1}, \mathbf{v}^t)\right\|_2^2] \\ & + \sum_{t=t_0}^T \frac{L\beta_t^2\sigma^2}{2} + \sum_{t=t_0}^T \frac{L\beta_t^2\rho^2}{2} \end{array} \tag{12}$$

Taking the limit $T \to \infty$ on both side of the above equation, we have

$$\begin{array}{rl} & \sum_{t=t_0}^\infty \frac{L\beta_t^2\sigma^2}{2} + \sum_{t=t_0}^\infty \frac{L\beta_t^2\rho^2}{2} - \sum_{t=t_0}^T (\beta_t - \frac{L\beta_t^2}{2})\mathbb{E}[\|\nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t)\|_2^2] \\ & -\sum_{t=t_0}^T (\beta_t - \frac{L\beta_t^2}{2})\mathbb{E}[\left\|\nabla_{\mathbf{v}} \mathrm{F}(\theta^{t+1}, \mathbf{v}^t)\right\|_2^2] \\ \geq & \lim_{T\to\infty} \mathbb{E}[\mathrm{F}(\theta^{T+1}, \mathbf{v}^{T+1})] - \mathrm{F}(\theta^1, \mathbf{v}^1) \\ \geq & \min_{\theta,\mathbf{v}} \mathrm{F}(\theta, \mathbf{v}) - \mathrm{F}(\theta^1, \mathbf{v}^1) \geq -\infty \end{array} \tag{13}$$

Since $\sum_{t=t_0}^\infty \frac{L\beta_t^2\sigma^2}{2} < \infty$ and $\sum_{t=t_0}^\infty \frac{L\beta_t^2\rho^2}{2} < \infty$, the above inequality implies that $\sum_{t=t_0}^\infty (\beta_t - \frac{L\beta_t^2}{2})\mathbb{E}[\|\nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t)\|_2^2] + \sum_{t=t_0}^\infty (\beta_t - \frac{L\beta_t^2}{2})\mathbb{E}[\left\|\nabla_{\mathbf{v}} \mathrm{F}(\theta^{t+1}, \mathbf{v}^t)\right\|_2^2] < \infty$.

Given that the number of training samples is limited, and all second moments are upper-bounded, we introduce an upper bound $\mathbb{E}\left[\|\nabla_{\mathbf{v}} \mathrm{F}(\theta^t, \mathbf{v}^t)\|_2^2\right] + \mathbb{E}\left[\left\|\nabla_{\mathbf{v}} \mathrm{F}(\theta^{t+1}, \mathbf{v}^{t+1})\right\|_2^2\right] + \mathbb{E}\left[\|\psi^t\|_2^2\right] \leq b^2$. To show that $\lim_{t\to\infty} \mathbb{E}[\|\nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t)\|_2^2] = 0$, according to Lemma A.5 in (Mairal 2013), it suffices to show $\left|\mathbb{E}[\left\|\nabla_\theta \mathrm{F}(\theta^{t+1}, \mathbf{v}^{t+1})\right\|_2^2] - \mathbb{E}[\|\nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t)\|_2^2]\right| \leq C\beta_t$ for some constant C, which is derived as follows:

$$\begin{array}{rl} & \left|\mathbb{E}[\left\|\nabla_\theta \mathrm{F}(\theta^{t+1}, \mathbf{v}^{t+1})\right\|_2^2] - \mathbb{E}[\|\nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t)\|_2^2]\right| \\ = & \left|\mathbb{E}[(\left\|\nabla_\theta \mathrm{F}(\theta^{t+1}, \mathbf{v}^{t+1})\right\|_2 + \|\nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t)\|_2)(\left\|\nabla_\theta \mathrm{F}(\theta^{t+1}, \mathbf{v}^{t+1})\right\|_2 - \|\nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t)\|_2)]\right| \\ \leq & \mathbb{E}\left[\left|\left\|\nabla_\theta \mathrm{F}(\theta^{t+1}, \mathbf{v}^{t+1})\right\|_2 + \|\nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t)\|_2\right| \cdot \left|\left\|\nabla_\theta \mathrm{F}(\theta^{t+1}, \mathbf{v}^{t+1})\right\|_2 - \|\nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t)\|_2\right|\right] \\ \leq & \mathbb{E}\left[\left\|\nabla_\theta \mathrm{F}(\theta^{t+1}, \mathbf{v}^{t+1}) + \nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t)\right\|_2 \cdot \left\|\nabla_\theta \mathrm{F}(\theta^{t+1}, \mathbf{v}^{t+1}) - \nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t)\right\|_2\right] \\ \leq & 2aL\mathbb{E}[\left\|(\theta^{t+1}, v^{t+1}) - (\theta^t, v^t)\right\|_2] \\ \leq & 2aL\beta_t\mathbb{E}\left[\|(\nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t) - \xi^t, \nabla_{\mathbf{v}} \mathrm{F}(\theta^{t+1}, \mathbf{v}^t) - \psi^t)\|_2\right] \\ = & 2aL\beta_t\mathbb{E}\left[\sqrt{\|\nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t) - \xi^t\|_2^2} + \sqrt{\|\nabla_{\mathbf{v}} \mathrm{F}(\theta^{t+1}, \mathbf{v}^t) - \psi^t\|_2^2}\right] \\ \leq & 2aL\beta_t[\sqrt{\mathbb{E}\left[\|\nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t) - \xi^t\|_2^2\right]} + \sqrt{\mathbb{E}\left[\|\nabla_{\mathbf{v}} \mathrm{F}(\theta^{t+1}, \mathbf{v}^t) - \psi^t\|_2^2\right]}] \\ \leq & 2aL\beta_t\sqrt{\mathbb{E}\left[\|\nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t)\|_2^2\right] + \mathbb{E}\left[\|\xi^t\|_2^2\right]} + \sqrt{\mathbb{E}\left[\|\nabla_{\mathbf{v}} \mathrm{F}(\theta^{t+1}, \mathbf{v}^t)\|_2^2\right] + \mathbb{E}\left[\|\psi^t\|_2^2\right]} \\ \leq & 2a(a+b)L\beta_t \end{array} \tag{14}$$

Similarly, we can show that $\lim_{t\to\infty}\mathbb{E}[\|\nabla_v \mathrm{F}(\theta^t, \mathbf{v}^t)\|_2^2] = 0$. Consequently, we show that $\lim_{t\to\infty}\mathbb{E}[\|\nabla \mathrm{F}(\theta^t, \mathbf{v}^t)\|_2^2] = 0$ □

For DCML, it should be noted that after a sufficiently large finite number of iterations t, the model focuses on learning from hard classes, indicating that the weight parameter $v_j = 1, \forall j \in J$, and the regularizer $R(\mathbf{v})$ is a constant, and we only need to consider the update of model parameter θ.

$$\theta^{t+1} = \theta^t - \beta_t \nabla_\theta \mathrm{F}(\theta^t, \mathbf{v}^t) \mid_{\mathcal{D}^{\text{que}}}$$

where $\mathcal{D}^{\text{que}}$ is the query set of a mini-batch of task sampled uniformly at random from hard class distribution $p^t(\mathcal{T})$.

D Proof of Theoretical Results

D.1 Proof of Theorem 1-Theorem 2

To prove Theorem 1, we first introduce the following Theorem 5.

Theorem 5 (Clean meta-learning task bound). *Let Q be a base learner, and $\mathcal{P}$ be some pre-defined hyper-prior distribution over priors P. Then for any $\delta \in (0, 1]$, the following inequality holds uniformly for all hyper-posterior distribution $\mathcal{Q}$ with probability at least* $1 - \delta$*,*

$$\mathbb{E}_{P\sim\mathcal{Q}} er(Q, \mathcal{D}) \leq \mathbb{E}_{P\sim\mathcal{Q}} \hat{er}(Q, S) + \sqrt{\frac{1}{2(m-1)}(D(\mathcal{Q}||\mathcal{P}) + \mathbb{E}_{P\sim\mathcal{Q}} D(Q(S, P)||P) + \log\frac{m}{\delta})}. \tag{15}$$

We first prove Theorem 5 by applying McAllaster's classical PAC-Bayes bound (McAllester 1999), which is provided below for completeness.

Theorem 6 (McAllaster's classical PAC-Bayes bound). *Let $\mathcal{X}$ be a sample space and $\mathbb{X}$ be some distribution over $\mathcal{X}$. Define a 'loss function' $g(f, X) : \mathcal{H} \times \mathcal{X} \to [0, 1]$. Let π be some prior distribution over $\mathcal{H}$. For K random variables sampled independently sampled from $\mathbb{X}$ and any $\delta \in (0, 1]$, the following bound hold for all posteriors ρ over $\mathcal{H}$,*

$$\mathrm{E}_{X\sim\mathbb{X}}\mathrm{E}_{f\sim\rho}\, g(f, X) \leq \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}_{f\sim\rho}\, g(f, X_k) + \sqrt{\frac{1}{2(K-1)}\left(D(\rho||\pi) + \log\frac{K}{\delta}\right)} \tag{16}$$

Theorem 5 can be proved by leveraging the above theorem, which is similar to (Amit and Meir 2018). Next, we provide the proof of Theorem 1.

Proof. First, by applying the change of measure, we have

$$D(\rho||\pi) = \mathbb{E}_{f\sim\rho}\log\frac{\rho(f)}{\pi(f)} = D(\mathcal{Q}||\mathcal{P}) + \mathbb{E}_{P\sim\mathcal{Q}} D(Q||P) \tag{17}$$

Second, by defining the loss function as $g(f, X) = \ell(h, z)$ we have the following result by the application of Theorem 6 with m number in each task and probability $1 - \delta$,

$$\mathbb{E}_{P\sim\mathcal{Q}} er(Q, \mathcal{D}) \leq \mathbb{E}_{P\sim\mathcal{Q}} \hat{er}(Q, S) + \sqrt{\frac{1}{2(m-1)}(D(\mathcal{Q}||\mathcal{P}) + \mathbb{E}_{P\sim\mathcal{Q}} D(Q(S, P)||P) + \log\frac{m}{\delta})} \tag{18}$$

Third, we consider the negative impact of the noisy examples in each task. For simplicity, we only reduce the number of training samples in each task according to the noise ratio $\hat{r}$, considering that all the noisy examples are removed without consider the negative effect of them remain in training. Assuming the noise ratio $\hat{r}$ to be the same for all tasks, the number of examples in each task becomes $m(1 - \hat{r})$. By utilizing the results from the second step, we have the inequality with probability $1 - \delta$:

$$\mathbb{E}_{P\sim\mathcal{Q}} er(Q, \mathcal{D}) \leq \mathbb{E}_{P\sim\mathcal{Q}} \hat{er}(Q, S) + \sqrt{\frac{(D(\mathcal{Q}||\mathcal{P}) + \mathbb{E}_{P\sim\mathcal{Q}} D(Q(S, P)||P) + \log\frac{m(1-\hat{r})}{\delta})}{2(m(1-\hat{r})-1)}} \tag{19}$$

□

The proof of Theorem 2 can be derived from Theorem 1 using the same technique by observing n_i tasks in the subject τ_i. We define $g(f, X) = \mathbb{E}_{h\sim Q}\mathbb{E}_{z\sim\mathcal{D}}\ell(h, z)$ and obtain the following results

$$\mathbb{E}_{P\sim\mathcal{Q}} er_i(P, \tau_i) \leq \frac{1}{n_i}\sum_{j=1}^{n_i}\mathbb{E}_{P\sim\mathcal{Q}} \hat{er}_i(P, S_1, \ldots, S_{n_i}) + \sqrt{\frac{1}{2(n_i-1)}(D(\mathcal{Q}||\mathcal{P}) + \log\frac{2n_i}{\delta})}$$

$$+ \frac{1}{n_i}\sum_{j=1}^{n_i}\sqrt{\frac{1}{2(m(1-\hat{r})-1)}(D(\mathcal{Q}||\mathcal{P}) + \mathbb{E}_{P\sim\mathcal{Q}} D(Q(S_i, P)||P) + \log\frac{2n_i m(1-\hat{r})}{\delta})}$$

D.2 Proof of Theorem 3

The proof requires the following Lemma 7 whose proof can be found in (Pentina and Lampert 2015).

Lemma 7. *For any fixed algorithm A and any λ the following holds:*

$$\mathbb{E}_{\mathbb{E}_1,...,\mathbb{E}_n} \exp(\lambda(er(A) - \frac{1}{n-1}\sum_{i=2}^{n} er_i(A))) \leq \exp(\frac{\lambda^2}{2(n-1)}) \tag{20}$$

We then provide proof for Theorem 3 as follows.

Proof. First we define the intermediate multi-task expected error as follows:

$$\tilde{er}(\mathcal{Q}, \mathcal{D}_1, ..., \mathcal{D}_C) = \mathbb{E}_{P\sim\mathcal{Q}} \frac{1}{C}\sum_{i=1}^{C} er_i(P_i, \tau_i)$$

And apply Donsker-Varadhan's variational formula (Seldin et al. 2012) to change the expectation over prior, that is, for any $\lambda > 0$, we have

$$\begin{aligned} er(\mathcal{Q}) - \tilde{er}(\mathcal{Q}) \leq \frac{1}{\lambda}(KL(\mathcal{Q}||\mathcal{P}) \\ + \log \mathbb{E}_{h_0\sim\mathcal{P}}\mathbb{E}_{h_1\sim P_1}\dots\mathbb{E}_{h_C\sim P_C} \exp(\lambda(er(P,\tau) - \frac{1}{C}\sum_{i=1}^{C}\hat{er}_i(P_i,\tau_i)))) \end{aligned} \tag{21}$$

By apply Lemma 7, we have:

$$\mathbb{E}_{\mathbb{E}_1...\mathbb{E}_C} \exp(\lambda(er(P,\tau) - \frac{1}{C}\sum_{i=1}^{C}\hat{er}_i(P_i,S_i))) \leq \exp(\frac{\lambda^2}{2C}) \tag{22}$$

By Markov's inequality, with probability $1-\delta_0$, we obtain:

$$\mathbb{E}_{h_0\sim\mathcal{P}}\mathbb{E}_{h_1\sim P_1}\dots\mathbb{E}_{h_C\sim P_C} \exp(\lambda(er(P,\tau) - \frac{1}{C}\sum_{i=1}^{C}\hat{er}_i(P_i,S_i))) \leq \frac{1}{\delta_0}\exp(\frac{\lambda^2}{2C}) \tag{23}$$

Setting $\lambda = \sqrt{C}$, we obtain, with probability $1-\delta_0$:

$$er(\mathcal{Q}) - \tilde{er}(\mathcal{Q}) \leq \frac{1}{\sqrt{C}}(D(\mathcal{Q}||\mathcal{P}) + \frac{1}{2} - \log\delta_0) \tag{24}$$

Second, we use Eqn. (9) for the following result, with probability $1-\delta_i$,

$$\begin{aligned} &\tilde{er}(\mathcal{Q}) \leq \hat{er}(\mathcal{Q}) \\ &+ \frac{1}{C}\sum_{i=1}^{C}\frac{1}{n_i}\sum_{j=1}^{n_i}\sqrt{\frac{1}{2(m(1-r)-1)}(D(\mathcal{Q}||\mathcal{P}) + \mathbb{E}_{P\sim\mathcal{Q}}D(Q(S_i,P)||P) + \log\frac{2n_i m(1-r)}{\delta_i})} \\ &+ \frac{1}{C}\sum_{i=1}^{C}\sqrt{\frac{1}{2(n_i-1)}(D(\mathcal{Q}||\mathcal{P}) + \log\frac{2n_i}{\delta_i}} \end{aligned} \tag{25}$$

Third, by using the union bound, and Eqn. (24) and Eqn. (25), and set $\delta_0 = \frac{\delta}{2}$, $\delta_i = \frac{\delta}{2C}$, we obtain the final result:

$$\begin{aligned} &er(\mathcal{Q}) \leq \hat{er}(\mathcal{Q}) + \frac{1}{\sqrt{C}}(D(\mathcal{Q}||\mathcal{P}) + \frac{1}{2} - \log\frac{\delta}{2}) \\ &+ \frac{1}{C}\sum_{i=1}^{C}\frac{1}{n_i}\sum_{j=1}^{n_i}\sqrt{\frac{1}{2(m(1-r)-1)}(D(\mathcal{Q}||\mathcal{P}) + \mathbb{E}_{P\sim\mathcal{Q}}D(Q(S_i,P)||P) + \log\frac{4Cn_i m(1-r)}{\delta})} \\ &+ \frac{1}{C}\sum_{i=1}^{C}\sqrt{\frac{1}{2(n_i-1)}(D(\mathcal{Q}||\mathcal{P}) + \log\frac{4Cn_i}{\delta})} \end{aligned} \tag{26}$$

□

D.3 Interpretation of the Theoretical Results

The generalization bound in Theorem 1 expresses that the expected risk of a single noisy task is bounded by the empirical risk of a single task, and a task complexity term of the observed task that consists of the KL-divergence of the hyper-posterior and the hyper-prior, the expectation of the prior and the posterior of each task, and a term related to the noise ratio. The task complexity term converges to zero when there is an infinite number of examples in the task. In Theorem 2, the right-hand side of the subject bound consists of the expectation of the empirical risk of each task in the subject, the average of the task complexity term, and an extra subject environment complexity term that consists of the KL-divergence of the hyper-posterior and the hyper-prior. This term converges to zero if an infinite number of tasks is observed from the subject environment. Similarly, the generalization bound of the curriculum is defined by the empirical transfer error, the average of the task complexity term, the average of the subject environment complexity term, and a curriculum complexity term that converges to zero when there is an infinite number of subjects.

E Additional Experimental Details and Results

In this section, we first provide details of the datasets including datasets Omniglot, miniImageNet, FC100, Food101, miniWV, CIFAR-100N and the details of the comparison baselines. We also provide the additional 5-way 5-shot classification results for both clean few-shot task evaluation and noisy few-shot task evaluation. We then present extra ablations including the impact of class-level and example threshold, the impact of the number of proxy models, the impact of subject numbers, and some qualitative results.

E.1 Details of the Datasets

Details of the datasets are listed below:

- **Omniglot** involves 1623 characters of 50 alphabets (Lake et al. 2011). Each alphabet contains different numbers of characters, ranging from 14 to 55. Each character contains 20 examples drawn by different people. These characters are split into 1040, 258, and 325 for training, testing, and validation, respectively.
- **miniImageNet** contains 100 different classes with 600 images/class (Ravi and Larochelle 2017), split into 64, 24, and 12 for train, testing, and validation, respectively.
- **FC100** includes 100 different classes with 600 images/class (Oreshkin, López, and Lacoste 2018). These classes are further grouped into 20 super-classes. The 100 classes are split into 60, 20, and 20 for train, testing, and validation, respectively, without super-class overlapped.

We also evaluate the real-world noisy datasets:

- **miniWV** The Webvision dataset contains 2.4 million images crawled from the websites using the 1,000 concepts in ImageNet ILSVRC12 (Li et al. 2017). We only use part of the Google subset, which includes 1623 classes with 20% noisy labels. Specifically, we use the 50, 20, and 20 classes for training, validation, and testing. Each training class contains about 1000 images, and each validation and test class contains 50 clean images. We called this dataset miniWV.
- **Food101** consists of 101 food categories, with 101,000 images (Bossard, Guillaumin, and Van Gool 2014). For each class, 250 manually reviewed clean test images and 750 training images with real-world label noise are provided. The 101 categories are split into 61, 20, and 20 for training, testing, and validation. For training categories, we only use the 750 images with real-world label noise and validate and test on the 250 clean images on the test and validation categories.
- **CIFAR-100N (fine-grained)** contains 60,000 images which are with fine-grained labels in 100 classes labeled by the human experts with 40.2% human annotated noisy labels (Wei et al. 2022). We used the 60 noisy classes for training, and for the rest 40 classes, we adopt the label from the original clean CIFAR100 dataset (Krizhevsky, Hinton et al. 2009) and use the rest 20, 20 for test and validation, respectively.

E.2 Details of different synthetic noise

The details of the symmetric noise are listed as follows:

- **miniImagenet**: in the total 64 meta-training classes, we randomly select 10 classes and randomly select 50 images out of a total of 600 images and randomly assigned a different label in the rest classes. In total, we have 14 classes containing 50 noisy images, 10 classes containing 100 noisy images, 10 classes containing 200 noisy images, 10 classes containing 300 noisy images, 10 classes containing 400 noisy images, and 10 classes containing 500 noisy images. The noise ratio is around 45.57%.
- **FC100**: in the total 60 meta-training classes, we randomly select 10 classes and randomly select 50 images out of a total of 600 images and randomly assigned a different label in the rest classes. In total, we have 10 classes containing 50 noisy images, 10 classes containing 100 noisy images, 10 classes containing 200 noisy images, 10 classes containing 300 noisy images, 10 classes containing 400 noisy images, and 10 classes containing 500 noisy images. The noise ratio is around 43%.

- **Omniglot**: in the total 1040 meta-training classes, we randomly select 500 classes and randomly select 10 images out of a total of 20 images and randomly assigned a different label in the rest classes. In total, we have 200 classes containing 8 noisy images, 200 classes containing 5 noisy images, 100 classes containing 3 noisy images, and the remaining 40 classes staying clean. The noise ratio is around 37.98%.

The details of the asymmetric noise are listed as follows:

- **miniImagenet**: in this dataset, we have 64 classes in meta-training, among which 28 classes have similar classes to each other, for example, there are nine different types of dogs and three different types of birds. 200 out of 600 examples from similar classes are randomly selected and assigned with the label of the similar classes. The noise ratio is around 15.5%.
- **FC100**: in this dataset, we have 60 meta-train classes where 49 classes have similar classes, and some of them have multiple similar classes. For example, there are five different types of flowers, and six different types of trees, and five different types of fish. 200 out of 600 examples from similar classes are randomly selected and assigned with the label of the similar classes. The noise ratio is around 27.2%.
- **Omniglot**: in the total of 1040 meta-train classes, 770 characters are similar to each other. 6 out of 20 examples from similar classes are randomly selected and assigned with the label of the similar classes. The noise ratio is around 22.21%.

The details of the BackgroundFlip noise are listed as follows:

- **miniImagenet**: in each meta-training class, 60 out of 600 images are randomly sampled and their label are flipped to a single background class to form a new background class. The background class therefore contains 3840 images, and the noise ratio is 10%.
- **FC100**: in each meta-training class, 60 out of 600 images are randomly sampled and their label are flipped to a single background class to form a new background class. The background class therefore contains 3600 images, and the noise ratio is 10%.
- **Omniglot**: in each meta-training class, 2 out of 20 images are randomly sampled and their label is flipped to a single background class to form a new background class. The background class therefore contains 2080 images, and the noise ratio is 10%.

The details of a mixture of label noises including symmetric, asymmetric, and BackgroundFlip:

- **miniImagenet**: in this dataset, we first add the asymmetric label noise to the 28 similar class pairs following the same setting of the aforementioned symmetric label noise. For the rest 36 not similar classes, we apply 200 symmetric label noise to each of the classes. Finally, we randomly sample 10 images of each of the classes in the training dataset to form one single background class. The noise ratio is around 31.8%.
- **FC100**: in this dataset, we first add the asymmetric label noise to the 49 similar class pairs following the same setting of the aforementioned symmetric label noise. For the rest 11 not similar classes, we apply 200 symmetric label noise to each of the classes. Finally, we randomly sample 10 images of each of the classes in the training dataset to form one single background class. The noise ratio is around 34.9%.
- **Omniglot**: in this dataset, we first add the asymmetric label noise to the 770 similar class pairs following the same setting of the aforementioned symmetric label noise. For the rest non-similar classes, we apply 10 symmetric label noise to 300 classes, 8 symmetric label noise to 100 classes, 5 symmetric label noise to 100 classes, 3 symmetric label noise to 100 classes, and the rest classes remain clean. Finally, we randomly sample 1 image of each of the classes in the training dataset to form one single background class. The noise ratio is around 49.32%.

E.3 Details of the comparative methods

The details of the comparative baselines are listed as follows:

- MAML+spld (Jiang et al. 2014b), spld is an self-paced learning (SPL) variant that leverages a general regularization term giving preference to both easy and diverse samples to form the curriculum. We implement MAML+spld by selecting the easy and diverse samples for the few-shot tasks in MAML.
- MAML+focal, focal (Lin et al. 2017) develops a sample weighting scheme that prioritizes relatively higher training losses. We implement MAML+focal by providing focal weight to the example in each task of MAML.
- MAML+dih (Zhou, Wang, and Bilmes 2020a) is another SPL variant that selects examples with less dynamic instance hardness (DIH) to form the curriculum. Instead of using a sample's instantaneous hardness, DIH utilizes the exponential moving average of a sample's instantaneous hardness over the training history. We implement MAML+dih by forming curriculum based on the DIH values for MAML model.
- CMAML (Agrawal, Squire et al. 2021) is a curriculum meta-learning method that forms its curriculum by gradually reducing the number of training examples in the support set to increase the training hardness for the model.
- RNNP (Mazumder, Singh, and Namboodiri 2021) is a robust meta-learning method, which forms robust prototypes during evaluation to enhance robustness to label noise in few-shot learning.

- TraNFS (Mazumder, Singh, and Namboodiri 2021) uses weighted prototypes for learning a model robust to label noise. In our implementation, the spatial median prototypes are utilized based on ProtoNets.
- CT (Luo, Xu, and Xu 2022) applies a simple channel transformation function during testing, which can greatly improve the generalization ability of learned image representations.
- RapNets (Lu et al. 2020) designs a correlation module to compute the correlation features between the embeddings of congener support data, and a attentive module to compute the weight of the congener support data.
- IDEAL (An et al. 2023) aggregates prototypes with intra-class and inter-class instance reweighting as well as scale the per-class prediction by fusing two spatial metrics.

For a fair comparison, we do not consider the methods that require a separate clean validation set such as L2RW (Ren et al. 2018) and MWN (Shu et al. 2019). Other methods such as (Chen et al. 2021) which also used the term dual-level, while the two levels in that work refer to very different concepts, is also not in the consideration of comparison. Different from DCML which jointly considers class- and example-level curricula, (Chen et al. 2021) leverages both city- and user-level hardness to help with the next POI recommendation. As a result, the primary focuses of these two works are fundamentally different from each other.

E.4 Details of training and hyperparameters setting

For all curriculum-based algorithms, it starts with a warm start training using all data for the first 5,000 episodes. After that, it begins to build a curriculum and using only easy training data using different selection criteria. The total number of training episodes is 40,000 for all experiments and all the methods come into effect after 5,000 iterations. For DCML, we save the proxy models for every 2,500 episodes for the first 30,000 episodes (at the last 10,000, the model might start to overfit the noisy data, which might affect the effectiveness of the proxy model), the total number of proxy models is set as 12 if without additional specified. After warm start, in the class-level, we first select 60% of easy classes and then increase the threshold by 10% for every 5,000 episodes, the total number of subjects is set as 8 if without additional specified. In the example-level, we sample one out of five examples in the support set for 5-way 1-shot (5w1s) classification tasks and five out of ten for that of 5-way 5-shot (5w5s) case. For other hyperparameters, such as meta learning rate and task learning rate, we follow the experimental setup of (Finn, Abbeel, and Levine 2017) unless explicitly stated.

E.5 Additional results for the mixture noise

In Tab. 7, we present the results of 5 way 1 shot experiment of mixture label noise during meta-training, under the clean support set and 40% synthetic label noise on the support set during meta-test. The results are consistent with the other three synthetic label noises presented in the main text. Our method outperforms other baselines and shows stable performance across different datasets.

Table 7: 5-way 1-shot few-shot classification on datasets with mixture noise

Method	miniImageNet	FC100	Omniglot	miniImageNet	FC100	Omniglot
Noise Type	w/o noise at support set at meta test			w/ noise at support set at meta test		
MAML(2017)	0.4206 ± 0.0063	0.3548 ± 0.0066	0.9323 ± 0.0018	0.3397 ± 0.0027	0.3061 ± 0.0036	0.6585 ± 0.0302
MAML+spld(2014)	0.4188 ± 0.0155	0.3438 ± 0.0004	0.9252 ± 0.0070	0.3318 ± 0.0105	0.2876 ± 0.0050	0.6534 ± 0.0047
MAML+focal(2017)	0.4235 ± 0.0066	0.3538 ± 0.0065	0.9297 ± 0.0057	0.3311 ± 0.0025	0.3033 ± 0.0069	0.6586 ± 0.0092
MAML+dih(2020)	0.4261 ± 0.0130	0.3570 ± 0.0074	0.9223 ± 0.0033	0.3365 ± 0.0090	0.2953 ± 0.0005	0.6554 ± 0.0045
CMAML(2021)	0.4352 ±0.0106	0.3467 ± 0.0079	0.9132 ± 0.0030	0.3344 ± 0.0039	0.2887 ± 0.0020	0.6580 ± 0.0051
RNNP(2021)	0.4604 ± 0.0161	0.4167 ± 0.0157	0.9532 ± 0.0134	0.3443 ± 0.0089	0.2967 ± 0.0042	0.6680 ± 0.0087
TraNFS(2022)	0.4549 ± 0.0513	0.4014 ± 0.0227	0.9518 ± 0.0014	0.3496 ± 0.0077	0.3009 ± 0.0038	0.6617 ± 0.0056
CT(2022)	0.4204 ± 0.0025	0.3502 ± 0.0073	0.9340 ± 0.0010	0.3336 ± 0.0012	0.2978 ± 0.0033	0.6669 ± 0.0049
IDEAL(2023)	0.4901 ± 0.0209	0.4553 ± 0.0090	0.9566 ± 0.0010	0.3207 ± 0.0200	0.3100 ± 0.0190	0.6609 ± 0.0298
DCML	**0.5019** ± 0.0113	**0.4714** ± 0.0073	**0.9629** ± 0.0025	**0.3558** ± 0.0057	**0.3149** ± 0.0039	**0.6716** ± 0.0033

E.6 Additional results for the 5-way 5-shot few-shot classification

From Tab. 8 and Tab. 9, we can draw similar conclusions for the 5-way 5-shot experiment results as the 5-way 1-shot ones. That is, DCML outperforms other methods in most cases. On the one hand, MAML+focal, TraNFS have pretty good performance over Asymmetric, Symmetric and Background Flip noise on miniImageNet dataset. However, in the most complicated mixture noise, and real-world noisy datasets, our method ranks the top, showing that our method is better when dealing with complicated and real-world situations, which is practically more useful. On the other hand, our method's performance achieve the highest average rank across all experiments over all datasets, which shows consistent superiority across different situations.

Table 8: 5-way 5-shot few-shot classification test accuracy on datasets with synthetic noise

Datasets	miniImageNet	FC100	Omniglot
Noise Type		Asymmetric	
MAML(2017)	0.6242 ± 0.0014	0.4689 ± 0.0057	0.9842 ± 0.0012
MAML+spld(2014)	0.5390 ± 0.0208	0.4020 ± 0.0266	0.9853 ± 0.0005
MAML+focal(2017)	0.6245 ± 0.0031	0.4779 ± 0.0034	0.9829 ± 0.0036
MAML+dih(2020)	0.5220 ± 0.0418	0.4359 ± 0.0200	0.9842 ± 0.0006
RapNets(2020)	0.5874 ± 0.0206	0.4638 ± 0.0081	0.9788 ± 0.0026
CMAML(2021)	0.6228 ± 0.0156	0.4723 ± 0.0090	0.9739 ± 0.0036
RNNP(2021)	0.6128 ± 0.0256	0.4753 ± 0.0083	0.9742 ± 0.0012
TraNFS(2022)	**0.6336** ± 0.0172	0.4838 ± 0.0294	0.9840 ± 0.0017
CT(2022)	0.6235 ± 0.0042	0.4676 ± 0.0045	0.9850 ± 0.0024
IDEAL(2023)	0.6308 ± 0.0200	0.4800 ± 0.0193	09822 ± 0.0085
DCML	0.6071 ± 0.0097	**0.4851** ± 0.0374	**0.9853** ± 0.0017
Noise Type		Symmetric	
MAML(2017)	0.5493 ± 0.0177	0.4449± 0.0061	0.9743 ± 0.0032
MAML+spld(2014)	0.5196 ± 0.0074	0.4533 ± 0.0046	0.9734 ± 0.0008
MAML+focal(2017)	**0.5626** ± 0.0088	0.4743 ± 0.0038	0.9729 ± 0.0024
MAML+dih(2020)	0.5156 ± 0.0044	0.4566 ± 0.0090	0.9734 ± 0.0036
RapNets(2020)	0.5086 ± 0.0401	0.4471 ± 0.0281	0.9669 ± 0.0266
CMAML(2021)	0.5549 ± 0.0073	0.4625 ± 0.0129	0.9630 ± 0.0033
RNNP(2021)	0.5249 ± 0.0030	0.4913 ±0.0080	0.9713 ± 0.0015
TraNFS(2022)	0.5059 ± 0.0077	0.4855 ± 0.0175	0.9739 ± 0.00172
CT(2022)	0.5409 ± 0.0149	0.4401 ± 0.0112	0.9750 ± 0.0020
IDEAL(2023)	0.5608 ± 0.0373	0.4888 ± 0.0160	0.9764 ± 0.0288
DCML	0.5354 ± 0.0041	**0.4917** ± 0.0133	**0.9784** ± 0.0014
Noise Type		Background Flip	
MAML(2017)	0.6246 ± 0.0140	0.4572± 0.0132	**0.9844** ± 0.0015
MAML+spld(2014)	0.5663 ± 0.0128	0.4288 ± 0.0266	0.9800 ± 0.0018
MAML+focal(2017)	0.6273 ± 0.0090	0.4760 ± 0.0059	0.9764 ± 0.0036
MAML+dih(2020)	0.5765 ± 0.0121	0.4363 ± 0.0031	0.9798 ±0.0012
RapNets(2020)	0.6278 ± 0.0150	0.4630± 0.0107	0.9769 ± 0.0115
CMAML(2021)	0.6292 ± 0.0067	0.4460 ± 0.0139	0.9652 ± 0.0039
RNNP(2021)	0.6192 ± 0.0057	0.4604 ± 0.0093	0.9735 ± 0.0026
TraNFS(2022)	**0.6336** ± 0.0172	0.4902 ± 0.0062	0.9836 ± 0.0002
CT(2022)	0.6174 ± 0.0159	0.4519 ± 0.0133	0.9758 ± 0.0031
IDEAL(2023)	0.6332 ± 0.0207	0.4898 ± 0.0380	0.9811 ± 0.0041
DCML	0.6255 ± 0.0345	**0.4903** ± 0.0058	0.9843 ± **0.0006**
Noise Type		Mixture	
MAML(2017)	0.5928 ± 0.0084	0.4666 ± 0.0115	0.9773 ± 0.0028
MAML+spld(2014)	0.5365 ± 0.0163	0.4049 ± 0.0123	0.9800 ± 0.0018
MAML+focal(2017)	0.5948 ± 0.0027	0.4755 ± 0.0099	0.9764 ± 0.0036
MAML+dih(2020)	0.5551 ± 0.0104	0.4570 ± 0.0086	0.9798 ± 0.0012
RapNets(2020)	0.5600 ± 0.0209	0.4436 ± 0.0183	0.9788 ± 0.0137
CMAML(2021)	0.5947 ± 0.0092	0.4760 ± 0.0070	0.9652 ± 0.0039
RNNP(2021)	0.5467 ± 0.0016	0.4516 ± 0.1793	0.9735 ± 0.0026
TraNFS(2022)	0.5749 ± 0.0619	0.4654 ± 0.0062	0.9817 ± 0.0025
CT(2022)	0.5899 ± 0.0049	0.4596 ± 0.0062	0.9758 ± 0.0031
IDEAL(2023)	**0.6010** ± 0.0255	0.4722 ± 0.0110	0.9802 ± 0.0023
DCML	0.5984 ± 0.0215	**0.4988** ± 0.0081	**0.9843** ± 0.0006

Table 9: 5-way 5-shot few-shot classification test accuracy on datasets with real-world noise

Datasets	CIFAR-100N	Food101FS	miniWV
MAML(2017)	0.7305 ± 0.0093	0.5784 ± 0.0084	0.4191 ± 0.0093
MAML+spld(2014)	0.6158 ± 0.0712	0.5027 ± 0.0142	0.3923 ± 0.0021
MAML+focal(2017)	0.7313 ± 0.0070	0.5761 ± 0.0038	0.4268 ± 0.0052
MAML+dih(2020)	0.6402 ± 0.0308	0.4939 ± 0.0130	0.3891 ±0.0063
RapNets(2020)	0.7035 ± 0.0119	0.5670 ± 0.0121	0.4101 ± 0.0165
CMAML(2021)	0.7203 ± 0.0209	0.5838 ± 0.0234	0.4154 ± 0.0137
RNNP(2021)	0.7051 ± 0.0026	0.5843 ± 0.0064	0.4291 ± 0.0100
TraNFS(2022)	0.7339 ± 0.0041	0.5637 ± 0.0615	0.4230 ± 0.0260
CT(2022)	0.7299 ± 0.0094	0.5723 ± 0.0081	0.4124 ± 0.0111
IDEAL(2023)	**0.7410** ± 0.0209	0.5790 ± 0.0109	0.4299 ± 0.0098
DCML	0.7398 ± 0.0105	**0.5973** ± 0.0066	**0.4314** ± 0.0070

E.7 Additional results for noise evaluation on few-shot classification

In this section, we present extra experimental results for 5-way 5-shot few-shot classification task with 40% symmetric noise in each task during evaluation in Tab. 10 and Tab. 11. From these results, we first observe a drastic performance drop, showing that few-shot learning tasks can be greatly affected by the noisy data in each task. Then we observe that our method is consistently outperforming other comparative baselines, indicating the robustness of our method under extreme label corruption.

Table 10: 5-way 5-shot few-shot classification test accuracy with evaluation symmetric noise on synthetic noisy datasets

Datasets	miniImageNet	FC100	Omniglot
Noise Type		Asymmetric	
MAML(2017)	0.4201 $\pm$ 0.0211	0.3841 $\pm$ 0.0102	0.7361 $\pm$ 0.0100
MAML+spld(2014)	0.3604 $\pm$ 0.0024	0.3254 $\pm$ 0.0095	0.7535 $\pm$ 0.0051
MAML+focal(2017)	0.4213 $\pm$ 0.0143	0.3558 $\pm$ 0.0117	0.7553 $\pm$ 0.0141
MAML+dih(2020)	0.3646 $\pm$ 0.0068	0.3397 $\pm$ 0.0098	0.7482 $\pm$ 0.0052
RapNets(2020)	0.4573 $\pm$ 0.0039	0.3789 $\pm$ 0.0049	0.7542 $\pm$ 0.0030
CMAML(2021)	0.4461 $\pm$ 0.0149	0.3716 $\pm$ 0.0143	0.7350 $\pm$ 0.0049
RNNP(2021)	0.4611 $\pm$ 0.0249	0.3816 $\pm$ 0.0243	0.7350 $\pm$ 0.0049
TraNFS(2022)	0.4633 $\pm$ 0.0036	0.3667 $\pm$ 0.0063	0.7478 $\pm$ 0.0036
CT(2022)	0.4420 $\pm$ 0.0100	0.3584 $\pm$ 0.0059	0.7596 $\pm$ 0.0055
IDEAL(2023)	0.4700 $\pm$ 0.0333	0.3888 $\pm$ 0.0080	0.7666 $\pm$ 0.0049
DCML	**0.4721** $\pm$ 0.0092	**0.3921** $\pm$ 0.0073	**0.7687** $\pm$ 0.0045
Noise Type		Symmetric	
MAML(2017)	0.3770 $\pm$ 0.0098	0.3500 $\pm$ 0.0209	0.7258 $\pm$ 0.0100
MAML+spld(2014)	0.3455 $\pm$ 0.0111	0.3334 $\pm$ 0.0114	0.7358 $\pm$ 0.0021
MAML+focal(2017)	0.3887 $\pm$ 0.0088	0.3635 $\pm$ 0.0143	0.7438 $\pm$ 0.0045
MAML+dih(2020)	0.3444 $\pm$ 0.0071	0.3331 $\pm$ 0.0154	0.7330 $\pm$ 0.0058
RapNets(2020)	0.3696 $\pm$ 0.0029	0.3606 $\pm$ 0.0040	0.7400 $\pm$ 0.0039
CMAML(2021)	0.3867 $\pm$ 0.0151	0.3512 $\pm$ 0.0085	0.7209 $\pm$ 0.0033
RNNP(2021)	0.3767 $\pm$ 0.0251	0.3712 $\pm$ 0.0095	0.7407 $\pm$ 0.0053
TraNFS(2022)	0.3856 $\pm$ 0.0096	0.3801 $\pm$ 0.0091	0.7398 $\pm$ 0.0066
CT(2022)	0.3695 $\pm$ 0.0108	0.3408 $\pm$ 0.0123	0.7384 $\pm$ 0.0018
IDEAL(2023)	0.3880 $\pm$ 0.0200	0.3711 $\pm$ 0.0200	0.7444 $\pm$ 0.0085
DCML	**0.3994** $\pm$ 0.0132	**0.3860** $\pm$ 0.0069	**0.7554** $\pm$ 0.0118
Noise Type		Background Flip	
MAML(2017)	0.4433 $\pm$ 0.0112	0.3501 $\pm$ 0.0209	0.7505 $\pm$ 0.0100
MAML+spld(2014)	0.3688 $\pm$ 0.0080	0.3420 $\pm$ 0.0130	0.7367 $\pm$ 0.0031
MAML+focal(2017)	0.4277 $\pm$ 0.0099	0.3440 $\pm$ 0.0130	0.7710 $\pm$ 0.0032
MAML+dih(2020)	0.3683 $\pm$ 0.0012	0.3363 $\pm$ 0.0190	0.7393 $\pm$ 0.0036
RapNets(2020)	0.3801 $\pm$ 0.0030	0.3439 $\pm$ 0.0039	0.7458 $\pm$ 0.0030
CMAML(2021)	0.4687 $\pm$ 0.0149	0.3660 $\pm$ 0.0032	0.7517 $\pm$ 0.0036
RNNP(2021)	0.4439 $\pm$ 0.0049	0.3560 $\pm$ 0.0132	0.7500 $\pm$ 0.0064
TraNFS(2022)	0.4523 $\pm$ 0.0078	0.3609 $\pm$ 0.0091	0.7498 $\pm$ 0.0052
CT(2022)	0.4224 $\pm$ 0.0044	0.3458 $\pm$ 0.0138	0.7504 $\pm$ 0.0028
IDEAL(2023)	0.4650 $\pm$ 0.0020	0.3600 $\pm$ 0.0039	0.7601 $\pm$ 0.0050
DCML	**0.4842** $\pm$ 0.0106	**0.3788** $\pm$ 0.0149	**0.7771** $\pm$ 0.0058
Noise Type		Mixture	
MAML(2017)	0.3960 $\pm$ 0.0109	0.3430 $\pm$ 0.0101	0.7399 $\pm$ 0.0050
MAML+spld(2014)	0.3542 $\pm$ 0.0152	0.3384 $\pm$ 0.0072	0.7442 $\pm$ 0.0017
MAML+focal(2017)	0.3988 $\pm$ 0.0118	0.3438 $\pm$ 0.0071	0.7498 $\pm$ 0.0018
MAML+dih(2020)	0.3605 $\pm$ 0.0054	0.3270 $\pm$ 0.0032	0.7436 $\pm$ 0.0023
RapNets(2020)	0.3892 $\pm$ 0.0036	0.3379 $\pm$ 0.0030	0.7490 $\pm$ 0.0032
CMAML(2021)	0.4167 $\pm$ 0.0038	0.3608 $\pm$ 0.0063	0.7261 $\pm$ 0.0071
RNNP(2021)	0.4067 $\pm$ 0.0138	0.3708 $\pm$ 0.0161	0.7513 $\pm$ 0.0172
TraNFS(2022)	0.4122 $\pm$ 0.0023	0.3668 $\pm$ 0.0090	0.7530 $\pm$ 0.0132
CT(2022)	0.3966 $\pm$ 0.0172	0.3462 $\pm$ 0.0089	0.7406 $\pm$ 0.0036
IDEAL(2023)	**0.4406** $\pm$ 0.0088	0.3790 $\pm$ 0.0033	**0.7677** $\pm$ 0.0056
DCML	0.4289 $\pm$ 0.0095	**0.3880** $\pm$ 0.0157	0.7624 $\pm$ 0.0104

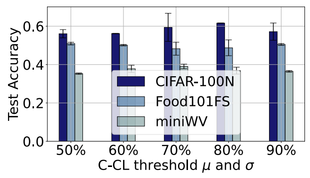
E.8 Additional ablation study

In this section, we provide the experimental results to study the impact of the class-level thresholds μ and σ, example-level threshold λ, the impact of the number of proxy models, and the impact of total subject numbers C.

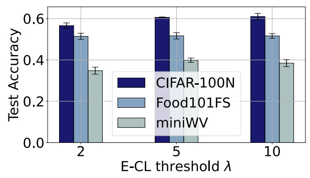
Impact of C-CL threshold μ and σ In this ablation, we've tested the impact of the percentage of classes we used in the first subject, which is defined by the values of the class-level thresholds μ and σ. Following the setting of curriculum learning (Bengio et al. 2009) and self-paced learning (Kumar, Packer, and Koller 2010), we simplified the choices of the threshold values of μ and σ as the percentage of classes. We have tried with 50%, 60%, 70%, 80%, and 90% in different trials. In this series of trials, the

Table 11: 5-way 5-shot few-shot classification test accuracy with evaluation symmetric noise on real-world noisy datasets

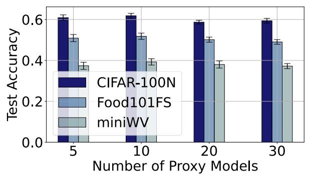
Datasets	CIFAR-100N	Food101FS	miniWV
MAML(2017)	0.5207 $\pm$ 0.0062	0.4333 $\pm$ 0.0088	0.3100 $\pm$ 0.0098
MAML+spld(2014)	0.4121 $\pm$ 0.0082	0.3346 $\pm$ 0.0061	0.2959 $\pm$ 0.0034
MAML+focal(2017)	0.4963 $\pm$ 0.0182	0.3812 $\pm$ 0.0114	0.3225 $\pm$ 0.0039
MAML+dih(2020)	0.4159 $\pm$ 0.0117	0.3355 $\pm$ 0.0009	0.2949 $\pm$ 0.0060
RapNets(2020)	0.5180 $\pm$ 0.0101	0.4010 $\pm$ 0.0126	0.3001 $\pm$ 0.0044
CMAML(2021)	0.5051 $\pm$ 0.0083	0.3867 $\pm$ 0.0154	0.3172 $\pm$ 0.0095
RNNP(2021)	0.5036 $\pm$ 0.0106	0.3967 $\pm$ 0.0541	0.3272 $\pm$ 0.0395
TraNFS(2022)	0.5256 $\pm$ 0.0618	0.3966 $\pm$ 0.0063	0.3238 $\pm$ 0.0230
CT(2022)	0.5136 $\pm$ 0.0406	0.3766 $\pm$ 0.0071	0.3124 $\pm$ 0.0119
IDEAL(2023)	0.5255 $\pm$ 0.0216	0.4301 $\pm$ 0.0203	0.3209 $\pm$ 0.0190
DCML	**0.5304** $\pm$ 0.0307	**0.4422** $\pm$ 0.0483	**0.3366** $\pm$ 0.0230



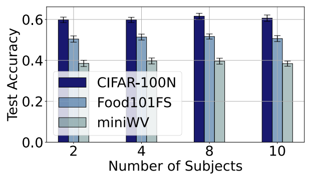
(a)



(b)



(c)



(d)

Figure 7: Model test accuracy with different settings. (a) DCML with different C-CL threshold values; (b) DCML with different E-CL threshold values; (c) DCML with different numbers of proxy models; (d) DCML with different numbers of subjects.

number of proxy models is fixed at 12, and the number of subjects is fixed at 4. And the example-level threshold is fixed as 5. As shown in Fig. 7a, when the class-level threshold values of μ and σ are set as 60% or 70% of classes, the model achieves the best performance. According to the result, setting the correct μ and σ is very important to the performance of the model. Inappropriate thresholds μ and σ are likely to lead to inferior performance. Different datasets also require different thresholds. This is due to the various data distribution and noise distributions in different datasets. Incorporating noisy data or providing insufficient data during early training might be the reason that leads to inferior model performance. Setting the class-level threshold as 60% or 70% provides the model enough knowledge to learn to build a curriculum and leaves plenty of potential to re-organize. Therefore, it serves as the best choice for different datasets.

Impact of the E-CL threshold λ In this ablation, we tested the impact of the E-CL threshold λ by setting different sampling pool sizes for the support set. Similar to the class-level thresholds μ and σ, we use percentage instead of loss values following (Bengio et al. 2009; Kumar, Packer, and Koller 2010), which makes the hyper-parameter setting direct and easy. Specifically, for the support set, we first formulate a N-way K'-shot task and then sample K examples out of K' examples by computing the loss values of the K' examples. For a 5-way 1-shot few-shot classification task, we've tested K' equal to 2, 5, and 10 for the support set and used 1 least loss example to update the task-specific model. The experiments are conducted on three real-world noisy datasets. In this series of trials, the number of proxy models is fixed at 12, and the number of subjects is fixed at 4. And the class-level threshold is fixed at 70%. As shown in Fig. 7b, the performance increase as K' increases. However, the increasing K' will subsequently increase the computational complexity. Since the performance of $K' = 5$ is close to that of $K' = 10$, we choose $K' = 5$ as our optimal choice for efficiency consideration.

Impact of the proxy models' number In this ablation, we've tested the impact of the number of proxy models. We have tried with 5, 10, 20, and 30 proxy models in different trials. The number of subjects is fixed at 4. The class-level threshold is set as 70%. And the example-level threshold is fixed at 5. From Fig. 7c, we observe that the performance of DCML doesn't change much as the number of the proxy model varies. In general, when the number is set to 10, the model achieves the best performance. Although there is no extra computation required for the proxy models, fewer proxy models require fewer storage, which is more memory efficient. Since the performance won't be affected too much by the number of proxy models, we can use fewer proxy models during training to achieve comparable performance.

Impact of the subjects number C In this ablation, we've tested the impact of the number of subjects. We have tried with 2, 4, 8, and 10 subjects in different trials. The number of proxy models is fixed at 12. The class-level threshold is set as 70%. And the example-level threshold is fixed at 5. From Fig. 7d, we observe that when the number of subjects is set around 4 or 8, the model achieves the best performance.

Generalization to metric-based method. The baselines and DCML are all implemented based on MAML to achieve a fair comparison. MAML is a well-recognized representative meta-learning method for FSL, which is proven to be effective and has been extensively studied. We've conducted an ablation on a more recent metric-based meta-learning method: DeepEMD (Zhang et al. 2020), and implement DCML based on DeepEMD. The comparison results of three real-world noisy datasets are presented in the table below. From the results, we observe that label noise has negatively impacted the generalization ability of the metric-based meta-learning model, especially when there is evaluation label noise. On the other hand, DCML has achieved consistent advantage over DeepEMD across different datasets and settings.

Table 12: Test accuracies on real-world noisy datasets

Method		CIFAR-100N	Food101FS	miniWV
DeepEMD	5w1s	0.4874 $\pm$ 0.0066	0.4001 $\pm$ 0.0069	0.3004 $\pm$ 0.0090
DCML		0.5008 $\pm$ 0.0045	0.4311 $\pm$ 0.0071	0.3501 $\pm$ 0.0088
DeepEMD	5w5s	0.6021 $\pm$ 0.0089	0.4709 $\pm$ 0.0032	0.3700 $\pm$ 0.0065
DCML		0.6432 $\pm$ 0.0070	0.5100 $\pm$ 0.0053	0.4069 $\pm$ 0.0089
DeepEMD	5w1s 40% sym noise	0.3941 $\pm$ 0.0078	0.3144 $\pm$ 0.0050	0.2545 $\pm$ 0.0077
DCML		0.4377 $\pm$ 0.0060	0.3418 $\pm$ 0.0064	0.2903 $\pm$ 0.0101
DeepEMD	5w5s 40% sym noise	0.4568 $\pm$ 0.0091	0.3699 $\pm$ 0.0040	0.3201 $\pm$ 0.0045
DCML		0.4909 $\pm$ 0.0083	0.4090 $\pm$ 0.0038	0.3417 $\pm$ 0.0091

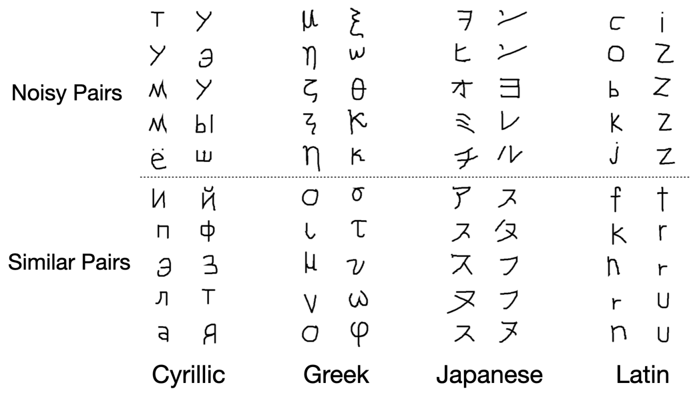


Figure 8: Noisy/similar pairs selected using the CP-metrics from different languages in the Omniglot dataset.

Qualitative Results In Fig. 8, we conduct the experiment on Omniglot dataset with symmetric label noise and we show some noisy class pairs and some similar class pairs chosen by the DCML framework using class-level CL from the same language including Cyrillic, Greek, Japanese and Latin. The results show that the similar pairs are visually similar characters in different languages in the Omniglot dataset, while noisy pairs are more random. For example, in the noisy pairs of Latin, the character z is assigned to different characters randomly, including o, b, j and k. As for similar pairs, some characters also appear repeatedly due to their similarity with multiple characters, for instance, the characters u, r, n in Latin are similar to each other. This result shows that our selection criteria are consistent and able to select those similar class pairs from the noisy data. By forming tasks with these similar pairs, we manage to build more informative tasks.