Task Groupings Regularization: Data-Free Meta-Learning with Heterogeneous Pre-trained Models

Yongxian Wei [1] **Zixuan Hu** [1 2] **Li Shen** [3] **Zhenyi Wang** [4] **Yu Li** [5] **Chun Yuan** [1] **Dacheng Tao** [2]

Abstract

Data-Free Meta-Learning (DFML) aims to derive knowledge from a collection of pre-trained models without accessing their original data, enabling the rapid adaptation to new unseen tasks. Current methods often overlook the heterogeneity among pre-trained models, which leads to performance degradation due to task conflicts. In this paper, we empirically and theoretically identify and analyze the model heterogeneity in DFML. We find that model heterogeneity introduces a heterogeneity-homogeneity trade-off, where homogeneous models reduce task conflicts but also increase the overfitting risk. Balancing this trade-off is crucial for learning shared representations across tasks. Based on our findings, we propose Task Groupings Regularization, a novel approach that benefits from model heterogeneity by grouping and aligning conflicting tasks. Specifically, we embed pre-trained models into a task space to compute *dissimilarity*, and group heterogeneous models together based on this measure. Then, we introduce implicit gradient regularization within each group to mitigate potential conflicts. By encouraging a gradient direction suitable for all tasks, the meta-model captures shared representations that generalize across tasks. Comprehensive experiments showcase the superiority of our approach in multiple benchmarks, effectively tackling the model heterogeneity in challenging multi-domain and multi-architecture scenarios.

[1]Tsinghua Shenzhen International Graduate School, Tsinghua University, China [2]College of Computing & Data Science, Nanyang Technological University, Singapore [3]School of Cyber Science and Technology, Sun Yat-sen University, China [4]University of Maryland, College Park, USA [5]Department of Computer Science and Engineering, The Chinese University of Hong Kong, China. Correspondence to: Li Shen <mathshenli@gmail.com>, Zhenyi Wang <zwang169@umd.edu>, Chun Yuan <yuanc@sz.tsinghua.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

1. Introduction

Data-Free Meta-Learning (DFML) (Kwon et al., 2020; Wang et al., 2022b; Hu et al., 2023a;b) aims to derive knowledge from a collection of pre-trained models without accessing their original data, enabling the rapid adaptation to new unseen tasks. Traditional meta-learning methods (Finn et al., 2017; Wang et al., 2020) assume access to a collection of tasks with available training and testing data. However, such data is often not accessible in real-world scenarios due to privacy concerns, security risks, or usage rights restrictions (Chen et al., 2019; Truong et al., 2021; Zheng et al., 2023). For instance, numerous pre-trained models from diverse domains are released on platforms like GitHub or Hugging Face without accompanying training data. This situation underscores the value of DFML: collect some pre-trained models with weaker generalization abilities, which likely originate from diverse domains online, to train a meta-model with enhanced generalization capacity for new tasks.

Current DFML methods predominantly focus on data recovery from pre-trained models. For example, PURER (Hu et al., 2023a) uses model inversion to optimize a learnable dataset for each pre-trained model and samples pseudo tasks for meta-learning. BiDf-MKD (Hu et al., 2023b) employs generators with multiple black-box APIs, generating support and query sets for meta-learning. Nonetheless, these methods often overlook the heterogeneity among pre-trained models, raised from differences in training data, model architecture, and optimization algorithm. Simultaneously learning from heterogeneous pre-trained models[1] can sometimes lead to performance degradation due to task conflicts (Standley et al., 2020; Shi et al., 2021; Ye et al., 2021).

In this paper, we first empirically demonstrate the existence and potential causes of model heterogeneity in DFML (see Fig. 1), and analyze the practical effects of model heterogeneity on generalization (see Sec. 3.2). Experiments suggest that model heterogeneity is a double-edged sword and can be beneficial and desirable in certain cases (see Fig. 2). A certain degree of heterogeneity plays a role similar to

[1]In this paper, we consider pre-trained models as distinct tasks. It means that the data recovered from pre-trained models form individual tasks; and also that pre-trained models resemble diverse tasks in meta-learning, on which the meta-model is trained.

regularization (Kurin et al., 2022), reducing the generalization error of the meta-model. Furthermore, we theoretically find that model heterogeneity introduces a heterogeneity-homogeneity trade-off. Conflicting tasks from heterogeneous models may compete for model capacity. Optimizing for task $\mathcal{T}_i$ could accidentally worsen the performance on task $\mathcal{T}_j$ and vice versa. In contrast, homogeneous models, while reducing task conflicts, also diminish data diversity due to their similar knowledge, thereby increasing the risk of overfitting. Balancing this trade-off is crucial for learning shared representations that generalize across tasks.

Based on our findings, we propose Task Groupings Regularization, a novel approach that benefits from model heterogeneity by balancing the heterogeneity-homogeneity trade-off. We divide the heterogeneous pre-trained models into task groups, each consisting of *the most dissimilar* pre-trained models. Then, we introduce implicit gradient regularization within each group, to mitigate potential *conflicts* of dissimilar tasks. Specifically, we propose to embed pre-trained models into a task space to compute task dissimilarity. We first recover task-specific data from pre-trained models via model inversion. Subsequently, we utilize the Fisher Information Matrix (FIM) (Amari, 1998) to extract task embeddings. Based on dissimilarity measures between these embeddings, we perform spectral clustering to divide models into multiple task groups. During the training phase of the meta-model, we periodically sample pre-trained models to inverse new tasks, facilitating the transfer of knowledge to the meta-model. Each time, we sample heterogeneous pre-trained models from the same task group. The tasks recovered from these heterogeneous models demonstrate conflicting optimization directions, failing to develop a shared representation. Therefore, we apply implicit gradient regularization among them, penalizing "non-uniform" regions to align conflicting tasks. By encouraging a gradient direction suitable for all tasks, the meta-model captures shared representations across different tasks from pre-trained models, enabling it to generalize effectively to unseen tasks.

Compared to existing DFML methods, our approach achieves improvements of 5.54%, 6.70% and 3.05% on *CIFAR-FS*, *miniImageNet* and *CUB*, respectively. Furthermore, we effectively tackle the model heterogeneity in challenging multi-domain and multi-architecture scenarios.

In summary, our main contributions are three-fold:

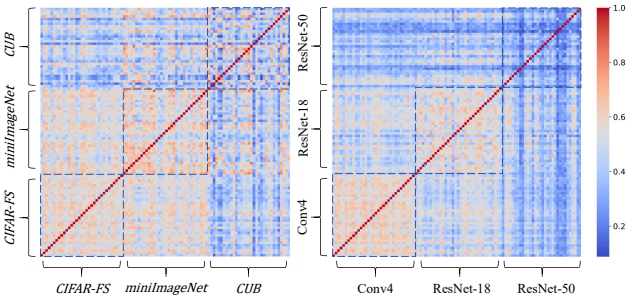
- For the first time, we systematically identify and analyze the heterogeneity-homogeneity trade-off from both empirical and theoretical perspectives in DFML.
- We propose Task Groupings Regularization, a novel approach that benefits from model heterogeneity. This approach aligns conflicting tasks, facilitating the learning of generalizable representations across tasks.
- We conduct comprehensive experiments and discussions. Further on two challenging model heterogeneity scenarios, our empirical results demonstrate a significant improvement over previous methods.

2. Related Work

Data-free meta-learning. Data-free learning (Chen et al., 2019; Liu et al., 2021; Li et al., 2023) facilitates the learning process without access to the raw data. This approach is of significant value in practical scenarios where data availability is limited due to privacy, safety, or ethics. It has been greatly influenced by model inversion (Fredrikson et al., 2015), which seeks to extract embedded data knowledge from pre-trained models. Recently, Data-Free Meta-Learning (DFML) (Wang et al., 2022b; Hu et al., 2023a;b) has gained popularity, by leveraging multiple pre-trained models with weaker generalization abilities to learn a meta-model with superior generalization. DRO (Wang et al., 2022b) predicts meta initialization through a black-box neural network emphasizing the parameter space. PURER (Hu et al., 2023a) progressively synthesizes a series of pseudo-tasks using episode curriculum inversion. BiDf-MKD (Hu et al., 2023b) learns the meta-model by transferring knowledge from black-box APIs via zero-order gradient estimation. However, they overlook the heterogeneity inherent in the collected pre-trained models.

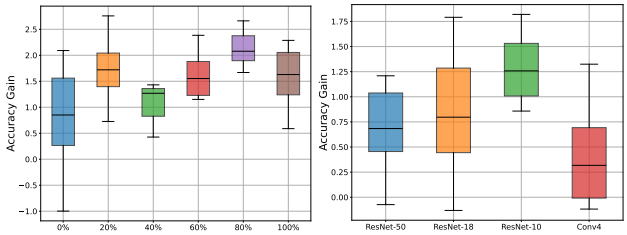
Task groupings. It is commonly believed that related tasks with similar underlying structures can benefit from training together. In multi-task learning (Caruana, 1993), deciding which tasks to train together has traditionally involved an extensive search (Standley et al., 2020) or subjective human judgment. Recently, TAG (Fifty et al., 2021) determines task groupings by jointly training all tasks and assessing the impact of one task's gradient on another task's loss. To mitigate Negative Transfer (Jiang et al., 2023), Zamir et al. (2018) suggest assigning unrelated tasks to different groups to minimize task interference. However, after establishing task groupings, most methods simply create a unified gradient by linearly combining gradients of tasks. Different from previous work, we propose grouping dissimilar tasks together and then applying regularization to align their gradients. Moreover, our task grouping is a pre-processing step, which does not increase the training overhead.

Gradient alignment. Gradient alignment (Shi et al., 2021; Wang et al., 2022a) highlights the importance of aligning or maintaining consistency in gradients during training. Empirical studies have demonstrated its effectiveness in various domains like meta-learning (Eshratifar et al., 2018), federated learning (Dandi et al., 2022), and multi-task learning (Shi et al., 2022). In our work, we design an implicit gradient regularization to align tasks from heterogeneous models, with low computation time and memory overheads.



(a) Multi-domain (b) Multi-architecture

Figure 1. Similarity heatmaps of pre-trained models measured by CKA. We compare (a) pre-trained models from different datasets, and (b) pre-trained models with different architectures. Coordinate axes indicate the corresponding model index, a total of 100 pre-trained models involved. Best viewed when zoomed in.



(a) Overlapping classes (b) Model architecture

Figure 2. Relationships between the model heterogeneity and Accuracy Gain. We select a pre-trained Conv4 as the basic model, combining it with an auxiliary pre-trained model to assess the AG.

3. Revisit DFML

In this section, we first describe the basic definition of DFML and then revisit the model heterogeneity challenges.

3.1. Problem Setup of DFML

We are given only a collection of pre-trained models $\mathcal{M}_{pool} = \{M_i\}_{i=1}^n$ without accessing their training data. DFML utilizes synthetic data $\hat{\boldsymbol{X}}$ recovered from $\mathcal{M}_{pool}$ to train a meta-model $M_{meta}(\cdot;\boldsymbol{\theta})$, which can be rapidly transferred to unseen tasks. During meta-testing, we sample "N-way K-shot" tasks. For such a task, there are N classes and each class only has K labeled samples named the support set, and U unlabelled samples per class called the query set. We use the few-shot support set to adapt the meta-model $M_{meta}(\cdot;\boldsymbol{\theta})$ to each specific task. The query set is then used to make predictions and assess the accuracy.

Next, we define the Accuracy Gain to measure the impact of joint training in DFML, using two pre-trained models.

Definition 3.1. (Accuracy Gain). Denote the meta-model trained from a single pre-trained model as $\boldsymbol{\theta}(M_{\text{bas}})$, and the meta-model trained from an auxiliary pre-trained model alongside the basic one as $\boldsymbol{\theta}(M_{\text{bas}}, M_{\text{aux}})$. Let $\mathcal{P}$ be the meta-testing accuracy as a performance measure. Then, the Accuracy Gain of joint training can be evaluated by:

$$AG = \mathcal{P}(\boldsymbol{\theta}(M_{\text{bas}}, M_{\text{aux}})) - \mathcal{P}(\boldsymbol{\theta}(M_{\text{bas}})). \tag{1}$$

3.2. Rethinking Model Heterogeneity in DFML

Model heterogeneity analysis. The heterogeneity in collected pre-trained models is inevitable, arising from differences in training data, model architecture, and optimization algorithm, particularly concerning the degree of model convergence. Models pre-trained on varied data distributions can be distinguished based on their specific domains. Even within the same dataset, models trained by different classes exhibit distinct sub-domain characteristics, *e.g.*, statistical properties like pixel intensity or texture variation (Chen et al., 2021; Song et al., 2020). Moreover, the diversity in model architectures also contributes to model heterogeneity, influencing how data is represented (Hermann & Lampinen, 2020). This in turn affects the patterns of the recovered data, leading to a gap in their distributions.

To demonstrate the heterogeneity among pre-trained models, we use Centered Kernel Alignment (CKA) (Kornblith et al., 2019; Hao et al., 2023) to measure the similarity of features extracted by various pre-trained models. CKA is well-suited for comparisons across different architectures since it can accommodate inputs of varying dimensions. The similarity heatmaps, presented in Fig. 1, distinctly showcase the substantial differences in feature distribution among models. Heterogeneity arises from different datasets and model architectures. This is particularly obvious in the fine-grained *CUB* dataset and the large-scale ResNet-50 architecture, showing increased internal heterogeneity.

Empirical observation. To analyze effects of model heterogeneity based on the defined Accuracy Gain, we evaluate model heterogeneity using two metrics: the ratio of overlapping training classes and the architecture of pre-trained models. This is based on the reasonable assumption that a lower overlap in classes indicates differing task distributions, and the representation ways of different model architectures vary. We select a pre-trained Conv4 as the basic model M_{bas}, combining it with various auxiliary models M_{aux} to assess the AG. For a fair comparison, each measure begins with the same model parameters. The process of training data recovery follows Eq. (2), which we will discuss later. Detailed designs of our experiment are provided in App. D.

Fig. 2 demonstrates that combining homogeneous models does not yield significant AG, *e.g.*, 100% overlapping classes in (a), and the same Conv4 architecture in (b). Conversely, choosing heterogeneous models results in higher AG. This is attributed to the fact that while conflicting gradients from heterogeneous models will increase training loss and slow down convergence speed, they may also play a

role similar to regularization (Kurin et al., 2022), reducing the overfitting risk, thereby reducing the generalization error of meta-model (Jiang et al., 2023). However, this can also result in significant performance degradation when tasks compete for model capacity or fail to develop a shared representation that generalizes across tasks, *e.g.*, 0% overlapping classes in (a), and the ResNet-50 architecture in (b).

Theoretical understanding. Inspired by the empirical observation, we will argue that there is a trade-off between minimizing the expected error over the merged distribution via the empirical error of joint learning.

Theorem 3.2. *Assume $M_{meta}(\cdot;\boldsymbol{\theta})$ is probably approximately correct (PAC),* i.e.*, there exists $\zeta(N,\delta) \geq 0$ monotonically decreasing with N, and the loss function $\ell(\cdot)$ is K-Lipschitz continuous. Then, with probability at least $1-2\delta$ the following bounds hold:*

$$|E(\boldsymbol{\theta}(M_{\text{bas}}),\boldsymbol{\theta}(M_{\text{aux}})) - \hat{E}(\boldsymbol{\theta}(M_{\text{bas}},M_{\text{aux}}))| \leq \zeta(N,\delta) + K\sum_{t=\text{bas}}^{\text{aux}} \mathbb{E}_{\mathcal{P}_t}[|M_{meta}(\boldsymbol{\theta}(M_t)) - M_{meta}(\boldsymbol{\theta}(M_{\text{bas}},M_{\text{aux}}))|]$$

where $E = \sum_{t=\text{bas}}^{\text{aux}} \mathbb{E}_{\mathcal{P}_t}[\ell(M_{meta}(x,\boldsymbol{\theta}(M_t)))]$ is the expected error over the merged distribution and $\hat{E} = \frac{1}{N}\sum_{i=1}^{N}\ell(M_{meta}(x_i,\boldsymbol{\theta}(M_{\text{bas}},M_{\text{aux}})))$ is the empirical error over the N training samples.

The bound consists of two terms: a homogeneity term $\zeta(N,\delta)$ dependent on the number of samples N; and one term $M_{meta}(\boldsymbol{\theta}(M_t)) - M_{meta}(\boldsymbol{\theta}(M_{\text{bas}},M_{\text{aux}}))$ based on the distance between meta-models trained from different models. This term can be seen as a heterogeneity term, arising from the discrepancy between M_{bas} and M_{aux}. When the distributions of M_{bas} and M_{aux} are close, a significant portion of the data recovered from them overlap (Yin et al., 2020), *i.e.*, a reduction in the number of distinct data sampled from the merged distribution $\mathcal{P}_{\text{bas}} + \mathcal{P}_{\text{aux}}$, thereby increasing the homogeneity term. On the other hand, when model heterogeneity is substantial, the discrepancy between $\boldsymbol{\theta}(M_{\text{bas}})$ and $\boldsymbol{\theta}(M_{\text{aux}})$ becomes significant (Cerviño et al., 2023), making the heterogeneity term larger. Thus, Theorem 3.2 highlights a heterogeneity-homogeneity trade-off that necessitates careful handling.

> **Finding.** *Model heterogeneity presents a heterogeneity-homogeneity trade-off, and balancing this trade-off is crucial for the meta-model's generalization across tasks.*

Consequently, it is a natural consideration to identify task groups, *i.e.*, decide which models should be trained together to balance the trade-off. The decision of task groupings is complex and not thoroughly explored, often relying on the judgment of human experts (Zhang & Yang, 2021).

4. Methodology

Guided by our findings, we propose Task Groupings Regularization (see Fig. 3) to learn from a collection of pre-trained models $\mathcal{M}_{pool}$. Initially, we recover specific tasks from these models via a generator, and use them to compute the FIM as task embeddings. Based on dissimilarity of these embeddings, we group heterogeneous models together. Then, implicit gradient regularization is applied within each group to mitigate potential conflicts among tasks.

Below, we describe the two parts in detail. In Sec. 4.1, we divide $\mathcal{M}_{pool}$ into c task groups $\mathcal{G}_i \subset \mathcal{M}_{pool}$, each consisting of the most dissimilar pre-trained models. Then in Sec. 4.2, we introduce conflicting task regularization to each group for optimizing the meta-model.

4.1. Heterogeneous Pre-trained Models Grouping

Our understanding of the heterogeneity in pre-trained models is limited since the original data is unknowable. Moreover, due to the exponential increase in the number of group combinations $2^n - 1$ with the number of pre-trained models n, it becomes both challenging and computationally demanding to group these models directly. To address these, we propose to embed pre-trained models into a task space to compute task dissimilarity. By performing spectral clustering based on the dissimilarity measure, we successfully group heterogeneous models without the need for training.

Task recovery via model inversion. Given a pre-trained model M, we optimize a generator $G(\cdot;\boldsymbol{\theta}_G)$ alongside a corresponding latent code $\boldsymbol{Z}$ to recover training data $\hat{\boldsymbol{X}}$. These synthetic data $\hat{\boldsymbol{X}}$ mimic similar responses from the pre-trained model as the real data $\boldsymbol{X}$. For recovering the task from pre-trained model M, we employ the inversion loss to update $\boldsymbol{\theta}_G$ and $\boldsymbol{Z}$, which comprises a classification loss $\mathcal{L}_{CE}$ and a feature regularization loss $\mathcal{L}_{BN}$:

$$\min_{\boldsymbol{Z},\boldsymbol{\theta}_G} \mathcal{L}_G(\boldsymbol{Z},\boldsymbol{Y};\boldsymbol{\theta}_G) = \mathcal{L}_{CE} + \mathcal{L}_{BN}, \tag{2}$$

where $\mathcal{L}_{CE}$ is a cross-entropy loss that conditions the input data on pre-defined task labels $\boldsymbol{Y}$. This enforces the synthetic data $\hat{\boldsymbol{X}} = G(\boldsymbol{Z};\boldsymbol{\theta}_G)$ to yield strong predictions (Chen et al., 2019), akin to a one-hot vector:

$$\mathcal{L}_{CE}(\hat{\boldsymbol{X}},\boldsymbol{Y};\boldsymbol{\theta}_G) = CE(M(\hat{\boldsymbol{X}}),\boldsymbol{Y}). \tag{3}$$

$\mathcal{L}_{BN}$ ensures the alignment of feature-map statistics of the synthetic data with those of the real training data (Yin et al., 2020), as preserved in the Batch Normalization (BN) layers of the pre-trained models:

$$\mathcal{L}_{BN}(\hat{\boldsymbol{X}};\boldsymbol{\theta}_G) = \sum_l \|\mu_l(\hat{\boldsymbol{X}}) - \mu_l^{\text{BN}}\| + \|\sigma_l^2(\hat{\boldsymbol{X}}) - \sigma_l^{\text{BN}}\|, \tag{4}$$

where $\mu_l(\cdot)$ and $\sigma_l^2(\cdot)$ denote their mean and variance at the lth convolutional layer. μ_l^{BN} and σ_l^{BN} refer to the mean and variance of prior statistics stored in the lth BN layer.

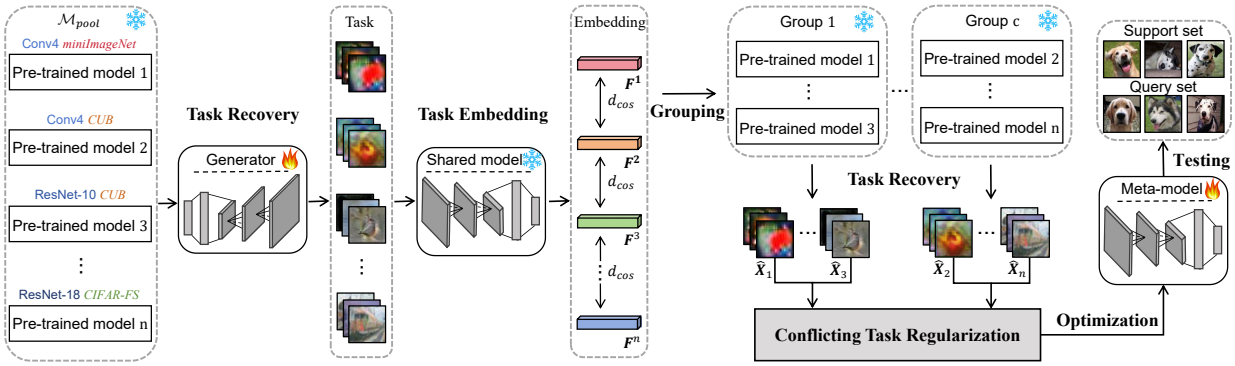


Figure 3. DFML training pipeline. We embed pre-trained models $\mathcal{M}_{pool}$ into a task space to compute dissimilarity, and divide heterogeneous models into task groups. Then, conflicting task regularization is introduced within each group to train the meta-model for new tasks.

We recover $\hat{\boldsymbol{X}}$ that minimize Eq. (2) by optimizing $(\boldsymbol{Z}, \boldsymbol{\theta}_G)$. In the absence of real tasks, we generate a certain amount of data (600 images per class in our experiment) to form a pseudo task that represents each pre-trained model.

Task grouping via dissimilarity measures. Fisher Information Matrix (FIM) serves as an indicator of a model's sensitivity to minor perturbations and the curvature of its loss surface. Furthermore, the FIM can be leveraged as the task embedding, capturing task domain and task difficulty (Achille et al., 2019). Formally, the empirical FIM is defined as the covariance of the gradients of the log-likelihood with respect to model parameters $\boldsymbol{\varphi}$:

$$\boldsymbol{F}_{\boldsymbol{\varphi}}^{i} = \frac{1}{N}\sum_{j=1}^{N}\left[\nabla_{\boldsymbol{\varphi}} \log P_{\boldsymbol{\varphi}}(y_j|x_j) \nabla_{\boldsymbol{\varphi}} \log P_{\boldsymbol{\varphi}}(y_j|x_j)^{\mathrm{T}}\right], \quad (5)$$

where $P_{\boldsymbol{\varphi}}(y|x)$ is the prediction function parametrized by $\boldsymbol{\varphi}$. We utilize the FIM to compute the embedding of task $\{(x_j, y_j)\}_{j=1}^{N}$ recovered from pre-trained model M_i.

The FIM offers a fixed-dimensional representation of a given task, unaffected by factors like the number of classes. We can select a shared pre-trained model and freeze its weights $\boldsymbol{\varphi}$ to compute the task embedding $\boldsymbol{F}$. Given the large dimensionality of $\boldsymbol{F}_{\boldsymbol{\varphi}}$, we approximate by considering only its diagonal entries, following the method of Wu et al. (2023b). Thus, the task embedding $\boldsymbol{F}$ is computed as $\boldsymbol{F} = \mathrm{diag}(\boldsymbol{F}_{\boldsymbol{\varphi}}) = [\boldsymbol{F}_{\boldsymbol{\varphi}_1}, \boldsymbol{F}_{\boldsymbol{\varphi}_2}, \cdots, \boldsymbol{F}_{\boldsymbol{\varphi}_n}]$, where $\boldsymbol{\varphi}_i$ represents the ith parameter of the shared model.

We then calculate the cosine similarity between the embeddings of each pair of tasks to obtain a similarity matrix (see App. E). By subtracting this matrix from an all-ones matrix $\boldsymbol{J}$, we derive the task dissimilarity matrix $\boldsymbol{W}$:

$$\boldsymbol{W} = \boldsymbol{J} - \left[d_{\cos}\left(\frac{\boldsymbol{F}^i}{\boldsymbol{F}^i + \boldsymbol{F}^j}, \frac{\boldsymbol{F}^j}{\boldsymbol{F}^i + \boldsymbol{F}^j}\right)\right]_{1 \le i,j \le n}. \quad (6)$$

The measures reflect the relations of tasks, correlating positively with natural distances in the task space. This allows us to identify task groupings that maximize total dissimilarity measures. While this problem is NP-hard (reduction from Graph Partitioning), it can be efficiently addressed using a clustering algorithm like spectral clustering (Ng et al., 2001). With the pre-computed $\boldsymbol{W}$, the objective function of spectral clustering is defined by:

$$\arg\min_{\boldsymbol{H}} \; Tr(\boldsymbol{H}^\top \boldsymbol{L} \boldsymbol{H}), \text{ s.t. } \boldsymbol{H}^\top \boldsymbol{H} = \boldsymbol{I}, \quad (7)$$

where $\boldsymbol{L}$ represents the Laplacian matrix defined by $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$, $\boldsymbol{D}$ is a diagonal matrix with $\boldsymbol{D}_{ii} = \sum_j \boldsymbol{W}_{ij}$, and $Tr(\cdot)$ denotes the trace of a matrix. The optimal data representation $\boldsymbol{H}$ to Eq. (7) comprises c eigenvectors corresponding to the c smallest eigenvalues of $\boldsymbol{L}$. With the optimal $\boldsymbol{H}$, the clustering assignment is obtained by performing k-means on it, dividing $\mathcal{M}_{pool}$ into c groups. The spectral clustering algorithm integrates the global relationships between tasks (*i.e.*, dissimilarity matrix $\boldsymbol{W}$), and performs clustering in the embedding space of tasks that are linearly inseparable (Huang et al., 2019). Dissimilar pre-trained models are grouped together, which diversifies recovered samples, thereby reducing the homogeneity term.

4.2. Conflicting Task Regularization

During the training phase of the meta-model, we periodically sample pre-trained models to inverse new tasks, facilitating the transfer of knowledge to the meta-model. Each time, we sample heterogeneous pre-trained models from the same group. Tasks from heterogeneous models demonstrate conflicting optimization directions, as evidenced by their gradients inner product < 0. For optimal generalization performance, it's crucial to not just converge to a solution that minimizes the mean loss, but also to steer towards regions where gradient discrepancy is minimized.

Meta-model optimization. For pre-trained model M_i, we recover the pseudo task $\hat{\boldsymbol{X}}_i$ as outlined in Eq. (2). This pseudo task is then utilized for knowledge distillation, allowing for the transfer of task-specific knowledge from

Algorithm 1: Task Groupings Regularization

Input: A collection of pre-trained models $\mathcal{M}_{pool}$; a shared model $\boldsymbol{\varphi}$; a generator $G(\cdot;\boldsymbol{\theta}_G)$; the memory bank $\mathcal{B}$.
Output: A meta-model $M_{meta}(\cdot;\boldsymbol{\theta})$ for new tasks.
// Pre-trained Models Grouping
Task recovery via $G(\cdot;\boldsymbol{\theta}_G)$ *w.r.t.* Eq. (2)
Task embedding via $\boldsymbol{\varphi}$ *w.r.t.* Eq. (5)
Grouping via dissimilarity measures *w.r.t.* Eqs. (6) and (7)
for epoch $\leftarrow 1$ **to** T **do**
  // Conflicting Task Regularization
  Sample a task group $\mathcal{G}_i$ from $\mathcal{M}_{pool}$
  Sample m pre-trained models from $\mathcal{G}_i$
  Recover new tasks $\{\hat{\boldsymbol{X}}_i\}_{i=0}^{m-1}$ via $G(\cdot;\boldsymbol{\theta}_G)$
  Compute the loss $\mathcal{L}_i(\boldsymbol{\theta})$ for each task *w.r.t.* Eq. (8)
  Optimize $M_{meta}(\cdot;\boldsymbol{\theta})$ with $\frac{1}{m}\sum_{i=0}^{m-1}\nabla\mathcal{L}_i(\boldsymbol{\theta}-\boldsymbol{v}_i(\boldsymbol{\theta}))$
  // Cross-Task Replay
  Put into memory bank $\mathcal{B}\leftarrow\mathcal{B}+\{\hat{\boldsymbol{X}}_i\}_{i=0}^{m-1}$
  Construct interpolated tasks $\hat{\mathcal{T}}$ from $\mathcal{B}$
  Optimize the meta-model M_{meta} with $\hat{\mathcal{T}}$

the pre-trained model (the teacher) to the meta-model (the student). The optimization of the meta-model focuses on minimizing the discrepancies in predictions between the teacher M_i and student M_{meta}. In addition to using soft-label predictions, we also implement hard-label supervision with pre-defined task labels $\boldsymbol{Y}$ to compute the loss $\mathcal{L}_i(\boldsymbol{\theta})$:

$$\begin{aligned}\mathcal{L}_i(\boldsymbol{\theta}) =& KL(M_i(\hat{\boldsymbol{X}}_i), M_{meta}(\hat{\boldsymbol{X}}_i;\boldsymbol{\theta})) \\ &+ CE(M_{meta}(\hat{\boldsymbol{X}}_i;\boldsymbol{\theta}), \boldsymbol{Y}).\end{aligned} \tag{8}$$

Once the newly recovered task $\hat{\boldsymbol{X}}_i$ has undergone knowledge transfer, it is stored in a memory bank $\mathcal{B}$. We introduce cross-task replay to broaden the pseudo task distribution. Specifically, we sample from $\mathcal{B}$ to obtain tasks that are interpolated across different tasks, *i.e.*, a random combination of classes. For each interpolated task $\hat{\mathcal{T}}$, we divide it into a support set and a query set. Then, we utilize MAML (Finn et al., 2017), as described in Eq. (9) in the Appendix, to optimize the meta-model. Broadening the range of training tasks is crucial for effective meta-learning.

Implicit gradient regularization among tasks. Regularization broadly refers to methods involving penalty functions that enhance model generalization (Zhao et al., 2022). Smith et al. (2020) discovered that the mean SGD iterate remains close to the path of gradient flow on a loss that includes an implicit regularizer $\sum_{i=0}^{m-1}||\nabla\mathcal{L}_i(\boldsymbol{\theta})-\nabla\bar{\mathcal{L}}(\boldsymbol{\theta})||^2$, where $\nabla\bar{\mathcal{L}}(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=0}^{m-1}\nabla\mathcal{L}_i(\boldsymbol{\theta})$ is the average gradient over minibatches. This regularizer penalizes "non-uniform" regions characterized by large norms of errors in the minibatch gradients. However, this assumption only holds true when the learning rate is small and finite.

It becomes intuitive to consider the explicit implementation of this regularizer, which measures the variance of gradients across conflicting tasks within a group. Minimizing this variance facilitates the alignment of these tasks. Nevertheless, optimizing it involves calculating the Hessian matrix. For DNNs, directly computing such a large-dimensional Hessian matrix entails prohibitive time and memory overheads. Therefore, an appropriate approximation method should be implemented in this calculation. Dandi et al. (2022) point out that gradient alignment in sequential updates for SGD originates from evaluating the gradient of a minibatch i after an additional displacement in the direction of $-(\nabla\bar{\mathcal{L}}(\boldsymbol{\theta})-\nabla\mathcal{L}_i(\boldsymbol{\theta}))$. Consequently, we can mimic this gradient alignment characteristic by using gradients for each task i that are computed following an initial displacement given by $\boldsymbol{v}_i(\boldsymbol{\theta}) = \beta\left(\nabla\bar{\mathcal{L}}(\boldsymbol{\theta})-\nabla\mathcal{L}_i(\boldsymbol{\theta})\right)$.

Theorem 4.1. *If $\mathcal{L}_i(\boldsymbol{\theta})$ has Lipschitz Hessian, then the update gradient $\boldsymbol{g}_{IGR} = \frac{1}{m}\sum_{i=0}^{m-1}\nabla\mathcal{L}_i(\boldsymbol{\theta}-\boldsymbol{v}_i(\boldsymbol{\theta}))$, calculated following an initial displacement applied to the parameters $\boldsymbol{\theta}$, can be expressed as follows:*

$$\boldsymbol{g}_{IGR}=\nabla\bar{\mathcal{L}}(\boldsymbol{\theta})+\underbrace{\frac{\beta}{2m}\nabla(\sum_{i=0}^{m-1}\|\nabla\mathcal{L}_i(\boldsymbol{\theta})-\nabla\bar{\mathcal{L}}(\boldsymbol{\theta})\|^2)}_{Gradient\ Regularization}+\mathcal{O}(\beta^2),$$

i.e., *implicitly minimizes the trace of the covariance matrix for the conflicting task gradients, along with the Empirical Risk Minimization (ERM) gradient on the mean loss $\bar{\mathcal{L}}(\boldsymbol{\theta})$.*

The above theorem implicitly penalizes "non-uniform" regions to align conflicting tasks, minimizing the gradient discrepancy (effects of implicit gradient regularization are validated in Fig. 5). It guides the meta-model towards a unified direction for learning task-shared representations. Thus, we mitigate the discrepancy of $\boldsymbol{\theta}$ trained from different pre-trained models, reducing the heterogeneity term.

To conclude, we concisely outline the pipeline of the proposed framework in Alg. 1.

5. Experiments

In this section, we empirically verify the effectiveness of our proposed approach via extensive experiments.

5.1. Experimental Setup

Datasets and pre-trained models. We conduct experiments on two widely-used DFML benchmark datasets, and one fine-grained dataset, including *miniImageNet* (Ravi & Larochelle, 2017), *CIFAR-FS* (Bertinetto et al., 2019) and *CUB* (Wah et al., 2011). Following standard splits, we split each dataset into the meta-training, meta-validating and meta-testing subsets with disjoint label spaces. In the DFML setting, we have no access to the meta-training data. Following Wang et al. (2022b); Hu et al. (2023b) for a fair comparison, we collect 100 models pre-trained on 100 N-

Table 1. Compare to existing baselines in DFML. We report on two categories of DFML algorithms: the non-inversion methods and the inversion-based methods. All results are the average accuracy with 95% confidence intervals. The best results are shown in bold.

Method	*CIFAR-FS*		*miniImageNet*		*CUB*	
	5-way 1-shot	5-way 5-shot	5-way 1-shot	5-way 5-shot	5-way 1-shot	5-way 5-shot
Finetune	28.59 ± 0.56	34.77 ± 0.62	25.06 ± 0.50	28.10 ± 0.52	26.26 ± 0.48	29.89 ± 0.55
Best-Model	21.68 ± 0.66	25.05 ± 0.67	22.86 ± 0.61	26.26 ± 0.63	24.16 ± 0.73	29.16 ± 0.73
Average	23.96 ± 0.53	27.04 ± 0.51	23.79 ± 0.48	27.49 ± 0.50	24.53 ± 0.46	28.00 ± 0.47
OTA (Singh & Jaggi, 2020)	29.10 ± 0.65	34.33 ± 0.67	24.22 ± 0.53	27.22 ± 0.59	24.23 ± 0.60	25.42 ± 0.63
DRO (Wang et al., 2022b)	30.43 ± 0.43	36.21 ± 0.51	27.56 ± 0.48	30.19 ± 0.43	28.33 ± 0.69	31.24 ± 0.76
PURER (Hu et al., 2023a)	38.66 ± 0.78	51.95 ± 0.79	31.14 ± 0.63	40.86 ± 0.64	30.08 ± 0.59	40.93 ± 0.66
BiDf-MKD (Hu et al., 2023b)	37.66 ± 0.75	51.16 ± 0.79	30.66 ± 0.59	42.30 ± 0.64	31.62 ± 0.60	44.32 ± 0.69
Ours	**43.58** ± **0.82**	**57.49** ± **0.79**	**36.20** ± **0.71**	**49.00** ± **0.71**	**33.87** ± **0.63**	**47.37** ± **0.73**

Table 2. Compare to baselines in a multi-domain scenario.

Method	*CIFAR-FS/miniImageNet/CUB*	
	5-way 1-shot	5-way 5-shot
Finetune	24.85 ± 0.54	28.35 ± 0.61
PURER (Hu et al., 2023a)	29.67 ± 0.59	37.96 ± 0.60
BiDf-MKD (Hu et al., 2023b)	31.44 ± 0.64	40.96 ± 0.59
Ours	**32.04** ± **0.63**	**44.88** ± **0.65**

Table 3. Compare to baselines in a multi-architecture scenario.

Method	Conv4/ResNet-10/ResNet-18	
	5-way 1-shot	5-way 5-shot
Finetune	28.59 ± 0.56	34.77 ± 0.62
PURER (Hu et al., 2023a)	39.15 ± 0.70	49.08 ± 0.74
BiDf-MKD (Hu et al., 2023b)	38.08 ± 0.80	50.58 ± 0.81
Ours	**44.18** ± **0.87**	**57.15** ± **0.81**

way tasks sampled from the meta-training subset, and those models are used as the meta-training resources.

Implementation details. For model architecture, we use Conv4 as the meta-model's backbone, a standard choice in meta-learning (Wu et al., 2023a). The architecture of pre-trained models includes Conv4, ResNet-10, and ResNet-18. If not specifically stated, Conv4 is the default architecture. We use an off-the-shelf ResNet-34 pre-trained on ImageNet as our shared pre-trained model to compute the FIM. For hyperparameters, we configure the number of task groups c to 5. The step size β for the displacement is set to 0.001, and the size of minibatches m is set to 4. The memory bank $\mathcal{B}$ is limited to store 20 tasks. We report the average accuracy over 600 meta-testing tasks. App. A details other setups.

Compared baselines. (i) Finetune. Train a classifier from scratch using the support set for each meta-testing task. **(ii) Best-Model.** We select the pre-trained model with the highest reported accuracy to predict the query set. **(iii) Average.** Average all pre-trained models and then finetune it using the support set. **(iv) OTA** (Singh & Jaggi, 2020). Compute the weighted average of all pre-trained models and then fine-tune using the support set. **(v) DRO** (Wang et al., 2022b). Meta-learn a hyper-network to fuse all pre-trained models into one single model, which serves as the meta-initialization and can be adapted to each meta-testing task using the support set. **(vi) PURER** (Hu et al., 2023a). Adversarially train the meta-model with a learnable dataset, where a batch of pseudo tasks is sampled for meta-training at each iteration. **(vii) BiDf-MKD** (Hu et al., 2023b). A bi-level data-free knowledge distillation framework implemented in the white-box setting to transfer meta knowledge.

5.2. Main Results

Comparisons with baselines. Tab. 1 presents the results for few-shot classification compared with baselines. To evaluate the effectiveness of our approach, we compare it against two categories of DFML algorithm: the non-inversion methods, which enable model fusion in the parameter space, and the inversion-based methods that generate pseudo data for training. As indicated in Tab. 1, we significantly outperform all other methods. For 1-shot learning, we outperform the leading baseline BiDf-MKD by 5.92%, 5.54% and 2.25% on three datasets. For 5-shot learning, we achieve an improvement of 6.33%, 6.70% and 3.05%, respectively.

Multi-domain scenario. We carry out experiments under a challenging multi-domain scenario, where each pre-trained model is designed to address different tasks from *CIFAR-FS*, *miniImageNet*, and *CUB*. For meta-testing, the meta-model is evaluated on unseen tasks that span across three datasets. Our results are shown in Tab. 2, outperforming the baseline by 7.19% and 16.53% in 1-shot and 5-shot learning, respectively. This demonstrates that our approach effectively aligns heterogeneous models from different domains, optimizing them towards a unified direction, thereby achieving robust generalization across various domains.

Multi-architecture scenario. We also conduct experiments in a multi-architecture scenario where each pre-trained model differs in architecture. For each pre-trained

Table 4. Effects of the proposed modules.

Setting	Module		*CIFAR-FS*		*miniImageNet*		*CUB*	
	Group	**IGR**	5-way 1-shot	5-way 5-shot	5-way 1-shot	5-way 5-shot	5-way 1-shot	5-way 5-shot
Vanilla			41.84 ± 0.82	55.01 ± 0.77	34.96 ± 0.70	47.24 ± 0.69	32.28 ± 0.65	44.82 ± 0.67
	✓		43.06 ± 0.81	55.84 ± 0.76	36.01 ± 0.70	48.53 ± 0.69	32.88 ± 0.60	47.25 ± 0.66
		✓	43.25 ± 0.85	55.54 ± 0.74	**36.78 ± 0.70**	48.35 ± 0.71	33.06 ± 0.64	**47.92 ± 0.69**
Ours	✓	✓	**43.58 ± 0.82**	**57.49 ± 0.79**	36.20 ± 0.71	**49.00 ± 0.71**	**33.87 ± 0.63**	47.37 ± 0.73

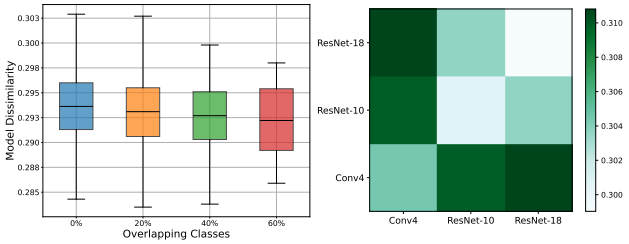
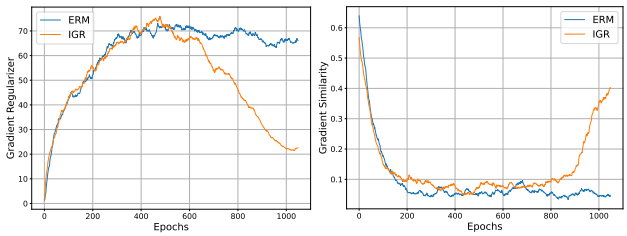


Figure 4. Model dissimilarity measured by FIM accurately reflects the model heterogeneity. We assess model heterogeneity by overlapping training classes and distinct model architectures.



(a) Gradient regularizer loss (b) Gradient cosine similarity

Figure 5. The gradient discrepancy across tasks. We plot the progression of gradient regularizer loss and gradient cosine similarity during training, confirming the efficacy of IGR in implicitly minimizing the gradient regularizer and aligning gradient directions.

model, the architecture is randomly chosen from Conv4, ResNet-10, and ResNet-18. The results are presented in Tab. 3. It outperforms all baselines and can apply to multi-architecture scenario without any change. This flexibility is attributable to our method's lack of constraints on the underlying architecture or scale of the pre-trained models, which enables effective alignment across heterogeneous models.

5.3. Ablation Studies

Effects of each module. Tab. 4 assesses the contribution of each module to performance. Initially, we introduce the vanilla only performing meta-model optimization. The vanilla still manages to outperform the baselines mentioned in Tab. 1, indicating its capacity for rapid adaptation to unseen tasks. When we integrate task grouping into the approach, we observe significant improvements. This indicates that training heterogeneous models together enables the meta-model to learn shared representations across tasks. However, this method (*i.e.*, ERM) merely aggregates losses from multiple tasks to update the meta-model, without aligning the optimization directions across these tasks. By adding implicit gradient regularization, we observe a significant improvement, demonstrating the effectiveness of the conflicting task regularization. With all modules, we achieve the best performance with a boosting improvement of 2%-3%, showing their complementarity.

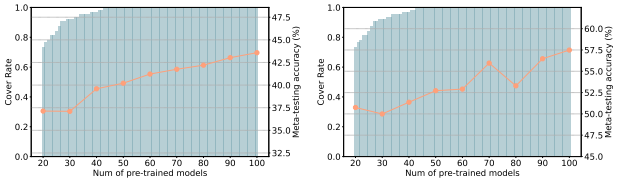
Model dissimilarity measurement. We also confirm that our model dissimilarity measurement accurately reflects model heterogeneity, facilitating the grouping of pre-trained models. We assess the model heterogeneity using the ratio of overlapping classes and model architectures. The premise is that fewer overlapping training classes or differing architectures imply higher model heterogeneity. We conduct experiments with 100 pre-trained Conv4 models, as shown in Fig. 4 (left). There is a negative correlation between the ratio of overlapping classes and the model dissimilarity. In Fig. 4 (right), we select 100 each of Conv4, ResNet-10, and ResNet-18. Results indicate that models with different architectures exhibit higher dissimilarity, with the greatest dissimilarity observed between Conv4 and ResNet-18.

Gradient discrepancy minimization. We incorporate implicit gradient regularization (IGR) to align gradients across different tasks, steering the meta-model in a unified direction. To ascertain whether our approach indeed minimizes gradient discrepancies, we plot the progression of gradient regularizer loss and gradient cosine similarity during training with different objectives, as depicted in Fig. 5. We train both IGR (blue) and ERM (orange) on *CIFAR-FS* until convergence, while tracking the gradient regularizer $\frac{1}{2m}\sum_{i=0}^{m-1}||\nabla\mathcal{L}_i(\boldsymbol{\theta})-\nabla\bar{\mathcal{L}}(\boldsymbol{\theta})||^2$ across tasks from heterogeneous pre-trained models. To ensure a fair comparison, we use the exact same sequence of tasks for IGR and ERM.

As shown in Fig. 5, the gradient regularizer of IGR gradually decreases with ongoing training. Conversely, ERM maintains high values until the end. Moreover, we evaluate gradient cosine similarity, which measures the angle between gradients of two tasks. It is clear that during training, the gradient similarity of IGR increases, whereas it remains constant for ERM. These observations confirm the efficacy of IGR in implicitly minimizing the gradient regularizer and aligning gradient directions.

Table 5. Ablation studies on the four loss components.

Method	*CIFAR-FS*		*miniImageNet*		*CUB*	
	5-way 1-shot	5-way 5-shot	5-way 1-shot	5-way 5-shot	5-way 1-shot	5-way 5-shot
w/o $\mathcal{L}_{CE}$	32.36 $\pm$ 0.59	46.22 $\pm$ 0.73	31.43 $\pm$ 0.56	44.36 $\pm$ 0.65	28.20 $\pm$ 0.55	36.58 $\pm$ 0.58
w/o $\mathcal{L}_{BN}$	41.28 $\pm$ 0.80	54.71 $\pm$ 0.76	34.18 $\pm$ 0.59	45.34 $\pm$ 0.64	33.41 $\pm$ 0.61	47.29 $\pm$ 0.69
w/o soft labels	42.87 $\pm$ 0.84	55.53 $\pm$ 0.78	36.03 $\pm$ 0.70	47.49 $\pm$ 0.74	32.71 $\pm$ 0.64	46.74 $\pm$ 0.69
w/o hard labels	43.54 $\pm$ 0.82	55.75 $\pm$ 0.77	35.70 $\pm$ 0.70	47.96 $\pm$ 0.74	33.52 $\pm$ 0.62	**47.72** $\pm$ **0.70**
w/ all components	**43.58** $\pm$ **0.82**	**57.49** $\pm$ **0.79**	**36.20** $\pm$ **0.71**	**49.00** $\pm$ **0.71**	**33.87** $\pm$ **0.63**	47.37 $\pm$ 0.73



(a) 5-way 1-shot (b) 5-way 5-shot

Figure 6. The relationship between the accuracy and the number of pre-trained models.

Effects of the number of pre-trained models. We carry out experiments with varying numbers of pre-trained models, each individually pre-trained on a 5-way subset of *CIFAR-FS*. Our method demonstrates scalability to any number of pre-trained models, with a consistent improvement in performance as more models are added (refer to Fig. 6). We also introduce an intrinsic factor termed the "cover rate", which reflects the coverage rate of classes within the meta-training subset. As the number of pre-trained models grows, so does the cover rate. Notably, even when the cover rate reaches 100% (which occurs with more than 50 pre-trained models), we continue to observe performance enhancements. This improvement is attributed to the increased diversity of pre-trained models, providing a more comprehensive understanding of various classes. This, in turn, strengthens the generalization ability of the meta-model when encountering previously unseen classes.

Effects of the number of task groupings. For task grouping algorithms, we conduct experiments with varying numbers of task groupings. As shown in Table 6, performance initially improves with an increasing number of groupings, while it diminishes when the number of groupings becomes overly large. This is because a proper number of groupings enables learning of shared representations among heterogeneous models. However, too many groups result in a decreased number of tasks per group, thereby reducing the benefits derived from inter-task information sharing. In future work, we plan to explore methods for automatically selecting an appropriate number of groupings, *e.g.*, by utilizing cluster evaluation metrics (Maulik & Bandyopadhyay, 2002) such as the Calinski-Harabasz index.

Table 6. Effects of the number of task groupings.

Number	1	2	3	4	5	6	7	8	9
5-way 5-shot	48.11	48.79	48.20	48.75	49.00	49.37	49.31	49.24	48.02

Effects of the loss components. The classification loss $\mathcal{L}_{CE}$ is introduced for class-specific generation, crucial for the following meta-learning process. It prevents the generated images from becoming a blend of attributes from multiple classes. The regularization loss $\mathcal{L}_{BN}$ utilizes Batch Normalization statistics to ensure the synthetic images' distribution aligns closer to the original images. Soft labels convey the teacher model's knowledge, utilizing semantic relationships for richer supervision. Hard labels, as the ground truth, offer precise supervisory information to prevent errors introduced by soft labels. As shown in Tab. 5, removing $\mathcal{L}_{CE}$, $\mathcal{L}_{BN}$, soft labels, and hard labels results in average performance decreases of 8.06%, 1.88%, 1.02%, and 0.55%, respectively. On the *CUB* dataset, hard labels even have a negative impact, possibly due to the overly strong one-hot assumption when distinguishing fine-grained categories. Tab. 5 indicate that the classification loss and soft labels loss have a considerable impact on the results.

6. Conclusion

Current DFML methods often overlook the heterogeneity among pre-trained models. In this paper, we empirically and theoretically rethink the model heterogeneity in DFML. We find that the model heterogeneity introduces a heterogeneity-homogeneity trade-off. Based on our findings, we propose Task Groupings Regularization that benefits from model heterogeneity by grouping and aligning conflicting tasks within each group, to capture shared representations that generalize across tasks. Comprehensive experiments showcase the superiority of our approach in multiple benchmarks. Exploring its extension to other domains, such as NLP, represents a valuable direction for future research.

Acknowledgements

This work was supported by the National Key R&D Program of China (2022YFB4701400/4701402), SSTIC Grant(KJZD20230923115106012), Shenzhen Key Labora-

tory (ZDSYS20210623092001004), and Beijing Key Lab of Networked Multimedia. Dr Tao's research is partially supported by NTU RSR and Start Up Grants.

Impact Statement

Data-Free Meta-Learning (DFML) provides an effective resolution to the ethical, privacy, and labor challenges posed by large-scale image datasets, which often result in the retraction or restriction of these resources and prompt a shift toward using pre-trained models for research. However, while DFML mitigates many of these concerns, the model inversion technique it employs could potentially expose information from the original training data, presenting a risk of data leakage. Despite this, DFML significantly benefits socially by enabling few-shot learning in data-limited environments, thus promoting substantial positive impacts in fields where data scarcity is a challenge.

References

Achille, A., Lam, M., Tewari, R., Ravichandran, A., Maji, S., Fowlkes, C. C., Soatto, S., and Perona, P. Task2vec: Task embedding for meta-learning. In *ICCV*, pp. 6430–6439, 2019.

Amari, S.-I. Natural gradient works efficiently in learning. *Neural Computation*, pp. 251–276, 1998.

Bertinetto, L., Henriques, J. F., Torr, P., and Vedaldi, A. Meta-learning with differentiable closed-form solvers. In *ICLR*, 2019.

Caruana, R. Multitask learning: A knowledge-based source of inductive bias1. In *ICML*, pp. 41–48, 1993.

Cerviño, J., Bazerque, J. A., Calvo-Fullana, M., and Ribeiro, A. Multi-task bias-variance trade-off through functional constraints. In *ICASSP*, pp. 1–5, 2023.

Chen, B., Yan, Z., Li, K., Li, P., Wang, B., Zuo, W., and Zhang, L. Variational attention: Propagating domain-specific knowledge for multi-domain learning in crowd counting. In *ICCV*, pp. 16065–16075, 2021.

Chen, H., Wang, Y., Xu, C., Yang, Z., Liu, C., Shi, B., Xu, C., Xu, C., and Tian, Q. Data-free learning of student networks. In *ICCV*, pp. 3514–3522, 2019.

Chen, J., Zhan, L.-M., Wu, X.-M., and Chung, F.-l. Variational metric scaling for metric-based meta-learning. In *AAAI*, pp. 3478–3485, 2020.

Dandi, Y., Barba, L., and Jaggi, M. Implicit gradient alignment in distributed and federated learning. In *AAAI*, pp. 6454–6462, 2022.

Eshratifar, A. E., Eigen, D., and Pedram, M. Gradient agreement as an optimization objective for meta-learning. *arXiv preprint arXiv:1810.08178*, 2018.

Fifty, C., Amid, E., Zhao, Z., Yu, T., Anil, R., and Finn, C. Efficiently identifying task groupings for multi-task learning. In *NeurIPS*, pp. 27503–27516, 2021.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, pp. 1126–1135, 2017.

Fredrikson, M., Jha, S., and Ristenpart, T. Model inversion attacks that exploit confidence information and basic countermeasures. In *SIGSAC*, pp. 1322–1333, 2015.

Han, M., Zhan, Y., Yu, B., Luo, Y., Hu, H., Du, B., Wen, Y., and Tao, D. Region-adaptive concept aggregation for few-shot visual recognition. *MIR*, pp. 554–568, 2023.

Hao, Z., Guo, J., Han, K., Tang, Y., Hu, H., Wang, Y., and Xu, C. One-for-all: Bridge the gap between heterogeneous architectures in knowledge distillation. In *NeurIPS*, 2023.

Hermann, K. and Lampinen, A. What shapes feature representations? Exploring datasets, architectures, and training. In *NeurIPS*, pp. 9995–10006, 2020.

Hu, Z., Shen, L., Wang, Z., Liu, T., Yuan, C., and Tao, D. Architecture, dataset and model-scale agnostic data-free meta-learning. In *CVPR*, pp. 7736–7745, 2023a.

Hu, Z., Shen, L., Wang, Z., Wu, B., Yuan, C., and Tao, D. Learning to learn from apis: Black-box data-free meta-learning. In *ICML*, 2023b.

Huang, Z., Zhou, J. T., Peng, X., Zhang, C., Zhu, H., and Lv, J. Multi-view spectral clustering network. In *IJCAI*, pp. 2563–2569, 2019.

Jiang, J., Chen, B., Pan, J., Wang, X., Liu, D., Long, M., et al. Forkmerge: Mitigating negative transfer in auxiliary-task learning. In *NeurIPS*, 2023.

Kornblith, S., Norouzi, M., Lee, H., and Hinton, G. Similarity of neural network representations revisited. In *ICML*, pp. 3519–3529, 2019.

Kurin, V., De Palma, A., Kostrikov, I., Whiteson, S., and Mudigonda, P. K. In defense of the unitary scalarization for deep multi-task learning. In *NeurIPS*, pp. 12169–12183, 2022.

Kwon, N., Na, H., Huang, G., and Lacoste-Julien, S. Repurposing pretrained models for robust out-of-domain few-shot learning. In *ICLR*, 2020.

Li, W., Peng, Y., Zhang, M., Ding, L., Hu, H., and Shen, L. Deep model fusion: A survey. *arXiv preprint arXiv:2309.15698*, 2023.

Liu, C., Wang, Z., Sahoo, D., Fang, Y., Zhang, K., and Hoi, S. C. Adaptive task sampling for meta-learning. In *ECCV*, pp. 752–769, 2020.

Liu, Y., Zhang, W., Wang, J., and Wang, J. Data-free knowledge transfer: A survey. *arXiv preprint arXiv:2112.15278*, 2021.

Maulik, U. and Bandyopadhyay, S. Performance evaluation of some clustering algorithms and validity indices. *TPAMI*, pp. 1650–1654, 2002.

McAllester, D. A. Pac-bayesian model averaging. In *COLT*, pp. 164–170, 1999.

Ng, A., Jordan, M., and Weiss, Y. On spectral clustering: Analysis and an algorithm. In *NeurIPS*, 2001.

Ravi, S. and Larochelle, H. Optimization as a model for few-shot learning. In *ICLR*, 2017.

Shi, G., Li, Q., Zhang, W., Chen, J., and Wu, X.-M. Recon: Reducing conflicting gradients from the root for multi-task learning. In *ICLR*, 2022.

Shi, Y., Seely, J., Torr, P., Siddharth, N., Hannun, A., Usunier, N., and Synnaeve, G. Gradient matching for domain generalization. In *ICLR*, 2021.

Singh, S. P. and Jaggi, M. Model fusion via optimal transport. In *NeurIPS*, pp. 22045–22055, 2020.

Smith, S. L., Dherin, B., Barrett, D., and De, S. On the origin of implicit regularization in stochastic gradient descent. In *ICLR*, 2020.

Song, T., Feng, J., Luo, L., Gao, C., and Li, H. Robust texture description using local grouped order pattern and non-local binary pattern. *TCSVT*, pp. 189–202, 2020.

Standley, T., Zamir, A., Chen, D., Guibas, L., Malik, J., and Savarese, S. Which tasks should be learned together in multi-task learning? In *ICML*, pp. 9120–9132, 2020.

Truong, J.-B., Maini, P., Walls, R. J., and Papernot, N. Data-free model extraction. In *CVPR*, pp. 4771–4780, 2021.

Wah, C., Branson, S., Welinder, P., Perona, P., and Belongie, S. The Caltech-UCSD birds-200-2011 dataset. Technical report, California Institute of Technology, 2011.

Wang, Z., Zhao, Y., Yu, P., Zhang, R., and Chen, C. Bayesian meta sampling for fast uncertainty adaptation. In *International Conference on Learning Representations*, 2020.

Wang, Z., Shen, L., Duan, T., Zhan, D., Fang, L., and Gao, M. Learning to learn and remember super long multi-domain task sequence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7982–7992, 2022a.

Wang, Z., Wang, X., Shen, L., Suo, Q., Song, K., Yu, D., Shen, Y., and Gao, M. Meta-learning without data via wasserstein distributionally-robust model fusion. In *UAI*, pp. 2045–2055, 2022b.

Wei, Y. and Wei, X.-S. Task-specific part discovery for fine-grained few-shot classification. *MIR*, 2024. in press.

Wu, B., Fang, J., Zeng, X., Liang, S., and Zhang, Q. Adaptive compositional continual meta-learning. In *ICML*, pp. 37358–37378, 2023a.

Wu, C., Wang, T., Ge, Y., Lu, Z., Zhou, R., Shan, Y., and Luo, P. π-tuning: Transferring multimodal foundation models with optimal multi-task interpolation. In *ICML*, pp. 37713–37727, 2023b.

Ye, F., Lin, B., Yue, Z., Guo, P., Xiao, Q., and Zhang, Y. Multi-objective meta learning. In *NeurIPS*, pp. 21338–21351, 2021.

Yin, H., Molchanov, P., Alvarez, J. M., Li, Z., Mallya, A., Hoiem, D., Jha, N. K., and Kautz, J. Dreaming to distill: Data-free knowledge transfer via deepinversion. In *CVPR*, pp. 8715–8724, 2020.

Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., and Finn, C. Gradient surgery for multi-task learning. In *NeurIPS*, pp. 5824–5836, 2020.

Zamir, A. R., Sax, A., Shen, W., Guibas, L. J., Malik, J., and Savarese, S. Taskonomy: Disentangling task transfer learning. In *CVPR*, pp. 3712–3722, 2018.

Zhang, Y. and Yang, Q. A survey on multi-task learning. *TKDE*, pp. 5586–5609, 2021.

Zhao, Y., Zhang, H., and Hu, X. Penalizing gradient norm for efficiently improving generalization in deep learning. In *ICML*, pp. 26982–26992, 2022.

Zheng, H., Shen, L., Tang, A., Luo, Y., Hu, H., Du, B., and Tao, D. Learn from model beyond fine-tuning: A survey. *arXiv preprint arXiv:2310.08184*, 2023.

A. Implementation Details

The architecture for the meta-model is based on Conv4. Common in meta-learning studies (Chen et al., 2020; Liu et al., 2020), Conv4 consists of four convolutional blocks, with each block comprising 32 3×3 filters, a BatchNorm layer, a ReLU function, and a 2×2 max-pooling layer. All pre-trained models are constructed to solve various 5-way tasks by randomly choosing 5 classes from the meta-training set (be it *CIFAR-FS*, *miniImageNet*, or *CUB*). With the Adam optimizer, these models are pre-trained using standard supervised learning at a learning rate of 0.01. We collect a set of 100 pre-trained models following previous works. For computing the FIM, we use the off-the-shelf ResNet-34[2] pre-trained on ImageNet as our shared model. The total epoch T is set to 1000 for *CIFAR-FS* and *CUB*, and 1200 for *miniImageNet*. For the task recovery, we generate $\hat{\boldsymbol{X}}_i$ with 30 images in each epoch. The generator parameters $\boldsymbol{\theta}_G$ and latent code $\boldsymbol{Z}$ are optimized simultaneously using the Adam optimizer, aiming to minimize Eq. (2) over 200 steps with a learning rate of 0.001. Similarly, the meta-model parameters $\boldsymbol{\theta}$ are updated with gradient $\boldsymbol{g}_{IGR}$ with a learning rate set at 0.001. For the cross-task replay, we employ MAML to execute meta-learning on tasks sampled in the memory bank. Following Hu et al. (2023a;b), we use the Adam optimizer, set with inner and outer loop learning rates of 0.01 and 0.001, respectively.

In the multi-domain scenario, pre-trained models are designed to solve different 5-way tasks, which are devised by randomly sampling 5 classes from the meta-training sets, including *CIFAR-FS*, *miniImageNet*, and *CUB* (Han et al., 2023; Wei & Wei, 2024). We also take Conv4 as the architecture of pre-trained models and the meta-model. For meta-testing, we evenly construct 1800 meta-testing tasks across all meta-testing sets, including *CIFAR-FS*, *miniImageNet*, and *CUB*.

In the multi-architecture scenario, we focus on pre-trained models with different architectures. Specifically, we consider Conv4, ResNet-10, and ResNet-18 as the diverse architectures represented within these pre-trained models. ResNet-10 and ResNet-18 are larger-scale neural networks compared to Conv4. We adopt Conv4 as the architecture for the meta-model. All pre-trained models are designed for solving diverse 5-way tasks, which are constructed by randomly sampling 5 classes from *CIFAR-FS*. All other configurations remain consistent with the above settings.

B. Architecture of the Generator

The structure of the generator for task recovery is detailed in Tab. 7. This generator processes standard Gaussian noise as input to produce the recovered data. The dimension of the Gaussian noise data, $\boldsymbol{z}$, is designated as 256 ($d_{\boldsymbol{z}}$). For the LeakyReLU activation, the $negative_slope$ parameter is set to 0.2. We specify the image size to be 32 for models pre-trained on *CIFAR-FS* and 84 for those pre-trained on *miniImageNet* and *CUB*. The number of channels (nc) for color image recovery is set to 3, and the number of convolutional filters (nf) is set to 64.

Table 7. Detailed structure of the generator. We highlight the dimension change in red.

Notion	Description
$img_size \times img_size$	resolution of recovered image
bs	batch size
nc	number of channels of recovered image
nf	number of convolutional filters
FC($\cdot$)	fully connected layer;
BN($\cdot$)	batch normalization layer
Conv2D($input$, $output$,$filter_size$, $stride$, $padding$)	convolutional layer

Structure	Dimension	
	Before	After
$\boldsymbol{z} \in \mathbb{R}_{d_{\boldsymbol{z}}} \sim \mathcal{N}(\mathbf{0}, \mathbf{1})$	$[\,1, d_{\boldsymbol{z}}\,]$	$[\,bs, d_{\boldsymbol{z}}\,]$
FC($\boldsymbol{Z}$)	$[\,bs, d_{\boldsymbol{z}}\,]$	$[\,bs, 2 \times nf \times (img_size//4) \times (img_size//4)\,]$
Reshape	$[\,bs, 2 \times nf \times (img_size//4) \times (img_size//4)\,]$	$[\,bs, 2 \times nf, (img_size//4), (img_size//4)\,]$
BN	$[\,bs, 2 \times nf, (img_size//4), (img_size//4)\,]$	$[\,bs, 2 \times nf, (img_size//4), (img_size//4)\,]$
Upsampling	$[\,bs, 2 \times nf, (img_size//4), (img_size//4))\,]$	$[\,bs, 2 \times nf, (img_size//2), (img_size//2))\,]$
Conv2D($2 \times nf$, $2 \times nf$, 3, 1, 1)	$[\,bs, 2 \times nf, (img_size//2), (img_size//2))\,]$	$[\,bs, 2 \times nf, (img_size//2), (img_size//2))\,]$
BN, LeakyReLU	$[\,bs, 2 \times nf, (img_size//2), (img_size//2))\,]$	$[\,bs, 2 \times nf, (img_size//2), (img_size//2))\,]$
Upsampling	$[\,bs, 2 \times nf, (img_size//2), (img_size//2))\,]$	$[\,bs, 2 \times nf, img_size, img_size\,]$
Conv2D($2 \times nf$, nf, 3, 1, 1)	$[\,bs, 2 \times nf, img_size, img_size\,]$	$[\,bs, nf, img_size, img_size\,]$
BN, LeakyReLU	$[\,bs, nf, img_size, img_size\,]$	$[\,bs, nf, img_size, img_size\,]$
Conv2D(nf, nc, 3, 1, 1)	$[\,bs, nf, img_size, img_size\,]$	$[\,bs, nc, img_size, img_size\,]$
Sigmoid	$[\,bs, nc, img_size, img_size\,]$	$[\,bs, nc, img_size, img_size\,]$

[2]https://pytorch.org/vision/stable/models/generated/torchvision.models.resnet34

C. Theoretical Proofs

Theorem C.1. *Assume $M_{meta}(\cdot;\boldsymbol{\theta})$ is probably approximately correct (PAC),* i.e.*, there exists $\zeta(N,\delta)\geq 0$ monotonically decreasing with N, and the loss function $\ell(\cdot)$ is K-Lipschitz continuous. Then, with probability at least $1-2\delta$ the following bounds hold:*

$$\begin{aligned}&|E(\boldsymbol{\theta}(M_{\text{bas}}),\boldsymbol{\theta}(M_{\text{aux}}))-\hat{E}(\boldsymbol{\theta}(M_{\text{bas}},M_{\text{aux}}))|\\ \leq&\zeta(N,\delta)+K\sum_{t=\text{bas}}^{\text{aux}}\mathbb{E}_{\mathcal{P}_t}[|M_{meta}(\boldsymbol{\theta}(M_t))-M_{meta}(\boldsymbol{\theta}(M_{\text{bas}},M_{\text{aux}}))|],\end{aligned}$$

where $E=\mathbb{E}_{\mathcal{P}_{\text{bas}}}[\ell\left(M_{meta}(x,\boldsymbol{\theta}(M_{\text{bas}}))\right)]+\mathbb{E}_{\mathcal{P}_{\text{aux}}}[\ell\left(M_{meta}(x,\boldsymbol{\theta}(M_{\text{aux}}))\right)]$ is the expected error over the merged distribution and $\hat{E}=\frac{1}{N}\sum_{i=1}^{N}\ell\left(M_{meta}(x_i,\boldsymbol{\theta}(M_{\text{bas}},M_{\text{aux}}))\right)$ is the empirical error over the N training samples.

Proof. Using PAC-Bayesian generalization bound (McAllester, 1999), for all $\boldsymbol{\theta}\in\boldsymbol{\Theta}$, with probability $1-\delta$ over independent draws $x_i\sim\mathcal{P}_t$, we have:

$$\left|\mathbb{E}_{\mathcal{P}_t}[\ell(M_{meta}(x,\boldsymbol{\theta}))]-\frac{1}{N}\sum_{i=1}^{N}\ell(M_{meta}(x_i,\boldsymbol{\theta}))\right|\leq\zeta(N,\delta)=\sqrt{\frac{\log|\boldsymbol{\Theta}|+\log\frac{1}{\delta}}{2N}}.$$

This comes as a generalization of the law of large numbers for the case in which samples are *i.i.d.*, where the error is of order $1/N$. If the loss function $\ell(\cdot)$ is K-Lipschitz continuous, we have:

$$|\ell(M_{meta}(x,\boldsymbol{\theta}_1))-\ell(M_{meta}(x,\boldsymbol{\theta}_2))|\leq K\left|M_{meta}(x,\boldsymbol{\theta}_1)-M_{meta}(x,\boldsymbol{\theta}_2)\right|.$$

By a direct application of the triangle inequality and the PAC bound over the merged probability distribution, *i.e.*, $\bar{\mathcal{P}}=\mathcal{P}_{\text{bas}}+\mathcal{P}_{\text{aux}}$, and taking the Lipschitz assumption over the solutions, then:

$$\begin{aligned}&|E(\boldsymbol{\theta}(M_{\text{bas}}),\boldsymbol{\theta}(M_{\text{aux}}))-\hat{E}(\boldsymbol{\theta}(M_{\text{bas}},M_{\text{aux}}))|\\ =&\left|\mathbb{E}_{\mathcal{P}_{\text{bas}}}[\ell\left(M_{meta}(x,\boldsymbol{\theta}(M_{\text{bas}}))\right)]+\mathbb{E}_{\mathcal{P}_{\text{aux}}}[\ell\left(M_{meta}(x,\boldsymbol{\theta}(M_{\text{aux}}))\right)]-\frac{1}{N}\sum_{i=1}^{N}\ell\left(M_{meta}(x_i,\boldsymbol{\theta}(M_{\text{bas}},M_{\text{aux}}))\right)\right|\\ \leq&\zeta(N,\delta)+K\mathbb{E}_{\mathcal{P}_{\text{bas}}}[|M_{meta}(x,\boldsymbol{\theta}(M_{\text{bas}}))-M_{meta}(x,\boldsymbol{\theta}(M_{\text{bas}},M_{\text{aux}}))|]\\ &+K\mathbb{E}_{\mathcal{P}_{\text{aux}}}[|M_{meta}(x,\boldsymbol{\theta}(M_{\text{aux}}))-M_{meta}(x,\boldsymbol{\theta}(M_{\text{bas}},M_{\text{aux}}))|]\\ =&\zeta(N,\delta)+K\sum_{t=\text{bas}}^{\text{aux}}\mathbb{E}_{\mathcal{P}_t}[|M_{meta}(\boldsymbol{\theta}(M_t))-M_{meta}(\boldsymbol{\theta}(M_{\text{bas}},M_{\text{aux}}))|].\end{aligned}$$

Here, we choose the notation $M_{meta}(x,\boldsymbol{\theta}(\cdot))=M_{meta}(\boldsymbol{\theta}(\cdot))$ for brevity, referring to the prediction function for inputs x.

□

Theorem C.2. *If $\mathcal{L}_i(\boldsymbol{\theta})$ has Lipschitz Hessian, then the update gradient $\boldsymbol{g}_{IGR}=\frac{1}{m}\sum_{i=0}^{m-1}\nabla\mathcal{L}_i(\boldsymbol{\theta}-\boldsymbol{v}_i(\boldsymbol{\theta}))$, calculated following an initial displacement applied to the parameters $\boldsymbol{\theta}$, can be expressed as follows:*

$$\boldsymbol{g}_{IGR}=\nabla\bar{\mathcal{L}}(\boldsymbol{\theta})+\underbrace{\frac{\beta}{2m}\nabla(\sum_{i=0}^{m-1}\|\nabla\mathcal{L}_i(\boldsymbol{\theta})-\nabla\bar{\mathcal{L}}(\boldsymbol{\theta})\|^2)}_{Gradient\ Regularization}+\mathcal{O}(\beta^2),$$

i.e.*, implicitly minimizes the trace of the covariance matrix for the conflicting task gradients, along with the Empirical Risk Minimization (ERM) gradient on the mean loss $\bar{\mathcal{L}}(\boldsymbol{\theta})$.*

Proof. If $\mathcal{L}_i(\boldsymbol{\theta})$ has Lipschitz Hessian, *i.e.*, $\|\nabla^2\mathcal{L}_i(\boldsymbol{\theta}_1) - \nabla^2\mathcal{L}_i(\boldsymbol{\theta}_2)\| \leq \rho\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|$ for some $\rho > 0$, we can apply the fundamental theorem of Taylor's expansion to the gradient $\nabla\mathcal{L}_i(\boldsymbol{\theta} - \boldsymbol{v}_i(\boldsymbol{\theta}))$, then:

$$\nabla\mathcal{L}_i(\boldsymbol{\theta} - \boldsymbol{v}_i(\boldsymbol{\theta})) = \nabla\mathcal{L}_i(\boldsymbol{\theta}) - \nabla^2\mathcal{L}_i(\boldsymbol{\theta})\boldsymbol{v}_i(\boldsymbol{\theta}) + \mathcal{O}(\|\boldsymbol{v}_i(\boldsymbol{\theta})\|^2).$$

For instance, when $\boldsymbol{v}_i(\boldsymbol{\theta}) = \beta\left(\nabla\bar{\mathcal{L}}(\boldsymbol{\theta}) - \nabla\mathcal{L}_i(\boldsymbol{\theta})\right)$, we have:

$$\nabla\mathcal{L}_i\left(\boldsymbol{\theta} - \beta(\nabla\bar{\mathcal{L}}(\boldsymbol{\theta}) - \nabla\mathcal{L}_i(\boldsymbol{\theta}))\right) = \nabla\mathcal{L}_i(\boldsymbol{\theta}) - \beta\nabla^2\mathcal{L}_i(\boldsymbol{\theta})(\nabla\bar{\mathcal{L}}(\boldsymbol{\theta}) - \nabla\mathcal{L}_i(\boldsymbol{\theta})) + \mathcal{O}(\beta^2).$$

Therefore, the update gradient $\boldsymbol{g}_{IGR}$ calculated after an displacement can be expressed as follows:

$$\begin{aligned}
\boldsymbol{g}_{IGR} &= \frac{1}{m}\sum_{i=0}^{m-1}\nabla\mathcal{L}_i(\boldsymbol{\theta} - \boldsymbol{v}_i(\boldsymbol{\theta})) \\
&= \frac{1}{m}\sum_{i=0}^{m-1}\nabla\mathcal{L}_i(\boldsymbol{\theta}) - \frac{\beta}{m}\sum_{i=0}^{m-1}\nabla^2\mathcal{L}_i(\boldsymbol{\theta})\left(\nabla\bar{\mathcal{L}}(\boldsymbol{\theta}) - \nabla\mathcal{L}_i(\boldsymbol{\theta})\right) + \mathcal{O}(\beta^2) \\
&= \frac{1}{m}\sum_{i=0}^{m-1}\nabla\mathcal{L}_i(\boldsymbol{\theta}) + \frac{\beta}{m}\sum_{i=0}^{m-1}(\nabla^2\mathcal{L}_i(\boldsymbol{\theta})\nabla\mathcal{L}_i(\boldsymbol{\theta}) - \nabla^2\mathcal{L}_i(\boldsymbol{\theta})\nabla\bar{\mathcal{L}}(\boldsymbol{\theta})) + \mathcal{O}(\beta^2),
\end{aligned}$$

given that $\sum_{i=0}^{m-1}(\nabla\mathcal{L}_i(\boldsymbol{\theta}) - \nabla\bar{\mathcal{L}}(\boldsymbol{\theta})) = 0$, then:

$$\begin{aligned}
\boldsymbol{g}_{IGR} &= \nabla\bar{\mathcal{L}}(\boldsymbol{\theta}) + \frac{\beta}{m}\sum_{i=0}^{m-1}(\nabla^2\mathcal{L}_i(\boldsymbol{\theta})\nabla\mathcal{L}_i(\boldsymbol{\theta}) - \nabla^2\mathcal{L}_i(\boldsymbol{\theta})\nabla\bar{\mathcal{L}}(\boldsymbol{\theta})) + \mathcal{O}(\beta^2) \\
&= \nabla\bar{\mathcal{L}}(\boldsymbol{\theta}) + \frac{\beta}{m}\sum_{i=0}^{m-1}(\nabla^2\mathcal{L}_i(\boldsymbol{\theta})\nabla\mathcal{L}_i(\boldsymbol{\theta}) - \nabla^2\mathcal{L}_i(\boldsymbol{\theta})\nabla\bar{\mathcal{L}}(\boldsymbol{\theta}) - \nabla^2\bar{\mathcal{L}}(\boldsymbol{\theta})\nabla\mathcal{L}_i(\boldsymbol{\theta}) + \nabla^2\bar{\mathcal{L}}(\boldsymbol{\theta})\nabla\bar{\mathcal{L}}(\boldsymbol{\theta})) + \mathcal{O}(\beta^2) \\
&= \nabla\bar{\mathcal{L}}(\boldsymbol{\theta}) + \frac{\beta}{m}\sum_{i=0}^{m-1}(\nabla^2\mathcal{L}_i(\boldsymbol{\theta}) - \nabla^2\bar{\mathcal{L}}(\boldsymbol{\theta}))(\nabla\mathcal{L}_i(\boldsymbol{\theta}) - \nabla\bar{\mathcal{L}}(\boldsymbol{\theta})) + \mathcal{O}(\beta^2) \\
&= \nabla\bar{\mathcal{L}}(\boldsymbol{\theta}) + \frac{\beta}{2m}\nabla(\sum_{i=0}^{m-1}\|\nabla\mathcal{L}_i(\boldsymbol{\theta}) - \nabla\bar{\mathcal{L}}(\boldsymbol{\theta})\|^2) + \mathcal{O}(\beta^2).
\end{aligned}$$

□

D. Empirical Observation Details

In the experiment (Fig. 2) on overlapping classes, we standardize the architecture of pre-trained models to Conv4. Initially, we select a basic model pre-trained on 5 classes of *CIFAR-FS*. Then, we choose pre-trained models with 0, 1, 2, 3, 4, and 5 overlapping training classes with the basic model, selecting 10 models for each category. In total, this amounts to 60 auxiliary models. Tasks are recovered from pre-trained models based on Eq. (2) and used as training data for the meta-model. For the recovered task $\hat{\mathcal{T}}$ with class labels $\boldsymbol{Y}$, we randomly divide it into a support set $\boldsymbol{S}$ and a query set $\boldsymbol{Q}$. Then, we employ MAML (Finn et al., 2017) to optimize the meta-model $\boldsymbol{\theta}$:

$$\begin{aligned}
\min_{\boldsymbol{\theta}} \mathbb{E}_{\hat{\mathcal{T}}}\mathcal{L}_{outer} =& CE(M_{meta}(\boldsymbol{Q};\boldsymbol{\theta}_c), \boldsymbol{Y}_Q), \\
\text{s.t. } \boldsymbol{\theta}_c = \min_{\boldsymbol{\theta}} \mathcal{L}_{inner} =& CE(M_{meta}(\boldsymbol{S};\boldsymbol{\theta}), \boldsymbol{Y}_S).
\end{aligned} \tag{9}$$

Initially, we train the meta-model with data from the basic model M_{bas} to establish a basic accuracy $\mathcal{P}(\boldsymbol{\theta}(M_{\text{bas}}))$. The accuracy is calculated by standard few-shot classification on meta-testing subsets. To measure the impact of the auxiliary model M_{aux} in joint training, we combine the losses of the basic and auxiliary models to update the meta-model, thereby obtaining a new accuracy $\mathcal{P}(\boldsymbol{\theta}(M_{\text{bas}}, M_{\text{aux}}))$ and calculating the Accuracy Gain.

In the experiment on model architecture, we keep the training classes of pre-trained models constant. The architecture of the basic model is Conv4, and we select pre-trained models with architectures of Conv4, ResNet-10, ResNet-18, and ResNet-50, 10 models for each category. The subsequent steps are the same as described above. We provide the pseudo-code for the empirical observation implementation in Alg. 2.

Algorithm 2: Pseudo-Code of the Empirical Observation

Input: A collection of auxiliary models $\mathcal{M}_{pool}$; a basic model M_{bas}; a meta-model $M_{meta}(\cdot;\boldsymbol{\theta})$.
Output: A list of the Accuracy Gain $[AG]$.
Recover data from M_{bas} to train $M_{meta}(\cdot;\boldsymbol{\theta})$
Calculate $\mathcal{P}(\boldsymbol{\theta}(M_{\text{bas}}))$ on meta-testing subset
for M_{aux} **in** $\mathcal{M}_{pool}$ **do**
  Re-initialize the meta-model $M_{meta}(\cdot;\boldsymbol{\theta})$
  Recover training data from M_{aux}
  for $\hat{\boldsymbol{X}}_{\text{bas}}, \hat{\boldsymbol{X}}_{\text{aux}}$ **in** DataLoader **do**
    Loss $= \text{MAML}(\hat{\boldsymbol{X}}_{\text{bas}};\boldsymbol{\theta}) + \text{MAML}(\hat{\boldsymbol{X}}_{\text{aux}};\boldsymbol{\theta})$
    Loss.backward()
    Update($M_{meta}(\cdot;\boldsymbol{\theta})$)
  Calculate $\mathcal{P}(\boldsymbol{\theta}(M_{\text{bas}}, M_{\text{aux}}))$ on meta-testing subset
  $AG = \mathcal{P}(\boldsymbol{\theta}(M_{\text{bas}}, M_{\text{aux}})) - \mathcal{P}(\boldsymbol{\theta}(M_{\text{bas}}))$

E. Task Grouping Strategies

It is commonly believed that related tasks with similar underlying structures can benefit from training together (Yu et al., 2020; Fifty et al., 2021). However, we argue that leveraging the model heterogeneity by grouping dissimilar pre-trained models can also be beneficial. To validate whether our insight holds true, we conduct an ablation study comparing groupings based on both dissimilarity and similarity measures. Specifically, we derive the task similarity matrix $\left[d_{\cos}\left(\frac{\boldsymbol{F}^i}{\boldsymbol{F}^i+\boldsymbol{F}^j}, \frac{\boldsymbol{F}^j}{\boldsymbol{F}^i+\boldsymbol{F}^j}\right)\right]_{1\leq i,j\leq n}$ to group the most similar pre-trained models together, keeping other steps of the process constant. As shown in Tab. 8, groupings based on dissimilarity achieve better results. This further supports our insight that grouping and aligning dissimilar tasks can effectively balance the heterogeneity-homogeneity trade-off.

Table 8. Different task grouping strategies. Grouping pre-trained models based on dissimilarity or similarity measures.

Strategy	*CIFAR-FS*		*miniImageNet*		*CUB*	
	5-way 1-shot	5-way 5-shot	5-way 1-shot	5-way 5-shot	5-way 1-shot	5-way 5-shot
Similarity	43.54 $\pm$ 0.85	55.71 $\pm$ 0.80	35.82 $\pm$ 0.68	48.73 $\pm$ 0.68	33.50 $\pm$ 0.62	46.77 $\pm$ 0.67
Dissimilarity	**43.58** $\pm$ **0.82**	**57.49** $\pm$ **0.79**	**36.20** $\pm$ **0.71**	**49.00** $\pm$ **0.71**	**33.87** $\pm$ **0.63**	**47.37** $\pm$ **0.73**