A. Proof of the Lemma 1

Lemma 2. *Let the sample volume of the dataset be m, the number of classes be C, the sample number per class be balanced, and the entropy consumed by annotation be H. Then, the expectation of correct labeled samples, i.e., m', is given by*

$$m' = \frac{m}{C} e^{\frac{H}{m}} \qquad \textit{s.t.} \;\; H \in [0, mlogC]. \tag{12}$$

Proof. Since the numbers of samples per class are balanced, the categories for each sample have C equal possibilities. The entropy of one sample is calculated by

$$\begin{aligned} H_i &= -\sum_{i=1}^{C} \frac{1}{C} \log \frac{1}{C} \\ &= \log C. \end{aligned}$$

Given that the samples demonstrate I.I.D., we can determine that the total entropy equals

$$H_{sum} = m \log C.$$

When the reduction of entropy due to annotation is H, the total entropy becomes

$$H_{sum} - H.$$

Then, after annotation, the number of possible states for each sample can be written as

$$C' = e^{\frac{H_{sum} - H}{m}}.$$

Then, the probability of correct labeling is

$$\begin{aligned} p =& \frac{1}{e^{\frac{H_{sum}-H}{m}}} \\ =& \frac{1}{e^{\frac{m \log C - H}{m}}} \\ =& e^{\frac{H}{m}} / C. \end{aligned}$$

The expectation of correct labeling samples is

$$m' = \frac{m}{C} e^{\frac{H}{m}}.$$

□

B. Proof of Theorem 1, Theorem 2, and Corollary 1

Definition 1 (Uniform stability of single-task learning)**.** *A single-task learning algorithm A has uniform stability β with respect to the loss function l if the following holds for any training set S and for all $i \in \{1, \dots, m\}$, where $S^{\setminus i}$ denotes the dataset with the i-th sample removed:*

$$|\hat{L}(A(S), S) - \hat{L}(A(S^{\setminus i}), S)| \leq \beta.$$

Definition 2 (Uniform stability of meta-learning)**.** *A meta-algorithm $\boldsymbol{\mathcal{A}}$ has uniform stability $\tilde{\beta}$ with respect to the loss function l if the following holds for any meta-sample $\boldsymbol{\mathcal{S}}$ and for all $i \in \{1, \dots, n\}$, $D \sim \tau$, $S^{tr} \sim D^m$:*

$$|\hat{L}(\boldsymbol{\mathcal{A}}(\boldsymbol{\mathcal{S}})(S^{tr}), S^{tr}) - \hat{L}(\boldsymbol{\mathcal{A}}(\boldsymbol{\mathcal{S}}^{\setminus i})(S^{tr}), S^{tr})| \leq \tilde{\beta}.$$

Theorem 3 (Generalization error of entropy-limited WCT)**.** *Let the sample volume of the dataset be m, the number of classes be C_1, the annotation entropy be H, and the single-task learning algorithm $\boldsymbol{A}$ have uniform stability β. Then the generalization error $R_{gen}(\boldsymbol{A})$ is bounded by the following equation with probability at least $1-\delta$ for any $\delta \in (0,1)$,*

$$R_{gen}(\boldsymbol{A}) \leq 2\beta + (4m\beta + M)\sqrt{\frac{C_1 \ln(1/\delta)}{2me^{H/m}}}. \tag{13}$$

Proof. According to Theorem 5, under the conventional supervised classification setting, the conventional single-task learning algorithm $\boldsymbol{A}$, *i.e.*, WCT, has the following generalization error upper bound,

$$R_{gen}\boldsymbol{A} \leq 2\beta + (4m\beta + M)\sqrt{\frac{\ln(1/\delta)}{2m}}.$$

Under entropy-limited setting, Lemma 2 derives the number of correct labeling samples. As indicated in Section 2.2, some algorithms are robust to label noise. However, we consider the worst-case scenario here, *i.e.*, only samples with correct labels are taken into account. As a result, we replace m in the above equation with m' given by Lemma 2, *i.e.*,

$$R_{gen}(\boldsymbol{\mathcal{A}}) \leq 2\beta + 2\tilde{\beta} + (4n\tilde{\beta} + M)\sqrt{\frac{kC_2^2 \ln(1/\delta)}{2me^{H/m}}}.$$

Note that we don't replace m in $4m\beta$, because it will be asymptotically eliminated by β in the derivation of Corollary 2. □

Theorem 4 (Generalization error of entropy-limited meta-learning)**.** *Let the sample volume of the dataset be m, the number of classes per task be C_2, the number of samples per class be k, the number of tasks be n, the annotation entropy be H, the base-learner $\boldsymbol{A}$ has uniform stability β, and the meta-learner $\boldsymbol{\mathcal{A}}$ have uniform stability $\tilde{\beta}$. Then generalization error $R_{gen}(\boldsymbol{\mathcal{A}})$ is bounded by the following equation with probability at least $1-\delta$ for any $\delta \in (0,1)$,*

$$R_{gen}(\boldsymbol{\mathcal{A}}) \leq 2\beta + 2\tilde{\beta} + (4n\tilde{\beta} + M)\sqrt{\frac{kC_2^2 \ln(1/\delta)}{2me^{H/m}}}. \tag{14}$$

Proof. According to Theorem 5, under the conventional supervised classification setting, the meta-learning algorithm $\boldsymbol{\mathcal{A}}$, has the following generalization error upper bound,

$$R_{gen}(\boldsymbol{\mathcal{A}}) \leq 2\tilde{\beta} + (4n\tilde{\beta} + M)\sqrt{\frac{\ln(1/\delta)}{2n}} + 2\beta.$$

The number of tasks n, in the worst-case scenario, satisfies

$$n = \frac{m}{kC_2}$$

Similar to the proof of Theorem 3, replace m in the above equation with m' given by Lemma 2, we have

$$R_{gen}(\boldsymbol{\mathcal{A}}) \leq 2\beta + 2\tilde{\beta} + (4n\tilde{\beta} + M)\sqrt{\frac{kC_2^2 \ln(1/\delta)}{2me^{H/m}}}.$$

Similar to the proof of Theorem 3, we don't replace n in $4n\tilde{\beta}$. □

Corollary 2. *Let the base-level stability $\beta \sim o(\sqrt{1/m})$, the meta-level stability $\tilde{\beta} \sim o(\sqrt{1/n})$, and the entropy resource H be equal for each algorithm. Then, the meta-learning algorithm $\boldsymbol{\mathcal{A}}$ has a tighter generalization error upper bound than the single-task learning algorithm $\boldsymbol{A}$ when*

$$C_2^2 \cdot k < C_1. \tag{15}$$

Proof. According to Bousquet [1] and Maurer [38] et.al., the uniform stability β and $\tilde{\beta}$ of algorithms decrease as the dataset scale increases, typically satisfying $\beta \sim o(\sqrt{1/m})$ and $\tilde{\beta} \sim o(\sqrt{1/n})$, respectively. To ensure that the generalization error formula holds with high probability, δ is typically minimized. When m is sufficiently large, these conditions ensure that the generalization error of $\boldsymbol{A}$ is dominated by

$$M\sqrt{\frac{C_1 \ln(1/\delta)}{2me^{H/m}}},$$

and the generalization error of $\boldsymbol{\mathcal{A}}$ is dominated by

$$M\sqrt{\frac{kC_2^2 \ln(1/\delta)}{2me^{H/m}}}.$$

In Theorem 4, n is significantly underestimated. Therefore, as long as $kC_2^2 < C_1$, or even when $kC_2^2 \sim O(C_1)$, the meta-learning algorithm $\boldsymbol{\mathcal{A}}$ admits a much tighter upper bound on the generalization error. □

Theorem 5 (Generalization error of single-task learning [38])**.** *For any data distribution $\mathcal{D}$ and training set S with m samples, if a single-task learning algorithm $\boldsymbol{A}$ has uniform stability β with respect to a loss function l bounded by M, then the following statement holds with probability at least $1-\delta$ for any $\delta \in (0,1)$:*

$$R(\boldsymbol{A}, \mathcal{D}) \leq \hat{R}(\boldsymbol{A}, S) + 2\beta + (4m\beta + M)\sqrt{\frac{\ln(1/\delta)}{2m}}.$$

Theorem 6 (Generalization error of meta-learning [38])**.** *For any task distribution τ and meta-sample $\boldsymbol{\mathcal{S}}$ with n tasks, if a meta-algorithm $\boldsymbol{\mathcal{A}}$ has uniform stability $\tilde{\beta}$ and the inner-task algorithm $A(S)$ has uniform stability β with respect to a loss function l bounded by M, then the following statement holds with probability at least $1-\delta$ for any $\delta \in (0,1)$:*

$$R(\boldsymbol{\mathcal{A}}(\boldsymbol{\mathcal{S}}), \tau) \leq \hat{R}(\boldsymbol{\mathcal{A}}(\boldsymbol{\mathcal{S}}), S) + 2\tilde{\beta} + (4n\tilde{\beta} + M)\sqrt{\frac{\ln(1/\delta)}{2n}} + 2\beta.$$

C. Experimental Details

C.1. Dataset setup

Omniglot. The raw dataset contains 1628 classes, we split the classes of training set, evaluation set, test set into 800: 400: 432. We use Omniglot in three scenarios. The first scenario is in Section 2.2. We perform supervised few-shot learning with label noise. We randomly mask the labels of the samples in the training set according to the noise ratio (*i.e.*, 0%, 15%, 30%). Depending on the training method, we can construct these raw data into task followed by Finn et al. [15], or use them directly for whole class training followed by Tian et al. [45]. The second scenario is in Section 2.3. We perform supervised few-shot learning with heterogeneous tasks. When constructing heterogeneous tasks, we sample a variable number of classes, to ensure the difference in the way of tasks (*i.e.*, 5-20 way), and further to ensure the heterogeneity. The third scenario is in Section 4.1. We perform unsupervised few-shot learning. We follow the protocol given by Hsu et al. [19].

Mini-Imagenet. The raw Mini-Imagenet contains 100 classes, we the split classes of training set, evaluation set, test set into 64: 16: 20. We use Mini-Imagenet in three scenario. The details of the setup of the three experimental scenarios are the same as Omniglot. With the except that we construct 5-10 way heterogeneous task in Section 2.3.

CIFAR-10, CIFAR-100, STL-10, Imagennet, and Tiny Imagenet. For CIFAR-10, CIFAR-100, STL-10, Imagennet, and Tiny Imagenet datasets, we follow the protocol given by Zheng et al. [54]. They are used for unsupervised zero-shot learning, so we mask all the labels in training set.

DomainNet. DomainNet is a domain adaption dataset. We use it to evaluate algorithms' ability of unsupervised zero-shot domain adaption. It contains 6 domain with 345 classes for each domain. We use one domain for test and the remain 5 domain for both training and validating. Note that when constructing tasks, we sample classes from the same domain and we mask all the labels in training set.

MobileNet40 and ShapeNetCore. ModelNet40 contains 12311 CAD models across 40 categories, primarily used for 3D shape classification and point cloud analysis. ShapeNetCore includes over 51300 3D models spanning 55 categories, serving as a benchmark for 3D classification, segmentation, and reconstruction. For the few-shot classification task, we follow the settings given by Khan et al. [27].

C.2. Algorithm Setup

MINO. We use MINO in both unsupervised zero-shot and few-shot scenario. In unsupervised few-shot datasets, we follow the same backbone architecture given by github.com/dragen1860/MAML-Pytorch. We set epoch, inner-loop learning rate, outer-loop learning rate, meta-batch size, inner-loop step, and number of sample per task, as 30000, 0.05, 0.001, 8, 5, and 50 respectively. For DBSCAN, we set min_samples and eps as 15 and 1.0, respectively. In unsupervised zero-shot datasets (except of DomainNet), we follow the same backbone architecture given by github.com/xu-ji/IIC, *i.e.*, ResNet and VGG11. We set epoch, inner-loop learning rate, outer-loop learning rate, meta-batch size, adaption steps for evaluation and sub-sample size, as 80000, 0.001, 0.001, 8, 0, and 100 respectively. For DBSCAN , we set min_samples and eps as 15 and 1.0, respectively. For DomainNet dataset, we use ResNet-9 as backbone architecture, which is the same as github.com/liyunsheng13/DRT. The other configuration is the same as other unsupervised zero-shot datasets. Note that for fair comparison, we keep the network structure of other methods consistent with that of MINO to ensure that the neural networks have the same scale. As for the hyperparameters, we follow the settings provided in the original paper.

WCT, MAML, ANIL, and MTL. In Section 2.2, Section 2.3, and Section 4.3, we use WCT, MAML, ANIL, and MTL to perform experiments, on Omniglot and Mini-Imagenet datasets. For MAML, we leverage the ”body” given by [48], which has 4 convolution modules. Each module consist of a 3×3 convolutions and 64 filters, followed by a batch normalization, a ReLU nonlinearity, and 2×2 max-pooling. We utilize the ”head” followed the baseline classifier given by Finn et al. [15]. For Omniglot, we used strided convolutions instead of max-pooling. For ANIL, its difference with MAML lies solely in the strategy for updating the head [39]. For WCT, we use the same neural network architecture and learning configuration, except that its head is fixed, and the length of the head equals the number of whole classes. For MTL, we also maintain the same neural network architecture and learning configuration, except that it uses a single-level optimization strategy. In Section 2.3, for SHM, we train a model for each way of tasks, and ultimately take the average testing performance of the models. For DHM and MTL, we train the model with the train set consisting of a mixture of the heterogeneous tasks, and ultimately evaluating its performance directly on the test set.

PsCo, Meta-GMVAE, UMTRA, and CACTUs. We reuse the configuration given by Jang et al. [20],Lee et al. [32],Khodadadeh et al. [28], and Hsu et al. [19], since our test scenarios are the same as theirs.

ReSSL and IIC. In CIFAR-10, CIFAR-100, STL-10, ImageNet, and Tiny ImageNet datasets, we reuse the configuration given by Zheng et al. [54] and Ji et al. [23]. In DomainNet dataset, for a fair comparison, we use ResNet-9 as backbone and maintain the same learning configuration as mentioned above.

MAE and NVAE. For a fair comparison, we use the same backbone provided by Masci et al. [37], which has a similar number of parameters as other models.

DeepCluster. We run DeepCluster for each unsupervised zero-shot dataset, which we respectively randomly crop and resize to the appropriate image size. We modify the first layer of the AlexNet architecture used by the authors to accommodate this input size. We follow the authors and use the input to the (linear) output layer as the embedding. These are 4096-dimensional, so we follow the authors and apply PCA to reduce the dimensionality to 256, followed by whitening. The configuration is built upon github.com/facebookresearch/deepcluster. In DomainNet dataset, we also use ResNet-9 as backbone.

BiGAN. We follow the BiGAN authors and specify a uniform 50-dimensional prior on the unit hypercube for the latent. They use a 200 dimensional version of the same prior for their ImageNet experiments, so we follow suit for our unsupervised zero-shot dataset. Our configuration is built upon github.com/jeffdonahue/bigan. In DomainNet dataset, we also use ResNet-9 as backbone.

D. Discussion and Future work.

Equal Classes Probability Assumption. We hold a balanced classes probability assumption in Lemma 2, which may raise scrutiny and challenge. Because in the unsupervised setting, the samples included in the constructed tasks are random. However, according the conclusion given by Khodadadeh et al. [28], for a task T_i, the probability that all samples are in a different class is equal to

$$P = \frac{C_1! \cdot k^{C_2} \cdot (C_1 \cdot k - C_2)!}{(C_1 - C_2)! \cdot (C_1 \cdot k)!}.$$

When $C_1 \gg C_2$, we have $p \to 1$, which implies that the equal probability assumption holds.

Experiments Under Equal Entropy Conditions. Entropy is not a directly utilized metric in the algorithm training process. Instead, it represents the resources required for annotation. Therefore, we leverage the label noise ration given by Lemma 2, *i.e.*,

$$p = e^{\frac{H}{m}}/C_1,$$

to infer the expected label noise in the dataset. We then use this noised dataset for training different models to simulate equal entropy conditions.

The Underestimated Number of Tasks n. In the worst-case scenario, according to the definition in Theorem 4, the number of tasks should be

$$n = \frac{m}{kC_2}.$$

However, in a typical unsupervised setting, the samples in a task are independently and randomly sampled. Assuming the total number of task samples is kC_2 , the number of different tasks we can construct is

$$\binom{m}{kC_2} = \frac{m!}{(kC_2)!(m - kC_2)!}.$$

This number is significantly larger than what we use in the paper. As a result, the actual generalization error bound of meta-learning under limited entropy conditions is much tighter than the one derived in Theorem 4. This reinforces the conclusion given by Corollary 2.

Overhead. MINO incurs low computational overhead by replacing existing components rather than adding new ones. Specifically, it replaces CACTUs' K-means with DBSCAN, uses a dynamic head instead of a static one, and introduces a lightweight meta-scaler based on batch-level linear computation. As shown in Table 10, each component adds less than 7% training overhead compared to CACTUs, while achieving a 13.21% accuracy gain (see Table 4). Importantly, all components are used only in training and do not affect inference overhead.

Table 10. Per-sample latency (ms) on CIFAR-100 during training and inference, compared with CACTUs.

Setting	Training Latency (ms)	Inference Latency (ms)
CACTUs	4.98	1.56
W/O DBSCAN	5.32 $\uparrow$ 0.34	1.54 $\downarrow$ 0.02
W/O Meta-Learning (WCT)	4.26 $\downarrow$ 0.72	1.57 $\uparrow$ 0.01
W/O Meta-Scaler	5.29 $\uparrow$ 0.31	1.56 $\uparrow$ 0.00
MINO (Ours)	5.51 $\uparrow$ 0.53	1.57 $\uparrow$ 0.01

Future Work. We will establish a more rigorous theoretical framework. By improving the computation method for the entropy of samples, this framework will no longer rely on the equal classes probability assumption, enabling it to cover datasets with arbitrary class probability distributions. At the same time, based on the latest research [26, 35], we need to incorporate the impact of label noise on the generalization ability of the algorithm into this theoretical framework. This allows us to go beyond considering only the worst-case scenario in terms of sample size, as in Proof of Theorem 3, and instead provide a tighter generalization error upper bound.

E. Related Work

Theoretical analysis of meta-learning. In recent years, significant advances have been made in meta-learning theory research. Studies have revealed important insights about MAML's fast adaptation mechanisms [16, 39], while other research has demonstrated that well-designed embeddings can potentially outperform meta-learning approaches in few-shot classification tasks [45]. Further investigations have established the independence between meta-training and adaptation algorithms [36], and identified specific conditions where baseline methods can exceed meta-learning performance in few-shot classification tasks [5]. Additional research has examined the implications of class and novel class generalization in meta-learning [6], while theoretical work has established upper bounds for meta-learning generalization error [3, 4, 13, 14, 17, 25, 38]. This paper addresses two fundamental questions regarding meta-learning: Under what circumstances does meta-learning demonstrate superior performance compared to alternative algorithms in few-shot classification, and what underlying factors contribute to this advantage?

Unsupervised meta-learning. Unsupervised meta-learning [19–21, 28, 32, 52] links meta-learning and unsupervised learning by constructing synthetic tasks and extracting the meaningful information from unlabeled data. For example, CACTUs [19] cluster the data on the pretrained representations at the beginning of meta-learning to assign pseudo-labels. Instead of pseudo-labeling, UMTRA [28] and LASIUM [29] generate synthetic samples using data augmentations or pretrained generative networks like Big-BiGAN [10]. Meta-GMVAE [32] and Meta-SVEBM [30] represent unknown labels via categorical latent variables using variational autoencoders [46] and energy-based models, respectively. In this paper, we leverage the sights under entropy-limited supervised setting, improve meta-learning algorithm's robustness against label noise and heterogeneous tasks.