This CVPR 2021 paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Pareto Self-Supervised Training for Few-Shot Learning

Zhengyu Chen[1,2,3], Jixie Ge[1,2], Heshen Zhan[1,2], Siteng Huang[1,2], Donglin Wang[1,2*]
[1] Machine Intelligence Lab (MiLAB), AI Division, School of Engineering, Westlake University
[2] Institute of Advanced Technology, Westlake Institute for Advanced Study
[3] College of Computer Science & Technology, Zhejiang University
{chenzhengyu,gejixie,zhanheshen,huangsiteng,wangdonglin}@westlake.edu.cn

Abstract

While few-shot learning (FSL) aims for rapid generalization to new concepts with little supervision, self-supervised learning (SSL) constructs supervisory signals directly computed from unlabeled data. Exploiting the complementarity of these two manners, few-shot auxiliary learning has recently drawn much attention to deal with few labeled data. Previous works benefit from sharing inductive bias between the main task (FSL) and auxiliary tasks (SSL), where the shared parameters of tasks are optimized by minimizing a linear combination of task losses. However, it is challenging to select a proper weight to balance tasks and reduce task conflict. To handle the problem as a whole, we propose a novel approach named as Pareto self-supervised training (PSST) for FSL. PSST explicitly decomposes the few-shot auxiliary problem into multiple constrained multi-objective subproblems with different trade-off preferences, and here a preference region in which the main task achieves the best performance is identified. Then, an effective preferred Pareto exploration is proposed to find a set of optimal solutions in such a preference region. Extensive experiments on several public benchmark datasets validate the effectiveness of our approach by achieving state-of-the-art performance.

1. Introduction

Although deep learning has achieved great success in a variety of fields, limitations still exist in many practical applications where labeled samples are intrinsically rare or expensive. Different from humans who can easily learn to accomplish new tasks with a few examples, it is difficult for machines to rapidly generalize to new concepts with very little supervision, which draws considerable attention to the challenging *few-shot learning* (FSL) setting. As training large models with few labeled samples leads to overfitting or even non-convergence, conventional deep neural networks fail to address such a problem.

Recently, *self-supervised learning* (SSL) attracts many researchers for its soaring performance without involving manual labels. By defining pretext tasks to exploit the structural information of data itself, supervisory signals can be easily developed to learn useful general-purpose representations [2, 20, 31]. As self-supervised learning can improve the generalization of the network under the limitation of labeled data, some recent *few-shot auxiliary learning* (FSAL) works [16, 39] take few-shot learning as learning main task with self-supervised auxiliary tasks. To encourage that the few-shot task benefits from auxiliary tasks, some parameters are shared across tasks to inductive knowledge transfer. However, as the objectives of distinct tasks are different and the relationship between objectives is complicated and unknown, optimizing each task not only promotes each other but also naturally conflicts. A typical solution to suppress such conflict is to optimize the shared parameters by minimizing a weighted sum of the empirical risk for each task, where each weight of the empirical risk can be viewed as the trade-off. In previous works [16, 39], these trade-offs are usually set by experience in practical situations. It is difficult to find optimal trade-offs. Moreover, these works [16, 39] attempt to find one single solution for all objectives, which is likely to sacrifice the performance of the main task and be inconsistent with the goal of few-shot auxiliary learning.

According to the above discussion, few-shot auxiliary learning with conflicting objectives requires better modeling of the trade-off between tasks, which is beyond what a linear combination achieves. To overcome the issue, we propose a novel approach named **Pareto self-supervised training (PSST)** for few-shot learning. PSST explicitly casts few-shot auxiliary learning as a multi-objective optimization problem, with the overall objective of finding a Pareto optimal solution of network parameters [26, 27]. However, different from previous works that explore in the global space [27], PSST uses an effective preferred Pareto exploration for FSL. Specifically, PSST decomposes the

*Corresponding author.

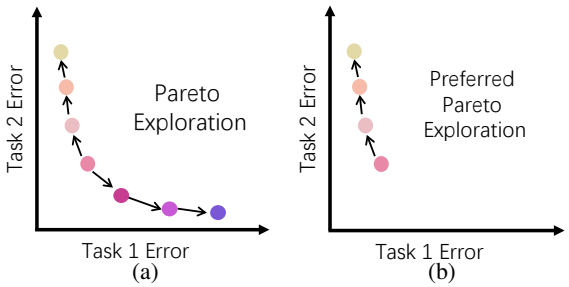


Figure 1: Illustrative examples of Pareto exploration in (a) previous works and (b) our PSST.

few-shot auxiliary problem into several constrained multi-objective subproblems with different trade-off preferences, and then identifies the preference region where the main task achieves the best performance. As illustrated in Fig. 1, the desired space of exploration is thus restricted by only exploring in the preference region where the given points achieve better performance in task 1 rather than task 2. Experiments demonstrate that this improvement can suppress the accumulation of residual error, which contributes to efficiently finding a more accurate Pareto solution. To summarize, our main contributions are as follows:

- We point out that existing few-shot auxiliary learning methods face with a linear combination of conflicting objectives, and propose a multi-objective optimization solution to address the issue.
- We propose a novel Pareto self-supervised training (PSST) approach for few-shot auxiliary learning. To achieve better performance for the main task and meanwhile improve the efficiency and accuracy of the exploration, PSST pioneers a preferred Pareto exploration that explores in the identified preference region.
- We conduct extensive experiments to demonstrate that our PSST can better model the trade-off between tasks, which leads to state-of-the-art performance on several benchmark datasets.

2. Related Work

2.1. Few-Shot Learning

Few-shot learning aims to generalize well to the novel classes where only a few labeled samples are available. Recently, *meta-learning* has been considered as the main solution to the few-shot problem due to the significant progress. Regarded as "learning to learn", meta-learning aims to improve its future learning performance with the experience of multiple learning episodes. Current meta-learning approaches for the few-shot problem can be roughly divided into three groups: optimization-based, model-based, and metric-based. *Optimization-based* approaches [5, 6, 13] learn a meta-learner to adjust the optimization algorithm, usually by providing better initialization or search steps for parameters. *Model-based* [1, 17, 19, 29] approaches depend on well-designed models, whose parameters are obtained with its internal architecture or a meta-learner for fast learning. *Metric-based* approaches [30, 38, 40] learn a generalizable embedding model to transform all samples into a common metric space, where specific distance measures can be employed with the nearest neighbor classifiers.

2.2. Self-Supervised Learning

As collecting enough human-annotated labels for large-scale unlabeled data is difficult and expensive, self-supervised learning methods aim to learn representations from unlabeled data. In computer vision applications, various pretext tasks have been utilized for pre-training the network including relative patch location [9], rotation prediction [18], image inpainting [31], and clustering [4]. Another line of works learns representations by contrasting positive pairs against negative pairs constructed on augmented samples [2, 10, 20, 44, 45, 46].

Regarded as auxiliary tasks, self-supervised learning can also be used to improve other tasks. For example, [47] shows that self-supervision can contribute to the recognition in a semi-supervised setting, and [3] uses self-supervision to improve domain generalization. Recently, few-shot auxiliary learning that exploits the complementarity of both few-shot learning and self-supervised learning has drawn much attention. Showing that the auxiliary loss without labels can extract discriminative features for few-shot learning, [16] considers rotation prediction and relative patch location as self-supervised tasks, and [39] uses image jigsaw puzzle. In our work, instead of being limited to specific self-supervised auxiliary tasks, we propose a general and effective preferred Pareto exploration that applies to arbitrary auxiliary tasks.

2.3. Multi-Objective Optimization

Multi-objective optimization [50] aims at finding a set of Pareto solutions with different trade-offs rather than one single solution. It has been used in many machine learning applications such as reinforcement learning [42], Bayesian optimization [21] and neural architecture search [12]. As the gradient information is usually not available in these applications, population-based and gradient-free multi-objective evolutionary algorithms [8, 49] are popular methods to find a set of well-distributed Pareto solutions in a single run. However, it can not be used for solving large-scale and gradient-based multi-task learning (MTL) problems. Multi-objective gradient descent [14] is an efficient approach for multi-objective optimization when gradient information is available. [37] proposes a novel method for solving MTL by treating it as multi-objective optimiza-

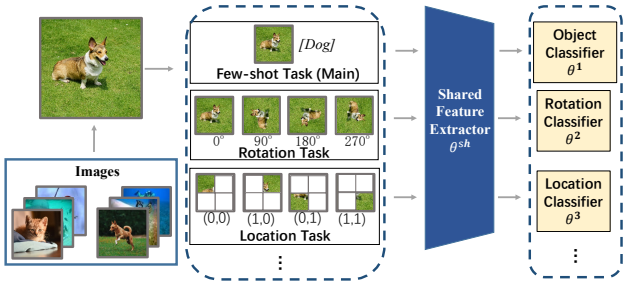


Figure 2: The framework used in our Pareto self-supervised training.

tion. [26] presents an MGDA-based method to generate a discrete set of solutions evenly distributed on the Pareto front. And [27] proposes to replace discrete solutions with continuous solution families, allowing for a much denser set of solutions and continuous analysis on them. However, these methods still focus on how to find a Pareto solution where the performance of each task is equally important. How to effectively find a Pareto solution that is concerned only with the performance of the main task still remains a challenge for auxiliary learning.

3. Pareto Self-Supervised Training

In this section, we firstly introduce a general framework to state our problem to be solved. Then we formulate the problem as a multi-objective optimization for FSAL. After that, we elaborate how to solve the optimization problem.

3.1. Framework

The general framework in Fig. 2 is used in our PSST for problem clarification, which combines self-supervised task and supervised class recognition task in a few-shot setting. We train the feature extractor θ^{sh} with both labeled (top branch) and unlabeled (bottom branches) data in a multi-task setting. We use the labeled data to train the object classifier θ^1 with few-shot classification loss. For the self-supervised task, we sample images from the unlabeled dataset. For example, we generate four rotations for each input image in rotation task, process them with θ^{sh} and train the rotation classifier θ^2 with the self-supervised loss. The pipeline for other self-supervised tasks is analog to this one.

3.2. Multi-Objective FSAL

Our goal is to improve few-shot learning performance via self-supervised training. Different from previous works using a weighted sum of loss functions for few-shot auxiliary learning with self-supervised learning [11, 16], we cast few-shot auxiliary learning as multi-objective optimization rather than a weighted sum of loss function.

We consider the few-shot auxiliary learning over an input space $\mathcal{X}$ and a collection of task spaces $\{\mathcal{Y}^m\}_{m\in[M]}$, such that a large dataset of independent and identically distributed (i.i.d.) data points $\left\{\mathbf{x}_i, y_i^1, \ldots, y_i^M\right\}_{i\in[N]}$ is given, where M is the number of tasks, N is the number of data points, and y_i^m is the label of the m^{th} task for the i^{th} data point. As shown in Fig. 2, we further consider a parametric hypothesis class per task as $f^m\left(\mathrm{x};\theta^{sh},\theta^m\right) : \mathcal{X} \to \mathcal{Y}^m$, such that some parameters $\left(\theta^{sh}\right)$ are shared between tasks and others (θ^m) are task-specific. Define the task-specific loss function by $\mathcal{L}_m(\cdot,\cdot) : \mathcal{Y}^m \times \mathcal{Y}^m \to \mathbb{R}^+$.

Previous works [11, 16] are to optimize a proxy objective that minimizes a weighted linear combination of per-task losses:

$$\min_{\theta} L(\theta) = \sum_{m=1}^{M} \omega_m L_m(\theta^{sh}, \theta^m), \tag{1}$$

where ω_m is the weight for the m-th task, and $L_m\left(\theta^{sh},\theta^m\right)$ is the empirical loss of the task m, defined as $L_m\left(\theta^{sh},\theta^m\right) \triangleq \frac{1}{N}\sum_{i=1}^{N} L_m\left(f^m\left(\mathbf{x}_i;\theta^{sh},\theta^m\right), y_i^m\right)$. This approach is simple and straightforward, however, it has some drawbacks which have been pointed out by many recent works [26, 27].

In a typical few-shot auxiliary learning application, the weight ω_m is needed to be assigned manually before optimization, and the overall performance is highly dependent on the assigned weights. Choosing a proper weight vector could be very difficult even for an expert. To accurately model the trade-off between tasks, which is beyond what a linear combination can achieve, multi-objective optimization is widely used in recent works [26, 27].

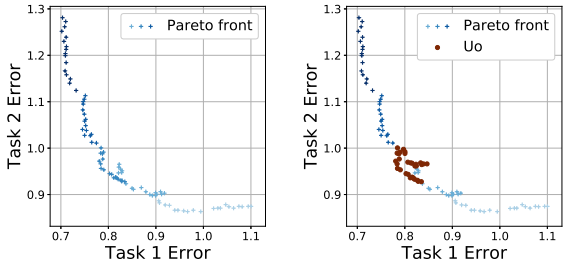
Previous works focus on weighted summation due to its intuitively appealing [11, 16, 7], however, these works typically require either an expensive grid search over various scalings or the use of some heuristics. Recent work formulates the weighted summation as multi-objective optimization, which optimizes a collection of possibly conflicting objectives [26]. By adopting this, we reformulate our few-shot auxiliary learning as multi-objective optimization problem with a vector-valued loss $\mathcal{L}$:

$$\min_{\theta} \mathcal{L}(\theta) = \min_{\theta} \left(\mathcal{L}_1\left(\theta^{sh},\theta^1\right), \ldots, \mathcal{L}_M\left(\theta^{sh},\theta^M\right)\right)^{\top}, \tag{2}$$

where $\theta = \left\{\theta^{sh},\theta^1,\ldots,\theta^M\right\}$. The goal of multi-objective optimization is to reach Pareto optimality.

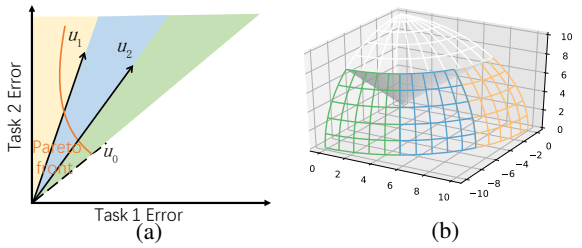
Definition 1 (Pareto dominance): A solution θ dominates a solution $\bar{\theta}$ if $\mathcal{L}_m\left(\theta^{sh},\theta^m\right) \leq \mathcal{L}_m\left(\bar{\theta}^{sh},\bar{\theta}^m\right)$ for all tasks m and $\mathcal{L}\left(\theta^{sh},\theta^1,\ldots,\theta^M\right) \neq \mathcal{L}\left(\bar{\theta}^{sh},\bar{\theta}^1,\ldots,\bar{\theta}^M\right)$.

Definition 2 (Pareto optimality): A solution θ^* is called Pareto optimal if there exists no solution θ that dominates θ^*. The set of Pareto optimal solutions is called the Pareto front $(\mathcal{P} = \{\mathcal{L}(\theta^*)\}_{\theta^*\in\mathcal{P}_{\theta^*}})$.



(a) Pareto front (b) U_0 points in Pareto front.

Figure 3: The convergence behaviors of PSST. Each U_0 point is the solution of PSST from different initial parameters of neural network. The proposed PSST successfully find U_0 in the balance space that maximizes the overall performance of two tasks.



(a) (b)

Figure 4: Illustrative examples of preference region in (a) 2 tasks (b) 3 tasks, where the white space will not be explored.

3.3. Optimization of Multi-Objective FSAL

3.3.1 Gradient-based multi-objective optimization

Many gradient-based methods have been proposed for solving multi-objective optimization problems [15, 26]. Fliege and Svaiter [14] have proposed a simple gradient-based method, which is a generalization of a single objective steepest descent algorithm. The update rule of the algorithm is $\theta_{t+1} = \theta_t + \eta d_t$, where η is the step size and the search direction d_t is obtained as follows:

$$\begin{aligned} &(d_t, \alpha_t) = \arg\min_{d \in R^M, \alpha \in R} \alpha + \tfrac{1}{2}\|d\|^2, \\ &\text{s.t.} \quad \nabla \mathcal{L}_m(\theta_t)^T d \leq \alpha, m = 1, \ldots, M. \end{aligned} \tag{3}$$

The solutions of the above problem will satisfy:

Lemma 1 [14]: Let $\left(d^k, \alpha^k\right)$ be the solution of problem in Eq. 3. If θ_t is Pareto optimal, then $d_t = 0 \in \mathbb{R}^n$ and $\alpha_t = 0$. If θ_t is not Pareto optimal, then

$$\begin{aligned} &\alpha_t \leq -(1/2)\|d_t\|^2 < 0, \\ &\nabla \mathcal{L}_m(\theta_t)^T d_t \leq \alpha_t, m = 1, \ldots, M, \end{aligned} \tag{4}$$

where θ is called Pareto optimal if no other solution in its neighborhood has better value in all objective functions. In other words, if $d_t = 0$, no direction can improve the performance for all tasks at the same time. If we want to improve the performance for one specific task, another task will be deteriorated (e.g., $\exists m, \nabla \mathcal{L}_m(\theta_t)^T d_t > 0$). Therefore, the current solution is a Pareto optimal point. When $d_t \neq 0$, we have $\nabla \mathcal{L}_m(\theta_t)^T d_t < 0, m = 1, \ldots, M$, which means d_t is a valid descent direction for all tasks. The current solution should be updated along the obtained direction $\theta_{t+1} = \theta_t + \eta d_t$, and will be stopped once a feasible solution is found.

As shown in Fig. 3, using gradient-based method in Eq. 3 to solve multi-objective optimization problems, the interaction between gradients within their shared parameters is balanced, which leads to balanced performances for different tasks. Thus, the solutions where given multiple different initial parameters achieve balanced performances for different tasks, and these solutions are in the space where different tasks are balanced.

3.3.2 Optimization in preference region

Decomposition-based multi-objective evolutionary algorithm [41, 48] is one of the most popular gradient-free multi-objective optimization methods, which decomposes a multi-objective optimization problem into K subproblems via preference vectors $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_K\}$ in R_+^M and solves them simultaneously. The preference vectors in these methods are handcrafted, and all subproblems are solved. However, in few-shot auxiliary learning, we focus on the main task, the challenge lies in how to identify the preference region where the main task achieves the best performance.

Because of the distinct optimization of different tasks, it is difficult to directly determine the space where the main task achieves optimal performance. Therefore, the goal is to exclude the parts that are not interested, where the auxiliary tasks hamper the performance of the main task. We try to find the space where distinct tasks are balanced. If the performance of the main task is not better than the balanced performance of the main task, we will remove this part in the space. We identify and empirically demonstrate the solution of gradient-based optimization method for multi-objective optimization lying in the set where the overall performance of different tasks are balanced. An illustrative example is shown in Fig. 4.

Assuming the best overall performence is achieved when all tasks are balanced, we state that if the performance of the main task is no better than the balanced performance of the main task, the auxiliary tasks hamper the main task. Let $\rho(\theta)$ be the proportion of performance in the main task over the performance of all auxiliary tasks in θ, i.e., $\rho(\theta) = \frac{\mathcal{L}_1(\theta)}{\sum_{m=2}^{M} \mathcal{L}_m(\theta)}$, we formalize the result as follows.

Lemma 2: If $\theta_{\pi 0}^*$ achieves the best overall performance, i.e. $min_\theta \sum_{m=1}^{M} \mathcal{L}_m(\theta) = \sum_{m=1}^{M} \mathcal{L}_m(\theta_{\pi 0}^*)$, and a better performance of the main task can be achieved in θ', i.e.,

$\mathcal{L}_1(\theta') < \mathcal{L}_1(\theta^*_{\pi 0})$, then $\rho(\theta') < \rho(\theta^*_{\pi 0})$. The proof of Lemma 2 can be found in supplemental materials.

Noting that $\rho(\theta) = \rho(\theta^*_{\pi 0})$ is a hyperplane dividing the objective space into two parts, we remove the part satisfying $\rho(\theta) > \rho(\theta^*_{\pi 0})$, and further divide the remaining part satisfying $\rho(\theta) \leq \rho(\theta^*_{\pi 0})$ into different regions with preference vectors to avoid residual error accumulation, which is detailed in Section 3.4. An illustrate example is shown in Fig. 4, the white space stands for $\rho(\theta) > \rho(\theta^*_{\pi 0})$.

Subproblem Decomposition. With the solution $\theta^*_{\pi 0}$, we have $\mathcal{L}(\theta^*_{\pi 0}) = [\mathcal{L}_1(\theta^*_{\pi 0}), \mathcal{L}_2(\theta^*_{\pi 0}), \cdots, \mathcal{L}_M(\theta^*_{\pi 0})]^{\mathrm{T}}$. In a 2-D space, $\rho(\theta) = \rho(\theta^*_{\pi 0})$ is a line, its direction vector is well defined. In a M-dimentional (M-D) space ($M \geq 3$), $\rho(\theta) = \rho(\theta^*_{\pi 0})$ is a hyperplane. This hyperplane will intersects the coordinate planes and form some lines. We define the direction vectors of $\rho(\theta) = \rho(\theta^*_{\pi 0})$ in M-D space as the direction vectors of the intersections formed by the hyperplane and coordinate planes. Coordinate plane is 2-D space, which corresponding two tasks with the name "two tasks scenario" comes from this place. Let unit vector u_0 be one of the direction vector of $\rho(\theta) = \rho(\theta^*_{\pi 0})$ for two-task scenario, we have $u_0 = (\cos \pi_0, \sin \pi_0)$, where $\cos \pi_0 = \frac{e_1 \mathcal{L}(\theta^*_{\pi 0})}{\left\| T_2 \mathcal{L}(\theta^*_{\pi 0}) \right\|_2}$ and $T_m = I - e^{mm}$. Here, e^{mm} is a single-entry matrix, i.e. the mth element in the mth column being one and the rest elements being zero; $e_1 = (1, 0)^T$; I is an identity matrix. Given the u_0, we further decompose the auxiliary problem into K subproblems with a set of unit preference vectors $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_K\}$, for the preference vector u_i:

$$\begin{aligned} & u_i = (\cos \pi_i, \sin \pi_i), \\ s.t. \quad & \pi_i = \tfrac{i}{K}\left(\tfrac{\pi}{2} - \pi_0\right) + \pi_0, i = 1, \ldots, K, \end{aligned} \tag{5}$$

where the derivation and proof of the preference vectors can be found in supplemental materials, followed by a further discussion on preference vector in multiple tasks.

Suppose all objectives in the multi-objective optimization are non-negative, the multi-objective subproblem corresponding to the preference vector u_i and u_{i+1} is:

$$\begin{aligned} & \min_\theta \mathcal{L}(\theta) = (\mathcal{L}_1(\theta), \mathcal{L}_2(\theta), \cdots, \mathcal{L}_M(\theta))^{\mathrm{T}}, \\ & \text{s.t. } u_i {e_1}^T \leq \tfrac{e_1 \mathcal{L}(\theta)}{\|T_2 \mathcal{L}(\theta)\|_2} \leq u_{i+1} {e_1}^T. \end{aligned} \tag{6}$$

The subproblem Eq.6 can be further reformulated as:

$$\begin{aligned} & \min_\theta \mathcal{L}(\theta) = (\mathcal{L}_1(\theta), \mathcal{L}_2(\theta), \cdots, \mathcal{L}_M(\theta))^{\mathrm{T}}, \\ & \text{s.t. } \mathcal{Q}_i(\theta) = \tfrac{e_1 \mathcal{L}(\theta)}{\|T_2 \mathcal{L}(\theta)\|_2} - u_{i+1} {e_1}^T \leq 0, \\ & \mathcal{R}_i(\theta) = u_i {e_1}^T - \tfrac{e_1 \mathcal{L}(\theta)}{\|T_2 \mathcal{L}(\theta)\|_2} \leq 0. \end{aligned} \tag{7}$$

The proof can be found in supplemental materials.

Preferred Pareto Optimality. To solve the constrained multi-objective subproblem with preference vector u_i and u_{i+1}, we need to find an initial solution which is feasible or at least satisfies most constraints. For a randomly generated solution θ, we define the index set of all activated constraints as $\mathcal{K}_\epsilon(\theta) = \{k \mid \mathcal{Q}_k(\theta) \geq -\epsilon, k = 0, \ldots, K-1\}$ and $\mathcal{J}_\epsilon(\theta) = \{j \mid \mathcal{R}_j(\theta) \geq -\epsilon, j = 0, \ldots, K-1\}$, where ϵ is a threshold. We can find a valid descent direction d_t to reduce the value of all activated constraints by solving:

$$\begin{aligned} & (d_t, \alpha_t) = \arg\min_{d \in R^M, \alpha \in R} \alpha + \tfrac{1}{2}\|d\|_2^2, \\ & \text{s.t.} \quad \nabla \mathcal{L}_m(\theta_t)^T d \leq \alpha, m = 1, \ldots, M. \\ & \nabla \mathcal{Q}_k(\theta_t)^T d \leq \alpha, k \in \mathcal{K}_\epsilon(\theta_t), \\ & \nabla \mathcal{R}_j(\theta_t)^T d \leq \alpha, j \in \mathcal{J}_\epsilon(\theta_t). \end{aligned} \tag{8}$$

The valid descent direction can be found by solving the constrained optimization problem in Eq. 8. However, the optimization problem itself is not scalable well for high dimensional decision space especially in deep neural networks. To solve the constrained optimization problem, we propose a scalable optimization method. Inspired by [14], we first rewrite the optimization problem Eq. 8 in its dual form. Based on the KKT conditions, we have the update direction as

$$\begin{aligned} & d_t(\omega_m, \beta_k, \gamma_j) = -\Big(\textstyle\sum_{m=1}^M \omega_m \nabla \mathcal{L}_m(\theta_t) \\ & + \textstyle\sum_{k \in \mathcal{K}_\epsilon(\theta_t)} \beta_k \nabla \mathcal{Q}_k(\theta_t) + \sum_{j \in \mathcal{J}_\epsilon(\theta_t)} \gamma_j \nabla \mathcal{R}_j(\theta_t)\Big), \end{aligned} \tag{9}$$

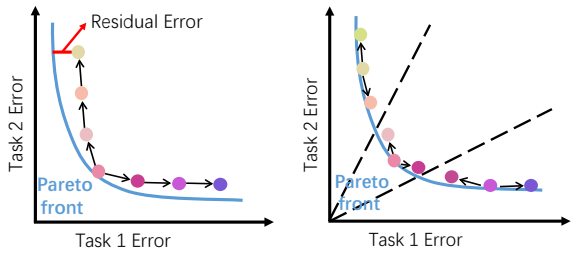
where $\omega_m \geq 0, \beta_k \geq 0$ and $\gamma_j \geq 0$ are the Lagrange multipliers for the linear inequality constraints. Therefore, the dual problem is:

$$\begin{aligned} & \max_{\omega_m, \beta_k, \gamma_j} -\tfrac{1}{2}\|d_t(\omega_m, \beta_k, \gamma_j)\|_2^2, \\ & \text{s.t. } \textstyle\sum_{m=1}^M \omega_m + \sum_{k \in \mathcal{K}_\epsilon(\theta_t)} \beta_k + \sum_{j \in \mathcal{J}_\epsilon(\theta_t)} \gamma_j = 1, \\ & \forall m = 1, \ldots, M, \forall k \in \mathcal{K}_\epsilon(\theta_t), \forall j \in \mathcal{J}_\epsilon(\theta_t), \\ & \omega_m \geq 0, \beta_k \geq 0, \gamma_j \geq 0. \end{aligned} \tag{10}$$

The details of derivations and proofs are shown in supplemental materials.

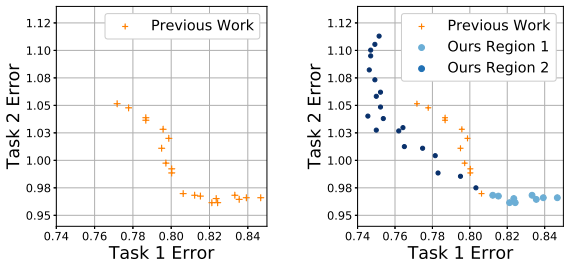
For the above problem, the decision space is no longer the parameter space, and it becomes the objective and constraint space. For a multi-objective optimization problem with three objective functions and seven activated constraints, the dimension of problem in Eq. 10 is ten, which is significantly smaller than the dimension of problem in Eq. 8 which is very large and could be more than a million.

As shown in Fig. 4, the preference vectors divide the objective space into different sub-regions, and with the restricted of u_0, the space where auxiliary tasks hamper the performance of the main task will not be considered, the computation at training time decreases with a smaller exploration space. The set of solutions for all subproblems would be in different sub-regions and represent different trade-offs among the tasks.



(a) Previous work (b) PSST

Figure 5: Illustrative examples of residual error accumulation problem in previous Pareto exploration, and PSST alleviate this issue by preferred Pareto exploration.



(a) (b)

Figure 6: Residual error comparison between (a) previous work and (b) PSST, details in Section 4.

3.4. Preferred Pareto Exploration

Pareto exploration aims at finding several Pareto solutions. Different from the grid search in the weighted sum of objectives, the Pareto exploration is more efficiency in finding Pareto solutions [27]. However, previous Pareto exploration has the residual error accumulation problem as shown in Figs. 5 and 6. To avoid such residual error accumulation, we explore the sub-regions instead of the whole region.

Find a Pareto solution. Firstly, our preferred Pareto exploration takes random parameters of neural network θ as input, and find a Pareto solution θ^* via direction d_t in Eq.9.

Find gradient on the tangent plane at θ^*. Secondly, once a Pareto solution θ^* is found, we explore its local Pareto set by spawning new points θ_t.

Lemma 3 [23]: If θ^* is Pareto optimal, there will be a $\boldsymbol{\lambda} \in \mathbb{R}^M$ such that $\lambda_m \geq 0$, $\sum_{m=1}^{M} \lambda_m = 1$, and $\sum_{m=1}^{M} \lambda_m \nabla \mathcal{L}_m(\theta^*) = 0$.

Proposition [22]: Assuming that $\mathcal{L}(\theta^*)$ is smooth and θ^* is Pareto optimal, consider any smooth curve $\theta_{(x)} : (-\epsilon, \epsilon) \to \mathbb{R}^M$ in the Pareto set and passing θ^* at $x = 0$, i.e., $\theta_{(0)} = \theta^*$, then for $\exists \beta \in \mathbb{R}^M$ we have:

$$\boldsymbol{H}(\theta^*) \frac{d\theta}{dx}(0) = \nabla \mathcal{L}(\theta^*)^\top \boldsymbol{\beta}, \tag{11}$$

where $\boldsymbol{H}(\theta^*) = \sum_{m=1}^{M} \lambda_m \nabla^2 \mathcal{L}_m(\theta^*)$, and we use $\frac{d\theta}{dx}(0)$ as the update direction and calculate $\theta_1 = \theta_{(0)} + \eta \frac{d\theta}{dx}(0)$. Solving such problem requires an efficient matrix solver.

Algorithm 1 Few shot Learning via Pareto Self-supervised Training

1: **Input:** A random initial neural network θ.
2: Find a Pareto solution $\theta^*_{\pi_0}$ via direction d_t in Eq. 3
3: Find $\pi_0 = \arccos \frac{e_1 \mathcal{L}(\theta^*_{\pi_0})}{\|T_2 \mathcal{L}(\theta^*_{\pi_0})\|_2}$, $u_0 = (\cos \pi_0, \sin \pi_0)$.
4: Calculate vectors $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_K\}$ via Eq. 5.
5: **for all** $i = 0$ to $K - 1$ **do**
6: Randomly generate parameters $\theta^{(i)}$.
7: Find Pareto solution $\theta^{(i)*}$ via direction d_t in Eq. 9.
8: $\boldsymbol{s}_i \leftarrow$ PreferredParetoExploration $\left(\theta^{(i)*}\right)$
9: **end for**
10: **Output:** The set of solutions for all subproblems $\{\boldsymbol{s}_i \mid i = 1, \cdots, K\}$.

Similar to [27], we use Krylov subspace iteration methods. The details of derivations and proofs are shown in supplemental materials, followed the method of obtain $\boldsymbol{\lambda}$.

Explore a set of Pareto solutions (line 8 in Algorithm 1). To explore a set of Pareto solutions based on θ^* in the region between u_i and u_{i+1}, we initialize a queue $q \leftarrow [\theta^*]$ and a set $s \leftarrow [\theta^*]$. Given a Pareto solution θ^*, with gradient $\frac{d\theta}{dx}(0)$ on the tangent plane at θ^*, we can have $\theta_1 = \theta^* + \eta \frac{d\theta}{dx}(0)$. Here, θ_1 is the first-order approximation of a Pareto solution close to θ^*, and we can have a Pareto solution θ^*_1 via direction d_t in Eq.9. The queue and set collect all the Pareto solutions: $q \leftarrow [\theta^*_t]$ and $s \leftarrow [\theta^*_t]$. Once we cannot find any approximate point at θ^*, we remove θ^* from queue q, and do the same to all points in the queue q. Thus, we explore all the Pareto solutions in the region between u_i and u_{i+1}. Pareto solutions in all regions are saved in set s. The Pareto solution which has the best performance in the main task will be the final solution for a specific few-shot task. We summarize the PSST in Algorithm 1.

To summarize, the efficiency of our exploration algorithm comes from two sources: exploration on the tangent plane and termination with restricted preference region. The time cost of getting one tangent direction is $O(kn)$, k is the iteration and n is the size of input data, which scales linearly to the network size.

4. Experiments

We design two experiments, which we call the sufficiency and necessity tests, to show the effectiveness and efficiency of our proposed PSST.

In the necessity test, which focuses on demonstrate the effectiveness of our proposed PSST in few-shot auxiliary learning, we compare our approach with prior few-shot methods on the MiniImageNet and CIFAR-FS datasets respectively. We call this experiment the necessity test as we use this experiment to establish that our proposed PSST, multi-objective optimization strategies rather than

Models	Backbone	1-shot	5-shot
MetaOptNet [25]	ResNet-12	62.64 ± 0.61	78.63 ± 0.46
+ rot, (*SLA* [24])	ResNet-12	62.93 ± 0.63	79.63 ± 0.47
+ rot **+ PSST**	ResNet-12	**64.05± 0.49**	**80.24 ± 0.45**
CC [17]	WRN-28-10	61.09 ± 0.44	78.43 ± 0.33
+ loc, (*BF3S* [16])	WRN-28-10	60.71 ± 0.46	77.64 ± 0.34
+ loc **+ PSST**	WRN-28-10	**63.26 ± 0.44**	**80.05 ± 0.34**
+ rot, (*BF3S* [16])	WRN-28-10	62.93 ± 0.45	79.87 ± 0.33
+ rot **+ PSST**	WRN-28-10	**64.16 ± 0.44**	**80.64 ± 0.32**
ProtoNet [38]	ResNet-18	54.16 ± 0.82	73.68 ± 0.65
+ rot, (*SSFSL* [39])	ResNet-18	57.92 ± 0.58	76.00 ± 0.58
+ rot **+ PSST**	ResNet-18	**59.28 ± 0.46**	**77.18± 0.43**
+ jig, (*SSFSL* [39])	ResNet-18	58.35 ± 0.50	76.20 ± 0.53
+ jig **+ PSST**	ResNet-18	**59.34 ± 0.45**	**77.27± 0.43**
+ jig + rot, (*SSFSL* [39])	ResNet-18	58.84 ± 0.49	76.60 ± 0.52
+ jig + rot **+ PSST**	ResNet-18	**59.52 ± 0.46**	**77.43± 0.46**

Table 1: Average accuracy (%) comparison with 95 confidence intervals before and after incorporating PSST into existing methods on MiniImageNet. Best results are displayed in boldface.

grid search in the weighted sum of objectives are indeed the source of efficiency in our method.

In the sufficiency test, we consider previous Pareto exploration method to show that our PSST is a fast and effective method. We call this experiment the sufficiency test as it demonstrates our method is able to quickly explore Pareto sets and Pareto front in few-shot auxiliary learning.

Datasets. We perform experiments on two few-shot datasets for necessity test: MiniImageNet [43] and CIFAR-FS [1]. MultiMNIST [35] is used for sufficiency test. MiniImageNet consists of 100 classes randomly picked from the ImageNet dataset [34] (i.e., 64 base classes, 16 validation classes, and 20 novel test classes); each class has 600 images with size 84×84 pixels. CIFAR-FS is a few-shot dataset created by dividing the 100 classes of CIFAR-100 into 64 base classes, 16 validation classes, and 20 novel test classes. The images in this dataset have size 32×32 pixels. Please refer to our supplemental materials for more information about the network architectures, task descriptions, and implementation details in each dataset.

Evaluation metrics. Few-shot classification algorithms are evaluated based on the classification accuracy in their testing stage (when the learned classifier is applied to test images from the novel classes). Specifically, a large number of N_n-way K-shot tasks are sampled from the available set of novel classes. Each task is created by randomly selecting N_n novel classes from the available test (validation) classes and then within the selected classes randomly selecting K train and M test images per class (making sure that train and test images do not overlap). The classification accuracy is measured on the $N_n \times M$ test images and is averaged over all the sampled few-shot tasks. Except otherwise stated, for all experiments we use M = 15, N_n = 5, and K = 1 or K = 5 (1-shot and 5-shot settings respectively).

Reduce the task conflict. To verify the effectiveness of our proposed PSST, we embed it into three self-supervised tasks : location prediction (loc), rotation prediction (rot) and jigsaw puzzle (jig), and three widely used meta-learning

Models	Backbone	1-shot	5-shot
MAML [13]	Conv-4-64	48.70 ± 1.84	63.10 ± 0.92
Prototypical Nets [38]	Conv-4-64	49.42 ± 0.78	68.20 ± 0.66
R2-D2 [1]	Conv-4-64	48.70 ± 0.60	65.50 ± 0.60
LwoF [17]	Conv-4-64	56.20 ± 0.86	72.81 ± 0.62
BF3S [16]	Conv-4-64	54.83 ± 0.43	71.86 ± 0.33
RelationNet [40]	Conv-4-64	50.40 ± 0.80	65.30 ± 0.70
GNN [36]	Conv-4-64	50.30	66.40
TADAM [30]	ResNet-12	58.50 ± 0.30	76.70 ± 0.30
Munkhdalai et al. [29]	ResNet-12	57.10 ± 0.70	70.04 ± 0.63
SNAIL [28]	ResNet-12	55.71 ± 0.99	68.88 ± 0.92
Shot-Free [33]	ResNet-12	59.04	77.64
MetaOptNet [25]	ResNet-12	62.64 ± 0.61	78.63 ± 0.46
Qiao et al. [32]	WRN-28-10	61.76 ± 0.08	77.59 ± 0.12
BF3S [16]	WRN-28-10	62.93 ± 0.45	79.87 ± 0.33
Su et al. [39]	WRN-28-10	60.43 ± 0.58	76.60 ± 0.29
AWGIM [19]	WRN-28-10	63.12 ± 0.08	78.40 ± 0.11
PSST	Conv-4-64	57.04 ± 0.51	73.17 ± 0.48
PSST	WRN-28-10	**64.16 ± 0.44**	**80.64 ± 0.32**

Table 2: Few-shot image classification average accuracy (%) comparison with state-of-the-arts with 95% confidence intervals on MiniImageNet.

Models	Backbone	1-shot	5-shot
Prototypical Nets [38]	Conv-4-64	62.82 ± 0.32	79.59 ± 0.24
MAML [13]	Conv-4-64	58.90 ± 1.90	71.50 ± 1.00
RelationNet [40]	Conv-4-64	55.00 ± 1.00	69.30 ± 0.80
BF3S [16]	Conv-4-64	63.45 ± 0.31	79.79 ± 0.24
GNN [36]	Conv-4-64	61.90	75.30
R2-D2 [1]	Conv-4-64	60.00 ± 0.70	76.10 ± 0.60
Shot-Free [33]	ResNet-12	69.20 ± 0.40	84.70 ± 0.40
MetaOptNet [25]	ResNet-12	72.00 ± 0.70	84.20 ± 0.50
BF3S [16]	WRN-28-10	76.09 ± 0.30	87.83 ± 0.21
PSST	Conv-4-64	64.37 ± 0.33	80.42 ± 0.32
PSST	WRN-28-10	**77.02 ± 0.38**	**88.45 ± 0.35**

Table 3: Few-shot image classification average accuracy (%) comparison with state-of-the-arts with 95% confidence intervals on CIFAR-FS.

Method	BF3S [16]		PSST	
Number of Solutions	1	10	1	10
Time (1 iteration)	1.02s	1.02s	1.05s	1.05s
Iterations ($\times 10^3$)	24	249	24	60

Table 4: Training time on the same GPU device. Multiple solutions are based on different trade-offs.

baselines: MetaOptNet [25], Cosine Classifier (CC) [17] and ProtoNet [38]. Table 1 shows that for all cases, incorporating PSST leads to a significant improvement which demonstrates the effectiveness of our PSST. Specifically, the performance gain is 5.36% on 1-shot and 3.75% on 5-shot in ProtoNet with rotation prediction task and jigsaw puzzle task. Particularly, the performance of BF3S method with location prediction task is lower than that without location prediction task in both 1-shot and 5-shot image classification task, which is caused by the *task conflict*. Incorporating PSST leads to a significant improvement . We believe the reason is that location prediction task hampers the performance of BF3S, and our PSST effectively find a proper trade-off to reduce the task conflict between image classification and location prediction.

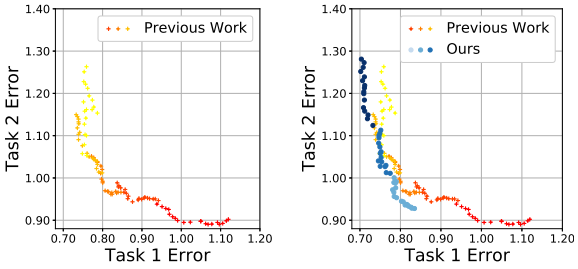
Comparison with prior works. In Tables 2 and 3, we compare our approach with prior few-shot methods on the MiniImageNet and CIFAR-FS datasets respectively. For our approach, we use CC and rotation prediction task, which gave the best results. In all cases we achieve state-of-the-art results surpassing prior methods.

Training time analysis. To further show the efficiency of PSST, we compare our approach with BF3S on the same GPU device as shown in Table 4. For a single solution, the training time of two method is close. For multiple solutions with different trade-offs, *e.g.* 10 solutions, BF3S with different weights via grid search needs 249×10^3 iterations, and PSST only needs 60×10^3 iterations, which is much faster than BF3S due to the effective preferred Pareto exploration.

In our sufficiency test, to evaluate our preferred Pareto exploration, similar to previous Pareto exploration work [27], we pick a subset of 2048 images from MultiMNIST.

Effective region decomposition. To demonstrate the effectiveness of our PSST, we assign PSST different initial parameters of neural network, and the results shows in Fig. 3. All u_0 points are in the balance space where the overall performance of two tasks achieve good performance, however, which may not be the best performance for a specific task. Given the u_0 points in the balance space, we can remove the space where the auxiliary tasks hamper the performance of the main task. The smaller exploration region is one of the sources of efficiency in our method.

Decrease residual error. To analyze how PSST decreases residual error, we record the exploration trajectories of both previous method and our PSST as shown in Fig. 6. Compared with previous method, our trajectories are pushed towards its lower left, indicating a better approximated Pareto front. We believe the reason is that the first-order approximation of a Pareto solution is closer to the Pareto front, and the exploration of θ^* is restricted in the sub-region with less residual error.



(a) (b)

Figure 7: Performance comparision between (a) previous Pareto exploration and (b) our PSST.

Performance comparison of Pareto exploration. As shown in Fig. 7, our method achieves a better performance on the main task with fewer explorations. Our trajectories are pushed towards its lower left, indicating a better approximated Pareto front. The training time of each iteration is the same for both methods, our PSST achieves better performance with 78 iterations, and previous work needs 130 iterations, which demonstrates both effectiveness and efficiency of our PSST. The efficiency of PSST is from less exploration space and the effectiveness is from the decreasing residual error.

5. Conclusion

In this paper, we study the problem of task conflict in few-shot auxiliary learning. We propose a novel Pareto self-supervised training to reduce the task conflict in few-shot auxiliary learning. We explicitly cast few-shot auxiliary learning as multi-objective optimization, with the overall objective of finding a Pareto optimal solution. We decompose the few-shot auxililary problem into several constrained multi-objective subproblems with different trade-off preferences allowing better exploration of the frontier. Experiments demonstrate both efficiency and effectiveness of our proposed PSST: where the efficiency of PSST is from the less exploration space and the effectiveness is from the decreasing residual error.

6. Acknowledgments

The authors gratefully acknowledge funding support from the Westlake University and Bright Dream Joint Institute for Intelligent Robotics.

References

[1] Luca Bertinetto, João F. Henriques, Philip H. S. Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019. 2, 7

[2] Piotr Bojanowski and Armand Joulin. Unsupervised learning by predicting noise. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 517–526, 2017. 1, 2

[3] Fabio M Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2229–2238, 2019. 2

[4] Mathilde Caron, Piotr Bojanowski, Julien Mairal, and Armand Joulin. Unsupervised pre-training of image features on non-curated data. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2959–2968, 2019. 2

[5] Zhengyu Chen and Donglin Wang. Multi-initialization meta-learning with domain adaptation. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021*, 2021. 2

[6] Zhengyu Chen, Donglin Wang, and Shiqian Yin. Improving cold-start recommendation via multi-prior meta-learning. In *43rd European Conference on Information Retrieval, ECIR 2021*, 2021. 2

[7] Zhengyu Chen, Ziqing Xu, and Donglin Wang. Deep transfer tensor decomposition with orthogonal constraint for recommender systems. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021*, 2021. 3

[8] Kalyanmoy Deb. *Multi-objective optimization using evolutionary algorithms*. Wiley-Interscience series in systems and optimization. Wiley, 2001. 2

[9] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430, 2015. 2

[10] Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with convolutional neural networks. In *Advances in neural information processing systems*, pages 766–774, 2014. 2

[11] Nikita Dvornik, Julien Mairal, and Cordelia Schmid. Diversity with cooperation: Ensemble methods for few-shot classification. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 3722–3730, 2019. 3

[12] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Efficient multi-objective neural architecture search via lamarckian evolution. In *International Conference on Learning Representations*, 2018. 2

[13] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, pages 1126–1135, 2017. 2, 7

[14] Jörg Fliege and Benar Fux Svaiter. Steepest descent methods for multicriteria optimization. *Mathematical Methods of Operations Research*, 51(3):479–494, 2000. 2, 4, 5

[15] Jörg Fliege and A. Ismael F. Vaz. A method for constrained multiobjective optimization based on SQP techniques. *SIAM J. Optim.*, 26(4):2091–2119, 2016. 4

[16] Spyros Gidaris, Andrei Bursuc, Nikos Komodakis, Patrick Pérez, and Matthieu Cord. Boosting few-shot visual learning with self-supervision. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 8058–8067, 2019. 1, 2, 3, 7, 8

[17] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4367–4375, 2018. 2, 7, 8

[18] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations*, 2018. 2

[19] Yiluan Guo and Ngai-Man Cheung. Attentive weights generation for few shot learning via information maximization. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 13496–13505, 2020. 2, 7

[20] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020. 1, 2

[21] Daniel Hernández-Lobato, Jose Hernandez-Lobato, Amar Shah, and Ryan Adams. Predictive entropy search for multi-objective bayesian optimization. In *International Conference on Machine Learning*, pages 1492–1501, 2016. 2

[22] Claus Hillermeier. Generalized homotopy approach to multiobjective optimization. *Journal of Optimization Theory and Applications*, 110(3):557–583, 2001. 6

[23] Claus Hillermeier et al. *Nonlinear multiobjective optimization: a generalized homotopy approach*, volume 135. Springer Science & Business Media, 2001. 6

[24] Hankook Lee, Sung Ju Hwang, and Jinwoo Shin. Self-supervised label augmentation via input transformations. In *Proceedings of Machine Learning and Systems 2020*, pages 3537–3547. 2020. 7

[25] Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019*, pages 10657–10665, 2019. 7, 8

[26] Xi Lin, Hui-Ling Zhen, Zhenhua Li, Qingfu Zhang, and Sam Kwong. Pareto multi-task learning. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 12037–12047, 2019. 1, 3, 4

[27] Pingchuan Ma, Tao Du, and Wojciech Matusik. Efficient continuous pareto exploration in multi-task learning. In *International Conference on Machine Learning*, 2020. 1, 3, 6, 8

[28] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner. In *International Conference on Learning Representations*, 2018. 7

[29] Tsendsuren Munkhdalai and Hong Yu. Meta networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017*, pages 2554–2563, 2017. 2, 7

[30] Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In *Advances in Neural Information Processing Systems*, pages 721–731, 2018. 2, 7

[31] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016. 1, 2

[32] Siyuan Qiao, Chenxi Liu, Wei Shen, and Alan L. Yuille. Few-shot image recognition by predicting parameters from activations. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 7229–7238, 2018. 7

[33] Avinash Ravichandran, Rahul Bhotika, and Stefano Soatto. Few-shot learning with embedded class models and shot-free meta training. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019*, pages 331–339, 2019. 7

[34] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 7

[35] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In *Advances in neural information processing systems*, pages 3856–3866, 2017. 7

[36] Victor Garcia Satorras and Joan Bruna Estrach. Few-shot learning with graph neural networks. In *6th International Conference on Learning Representations, ICLR 2018*, 2018. 7

[37] Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Advances in Neural Information Processing Systems*, pages 527–538, 2018. 2

[38] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 4077–4087, 2017. 2, 7, 8

[39] Jong-Chyi Su, Subhransu Maji, and Bharath Hariharan. When does self-supervision improve few-shot learning? In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part VII*, pages 645–666, 2020. 1, 2, 7

[40] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H. S. Torr, and Timothy M. Hospedales. Learning to compare: Relation network for few-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1199–1208, 2018. 2, 7

[41] Anupam Trivedi, Dipti Srinivasan, Krishnendu Sanyal, and Abhiroop Ghosh. A survey of multiobjective evolutionary algorithms based on decomposition. *IEEE Trans. Evol. Comput.*, 21(3):440–462, 2017. 4

[42] Kristof Van Moffaert and Ann Nowé. Multi-objective reinforcement learning using sets of pareto dominating policies. *The Journal of Machine Learning Research*, 15(1):3483–3512, 2014. 2

[43] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 3630–3638, 2016. 7

[44] Teng Xiao, Shangsong Liang, and Zaiqiao Meng. Dynamic collaborative recurrent learning. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1151–1160, 2019. 2

[45] Teng Xiao, Shangsong Liang, Hong Shen, and Zaiqiao Meng. Neural variational hybrid collaborative filtering. *arXiv preprint arXiv:1810.05376*, 2018. 2

[46] Xu Yan, Chaoda Zheng, Zhen Li, Sheng Wang, and Shuguang Cui. Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5589–5598, 2020. 2

[47] Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S4l: Self-supervised semi-supervised learning. In *Proceedings of the IEEE international conference on computer vision*, pages 1476–1485, 2019. 2

[48] Qingfu Zhang and Hui Li. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Trans. Evol. Comput.*, 11(6):712–731, 2007. 4

[49] Eckart Zitzler. *Evolutionary algorithms for multiobjective optimization: methods and applications*. PhD thesis, University of Zurich, Zürich, Switzerland, 1999. 2

[50] Marcela Zuluaga, Guillaume Sergent, Andreas Krause, and Markus Püschel. Active learning for multi-objective optimization. In *International Conference on Machine Learning*, pages 462–470, 2013. 2