Gradient-Based Multi-Objective Deep Learning: Algorithms, Theories, Applications, and Beyond

Anonymous authors
Paper under double-blind review

Abstract

Many modern deep learning applications require balancing multiple objectives that are often conflicting. Examples include multi-task learning, fairness-aware learning, and the alignment of Large Language Models (LLMs). This leads to multi-objective deep learning, which tries to find optimal trade-offs or Pareto-optimal solutions by adapting mathematical principles from the field of Multi-Objective Optimization (MOO). However, directly applying gradient-based MOO techniques to deep neural networks presents unique challenges, including high computational costs, optimization instability, and the difficulty of effectively incorporating user preferences. This paper provides a comprehensive survey of gradient-based techniques for multi-objective deep learning, with a primary focus on supervised learning settings. We systematically categorize existing algorithms based on their outputs: (i) methods that find a single, well-balanced solution, (ii) methods that generate a finite set of diverse Pareto-optimal solutions, and (iii) methods that learn a continuous Pareto set of solutions. In addition to this taxonomy, the survey covers theoretical analyses, key applications, practical resources, and highlights open challenges and promising directions for future research.

1 Introduction

Traditional deep learning often focuses on optimizing a single learning objective, such as minimizing the prediction error or maximizing the likelihood. However, many real-world applications require balancing multiple, often conflicting, objectives. For instance, a computer vision system might need to perform the tasks of segmentation, depth estimation, and surface normal prediction simultaneously (Vandenhende et al., 2021), thus moving from single-task learning to multi-task learning (Zhang & Yang, 2022). Similarly, Large Language Models (LLMs) are expected to excel at diverse tasks like reasoning and coding, while also being safe, fair, and harmless (Wang et al., 2023a). Multi-Objective Optimization (MOO) (Miettinen, 1999), a field originating from operations research (Ehrgott, 2005), provides a formal framework for navigating these trade-offs. It has been widely studied across science and engineering, with diverse applications such as in finance (Steuer & Na, 2003), engineering design (Marler & Arora, 2004), and transportation planning (Tzeng et al., 2005).

Recently, there has been a surge of interest in adapting MOO for deep learning, re-framing many popular learning problems as multi-objective optimization problems. Examples include multi-task learning where each task performance is considered as an objective (Sener & Koltun, 2018), fairness-aware learning where accuracy is one objective and fairness metrics constitute the other objectives (Martinez et al., 2020), LLM alignment where each alignment criterion (such as helpfulness, harmlessness, and honesty) represents a separate objective (Wang et al., 2023a), and federated learning where the performance on each client is treated as an individual objective (Hu et al., 2022). All these problems can be formally expressed as:

$$\min_{\boldsymbol{\theta} \in \mathcal{K} \subset \mathbb{R}^d} \quad \boldsymbol{f}(\boldsymbol{\theta}) := [f_1(\boldsymbol{\theta}), \ldots, f_m(\boldsymbol{\theta})]^\top, \tag{1}$$

where $m \geq 2$ is the number of objectives, $\boldsymbol{\theta}$ is the decision variable, $\mathcal{K} \subset \mathbb{R}^d$ is the feasible set of the decision variable, and each $f_i : \mathcal{K} \to \mathbb{R}$ represents an individual objective function to be minimized.

Unlike single-objective optimization, which focuses on finding a single best solution, MOO recognizes that no single solution is optimal for all objectives simultaneously. Instead, MOO aims to identify a set of solutions that represent different trade-offs between objectives, collectively known as the Pareto set (Miettinen, 1999). The Pareto set consists of all solutions where improving any objective would necessarily worsen at least one other objective, representing the best possible trade-offs available. In real-world applications, users often have varying preferences for these trade-offs. For instance, in the development of LLMs, a customer service application might emphasize harmlessness and safety, whereas an educational tool might prioritize reasoning and factual accuracy. To address this variability, users can specify their preferences using a vector, $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_m]^\top \in \Delta_{m-1}$, where $\Delta_{m-1} = \{\boldsymbol{\alpha} \in \mathbb{R}_+^m : \sum_{i=1}^m \alpha_i = 1\}$ is the probability simplex and each α_i represents the importance assigned to the i-th objective. MOO approaches can accommodate these user-defined preferences, enabling the discovery of solutions that align with individual needs.

MOO methods can be broadly divided into gradient-free and gradient-based approaches. Gradient-free methods, such as evolutionary algorithms (Deb et al., 2002; Zhang & Li, 2007) and particle swarm optimization (Coello et al., 2004), explore the solution space using population-based sampling. While effective for traditional low-dimensional MOO problems (Deb, 2011), they struggle with deep neural networks, which often have millions or even billions of parameters. Due to the curse of dimensionality, the parameter space becomes too large to explore efficiently without gradient information, leading to slow or failed convergence. This makes gradient-based methods (Désidéri, 2012; Mukai, 1980; Fliege & Svaiter, 2000; Liu et al., 2021a) preferable for deep neural networks, as they efficiently guide the search using objective gradients.

However, gradient-based MOO faces several significant challenges in deep learning. First, incorporating user preferences is difficult because directly weighting objectives often produces unaligned solutions. Second, training even one neural network is computationally expensive, posing a major challenge to efficiently approximating the entire Pareto set of networks. Finally, the use of mini-batch optimization in deep learning introduces high variance, which can further destabilize the optimization process. These challenges have motivated extensive research efforts to develop specialized MOO techniques for deep learning. This paper provides a comprehensive survey of these advances, systematically categorizing existing approaches and identifying key open problems in the field.

1.1 Scope of the Survey

This survey focuses on gradient-based multi-objective optimization methods within the context of supervised deep learning. We concentrate on gradient-based approaches because, as discussed above, the high dimensionality of modern deep neural networks renders gradient-free methods impractical in these settings. Within this scope, we cover methods spanning the full spectrum of solution outputs: from finding a single balanced model, finding a finite set of the Pareto set, to learning a continuous representation of the entire Pareto set. While multi-objective reinforcement learning is a closely related and highly active field, it possesses distinct characteristics and is already well-documented in existing comprehensive surveys (Felten et al., 2024; Roijers et al., 2013; Hayes et al., 2022). Consequently, although certain methods discussed herein can be adapted for MORL, algorithms designed exclusively for reinforcement learning fall outside the scope of this paper.

1.2 Comparison with Related Works

Previous literature has explored various facets of multi-objective optimization and multi-task learning, though none fully address the specific intersection of gradient-based methods and deep learning. Foundational books (Miettinen, 1999; Ehrgott, 2005) and general surveys (Eichfelder, 2021) provide introductions to classic MOO from a broad optimization perspective, but they primarily focus on early methods and lack an emphasis on deep learning. Other reviews, such as Wei et al. (2021), focus extensively on evolutionary MOO algorithms. While effective in lower-dimensional spaces, these gradient-free approaches are generally unsuitable for deep neural networks due to the massive parameter spaces.

In the machine learning domain, several surveys explore multi-task learning. Zhang & Yang (2022) provide an overview of traditional multi-task methods, while Crawshaw (2020) summarizes deep multi-task learning advancements up to 2020. More recently, Yu et al. (2024) presented a comprehensive review spanning both traditional and deep multi-task learning. However, these works primarily treat the problem from a standard

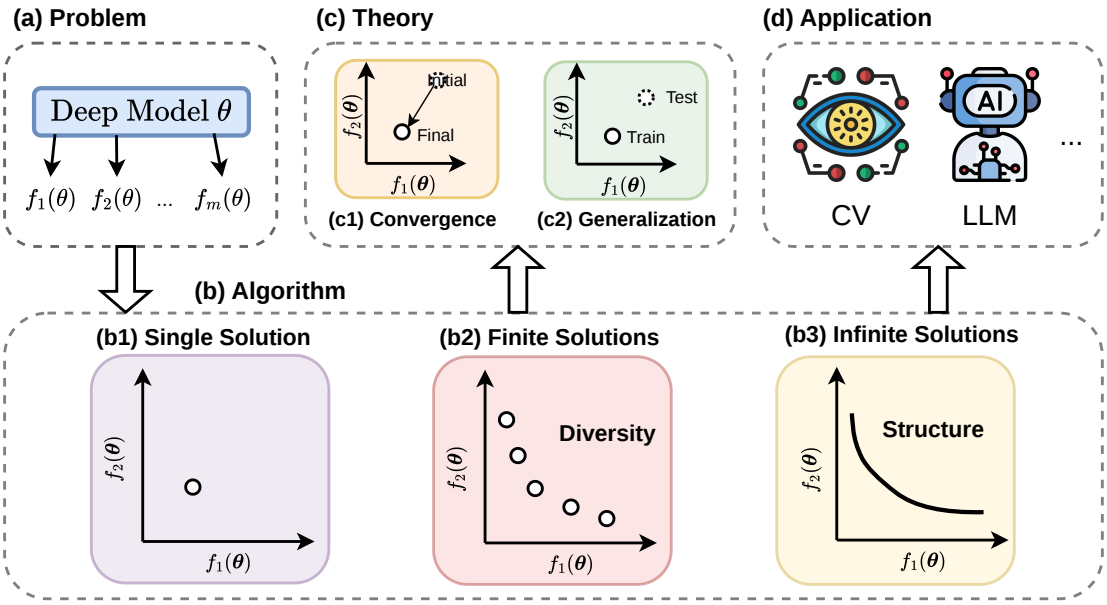


Figure 1: An overview of gradient-based multi-objective deep learning.

multi-task perspective rather than formally grounding the optimization process within the framework of MOO.

The most closely related work to ours is by Peitz & Hotegni (2024), who survey MOO algorithms for deep learning. However, their work focuses on general MOO algorithms and covers only a limited selection of gradient-based methods. Crucially, it lacks a detailed structural taxonomy of gradient-based approaches and omits in-depth discussions on theoretical foundations and practical applications. Finally, several other surveys concentrate strictly on the application of MOO in specific domains, such as dense prediction tasks (Vandenhende et al., 2021), recommender systems (Zhang et al., 2023a), and natural language processing (Chen et al., 2024b).

To bridge these gaps, this survey provides a dedicated, in-depth review of recent gradient-based MOO methods in supervised deep learning. Our main contributions are threefold:

- To the best of our knowledge, this is the first survey focused on gradient-based MOO methods in deep learning, addressing the limitations of previous literature that covers only a fraction of these algorithms.
- We propose a taxonomy that systematically categorizes gradient-based MOO methods based on their target output: finding a single solution, a finite set of solutions, or an infinite set of solutions. We further divide these into detailed subcategories, providing the first structured classification of its kind.
- Beyond a comprehensive algorithmic review, we provide an overview of the theoretical foundations, applications, resources, and future directions.

1.3 Organization of the Survey

In this survey, we categorize gradient-based multi-objective deep learning algorithms based on the type of output they obtain (Figure 2): (i) Algorithms that obtain a single, well-balanced model that is not overly biased toward any one objective: This is achieved through techniques like loss balancing and gradient balancing, which adjust the weight of each objective to resolve conflicts during optimization. (ii) Algorithms that obtain a finite set of Pareto-optimal solutions that allows users to select from multiple options based on

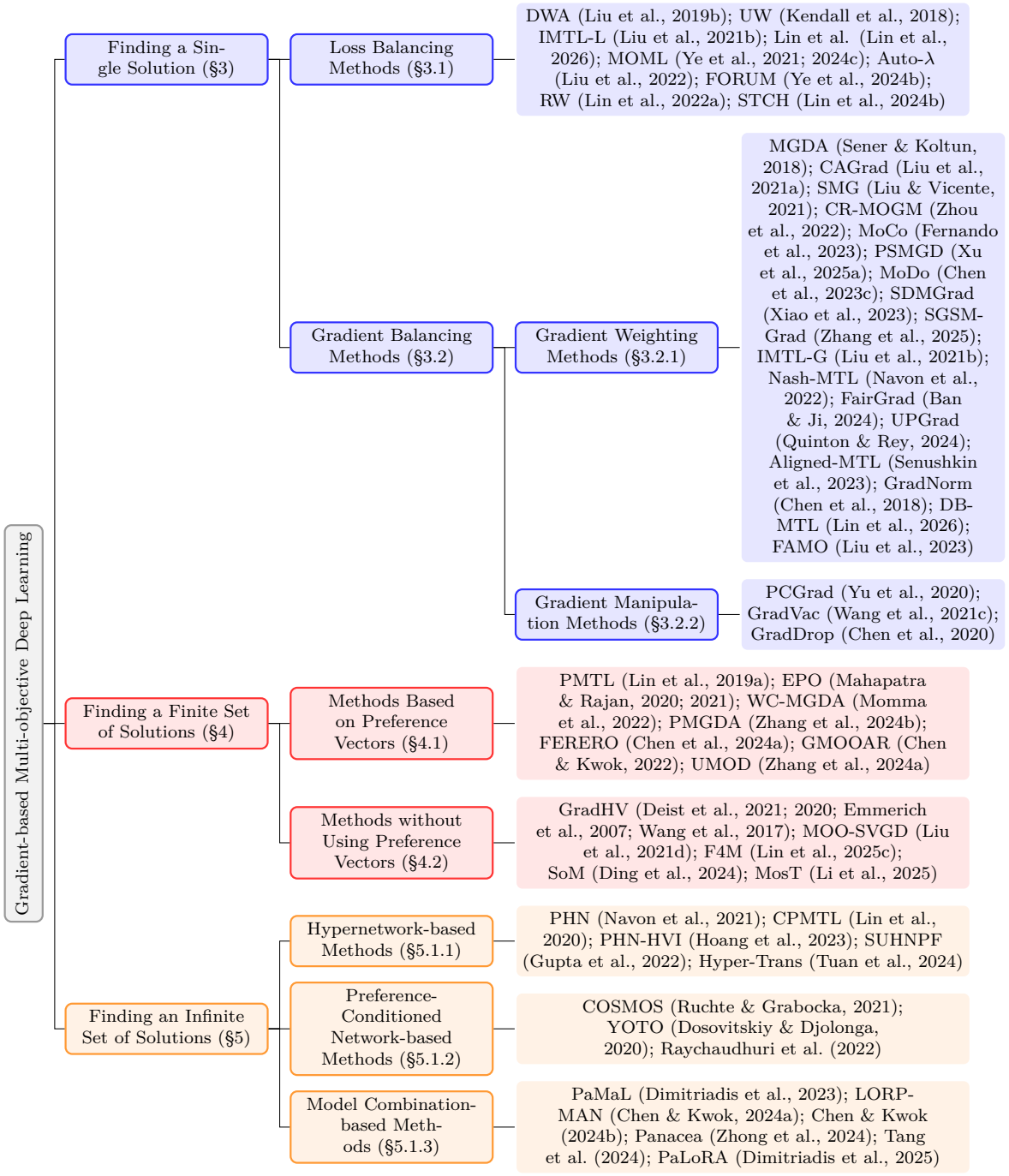


Figure 2: Taxonomy of existing gradient-based multi-objective deep learning algorithms.

their specific needs: This is accomplished by either dividing the problem into subproblems using preference vectors or directly optimizing for multiple solutions simultaneously. (iii) Algorithms that obtain a continuous set of Pareto-optimal models that can be generated on-demand for any given preference: This is achieved by learning a solution subspace using efficient structures, where preference vectors are randomly sampled during training and the optimization adapts techniques from single or finite solution methods.

The rest of this survey is organized as follows: Section 2 presents MOO preliminaries, including MOO definitions and concepts; Section 3 discusses methods for finding a single Pareto-optimal solution, covering

Table 1: Summary of Notations.

Notation	Description
$\boldsymbol{\theta} \in \mathcal{K} \subset \mathbb{R}^d$	Decision variable $\boldsymbol{\theta}$ in feasible set $\mathcal{K}$ with dimension d.
m	Number of objectives.
n	Number of finite Pareto solutions.
K	Number of iterations.
$\boldsymbol{f} = [f_1, \ldots, f_m]^\top$	Objective function.
$\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_m]^\top$	Objective weight vector.
$\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_m]^\top$	Preference vector.
$\boldsymbol{z}^* = [z_1^*, \ldots, z_m^*]^\top$	Ideal objective value.
$\boldsymbol{\theta}^{(k)}$	Solution at k-th iteration.
$\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(n)}$	Solution 1 to Solution n in a size-n solution set.
$\boldsymbol{g}_i^{(k)}$	The gradient vector of i-th objective at k-th iteration.
$\boldsymbol{G}^{(k)} = [\boldsymbol{g}_1^{(k)}, \ldots, \boldsymbol{g}_m^{(k)}]$	Gradient matrix at k-th iteration.
$\boldsymbol{d}^{(k)}$	Updating direction of $\boldsymbol{\theta}$ at k-th iteration.
$\boldsymbol{\phi}$	Parameters of Pareto set learning structures.
$\mathrm{HV}_{\boldsymbol{r}}(\cdot)$	Hypervolume with respect to reference point $\boldsymbol{r}$.
$[m]$	Index set $\{1, \ldots, m\}$.
Δ_{m-1}	$(m-1)$-D simplex $\{\boldsymbol{\alpha} \mid \sum_{i=1}^m \alpha_i = 1, \alpha_i \geq 0, i \in [m]\}$.
$\|\cdot\|$	ℓ_2 norm.
η	Step size for updating $\boldsymbol{\theta}$.
ϵ	Error tolerance.

loss balancing and gradient balancing approaches; Section 4 focuses on methods for identifying a set of finite Pareto-optimal solutions; Section 5 covers methods for identifying a set of infinite Pareto-optimal solutions; Section 6 delves into the theoretical analysis of convergence and generalization in gradient-based multi-objective optimization algorithms; Section 7 showcases various applications in deep learning, including computer vision, neural architecture search, recommender systems, and large language models; Section 8 offers useful resources on datasets and software tools for multi-objective deep learning; Section 9 highlights challenges in the field and suggests promising directions for future research; Section 10 summarizes this survey. An overview is shown in Figure 1.

1.4 Notations

The notations used in this survey are summarized in Table 1. Scalars are represented by non-bold letters; vectors and matrices are denoted by boldface type. Subscripts are used as indices for elements, and superscripts typically denote iteration numbers or indices within a solution set.

2 Preliminary: Multi-Objective Optimization

In this section, we first introduce some MOO concepts in Section 2.1 and then review two classical methods (i.e., linear scalarization and Tchebycheff scalarization) to find Pareto solutions of MOO problems in defined in Section 2.2.

2.1 Key Concepts in Multi-Objective Optimization

Unlike single-objective optimization, solutions in MOO cannot be directly compared based on a single criterion, but are compared using the concept of dominance as follows. Note that without loss of generality, we consider multi-objective minimization (i.e., problem (1)) here.

Definition 1 ((strict) Pareto dominance (Miettinen, 1999))**.** *A solution $\boldsymbol{\theta}^{(a)}$ dominates another solution $\boldsymbol{\theta}^{(b)}$ (denoted as $\boldsymbol{\theta}^{(a)} \preceq \boldsymbol{\theta}^{(b)}$) if and only if $f_i(\boldsymbol{\theta}^{(a)}) \leq f_i(\boldsymbol{\theta}^{(b)})$ for all $i \in [m]$, and there exists at least one $i \in [m]$ such that $f_i(\boldsymbol{\theta}^{(a)}) < f_i(\boldsymbol{\theta}^{(b)})$. A solution $\boldsymbol{\theta}^{(a)}$ strictly dominates another solution $\boldsymbol{\theta}^{(b)}$ if and only if $f_i(\boldsymbol{\theta}^{(a)}) < f_i(\boldsymbol{\theta}^{(b)})$ for all $i \in [m]$.*

Based on this definition, we further define Pareto-optimality (PO), Pareto set, and Pareto front as follows.

Definition 2 ((weak) Pareto optimality (Miettinen, 1999))**.** *A solution $\boldsymbol{\theta}^*$ is Pareto-optimal if no other solution dominates it. A solution $\boldsymbol{\theta}^*$ is weakly Pareto-optimal if no other solution strictly dominates it.*

Definition 3 (Pareto set (PS) and Pareto front (PF) (Miettinen, 1999))**.** *A PS is the set of all PO solutions. A PF is the set of all objective function values of the PO solutions.*

Figure 3 illustrates the Pareto concepts on the two-objective problem. The blue circles represent **Pareto-optimal solutions** (defined in Definition 2), which indicates that no solution can improve one objective without worsening the other. The blue curve connecting the blue circles denotes the **Pareto front**, as defined in Definition 3. The yellow circles indicate **weak Pareto-optimal solutions** since they can be improved in one objective without negatively impacting the other. For example, comparing $\boldsymbol{\theta}_2$ and $\boldsymbol{\theta}_1$, $\boldsymbol{\theta}_2$ is a weak Pareto-optimal solution since it can be improved in the second objective without affecting the first. The red circles represent solutions that are **not Pareto-optimal** because there exists another solution that strictly Pareto dominates them. For instance, comparing $\boldsymbol{\theta}_3$ and $\boldsymbol{\theta}_1$, $\boldsymbol{\theta}_3$ is not a Pareto-optimal solution as $\boldsymbol{\theta}_1$ outperforms it in both objectives.

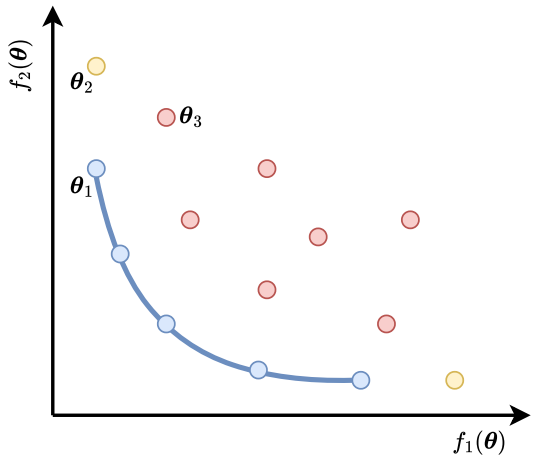


Figure 3: Illustration of Pareto optimality concepts. The blue, yellow, and red circles denote Pareto-optimal solutions, weakly Pareto-optimal solutions, and dominated solutions, respectively. The blue curve represents the Pareto front.

Definition 4 (Preference vector)**.** *$\boldsymbol{\alpha}$ denotes a preference vector. The value of the entries of $\boldsymbol{\alpha}$ denote the preference of a specific objective. Throughout this paper, a preference vector $\boldsymbol{\alpha}$ is constrained on a simplex Δ_{m-1}, where $\Delta_{m-1} = \{\boldsymbol{\alpha} | \sum_{i=1}^{m} \alpha_i = 1, \alpha_i \geq 0, i \in [m]\}$.*

In deep learning, the parameter $\boldsymbol{\theta}$ is typically optimized using gradient descent. In single-objective optimization, when the objective function is non-convex, the optimization process often reaches a stationary point. In the case of MOO, the solution generally converges to a *Pareto stationary* solution, which is defined as follows:

Definition 5 (Pareto stationary (Désidéri, 2012))**.** *A solution $\boldsymbol{\theta}^*$ is called* Pareto stationary *if there exists a vector $\boldsymbol{\lambda} \in \Delta_{m-1}$ such that $\| \sum_{i=1}^{m} \lambda_i \nabla f_i(\boldsymbol{\theta}^*) \| = 0$.*

Pareto stationarity is a necessary condition for achieving Pareto optimality. If all objectives are convex with $\lambda_i > 0, \forall i$, it also serves as the Karush-Kuhn-Tucker (KKT) sufficient and necessary condition for Pareto optimality (Désidéri, 2012).

When evaluating the performance of a set of obtained solutions, the Hypervolume (HV) (Zitzler & Thiele, 1998) indicator is one of the most widely used performance indicators, which is defined as follows.

Definition 6 (Hypervolume (Zitzler & Thiele, 1998)). *Given a solution set* $\mathbb{S} = \{\boldsymbol{q}^{(1)}, \ldots, \boldsymbol{q}^{(N)}\}$ *and a reference point* $\boldsymbol{r}$*, the hypervolume of* $\mathbb{S}$ *is calculated by:*

$$\mathrm{HV}_{\boldsymbol{r}}(\mathbb{S}) = \Lambda(\boldsymbol{p} \mid \exists \boldsymbol{q} \in \mathbb{S} : \boldsymbol{q} \preceq \boldsymbol{p} \preceq \boldsymbol{r}), \tag{2}$$

where $\Lambda(\cdot)$ *denotes the Lebesgue measure of a set.*

HV provides a unary measure of both convergence and diversity of a solution set. A larger HV indicates that the obtained solutions are more diverse and closer the PF in the objective space. In addition to HV, several other performance indicators are frequently utilized in MOO: (i) Generational Distance (GD) (Van Veldhuizen & Lamont, 1999): This metric measures the convergence of the obtained solution set by calculating the average Euclidean distance from each point in the solution set to the nearest point on the true PF. A smaller GD value indicates better convergence. (ii) Inverted Generational Distance (IGD) (Zitzler et al., 2003): Unlike GD, IGD measures the average distance from each point on the true PF to the nearest point in the obtained solution set. Consequently, IGD assesses both convergence and diversity simultaneously. A lower IGD value is desirable. For a more comprehensive review of performance indicators in MOO, we refer readers to (Jiang et al., 2014).

2.2 Scalarization Methods

Linear Scalarization (LS). The most straightforward way to solve MOO problems is to reformulate them as single-objective optimization problems weighted by the given preference vector $\boldsymbol{\alpha}$ as follows:

$$\min_{\boldsymbol{\theta} \in \mathcal{K}} g^{\mathrm{LS}}_{\boldsymbol{\alpha}}(\boldsymbol{\theta}) := \sum_{i=1}^{m} \alpha_i f_i(\boldsymbol{\theta}). \tag{3}$$

LS is widely used in practice due to its simplicity. However, this method suffers from a limitation: it can only identify solutions on the convex part of the PF. For PFs with a concave shape, using LS only finds the endpoints of a PF.

Tchebycheff Scalarization. Another way to convert a MOO problem to a single objective optimization problem is to use the Tchebycheff scalarization function (Bowman Jr, 1976; Steuer & Choo, 1983), defined as:

$$\min_{\boldsymbol{\theta} \in \mathcal{K}} g^{\mathrm{Tche}}_{\boldsymbol{\alpha}}(\boldsymbol{\theta}) := \max_{i \in [m]} \alpha_i (f_i(\boldsymbol{\theta}) - z_i^*), \tag{4}$$

where z_i^* are reference values, usually set as the ideal value of the i-th objective. Unlike linear scalarization, Tchebycheff scalarization can explore the entire PF by traversing all preference vectors within the simplex. This ensures that both convex and concave regions of the PF are covered. The relationship between Tchebycheff scalarization and weak Pareto optimality can be formally described as follows:

Theorem 1 ((Choo & Atkins, 1983)). *A solution* $\boldsymbol{\theta}^*$ *of the original MOO problem (1) is weakly Pareto-optimal if and only if there exists a preference vector* $\boldsymbol{\alpha}$ *such that* $\boldsymbol{\theta}^*$ *is an optimal solution of problem (4).*

If $\boldsymbol{\theta}^*$ is unique for a given $\boldsymbol{\alpha}$, it is considered Pareto-optimal. However, Tchebycheff scalarization poses challenges for gradient-based deep learning due to the nonsmooth nature of the $\max(\cdot)$ operator. This nonsmoothness leads to non-differentiability and a slow convergence rate of $\mathcal{O}(1/\epsilon^2)$, where ϵ represents the error tolerance (Lin et al., 2024b). Even when all objectives $\{f_i\}_{i=1}^m$ are differentiable, problem (4) is still non-differentiable.

3 Finding a Single Pareto-optimal Solution

In many real-world scenarios, such as multi-task learning (MTL) (Zhang & Yang, 2022; Ruder, 2017; Crawshaw, 2020), users genuinely require a single model that achieves a balance between objectives. We define "balance" as finding a fair compromise near the center of the Pareto front, ensuring that no single objective is disproportionately degraded in favor of others.

While using multiple task-specific networks offers flexibility, deploying a single balanced model is frequently necessary due to practical constraints. For example, in applications like autonomous driving, systems must perform multiple tasks simultaneously. Strict latency limits require these tasks to be executed in a single forward pass, making it impractical to run multiple models that require separate inferences. Similarly, in LLM alignment, a single generated response is evaluated simultaneously on multiple criteria. Since one output must satisfy all of these criteria at once, utilizing multiple separate models is not a viable solution.

In such cases, the goal is to identify a single, well-balanced Pareto-optimal solution. By default, objectives are treated as equally important, which corresponds to a uniform preference vector $\boldsymbol{\alpha} = [\frac{1}{m}, \ldots, \frac{1}{m}]^\top$. If the objectives carry unequal importance and a specific preference vector can be provided, users can simply rescale the objective functions accordingly. Once rescaled, the algorithms discussed in this section can then be applied to find the desired solution.

The most straightforward method is linear scalarization (i.e., problem (3)) with $\alpha_i = \frac{1}{m}$, which is known as Equal Weighting (EW) (Zhang & Yang, 2022) in MTL. However, EW may cause some tasks to have unsatisfactory performance (Standley et al., 2020; Lin et al., 2026). Therefore, to improve the performance, many methods have been proposed to dynamically tune the objective weights $\{\lambda_i\}_{i=1}^m$ during training, which can in general be formulated as:

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{m} \lambda_i f_i(\boldsymbol{\theta}), \tag{5}$$

where λ_i is the weight for the i-th objective $f_i(\boldsymbol{\theta})$ and the uniform preference vector $\boldsymbol{\alpha}$ is omitted for simplicity. These methods can be categorized as loss balancing methods and gradient balancing approaches. Some representative loss/gradient balancing methods are illustrated in Figure 4 (adapted from (Liu et al., 2021a;b)).

3.1 Loss Balancing Methods

This type of approach dynamically computes or learns the objective weights $\{\lambda_i\}_{i=1}^m$ during training using some measures on loss such as the decrease speed of loss (Liu et al., 2019b), homeostatic uncertainty of loss (Kendall et al., 2018), loss scale (Liu et al., 2021b; Lin et al., 2026), and validation loss (Ye et al., 2021; 2024c;b; Liu et al., 2022).

Dynamic Weight Average (DWA) (Liu et al., 2019b) estimates each objective weight as the ratio of the training losses in the last two iterations. Formally, the objective weight in k-th iteration ($k > 2$) is computed as follows:

$$\lambda_i^{(k)} = \frac{m \exp(\omega_i^{(k-1)}/\gamma)}{\sum_{j=1}^m \exp(\omega_j^{(k-1)}/\gamma)}, \quad \omega_j^{(k-1)} = \frac{f_j^{(k-1)}}{f_j^{(k-2)}}, \tag{6}$$

where γ is the temperature and $f_i^{(k)}$ is the loss value of i-th objective at k-th iteration.

Kendall et al. (2018) propose Uncertainty Weighting (UW) that considers the homoscedastic uncertainty of each objective and optimizes objective weights together with the model parameter as follows:

$$\min_{\boldsymbol{\theta}, \boldsymbol{s}} \sum_{i=1}^{m} \left(\frac{1}{2s_i^2} f_i(\boldsymbol{\theta}) + \log s_i \right), \tag{7}$$

where $\boldsymbol{s} = [s_1, \ldots, s_m]^\top$ is learnable and $\log s_i$'s are regularization terms.

Impartial Multi-Task Learning (IMTL) (Liu et al., 2021b) balances losses across tasks, abbreviated as IMTL-L(oss). Specifically, IMTL-L encourages all objectives to have a similar loss scale by transforming each objective $f_i(\boldsymbol{\theta})$ as $e^{s_i} f_i(\boldsymbol{\theta}) - s_i$. Similar to UW (Kendall et al., 2018), IMTL-L simultaneously learns the transformation parameter $\boldsymbol{s}$ and the model parameter $\boldsymbol{\theta}$ as follows:

$$\min_{\boldsymbol{\theta}, \boldsymbol{s}} \sum_{i=1}^{m} \left(e^{s_i} f_i(\boldsymbol{\theta}) - s_i \right). \tag{8}$$

Similar to IMTL-L (Liu et al., 2021b), Lin et al. (2026) also aim to make all objective losses have a similar scale. They achieve it by performing a logarithm transformation on each objective (i.e., $\log f_i(\boldsymbol{\theta})$). Moreover, they theoretically show that the transformation in IMTL-L is equivalent to the logarithm transformation when s_i is the exact minimizer of $\min_{s_i} e^{s_i} f_i(\boldsymbol{\theta}) - s_i$ in each iteration. Hence, the logarithm transformation can recover the transformation in IMTL-L.

Ye et al. (2021; 2024c) propose Multi-Objective Meta Learning (MOML), which adaptively tunes the objective weights $\boldsymbol{\lambda}$ based on the validation performance, reformulating problem (5) as a multi-objective bi-level optimization problem:

$$\min_{\boldsymbol{\lambda}} \ \left[f_1(\boldsymbol{\theta}^*(\boldsymbol{\lambda}); \mathcal{D}_1^{\text{val}}), \ldots, f_m(\boldsymbol{\theta}^*(\boldsymbol{\lambda}); \mathcal{D}_m^{\text{val}})\right]^\top \tag{9a}$$

$$\text{s.t.} \ \ \boldsymbol{\theta}^*(\boldsymbol{\lambda}) = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^m \lambda_i f_i(\boldsymbol{\theta}; \mathcal{D}_i^{\text{tr}}), \tag{9b}$$

where $\mathcal{D}_i^{\text{tr}}$ and $\mathcal{D}_i^{\text{val}}$ are the training and validation datasets for the i-th objective, respectively. In each training iteration, when given objective weights $\{\lambda_i\}_{i=1}^m$, MOML first learns the model $\boldsymbol{\theta}^*(\boldsymbol{\lambda})$ on the training dataset by solving lower-level (LL) subproblem (9b) with T iterations and then updates $\boldsymbol{\lambda}$ in the upper-level (UL) subproblem (9a) via minimizing the loss of the trained MTL model $\boldsymbol{\theta}^*(\boldsymbol{\lambda})$ on the validation dataset of each objective. However, when solving the UL subproblem (9a), MOML needs to calculate the complex hypergradient $\nabla_{\boldsymbol{\lambda}} \boldsymbol{\theta}^*(\boldsymbol{\lambda})$ which requires to compute many Hessian-vector products via the chain rule. Hence, the time and memory costs of MOML grow significantly fast with respect to the dimension of $\boldsymbol{\theta}$ and the number of LL iterations T.

Auto-λ (Liu et al., 2022) modifies problem (9) by replacing the multi-objective UL subproblem (9a) with a single-objective problem: $\min_{\boldsymbol{\lambda}} \sum_{i=1}^m f_i(\boldsymbol{\theta}^*(\boldsymbol{\lambda}); \mathcal{D}_i^{\text{val}})$. Besides, Auto-$\lambda$ approximates the hypergradient $\nabla_{\boldsymbol{\lambda}} \boldsymbol{\theta}^*(\boldsymbol{\lambda})$ via the finite difference approach. Thus, it is more efficient than MOML (Ye et al., 2021; 2024c).

FORUM (Ye et al., 2024b) is proposed to improve the efficiency of MOML (Ye et al., 2021; 2024c). FORUM first reformulates problem (9) as a constrained MOO problem via the value-function approach (Liu et al., 2021c) and then proposes a multi-gradient aggregation method to solve it. Compared with MOML, FORUM is a first-order method and does not need to compute the hypergradient $\nabla_{\boldsymbol{\lambda}} \boldsymbol{\theta}^*(\boldsymbol{\lambda})$. Thus, it is significantly more efficient than MOML.

Random Weighting (RW) (Lin et al., 2022a) randomly samples objective weights from a distribution (such as the standard normal distribution) and normalizes them into the simplex in each iteration. RW has a higher probability of escaping local minima than EW due to the randomness from objective weights, resulting in better generalization performance.

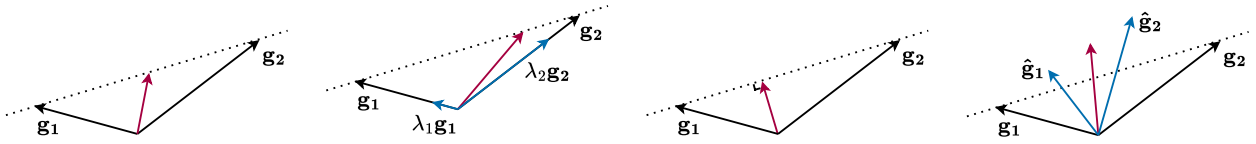
Different from the above methods based on linear scalarization, Lin et al. (2024b) propose Smooth Tchebycheff scalarization (STCH), which transforms the Tchebycheff scalarization into a smooth function:

$$\min_{\boldsymbol{\theta}} f_{\boldsymbol{\alpha}}^{\text{STCH}}(\boldsymbol{\theta}, \mu) := \mu \log \sum_{i=1}^m \exp\left(\frac{\alpha_i (f_i(\boldsymbol{\theta}) - z_i^*)}{\mu}\right), \tag{10}$$

where $\mu > 0$ is a smoothing parameter, and $\boldsymbol{z}^*$ is a reference point as in Tchebycheff scalarization (problem (4)). STCH improves upon the original Tchebycheff scalarization by making $f_{\boldsymbol{\alpha}}^{\text{STCH}}(\boldsymbol{\theta}, \mu)$ smooth when all objective functions $f_i(\boldsymbol{\theta})$'s are smooth, resulting in faster convergence. It also retains the advantages of the original Tchebycheff scalarization: (1) convexity when all objective functions are convex and (2) Pareto optimality by appropriately choosing μ.

3.2 Gradient Balancing Methods

Let $\boldsymbol{g}_i^{(k)} = \nabla_{\boldsymbol{\theta}} f_i(\boldsymbol{\theta})|_{\boldsymbol{\theta}^{(k)}} \in \mathbb{R}^d$ be the gradient of the i-th objective at the k-th iteration. This type of approach aims to find a common update direction $\boldsymbol{d}^{(k)}$ by adaptively aggregating the gradients of all objectives $\{\boldsymbol{g}_i^{(k)}\}_{i=1}^m$ at each iteration. Then, the model parameter is updated via $\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \eta \boldsymbol{d}^{(k)}$, where η is a step size. To simplify notations, we omit the superscript in this section. Compared with loss balancing methods in Section 3.1, gradient balancing approaches usually achieve better performance.



(a) EW. (b) DWA (Liu et al., 2019b). (c) MGDA (Sener & Koltun, 2018). (d) PCGrad (Yu et al., 2020).

Figure 4: Illustration of the update direction $\boldsymbol{d}$ of several representative methods: EW, DWA (Liu et al., 2019b) from loss balancing methods, MGDA (Sener & Koltun, 2018) from gradient weighting methods, and PCGrad (Yu et al., 2020) from gradient manipulation methods. They are illustrated in a two-objective learning problem (two specific gradients are labeled as $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$).

3.2.1 Gradient Weighting Methods

In most gradient weighting methods, $\boldsymbol{d}$ is computed by:

$$\boldsymbol{d} = \boldsymbol{G}\boldsymbol{\lambda}, \tag{11}$$

where $\boldsymbol{G} = [\boldsymbol{g}_1, \dots, \boldsymbol{g}_m] \in \mathbb{R}^{d\times m}$ is the gradient matrix and $\boldsymbol{\lambda} = [\lambda_1, \dots, \lambda_m]^\top$ is the gradient weights.

Sener & Koltun (2018) apply Multiple-Gradient Descent Algorithm (MGDA) (Désidéri, 2012), which aims to find a direction $\boldsymbol{d}$ at each iteration to maximize the minimal decrease across the losses. This is formulated as:

$$\max_{\boldsymbol{d}} \min_{i\in[m]} \left(f_i(\boldsymbol{\theta}) - f_i(\boldsymbol{\theta} - \eta\boldsymbol{d})\right) \approx \max_{\boldsymbol{d}} \min_{i\in[m]} \boldsymbol{g}_i^\top \boldsymbol{d}, \tag{12}$$

where η is the step size. The solution of problem (12) is $\boldsymbol{d} = \boldsymbol{G}\boldsymbol{\lambda}$, where $\boldsymbol{\lambda}$ is computed as:

$$\boldsymbol{\lambda} = \arg\min_{\boldsymbol{\lambda}\in\Delta_{m-1}} \left\|\boldsymbol{G}\boldsymbol{\lambda}\right\|^2, \tag{13}$$

where Δ_{m-1} denotes the simplex. Sener & Koltun (2018) use the Frank-Wolfe algorithm (Jaggi, 2013) to solve problem (13).

Conflict-Averse Gradient descent (CAGrad) (Liu et al., 2021a) improves MGDA (Sener & Koltun, 2018) by constraining the aggregated gradient $\boldsymbol{d}$ to be around the average gradient $\boldsymbol{g}_0 = \frac{1}{m}\sum_{i=1}^m \boldsymbol{g}_i$ with a distance $c\left\|\boldsymbol{g}_0\right\|$, where $c \in [0, 1)$ is a constant. Specifically, $\boldsymbol{d}$ is computed by solving the following problem:

$$\max_{\boldsymbol{d}} \min_{i\in[m]} \boldsymbol{g}_i^\top \boldsymbol{d}, \quad \text{s.t.} \quad \left\|\boldsymbol{d} - \boldsymbol{g}_0\right\| \le c\left\|\boldsymbol{g}_0\right\|. \tag{14}$$

Problem (14) is equivalent to first computing $\boldsymbol{\lambda}$ by solving the following problem:

$$\boldsymbol{\lambda} = \arg\min_{\boldsymbol{\lambda}\in\Delta_{m-1}} \boldsymbol{g}_{\boldsymbol{\lambda}}^\top \boldsymbol{g}_0 + \left\|\boldsymbol{g}_0\right\| \left\|\boldsymbol{g}_{\boldsymbol{\lambda}}\right\|, \tag{15}$$

where $\boldsymbol{g}_{\boldsymbol{\lambda}} = \frac{1}{m}\boldsymbol{G}\boldsymbol{\lambda}$ and then calculate the update direction $\boldsymbol{d} = \boldsymbol{g}_0 + \frac{c}{\left\|\boldsymbol{g}_{\boldsymbol{\lambda}}\right\|}\boldsymbol{g}_{\boldsymbol{\lambda}}$. Note that CAGrad is simplified to EW when $c = 0$ and degenerates into MGDA (Sener & Koltun, 2018) when $c \to +\infty$.

MGDA (Sener & Koltun, 2018) and its variant CAGrad (Liu et al., 2021a) can converge to a Pareto stationary point in the deterministic setting with full-gradient computations. However, in deep learning, where stochastic gradients are used, many works (Liu & Vicente, 2021; Zhou et al., 2022; Xu et al., 2025a; Fernando et al., 2023; Chen et al., 2023c; Xiao et al., 2023; Zhang et al., 2025) explore the convergence properties of MGDA under stochastic gradients, which are introduced in detail in Section 6.1.

IMTL-G(rad) (Liu et al., 2021b) aims to find $\boldsymbol{d}$ that has equal projections on all objective gradients. Thus, we have:

$$\boldsymbol{u}_1^\top \boldsymbol{d} = \boldsymbol{u}_i^\top \boldsymbol{d}, \quad 2 \le i \le m, \tag{16}$$

where $\boldsymbol{u}_i = \boldsymbol{g}_i/\|\boldsymbol{g}_i\|$. If constraining $\sum_{i=1}^m \lambda_i = 1$, problem (16) has a closed-form solution of $\boldsymbol{\lambda}$:

$$\boldsymbol{\lambda}_{(2,\dots,m)} = \boldsymbol{g}_1^\top \boldsymbol{U} \left(\boldsymbol{D}\boldsymbol{U}^\top\right)^{-1}, \quad \lambda_1 = 1 - \sum_{i=2}^m \lambda_i, \tag{17}$$

where $\boldsymbol{\lambda}_{(2,\dots,m)} = [\lambda_2, \dots, \lambda_m]^\top$, $\boldsymbol{U} = [\boldsymbol{u}_1 - \boldsymbol{u}_2, \dots, \boldsymbol{u}_1 - \boldsymbol{u}_m]$, and $\boldsymbol{D} = [\boldsymbol{g}_1 - \boldsymbol{g}_2, \dots, \boldsymbol{g}_1 - \boldsymbol{g}_m]$.

Nash-MTL (Navon et al., 2022) formulates problem (11) as a bargaining game (Nash, 1953; Thomson, 1994), where each objective is cast as a player and the utility function for each player is defined as $\boldsymbol{g}_i^\top \boldsymbol{d}$. Hence, $\boldsymbol{d}$ is computed by solving the following problem:

$$\max_{\boldsymbol{d}} \sum_{i=1}^m \log\left(\boldsymbol{g}_i{}^\top \boldsymbol{d}\right). \tag{18}$$

The solution of problem (18) is $\boldsymbol{d} = \boldsymbol{G}\boldsymbol{\lambda}$, where $\boldsymbol{\lambda}$ is obtained by solving the following problem:

$$\boldsymbol{G}^\top \boldsymbol{G}\boldsymbol{\lambda} = \boldsymbol{\lambda}^{-1}, \tag{19}$$

which can be approximately solved using a sequence of convex optimization problems.

FairGrad (Ban & Ji, 2024) extends Nash-MTL (Navon et al., 2022) by computing $\boldsymbol{\lambda}$ through the equation $\boldsymbol{G}^\top \boldsymbol{G}\boldsymbol{\lambda} = \boldsymbol{\lambda}^{-1/\gamma}$, where $\gamma \geq 0$ is a constant. Unlike Nash-MTL (Navon et al., 2022), FairGrad treats the problem as a constrained nonlinear least squares problem. Similarly, UPGrad (Quinton & Rey, 2024) calculates $\boldsymbol{\lambda}$ by optimizing $\min_{\boldsymbol{\lambda}} \boldsymbol{\lambda}^\top (\boldsymbol{G}^\top \boldsymbol{G})\boldsymbol{\lambda}$, a quadratic programming (QP) problem.

Aligned-MTL (Senushkin et al., 2023) considers problem (11) as a linear system and aims to enhance its stability by minimizing the condition number of $\boldsymbol{G}$. Specifically, the Gram matrix $\boldsymbol{C} = \boldsymbol{G}^\top \boldsymbol{G}$ is first computed. Then, the eigenvalues $\{\sigma_1, \dots, \sigma_R\}$ and eigenvectors $\boldsymbol{V}$ of $\boldsymbol{C}$ are obtained. Finally, $\boldsymbol{\lambda}$ is computed by:

$$\boldsymbol{\lambda} = \sqrt{\sigma_R} \boldsymbol{V} \boldsymbol{\Sigma}^{-1} \boldsymbol{V}^\top, \tag{20}$$

where $\boldsymbol{\Sigma}^{-1} = \operatorname{diag}\left(\sqrt{1/\sigma_1}, \dots, \sqrt{1/\sigma_R}\right)$.

Gradient Normalization (GradNorm) (Chen et al., 2018) aims to learn $\boldsymbol{\lambda}$ so that the scaled gradients have similar magnitudes. It updates $\boldsymbol{\lambda}$ by solving the following problem via one-step gradient descent:

$$\min_{\boldsymbol{\lambda}} \sum_{i=1}^m \left(\|\lambda_i \boldsymbol{g}_i\| - c \times r_i^\gamma\right), \tag{21}$$

where $c = \frac{1}{m}\sum_{i=1}^m \|\lambda_i \boldsymbol{g}_i\|$ is the average scaled gradient norm and is treated as a constant, $r_i = \frac{f_i/f_i^{(0)}}{\frac{1}{m}\sum_{j=1}^m (f_j/f_j^{(0)})}$ is the training speed of the i-th objective and is also considered as a constant, and $\gamma > 0$ is a hyperparameter.

Dual-Balancing Multi-Task Learning (DB-MTL) (Lin et al., 2026) also normalizes all objective gradients to the same magnitude. $\boldsymbol{\lambda}$ is computed as $\lambda_i = \gamma/\|\boldsymbol{g}_i\|$, where $\gamma = \max_{i\in[m]} \|\boldsymbol{g}_i\|$ is a scaling factor controlling the update magnitude. Thus, compared with GradNorm (Chen et al., 2018), this method guarantees all objective gradients have the same norm in each iteration. Moreover, they observe that the choice of the update magnitude γ significantly affects performance.

3.2.2 Gradient Manipulation Methods

To address gradient conflicts, gradient manipulation methods correct each objective gradient $\boldsymbol{g}_i$ to $\hat{\boldsymbol{g}}_i$ and then compute the update direction as $\boldsymbol{d} = \sum_{i=1}^m \hat{\boldsymbol{g}}_i$.

Projecting Conflicting Gradients (PCGrad) (Yu et al., 2020) considers two objective gradients $\boldsymbol{g}_i$ and $\boldsymbol{g}_j$ as conflicting if $\boldsymbol{g}_i^\top \boldsymbol{g}_j < 0$. Then, it corrects each objective gradient by projecting it onto the normal plane of

other objectives' gradients. Specifically, at each iteration, $\hat{\boldsymbol{g}}_i$ is initialized with its original gradient $\boldsymbol{g}_i$. Then, for each $j \neq i$, if $\hat{\boldsymbol{g}}_i^\top \boldsymbol{g}_j < 0$, $\hat{\boldsymbol{g}}_i$ is corrected as:

$$\hat{\boldsymbol{g}}_i = \hat{\boldsymbol{g}}_i - \frac{\hat{\boldsymbol{g}}_i^\top \boldsymbol{g}_j}{\|\boldsymbol{g}_j\|^2} \boldsymbol{g}_j. \tag{22}$$

This reduces gradient conflicts by removing the components of $\hat{\boldsymbol{g}}_i$ that oppose other objective gradients.

While PCGrad (Yu et al., 2020) corrects the gradient if and only if two objectives have a negative gradient similarity, Gradient Vaccine (GradVac) (Wang et al., 2021c) extends it to a more general and adaptive formulation. In each iteration, if $\phi_{ij} < \bar{\phi}_{ij}$, GradVac updates the corrected gradient $\hat{\boldsymbol{g}}_i$ as:

$$\hat{\boldsymbol{g}}_i = \hat{\boldsymbol{g}}_i - \frac{\|\hat{\boldsymbol{g}}_i\| \left(\bar{\phi}_{ij} \sqrt{1 - \phi_{ij}^2} - \phi_{ij} \sqrt{1 - \bar{\phi}_{ij}^2} \right)}{\|\boldsymbol{g}_j\| \sqrt{1 - \bar{\phi}_{ij}^2}} \boldsymbol{g}_j, \tag{23}$$

where ϕ_{ij} is the cosine similarity between $\hat{\boldsymbol{g}}_i$ and $\boldsymbol{g}_j$. $\bar{\phi}_{ij}$ is initialized to 0 and updated by Exponential Moving Average (EMA), i.e., $\bar{\phi}_{ij} \leftarrow (1 - \beta)\bar{\phi}_{ij} + \beta \phi_{ij}$, where β is a hyperparameter. Note that GradVac simplifies to PCGrad when $\bar{\phi}_{ij} = 0$. Gradient Sign Dropout (GradDrop) (Chen et al., 2020) considers that conflicts arise from differences in the sign of gradient values. Thus, a probabilistic masking procedure is proposed to preserve gradients with consistent signs during each update.

3.2.3 Practical Speedup Strategy

Gradient balancing methods suffer from high computational and storage costs. Specifically, in each iteration, almost all gradient balancing methods require performing m back-propagation processes to obtain all objective gradients w.r.t. the model parameter $\boldsymbol{\theta} \in \mathbb{R}^d$ and then store these gradients $\boldsymbol{G} \in \mathbb{R}^{d \times m}$. This can be computationally expensive when dealing with a large number of objectives or using a neural network with a large number of parameters. Moreover, many gradient balancing methods (such as MGDA (Sener & Koltun, 2018), CAGrad (Liu et al., 2021a), Nash-MTL (Navon et al., 2022), and FairGrad (Ban & Ji, 2024)) need to solve an optimization problem to obtain the objective weight $\boldsymbol{\lambda}$ in each iteration, which also increases the computational and memory costs. Hence, several strategies are proposed to alleviate this problem.

Sener & Koltun (2018) use feature-level gradients (i.e., gradients w.r.t. the representations from the last shared layer) to replace the parameter-level gradients (i.e., gradients w.r.t. the shared parameters $\boldsymbol{\theta}$). Since the dimension of the representation is much smaller than that of the shared parameters, it can significantly reduce the computational and memory costs. This strategy is also adopted by some gradient balancing methods, such as IMTL-G (Liu et al., 2021b) and Aligned-MTL (Senushkin et al., 2023).

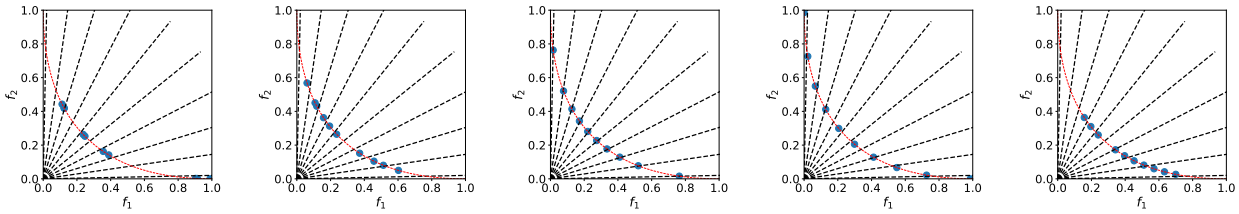
Liu et al. (2021a) randomly select a subset of objectives to calculate the update direction in each iteration. Navon et al. (2022) propose to update the objective weight $\boldsymbol{\lambda}$ every τ iterations instead of updating in each iteration. Although this strategy speeds up training, they observe that it may cause a noticeable drop in performance. Liu et al. (2023) consider optimizing the logarithm of the MGDA objective (Sener & Koltun, 2018) and propose a speedup strategy. Specifically, when solving problem (13) via one-step gradient descent, $\boldsymbol{\lambda}$ is updated as $\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} - \eta \nabla_{\boldsymbol{\lambda}} \|\boldsymbol{G}\boldsymbol{\lambda}\|^2$. Note that

$$\frac{1}{2} \nabla_{\boldsymbol{\lambda}} \|\boldsymbol{G}\boldsymbol{\lambda}\|^2 = \boldsymbol{G}^\top \boldsymbol{G}\boldsymbol{\lambda} = \boldsymbol{G}^\top \boldsymbol{d} \approx \frac{1}{\eta} \left[f_1^{(k)} - f_1^{(k+1)}, \ldots, f_m^{(k)} - f_m^{(k+1)} \right]^\top, \tag{24}$$

where $f_i^{(k)}$ is the loss value of the i-th objective in the k-th iteration. Hence, $\boldsymbol{\lambda}$ can be approximately updated using the change in losses without computing all objective gradients. Although this strategy significantly reduces computational and memory costs, it may cause performance degradation. Moreover, it is only applicable to MGDA-based methods.

3.3 Discussions

Loss balancing and gradient balancing methods effectively mitigate conflicts among different objectives and achieve a Pareto-optimal solution, which is important for MOO scenarios such as multi-task learning. How-



(a) MGDA (Sener & Koltun, 2018). (b) PMTL (Lin et al., 2019a). (c) EPO (Mahapatra & Rajan, 2020; 2021). (d) GradHV (Wang et al., 2017). (e) MOO-SVGD (Liu et al., 2021d).

Figure 5: Distributions of solutions obtained by using representative methods: PMTL (Lin et al., 2019a) and EPO (Mahapatra & Rajan, 2020; 2021) are preference vector-based methods, while GradHV (Wang et al., 2017) and MOO-SVGD (Liu et al., 2021d) are methods not using preference vectors. We also include MGDA (Sener & Koltun, 2018) as a reference; it does not use preference vectors nor incorporate any mechanism to promote diversity. The blue circles denote the solutions, the red curves denote the ground truth PF and the black lines denote the preference vectors. As can be seen, MGDA (Sener & Koltun, 2018) demonstrates poor diversity due to its lack of preference incorporation. While the preference vector-based method PMTL (Lin et al., 2019a) achieves better alignment with the preference vector, it still falls short of exact alignment. In contrast, EPO (Mahapatra & Rajan, 2020; 2021) successfully achieves exact alignment. GradHV (Wang et al., 2017) and MOO-SVGD (Liu et al., 2021d) can also generate diverse solutions, despite they do not use a preference vector.

ever, they are limited to finding a single Pareto-optimal solution and lack the ability to control the solution's position on the Pareto front, resulting in inapplicability in some scenarios such as multi-criteria learning. In Sections 4 and 5, we introduce methods to achieve more diverse and targeted solutions.

The computational cost of loss balancing methods (except MOML (Ye et al., 2021; 2024c), Auto-λ (Liu et al., 2022), and FORUM (Ye et al., 2024b)) is almost the same as that of EW since they only need one backpropagation process in each training iteration. However, gradient balancing methods are much more computationally expensive due to the need of m backpropagation processes to obtain each objective gradient, where m is the number of objectives. While various speedup strategies can address this problem, they often come at the cost of performance, as detailed in Section 3.2.3. Compared to loss balancing methods, gradient balancing methods generally exhibit better convergence properties such as guaranteeing to converge to a Pareto stationary point, detailed in Section 6.1.

4 Finding a Finite Set of Solutions

In some scenarios, a single solution may be insufficient to balance the objectives, as users may prefer to obtain multiple trade-off solutions and select one based on their specific needs. This section introduces algorithms to identify a finite set of solutions, providing a discrete approximation of the Pareto set. We first discuss methods based on preference vectors (Section 4.1), followed by approaches that do not require the use of preference vectors (Section 4.2). Figure 5 shows solutions obtained by some representative methods on LSMOP1 (Cheng et al., 2016) after 5000 iterations of each algorithm.

4.1 Methods Based on Preference Vectors

Preference vector-based methods rely on a preference vector set $\{\boldsymbol{\alpha}^{(1)}, \ldots, \boldsymbol{\alpha}^{(n)}\}$. These preference vectors partition the problem into n subproblems, where each subproblem involves finding the solution that corresponds to a specific preference vector in the set. By solving n subproblems, these methods obtain a set of n solutions.

Pareto Multi-Task Learning (PMTL) (Lin et al., 2019a), inspired by the idea of decomposition-based MOO algorithms (Zhang & Li, 2007), incorporates preference vectors as constraints on the objectives. For the i-th

subproblem with preference vector $\boldsymbol{\alpha}^{(i)}$, the objective vector $\boldsymbol{f}(\boldsymbol{\theta})$ is constrained to be closer to $\boldsymbol{\alpha}^{(i)}$ than to the other preference vectors:

$$\begin{aligned} &\min_{\boldsymbol{\theta}\in\mathcal{K}\subset\mathbb{R}^d} \boldsymbol{f}(\boldsymbol{\theta}) := [f_1(\boldsymbol{\theta}),\dots,f_m(\boldsymbol{\theta})]^\top, \\ &\text{s.t. } \; r^{(j)}(\boldsymbol{\theta}) := (\boldsymbol{\alpha}^{(j)} - \boldsymbol{\alpha}^{(i)})^\top \boldsymbol{f}(\boldsymbol{\theta}) \le 0, \quad j \in [n]. \end{aligned} \tag{25}$$

Let $\boldsymbol{R}$ be the matrix containing gradients of the active constraints, i.e, $\boldsymbol{R} = \left[\nabla r^{(j)}(\boldsymbol{\theta})\right]_{j\in\mathbb{I}(\boldsymbol{\theta})}$, with $\mathbb{I}(\boldsymbol{\theta}) = \{j \in [n] \mid r^{(j)}(\boldsymbol{\theta}) \ge -\epsilon\}$ containing indices of the active constraints. Using a method akin to MGDA (Sener & Koltun, 2018), PMTL derives a descent direction for problem (25) that optimizes all m objectives while keeping $\boldsymbol{f}(\boldsymbol{\theta})$ within the sector closer to the preference vector $\boldsymbol{\alpha}^{(i)}$. The descent direction is:

$$\boldsymbol{d} = -\boldsymbol{G}\boldsymbol{\lambda} + \boldsymbol{R}\boldsymbol{\beta}, \tag{26}$$

where $\boldsymbol{G} = [\nabla f_1(\boldsymbol{\theta}),\dots,\nabla f_m(\boldsymbol{\theta})]$ is the Jacobian matrix. $\boldsymbol{\lambda}$ and $\boldsymbol{\beta}$ are coefficients obtained by solving the following quadratic programming problem:

$$\min_{\boldsymbol{\lambda},\boldsymbol{\beta}} \|\boldsymbol{G}\boldsymbol{\lambda} + \boldsymbol{R}\boldsymbol{\beta}\|^2, \quad \text{s.t. } \; \boldsymbol{\lambda}^\top\mathbf{1} + \boldsymbol{\beta}^\top\mathbf{1} = 1,\ \lambda_i \ge 0,\ \beta_j \ge 0. \tag{27}$$

A limitation of PMTL is it can only constrain objective vectors within sectors, which lacks precise control over the position of Pareto solutions.

Exact Pareto Optimal search (EPO) (Mahapatra & Rajan, 2020; 2021) is designed to locate Pareto solutions which are aligned the objective vector exactly with given preference vectors. EPO achieves this using a uniformity function, $u_{\boldsymbol{\alpha}}(\boldsymbol{f}(\boldsymbol{\theta})) = \mathrm{KL}(\hat{\boldsymbol{f}}(\boldsymbol{\theta}) \parallel 1/m)$, where: $\hat{f}_i(\boldsymbol{\theta}) = \frac{\alpha_i f_i(\boldsymbol{\theta})}{\sum_{j=1}^m \alpha_j f_j(\boldsymbol{\theta})}$. Minimizing this function aligns $\boldsymbol{f}(\boldsymbol{\theta})$ with $\boldsymbol{\alpha}$, achieving precise alignment and Pareto optimality. Let $\boldsymbol{C} = \boldsymbol{G}^\top\boldsymbol{G}$ and $\boldsymbol{c}_i$ be the i-th column of $\boldsymbol{C}$. Let $\boldsymbol{a} = [a_1,\dots,a_m]^\top$, where $a_i = \alpha_i(\log(m\hat{f}_i(\boldsymbol{\theta})) - u_{\boldsymbol{\alpha}}(f(\boldsymbol{\theta})))$. At each iteration, EPO classifies objective indices into three sets: $\mathbb{J} = \{j \mid \boldsymbol{a}^\top\boldsymbol{c}_j > 0\}$ are the indices decreasing uniformity, $\overline{\mathbb{J}} = \{j \mid \boldsymbol{a}^\top\boldsymbol{c}_j \le 0\}$ are the indices increasing uniformity, and $\mathbb{J}^*$ are the indices with the maximum $\alpha_j f_j(\boldsymbol{\theta})$. Based on these sets, it determines $\boldsymbol{\lambda}$ for the common descent direction $\boldsymbol{d} = \boldsymbol{G}\boldsymbol{\lambda}$ by solving:

$$\max_{\boldsymbol{\lambda}\in\Delta_{m-1}} \boldsymbol{\lambda}^\top\boldsymbol{C}(\boldsymbol{a}\mathbb{1}_{u_{\boldsymbol{\alpha}}} + \mathbf{1}(1-\mathbb{1}_{u_{\boldsymbol{\alpha}}})), \quad \text{s.t.} \begin{cases} \boldsymbol{\lambda}^\top\boldsymbol{c}_j \ge \boldsymbol{a}^\top\boldsymbol{c}_j\mathbb{1}_{\mathbb{J}}, & \forall j \in \overline{\mathbb{J}}\setminus\mathbb{J}^*, \\ \boldsymbol{\lambda}^\top\boldsymbol{c}_j \ge 0, & \forall j \in \mathbb{J}^*. \end{cases} \tag{28}$$

This updating direction can balance objectives while reducing non-uniformity, resulting in solutions that exactly match each preference vector. However, EPO has three drawbacks: (1) objective vectors must be non-negative, (2) unnecessary complexity from dividing sets into three subsets, and (3) computational inefficiency due to repeated Jacobian calculations and solving linear programming problem.

To address these drawbacks, Weighted Chebyshev MGDA (WC-MGDA) (Momma et al., 2022) considers the dual form of Tchebycheff scalarization (problem (4)) of a linear programming problem. Optimizing Tchebycheff function addresses the previously mentioned drawbacks of EPO. However, its convergence rate is relatively slow. Two methods for improving the Tchebycheff scalarization include smooth Tchebycheff scalarization, which replaces the non-smooth max operation with a smooth approximation, and Preference-based MGDA (PMGDA), which only requires the update direction to have a negative inner product with the exact constraint gradient.

FERERO (Chen et al., 2024a) captures preference information within the MGDA framework, addressing both objective constraints with constants and the exactness constraint. It achieves fast convergence rates of $\mathcal{O}(\epsilon^{-1})$ for deterministic gradients and $\mathcal{O}(\epsilon^{-2})$ for stochastic gradients, where ϵ is the error tolerance.

The methods discussed above rely on a fixed set of preference vectors. However, in real-world applications where the shape of the Pareto front is unknown, a predefined set of preference vectors may not always result in well-distributed solutions. To overcome this limitation, GMOOAR (Chen & Kwok, 2022) formulates the problem as a bi-level optimization problem. In the upper level, preference vectors are optimized to maximize either the hypervolume or uniformity of the solutions. In the lower level, solutions are optimized based on these preference vectors. This approach enables the algorithm to dynamically adjust preference vectors

based on the given optimization problem, ensuring the desired solution distribution. UMOD (Zhang et al., 2024a) introduces an approach that aims to maximize the minimum distance between solutions:

$$\max_{\boldsymbol{\theta}^{(i)},\boldsymbol{\theta}^{(j)}\in \mathrm{PS}} \min_{1\le i<j\le n} \rho(\boldsymbol{f}(\boldsymbol{\theta}^{(i)}), \boldsymbol{f}(\boldsymbol{\theta}^{(j)})), \tag{29}$$

where $\rho(\cdot,\cdot)$ denotes the Euclidean distance. UMOD achieves desirable solution distributions with theoretical guarantees: (1) For bi-objective problems with a connected, compact PF, the optimal solution includes the PF endpoints, with equal distances between neighboring objective vectors; (2) As the number of solutions increases, the objective vectors asymptotically distribute on the PF.

4.2 Methods without Using Preference Vectors

Unlike methods that rely on preference vectors, another approach directly optimizes for a diverse set of Pareto-optimal solutions. As HV evaluates solution sets based on both convergence and diversity, it is commonly used in traditional evolutionary MOO (Shang et al., 2020). In the context of gradient-based MOO, gradient-based hypervolume maximization algorithms (GradHV) (Wang et al., 2017; Deist et al., 2021; 2020; Emmerich et al., 2007) have been developed. Let $\{\boldsymbol{\theta}^{(1)},\dots,\boldsymbol{\theta}^{(n)}\}$ be a set of solutions and $\{\boldsymbol{f}(\boldsymbol{\theta}^{(1)}),\dots,\boldsymbol{f}(\boldsymbol{\theta}^{(n)})\}$ be their corresponding objective vectors. These algorithms first calculate the hypervolume gradient of the i-th solution $\boldsymbol{\theta}^{(i)}$ as follows:

$$\boldsymbol{d}^{(i)} = \sum_{k=1}^{m} \underbrace{\frac{\partial \mathrm{HV}_{\boldsymbol{r}}(\{\boldsymbol{f}(\boldsymbol{\theta}^{(1)}),\dots,\boldsymbol{f}(\boldsymbol{\theta}^{(n)})\})}{\partial f_k(\boldsymbol{\theta}^{(i)})}}_{a_i:1\times 1} \underbrace{\frac{\partial f_k(\boldsymbol{\theta}^{(i)})}{\partial \boldsymbol{\theta}^{(i)}}}_{\boldsymbol{B}:1\times d}. \tag{30}$$

Once $\boldsymbol{d}^{(i)}$ is calculated, solutions $\boldsymbol{\theta}^{(i)}$'s are updated by gradient ascent, $\boldsymbol{\theta}^{(i)} \leftarrow \boldsymbol{\theta}^{(i)} + \eta \boldsymbol{d}^{(i)}$, where η is a learning rate. The term $\boldsymbol{B}$ in Equation (30) can be easily estimated via backpropagation. The main challenge lies in estimating the first term a_i. As proposed by Emmerich et al. (2007), index sets are categorized based on differentiability: $\mathbb{Z}$ includes subvectors with zero partial gradients, $\mathbb{U}$ contains subvectors with undefined or indeterminate gradients, and $\mathbb{P}$ has subvectors with positive gradients. This classification leads to many zero-gradient terms, improving efficiency. Then, non-zero terms are computed using a fast dimension-sweeping algorithm (Beume et al., 2009; Guerreiro et al., 2012). This method applies to bi-, tri-, and four-objective problems with time complexities of $\Theta(mn + n\log n)$ for $m = 2$ or 3 and $\Theta(mn + n^2)$ for $m = 4$, where m is the number of objectives and n is the number of solutions. To further enhance convergence, the Newton-hypervolume method (Sosa Hernández et al., 2020) incorporates second-order information for faster updates. Deist et al. (2021) apply the hypervolume gradient methods for deep learning tasks such as medical image classification with more than thousands of parameters.

Inspired by Stein Variational Gradient Descent (SVGD) (Liu & Wang, 2016), MOO-SVGD (Liu et al., 2021d) proposes a different approach to maintain diversity while enhancing convergence. The update direction for the i-th solution $\boldsymbol{\theta}^{(i)}$ is given by:

$$\boldsymbol{d}^{(i)} = -\sum_{j=1}^{n} \boldsymbol{g}^*(\boldsymbol{\theta}^{(j)})\kappa(\boldsymbol{f}(\boldsymbol{\theta}^{(i)}), \boldsymbol{f}(\boldsymbol{\theta}^{(j)})) + \mu \nabla_{\boldsymbol{\theta}^{(i)}} \kappa(\boldsymbol{f}(\boldsymbol{\theta}^{(i)}), \boldsymbol{f}(\boldsymbol{\theta}^{(j)})). \tag{31}$$

where $\boldsymbol{g}^*(\boldsymbol{\theta}^{(j)}) = \sum_{i=1}^{m} \lambda_i^* \nabla f_i(\boldsymbol{\theta}^{(j)})$, and $\{\lambda_i^*\}_{i=1}^m$ are the objective weights obtained by MGDA (Sener & Koltun, 2018) (i.e., problem (13)). $\kappa(\cdot,\cdot)$ represents a kernel function, often chosen as the radial basis function (RBF) kernel. The first term in Equation (31) drives the particles toward the target distribution, while the second term acts as a repulsive force that promotes diversity.

While the methods discussed above address scenarios with more solutions than objectives ($n > m$), recent work has begun to explore the converse case, where objectives outnumber solutions ($m > n$) (Lin et al., 2025c; Li et al., 2025; Liu et al., 2024c; Ding et al., 2024). These approaches assign each objective to a solution $\boldsymbol{\theta} \in \{\boldsymbol{\theta}^1,\dots,\boldsymbol{\theta}^n\}$ that achieves the smallest objective value for that particular objective. The goal is then to minimize an aggregation of these smallest objective values, typically either their sum or their

maximum. The Few for Many (F4M) framework (Lin et al., 2025c; Liu et al., 2024c), for example, employs a minimax strategy by minimizing the maximum value, which is formulated as:

$$\min_{\boldsymbol{\theta}^{(1)},\dots,\boldsymbol{\theta}^{(n)}} \max_{i\in[m]} \min_{\boldsymbol{\theta}\in\{\boldsymbol{\theta}^{(1)},\dots,\boldsymbol{\theta}^{(n)}\}} f_i(\boldsymbol{\theta}). \tag{32}$$

Different from F4M, Sum of Minimum (SoM) (Ding et al., 2024) uses the sum aggregation, formulated as:

$$\min_{\boldsymbol{\theta}^{(1)},\dots,\boldsymbol{\theta}^{(n)}} \sum_{i=1}^{m} \left(\min_{\boldsymbol{\theta}\in\{\boldsymbol{\theta}^{(1)},\dots,\boldsymbol{\theta}^{(n)}\}} f_i(\boldsymbol{\theta}) \right). \tag{33}$$

MosT (Li et al., 2025) frames the problem as a bilevel optimization task, with the outer loop using MGDA to optimize solutions and the inner loop solving an optimal transport problem to assign objectives to solutions.

4.3 Discussions

This section discusses two primary approaches for obtaining multiple Pareto-optimal solutions. Preference vector-based methods decompose a MOP into multiple independent subproblems based on predefined preference vectors. These subproblems can be solved independently, which enhances memory efficiency and enables parallel computation. However, in real-world applications, determining suitable preference vectors can be challenging due to the complex and often unpredictable nature of the Pareto front. Furthermore, since the subproblems are solved independently, they cannot benefit from shared information between one another.

On the other hand, methods not using preference vectors bypass the challenge of selecting preference vectors by directly optimizing a set of solutions. These methods generate a set of Pareto-optimal solutions that inherently share interdependent information. However, this approach typically requires higher memory resources and incurs greater computational costs, making it less efficient in resource-constrained scenarios.

5 Finding an Infinite Set of Solutions

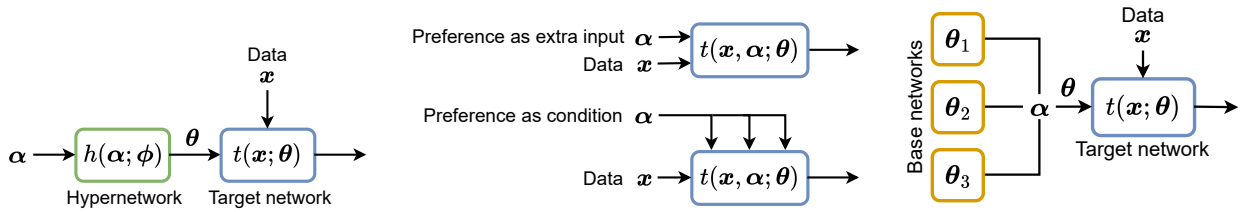
While a finite set of solutions can only provide a discrete approximation of the Pareto front, many applications require the ability to obtain solutions corresponding to any user preference, effectively representing the entire Pareto set. Directly learning an infinite number of solutions individually is impractical. Ma et al. (2020) initially investigated deriving an infinite set of solutions by approximating the Pareto set with first-order expansion around discrete Pareto-optimal solutions. However, this method has several drawbacks: (1) The approximation error grows when the solutions are widely spaced, (2) First-order approximations perform poorly in high-dimensional objective spaces, and (3) The approach requires solving a linear programming problem for updates, which can be computationally challenging. To address these issues, many methods have been introduced that leverage neural networks to learn mappings from user preferences to solutions directly, enabling the capture of the entire PS. These methods rely on designing efficient network architectures and implementing effective training strategies.

In Section 5.1, we introduce various network structures designed to capture the entire PS through preference-based mappings. Specifically, current algorithms mainly use three types of network structures: (1) Hypernetwork in Section 5.1.1, (2) Preference-Conditioned Network in Section 5.1.2, and (3) Model Combination in Section 5.1.3. An illustration of these methods is shown in Figure 6. Section 5.2 discusses training strategies for these network structures.

5.1 Network Structure

5.1.1 Hypernetwork

Hypernetwork is a neural network that generates the parameters for another target network (Ha et al., 2016). This idea has been leveraged by the Pareto Hypernetwork (PHN) (Navon et al., 2021) and Controllable Pareto



(a) Methods based on the hypernetwork.

(b) Methods based on preference-conditioned network.

(c) Methods based on model combination.

Figure 6: Illustration of different structures to learn an infinite number of solutions.

Multi-Task Learning (CPMTL) (Lin et al., 2020) to learn the entire Pareto set, where the input is the user preference $\boldsymbol{\alpha}$ and the output is the target network's parameters. The hypernetwork usually consists of several MLP layers, and has been adopted in many subsequent approaches such as PHN-HVI (Hoang et al., 2023) and SUHNPF (Gupta et al., 2022). Recently, Tuan et al. (2024) propose using a transformer architecture as the hypernetwork, demonstrating superior performance compared to MLP-based hypernetwork models.

A primary limitation of hypernetwork-based algorithms is their sizes. Since the output dimension matches the number of parameters in the target network, hypernetworks are often much larger than the base networks, limiting their applicability to large models. Some methods address this limitation by employing hypernetworks with chunking (Navon et al., 2021; Lin et al., 2020). Chunking involves partitioning the parameter space into smaller, more manageable segments, enabling the hypernetwork to generate parameters more efficiently and scalability. This approach reduces the overall size of the hypernetwork while preserving its capability to produce accurate parameters for the target network.

5.1.2 Preference-Conditioned Network

Instead of using a hypernetwork to generate weights, the original model can be directly modified to incorporate preferences. COSMOS (Ruchte & Grabocka, 2021) suggests adding preferences as an additional input to the model by combining user preference $\boldsymbol{\alpha}$ with input data $\boldsymbol{x}$, thereby increasing the input dimension of the original model.

However, such input-based conditioning has limited ability to produce diverse solutions. Some studies (Dosovitskiy & Djolonga, 2020; Chen & Kwok, 2022; Raychaudhuri et al., 2022) employ the Feature-wise Linear Modulation (FiLM) layer (Perez et al., 2018) to condition the network. The FiLM layer works by applying an affine transformation to the feature maps. Specifically, given a feature map $\boldsymbol{u}$ with C channels, we use an MLP to generate conditioning parameters $\boldsymbol{\gamma} \in \mathbb{R}^C$ and $\boldsymbol{\beta} \in \mathbb{R}^C$ based on the given preference $\boldsymbol{\alpha}$. Then, the FiLM layer modifies the features as $\boldsymbol{u}'_c = \gamma_c \cdot \boldsymbol{u}_c + \beta_c$, where $\boldsymbol{u}_c$ is the c-th channel of $\boldsymbol{u}$. This method achieves better conditioning ability compared to merely adding the preference to the input layer. Raychaudhuri et al. (2022) also propose leveraging another network to predict the network architecture. It dynamically adapts the model's structure according to user preferences, allowing for more flexible and efficient learning.

5.1.3 Model Combination

Methods based on model combination construct a composite model by integrating multiple individual models, thereby offering an effective way to introduce diversity. PaMaL (Dimitriadis et al., 2023) achieves this by learning several base models and combining them through a weighted sum of their parameters based on user preferences:

$$\boldsymbol{\theta}(\boldsymbol{\alpha}) = \sum_{i=1}^{m} \alpha_i \boldsymbol{\theta}_i. \tag{34}$$

Although this incorporates more parameters than methods based on preference-conditioned networks, PaMaL enhances diversity and provides users with a broader range of choices.

Despite its benefits, learning multiple models simultaneously may become inefficient when handling a large number of objectives. To mitigate this issue, LORPMAN (Chen & Kwok, 2024a) proposes learning a full-rank model $\boldsymbol{\theta}_0 \in \mathbb{R}^{p\times q}$ and m low-rank models $\boldsymbol{B}_i\boldsymbol{A}_i$'s, where $\boldsymbol{B}_i \in \mathbb{R}^{p\times r}$ and $\boldsymbol{A}_i \in \mathbb{R}^{r\times q}$. Given user preference $\boldsymbol{\alpha}$, the composite model is given by:

$$\boldsymbol{\theta}(\boldsymbol{\alpha}) = \boldsymbol{\theta}_0 + s\sum_{i=1}^{m} \alpha_i \boldsymbol{B}_i\boldsymbol{A}_i, \tag{35}$$

where s is a scaling factor that adjusts the impact of the low-rank components. This approach efficiently scales Pareto set learning to manage up to 40 objectives, achieving superior performance. A similar approach is also mentioned in (Dimitriadis et al., 2025). To further enhance parameter efficiency, Chen & Kwok (2024b) introduce a preference-aware tensor multiplication:

$$\boldsymbol{\theta}(\boldsymbol{\alpha}) = \boldsymbol{\theta}_0 + s\boldsymbol{C} \times_1 \boldsymbol{A} \times_2 \boldsymbol{B} \times_3 \boldsymbol{\alpha}, \tag{36}$$

where $\boldsymbol{C} \in \mathbb{R}^{r\times r\times m}$ is a learnable core tensor, $\boldsymbol{A} \in \mathbb{R}^{r\times p}$, $\boldsymbol{B} \in \mathbb{R}^{r\times q}$ are learnable matrices, and $\times_u$ denotes mode-u tensor multiplication (Kolda & Bader, 2009). Zhong et al. (2024) address scalability challenges for large-scale models, such as large language models, using SVD-LoRA:

$$\boldsymbol{\theta}(\boldsymbol{\alpha}) = \boldsymbol{\theta}_0 + \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}, \tag{37}$$

where $\boldsymbol{\Sigma}$ is a diagonal matrix defined as $\mathrm{diag}(\sigma_1, \ldots, \sigma_r, s\alpha_1, \ldots, s\alpha_m)$, $\{\sigma_i\}_{i=1}^r$ and s are learnable scalars. $\boldsymbol{U} \in \mathbb{R}^{p\times(r+m)}$ and $\boldsymbol{V} \in \mathbb{R}^{(r+m)\times q}$ are learnable matrices.

Instead of training new models, Tang et al. (2024) propose to utilize a mixture of experts (Cai et al., 2024) to combine pre-existing models in order to achieve the Pareto set. Given n models, $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_n$, each trained on different datasets, they are first integrated into a unified base model $\boldsymbol{\theta}_0$. For some modules in the network, differences between the individual models and the base model (i.e., $\boldsymbol{\theta}_i - \boldsymbol{\theta}_0$'s) are maintained. These differences are treated as experts within the framework. Then, a mixture of experts is applied, where a gating network conditioned on user preferences determines the weighting of these experts. It enables the generation of diverse models by applying different weighted combinations of experts to align with specific user preferences.

5.2 Training Strategy

This section outlines the training approach for the parameters (denoted $\boldsymbol{\phi}$) of the network structures introduced in Section 5.1. Specifically, $\boldsymbol{\phi}$ refers to the parameters of the hypernetwork in Section 5.1.1, parameters of the preference-conditioned network in Section 5.1.2, or parameters of the base models or low-rank models in Section 5.1.3. Current algorithms follow a similar approach: sample user preferences and optimize the structure parameters to generate networks that align with those preferences. The training objective can be written as:

$$\min_{\boldsymbol{\phi}} \mathbb{E}_{\boldsymbol{\alpha}\sim\Delta_{m-1}}\, \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}} \left[\tilde{g}_{\boldsymbol{\alpha}}(\boldsymbol{\ell}(t(\boldsymbol{x};\boldsymbol{\theta}(\boldsymbol{\alpha};\boldsymbol{\phi}),\boldsymbol{y}))\right], \tag{38}$$

where $(\boldsymbol{x}, \boldsymbol{y})$'s are training data sampled from dataset $\mathcal{D}$, $\boldsymbol{\ell}(\cdot,\cdot)$ is the multi-objective loss function and $\tilde{g}_{\boldsymbol{\alpha}}(\cdot)$ is an MOO algorithm that produces a single solution given preference vector $\boldsymbol{\alpha}$. In general, most optimization algorithms discussed in Sections 2.2 and 4 can be adopted. Below, we summarize the algorithms adopted by the existing methods.

- Scalarization: As outlined in Section 2.2, scalarization combines multiple objectives into one scalar objective. Linear scalarization is common in most algorithms, such as PHN (Navon et al., 2021), COSMOS (Ruchte & Grabocka, 2021), PAMAL (Dimitriadis et al., 2023), and LORPMAN (Chen & Kwok, 2024a). Alternatives like Tchebycheff and smooth Tchebycheff scalarization (Lin et al., 2024b) can also be adopted.
- Preference-Aware Weighting Methods: As discussed in Section 4.1, these methods incorporate user preferences into optimization. PHN (Navon et al., 2021) employs the Exact Pareto Optimal (EPO) solver (Mahapatra & Rajan, 2020) to find Pareto-optimal solutions aligned with preferences. CPMTL (Lin et al., 2020) uses a constrained approach inspired by PMTL (Lin et al., 2019a).

- Hypervolume Maximization: As discussed in Section 4.2, hypervolume maximization can optimize both diversity and convergence of a solution set. PHN-HVI (Hoang et al., 2023) leverages hypervolume maximization to optimize the structural parameters $\boldsymbol{\phi}$. In each iteration, it samples a set of preference vectors and generates solutions based on these vectors. Subsequently, hypervolume maximization is applied to promote both the diversity and convergence of the solutions.

5.3 Discussions

Selecting the appropriate structural framework and optimization algorithms is important for effectively learning an infinite Pareto set. Hypernetwork-based approaches offer significant flexibility by generating a diverse set of models tailored to varying user preferences. However, their main drawback is size: the output dimension matches the target network's parameters, making hypernetworks significantly larger than base models. This limits their applicability for large-scale networks or resource-constrained scenarios.

In contrast, methods based on preference-conditioned networks are highly parameter-efficient. They introduce a minimal number of additional parameters relative to the base model, making them well-suited for large-scale applications where adding extra parameters is costly. However, their limited parameter space may reduce diversity, hindering their ability to approximate complex PS accurately.

Methods based on model combinations serve as a middle ground between hypernetwork and preference-conditioned network methods. By integrating multiple models, they offer a good approximation of the PS while maintaining scalability for large models. Furthermore, some methods allow for parameter adjustments through the selection of appropriate ranks in low-rank approximations.

Regarding training strategies, most algorithms use linear scalarization for its simplicity and strong performance. However, when aligning solutions with user preferences is important, preference-dependent weighting schemes in Section 4.1 are more suitable. Additionally, to achieve HV-optimal solutions, HV-based optimization criteria in Section 4.2 can also be employed, as they better capture the PF by focusing on the dominated objective space volume.

6 Theories

6.1 Convergence of Gradient-Balancing Methods in Section 3.2

This section reviews the convergence analysis for gradient-balancing methods in Section 3.2. This is first examined under the deterministic gradient setting (i.e., full-batch gradient). Subsequently, it is analyzed under the stochastic gradient setting. Note that in non-convex MOO, convergence refers to reaching Pareto stationary (Definition 5).

6.1.1 Deterministic Gradient

With the use of deterministic gradients, it has been shown that under mild conditions, MGDA (Sener & Koltun, 2018) can converge to a Pareto-stationary point at a convergence rate of $\mathcal{O}(K^{-1/2})$ (Fliege et al., 2019), where K is the number of iterations. This rate is comparable to that of single-objective optimization (Nesterov, 2013). The fundamental idea is to assess the reduction in each individual objective function. When the current solution is not Pareto-stationary, it can be shown that the common descent direction $\boldsymbol{d}$ identified by MGDA ensures an improvement of at least $\mathcal{O}(\|\boldsymbol{d}\|^2)$ in each objective, which mirrors the situation in single-objective optimization (Fliege et al., 2019). Therefore, the method used in single-objective optimization can be applied to prove the convergence. Furthermore, other algorithms such as CAGrad (Liu et al., 2021a), Nash-MTL (Navon et al., 2022), and PCGrad (Yu et al., 2020) also offer convergence analyses in similar approaches.

6.1.2 Stochastic Gradient

Liu & Vicente (2021) are the first to analyze the convergence of gradient-based MOO algorithms with stochastic gradients. They propose a stochastic version of MGDA (Sener & Koltun, 2018) that computes

Table 2: Convergence of existing stochastic gradient MOO algorithms. “LS" and “GS" denote the objectives are L-smooth and generalized L-smooth, respectively. “BG" and “BF" represent the bounded gradient and bounded function value assumptions, respectively. “Complexity" denotes the sample complexity achieving ϵ-accurate Pareto stationary point. “Bounded CA" denotes the bounded conflict-avoidant distance.

Method	Batch Size	Assumption	Complexity	Bounded CA
SMG (Liu & Vicente, 2021)	$\mathcal{O}(\epsilon^{-2})$	LS, BG	$\mathcal{O}(\epsilon^{-4})$	✗
CR-MOGM (Zhou et al., 2022)	$\mathcal{O}(1)$	LS, BF, BG	$\mathcal{O}(\epsilon^{-2})$	✗
MoCo (Fernando et al., 2023)	$\mathcal{O}(1)$	LS, BF, BG	$\mathcal{O}(\epsilon^{-2})$	✗
PSMGD (Xu et al., 2025a)	$\mathcal{O}(1)$	LS, BF, BG	$\mathcal{O}(\epsilon^{-2})$	✗
MoDo (Chen et al., 2023c)	$\mathcal{O}(1)$	LS, BG	$\mathcal{O}(\epsilon^{-2})$	✓
SDMGrad (Xiao et al., 2023)	$\mathcal{O}(1)$	LS, BG	$\mathcal{O}(\epsilon^{-2})$	✓
SGSMGrad (Zhang et al., 2025)	$\mathcal{O}(1)$	GS	$\mathcal{O}(\epsilon^{-2})$	✓

a common descent direction using stochastic gradients for all objectives and prove convergence to Pareto-optimal solutions under the assumption of convex objective functions. However, due to the inherent bias of common descent direction $\boldsymbol{d}$ introduced by stochastic gradient estimations, their analysis requires the use of an increasing batch size that grows linearly with the number of iterations to ensure convergence. This requirement can be impractical in real-world applications, as large batch sizes lead to increased computational costs and memory usage.

To overcome this limitation, Zhou et al. (2022) propose Correlation-Reduced Stochastic Multi-Objective Gradient Manipulation (CR-MOGM). It addresses the bias in the common descent direction by introducing a smoothing technique on weight coefficient $\boldsymbol{\lambda}$. In CR-MOGM, $\boldsymbol{\lambda}^{(k)}$ at iteration k is updated using a moving average:

$$\boldsymbol{\lambda}^{(k)} = (1-\gamma)\hat{\boldsymbol{\lambda}}^{(k)} + \gamma\boldsymbol{\lambda}^{(k-1)}, \tag{39}$$

where $\gamma \in [0,1]$ is a smoothing factor, $\hat{\boldsymbol{\lambda}}^{(k)}$ is the weight vector obtained by the MOO solver (such as MGDA) at iteration k using the current stochastic gradients, and $\boldsymbol{\lambda}^{(k-1)}$ is the smoothed weight from the previous iteration. Smoothing reduces the variance in the weight, leading to a more stable and reliable common descent direction. Xu et al. (2025a) further analyze the convergence by updating the objective weights every τ iterations instead of at each iteration, and achieve the same convergence rate $\mathcal{O}(\epsilon^{-2})$ as CR-MOGM. They also match the convergence rate in stochastic single-objective optimization (Ghadimi & Lan, 2013).

Another approach to address gradient bias is MoCo (Fernando et al., 2023), which introduces a tracking variable $\hat{\boldsymbol{g}}_i^{(k)}$ that approximates the true gradient:

$$\hat{\boldsymbol{g}}_i^{(k+1)} = \prod_{L_i}\left(\hat{\boldsymbol{g}}_i^{(k)} - \gamma\left(\hat{\boldsymbol{g}}_i^{(k)} - \boldsymbol{g}_i^{(k)}\right)\right), \tag{40}$$

where $\prod_{L_i}$ is the projection to the set $\{\boldsymbol{g} \in \mathbb{R}^d \mid \|\boldsymbol{g}\| \leq L_i\}$, and L_i is the Lipschitz constant of $f_i(\boldsymbol{\theta})$. However, the analysis requires the number of iterations K to be sufficiently large (in the order of $\mathcal{O}(m^{10})$). This high dependency on m makes MoCo less practical for problems involving many objectives. Additionally, the convergence analysis of CR-MOGM and MoCo relies on the assumption that the objective functions have bounded values.

To address these limitations, Chen et al. (2023c) propose the Multi-objective gradient with double sampling algorithm (MoDo). MoDo mitigates bias in stochastic gradient-based MOO methods without assuming bounded function values. It introduces a double sampling technique to obtain unbiased estimates of the gradient products needed for updating the weight coefficients. Specifically, at each iteration, MoDo updates the weight vector $\boldsymbol{\lambda}^{(k)}$ using gradients computed on two independent mini-batches:

$$\boldsymbol{\lambda}^{(k+1)} = \prod_{\Delta_{m-1}}\left(\boldsymbol{\lambda}^{(k)} - \eta\boldsymbol{G}^{(k)}(\boldsymbol{z}_1^{(k)})^\top\boldsymbol{G}^{(k)}(\boldsymbol{z}_2^{(k)})\boldsymbol{\lambda}^{(k)}\right), \tag{41}$$

where η is the step size, $\prod_{\Delta_{m-1}}$ is the projection onto the simplex Δ_{m-1}, and $\boldsymbol{G}^{(k)}(\boldsymbol{z}_1^{(k)})$, $\boldsymbol{G}^{(k)}(\boldsymbol{z}_2^{(k)})$ are stochastic gradients evaluated on two independent mini-batches $\boldsymbol{z}_1^{(k)}$ and $\boldsymbol{z}_2^{(k)}$. By reducing the bias in estimating the common descent direction, MoDo achieves convergence guarantees without the bounded function value assumption. Additionally, it guarantees a bounded conflict-avoidant (CA) distance, which is the distance between the estimated update direction and the CA direction defined in problem (12).

Similarly, Xiao et al. (2023) propose the Stochastic Direction-oriented Multi-objective Gradient descent (SDMGrad) algorithm, which introduces a new direction-oriented multi-objective formulation by regularizing the common descent direction within a neighborhood of a target direction (such as the average gradient of all objectives). In SDMGrad, the weight vector $\boldsymbol{\lambda}^{(k)}$ is updated as:

$$\boldsymbol{\lambda}^{(k+1)} = \prod_{\Delta_{m-1}} \left(\boldsymbol{\lambda}^{(k)} - \eta \left[\boldsymbol{G}^{(k)}(\boldsymbol{z}_1^{(k)})^\top \left(\boldsymbol{G}^{(k)}(\boldsymbol{z}_2^{(k)}) \boldsymbol{\lambda}^{(k)} + \gamma \boldsymbol{g}_0(\boldsymbol{z}_2^{(k)}) \right) \right] \right), \tag{42}$$

where η is the step size, γ is a regularization factor controlling the proximity to the target direction $\boldsymbol{g}_0(\boldsymbol{z}_2^{(k)})$. By incorporating this regularization, SDMGrad effectively balances optimization across objectives while guiding the overall descent direction. Zhang et al. (2025) further consider the convergence under generalized L-smoothness (Li et al., 2023) and without bounded gradient assumption. A summary of the convergence results of existing stochastic gradient MOO algorithms is provided in Table 2.

6.2 Generalization

The generalization aspect of multi-objective deep learning remains relatively underexplored compared to its convergence properties. Cortes et al. (2020) analyze the generalization behavior of a specific scalarization approach that minimizes over convex combinations. Súkeník & Lampert (2024) consider a broader class of scalarization methods. They show that the generalization bounds for individual objectives extend to MOO with scalarization. Both studies rely on the Rademacher complexity of the hypothesis class to establish algorithm-independent generalization bounds, which are unaffected by the training process.

Another line of research investigates Tchebycheff scalarization, which focuses on the sample complexity required to achieve a generalization error within ϵ of the optimal objective value. Formally, given a set of m distributions $\{\mathcal{D}_i\}_{i=1}^m$ and a hypothesis class $\mathcal{H}$, the goal is to find a (possibly randomized) hypothesis h such that

$$\max_{i\in[m]} \ell_{\mathcal{D}_i}(h) \leq \min_{h^*\in\mathcal{H}} \max_{i\in[m]} \ell_{\mathcal{D}_i}(h^*) + \epsilon, \tag{43}$$

where $\ell_{\mathcal{D}_i}(h)$ is the loss of hypothesis h on $\mathcal{D}_i$, h^* is the optimal hypothesis and ϵ is the error tolerance. It is first formulated by Awasthi et al. (2023) as an open problem in 2023 and has since been addressed by several works (Peng, 2024; Zhang et al., 2024d). Haghtalab et al. (2022) are the first to show the sample complexity lower bound of $\widetilde{\Omega}\left(\frac{v+m}{\epsilon^2}\right)$, where $v = \mathrm{VCdim}(\mathcal{H})$ is the VC-dimension of the hypothesis class $\mathcal{H}$. By using a boosting framework, Peng (2024) gives an algorithm that achieves a sample complexity upper bound of $\widetilde{O}\left(\frac{v+m}{\epsilon^2} \cdot \left(\frac{m}{\epsilon}\right)^{o(1)}\right)$, which is nearly optimal. In a concurrent work, Zhang et al. (2024d) provide a variant of the hedge algorithm under an Empirical Risk Minimization (ERM) oracle access that achieves the optimal sample complexity bound of $\widetilde{O}\left(\frac{v+m}{\epsilon^2}\right)$.

In the online setting, Liu et al. (2024b) provide an adaptive online mirror descent algorithm that achieves $\mathcal{O}\left(\frac{mv}{\sqrt{K}}\right)$ regret, where K is the number of iterations. Using a plain online-to-batch conversion scheme, this algorithm leads to an $\mathcal{O}\left(\frac{m^2v^2}{\epsilon^2}\right)$ sample complexity, which still lags behind the optimal offline sample complexity of $\widetilde{O}\left(\frac{m+v}{\epsilon^2}\right)$ (Zhang et al., 2024d). It is interesting to see whether it is possible to use an online learner with proper online-to-batch conversion schemes that is able to match the optimal offline sample complexity.

MoDo (Chen et al., 2023c) introduces a different perspective by leveraging algorithm stability to derive algorithm-dependent generalization error bounds of gradient balancing algorithms.

7 Applications

In this section, we review key application domains where gradient-based MOO has been successfully applied, spanning computer vision, neural architecture search, recommendation systems, large language models, and other emerging areas.

7.1 Computer Vision

The most representative application of MOO in computer vision is multi-task dense prediction (Vandenhende et al., 2021), which aims to train a model for simultaneously dealing with multiple dense prediction tasks (such as semantic segmentation, monocular depth estimation, and surface normal estimation) and has been successfully applied in autonomous driving (Ishihara et al., 2021; Liang et al., 2023). Since the encoder in computer vision usually contains large number of parameters, sharing the encoder across different tasks can significantly reduce the computational cost but may cause conflicts among tasks, leading to a performance drop in some of the tasks. Hence, many methods are proposed to address the issue from the perspective of MOO, which use scalarization to balance multiple losses (Xu et al., 2022; Ye & Xu, 2022; Lin et al., 2024a; 2025a; Li et al., 2024a). A comprehensive survey on multi-task dense prediction is in (Vandenhende et al., 2021). Besides the dense prediction task, MOO is also useful in other computer vision tasks such as point cloud (Xie et al., 2023; 2024; Wang et al., 2024b), medical image denoising (Kyung et al., 2024), and pose estimation (Ye et al., 2024a).

7.2 Neural Architecture Search

Neural Architecture Search (NAS), which aims to design the architecture of neural networks automatically, has gained significant interest recently (Ren et al., 2021). Due to the complex application scenarios in the real world, recent works consider multiple objectives beyond just accuracy. For example, to search an efficient architecture for deployment on platforms with limited resources, many studies incorporate resource-constraint objectives such as the parameter size, FLOPs, and latency. These works are mainly based on gradient-based NAS methods (like DARTS (Liu et al., 2019a)) and formulate it as a multi-objective optimization problem. Among them, (Wu et al., 2019; Cai et al., 2019; Wu et al., 2021; Wang et al., 2021a; Yue et al., 2022) employ scalarization to identify a single solution, where the task and efficiency objectives are combined with fixed weights. However, altering the weights of the objectives needs a complete rerun of the search, which is computationally intensive. Therefore, Sukthanker et al. (2025) propose to learn a mapping from a preference vector to an architecture using a hypernetwork, enabling to provide the entire PS without the need for a new search.

7.3 Recommendation Systems

Beyond just focusing on accuracy, recommendation systems often incorporate additional quality metrics such as novelty, diversity, serendipity, popularity, and others (Zheng & Wang, 2022). These factors naturally frame the problem as a MOO problem. Traditionally, many methods have employed scalarization techniques to assign weights to the different objectives (Di Noia et al., 2017; Patil et al., 2021; Lacerda, 2017). In recent developments, gradient-balancing approaches are applied to address multi-objective optimization within recommendation systems to learn a single recommendation model (Milojkovic et al., 2019; Lin et al., 2019b; Mitrevski et al., 2020; Yang et al., 2024a). Additionally, some research efforts aim to identify finite Pareto sets (Xie et al., 2021) and infinite Pareto sets (Ge et al., 2022; Wilm et al., 2024), enabling the provision of personalized recommender models that cater to varying user preferences.

7.4 Large Language Models (LLMs)

Recently, there has been a growing trend of incorporating multi-objective optimization into the training of large language models (LLMs), including multi-task fine-tuning and multi-objective alignment.

Due to the powerful transferability of LLMs, users can fine-tune LLMs to specific downstream tasks or scenarios. This approach separates fine-tuning on each task, causing extensive costs in training and difficulties

Table 3: Summary of benchmark datasets in multi-objective deep learning.

Dataset	Description	#Objectives	#Samples
NYUv2 (Silberman et al., 2012)	indoor scene understanding	3	1,449
Taskonomy (Zamir et al., 2018)	indoor scene understanding	26	≈ 4M
Cityscapes (Cordts et al., 2016)	urban scene understanding	2	3,475
QM9 (Ramakrishnan et al., 2014)	molecular property prediction	11	130K
Office-31 (Saenko et al., 2010)	image classification	3	4,110
Office-Home (Venkateswara et al., 2017)	image classification	4	15,500
CelebA (Liu et al., 2015)	image classification	40	≈ 202K
CIFAR-100 (Krizhevsky & Hinton, 2009)	image classification	20	60K
XTREME (Hu et al., 2020)	multilingual learning	9	≈ 597K
PKU-SafeRLHF (Ji et al., 2024)	two-dimensional preference data	2	≈ 82K
UltraFeedBack (Cui et al., 2023)	multi-dimensional preference data	4	≈ 64K
HelpSteer2 (Wang et al., 2024c)	multi-attributes preference data	5	≈ 21K

in deployment. MFTCoder (Liu et al., 2024a) proposes a multi-task fine-tuning framework that enhances the coding capabilities of LLMs by addressing data imbalance, varying difficulty levels, and inconsistent convergence speeds across tasks. CoBa (Gong et al., 2024) studies multi-task fine-tuning for LLMs and balances task convergence with minimal computational overhead.

Aligning with multi-dimensional human preferences (such as helpfulness, harmfulness, humor, and conciseness) is essential for customizing responses to users' needs, as users typically have diverse preferences for different aspects. MORLHF (Wu et al., 2023) and MODPO (Zhou et al., 2024a) train an LLM for every preference configuration by linearly combining multiple (implicit) reward models. To avoid retraining, some methods train multiple LLMs separately for each preference dimension and merge their parameters (Rame et al., 2023; Jang et al., 2023) or output logits (Shi et al., 2024) to deal with different preference requirements. However, these methods need to train and store multiple LLMs, leading to huge computational and storage costs. To improve efficiency, several preference-aware methods are proposed to fine-tune a single LLM for varying preferences by incorporating the relevant coefficients into the input prompts (Wang et al., 2024a; Guo et al., 2024; Yang et al., 2024b) or model parameters (Zhong et al., 2024; Lin et al., 2025b).

7.5 Miscellaneous

In addition to the applications mentioned above, multi-objective optimization has been applied to a variety of other deep learning scenarios, including meta-learning (Yu et al., 2023; Wang et al., 2021b), federated learning (Hu et al., 2022; Askin et al., 2024; Kang et al., 2024), long-tailed learning (Li et al., 2024b; Zhao et al., 2024; Zhou et al., 2024b), continual learning (Lai et al., 2025), diffusion models (Hang et al., 2023; Go et al., 2023; Xu et al., 2025b; Yao et al., 2024), neural combinatorial optimization (Li et al., 2020; Lin et al., 2022b; Wang et al., 2023b; Chen et al., 2023b;a), GFlowNets (Jain et al., 2023; Zhu et al., 2023), multi-agent systems (Rădulescu et al., 2020), LLM pruning (Chen et al., 2025) and physics informed neural networks (PINNs) (Hwang & Lim, 2024; Liu et al., 2025). These applications highlight the versatility of MOO in addressing diverse challenges within deep learning.

8 Resources

In this section, we introduce some benchmark datasets and open-source libraries for gradient-based MOO in deep learning.

8.1 Datasets

Below, we provide details on the benchmark datasets, which are summarized in Table 3:

- **NYUv2** dataset (Silberman et al., 2012) is for indoor scene understanding. It has three tasks (i.e., semantic segmentation, depth estimation, and surface normal prediction) with 795 training and 654 testing samples.
- **Taskonomy** dataset (Zamir et al., 2018) is obtained from 3D scans of about 600 buildings and contains 26 tasks (such as semantic segmentation, depth estimation, surface normal prediction, keypoint detection, and edge detection) with about 4 million samples.
- **Cityscapes** dataset (Cordts et al., 2016) is for urban scene understanding. It has two tasks (i.e., semantic segmentation and depth estimation) with $2,975$ training and 500 testing samples.
- **QM9** dataset (Ramakrishnan et al., 2014) is for molecular property prediction with 11 tasks. Each task performs regression on one property. It contains $130,000$ samples.
- **Office-31** dataset (Saenko et al., 2010) contains $4,110$ images from three domains (tasks): Amazon, DSLR, and Webcam. Each task has 31 classes.
- **Office-Home** dataset (Venkateswara et al., 2017) contains $15,500$ images from four domains (tasks): artistic images, clipart, product images, and real-world images. Each task has 65 object categories collected under office and home settings.
- **CelebA** (Liu et al., 2015) is a large-scale face attribute dataset comprising $202,599$ face images. Each image is annotated with 40 binary attributes, resulting in 40 distinct binary classification tasks.
- **CIFAR-100** (Krizhevsky & Hinton, 2009) is an image classification dataset with 100 classes, containing 50K training and 10K testing images. These 100 classes are divided into 20 tasks, each of which is a 5-class image classification problem.
- **XTREME** benchmark (Hu et al., 2020) aims to evaluate the cross-lingual generalization abilities of multilingual representations. It contains 9 tasks in 40 languages, including 2 classification tasks, 2 structure prediction tasks, 3 question-answering tasks, and 2 sentence retrieval tasks.
- **PKU-SafeRLHF** dataset (Ji et al., 2024) focuses on safety alignment in LLMs, containing $82,118$ question-answering pairs with the annotations of helpfulness and harmlessness.
- **UltraFeedBack** dataset (Cui et al., 2023) contains $63,967$ prompts, each corresponding to four responses from different LLMs. Each response has 4 aspects of annotations, namely instruction-following, truthfulness, honesty, and helpfulness, generated by GPT-4.
- **HelpSteer2** dataset (Wang et al., 2024c) contains $21,362$ samples. Each sample contains a prompt and a response with 5 human-annotated attributes (i.e., helpfulness, correctness, coherence, complexity, and verbosity), each ranging between 0 and 4 where higher means better for each attribute.

8.2 Libraries

Two popular libraries, LibMTL[1] (Lin & Zhang, 2023) and LibMOON[2] (Zhang et al., 2024c), provide unified environments for implementing and fairly evaluating MOO methods. Both are built on PyTorch (Paszke et al., 2019) and feature modular designs, enabling flexible development of new methods or application of existing ones to new scenarios. LibMTL (Lin & Zhang, 2023) mainly focuses on finding a single Pareto-optimal solution and supports 24 methods introduced in Section 3 and 6 benchmark datasets. LibMOON (Zhang et al., 2024c) mainly focuses on exploring the whole Pareto set. It includes over 20 methods for obtaining a finite set of solutions (introduced in Section 4) or infinite set of solutions (introduced in Section 5).

[1] `https://github.com/median-research-group/LibMTL`
[2] `https://github.com/xzhang2523/libmoon`

9 Challenges and Future Directions

Despite significant progress in applying MOO in deep learning, both in learning a single model and in learning a Pareto set of models, several challenges remain.

Theoretical Understanding. While practical methods for MOO in deep learning have seen significant progress, their theoretical foundations remain relatively underexplored. Most existing research focuses on analyzing the convergence of MOO algorithms to stationary points, while generalization error, which is critical for evaluating performance on unseen data, has received less attention. For instance, Chen et al. (2023c), as discussed in Section 6.2, provide the algorithm-dependent generalization analysis. Extending this to a broader range of algorithms could offer deeper insights into how different techniques affect generalization. For Pareto set learning algorithms, there is limited theoretical understanding of how effectively existing algorithms can approximate Pareto sets. Zhang et al. (2023b) established a generalization bound for HV-based Pareto set learning. However, the effect of various network architectures (Section 5.1) on approximating Pareto sets remain unclear. Additionally, it is uncertain how effective current training strategies (Section 5.2) are in approximating the Pareto set.

Reducing Gradient Balancing Costs. While gradient balancing methods in Section 3.2 often show strong performance in various applications, they come with significant computational overhead. Despite the introduction of some practical speedup strategies (discussed in Section 3.2.3), existing approaches remain insufficiently efficient. A deeper understanding of the optimization differences between gradient balancing, linear scalarization, and loss balancing is crucial. This insight could facilitate the integration of gradient balancing with linear scalarization and loss balancing, reducing computational overhead significantly and enabling its application in large-scale training scenarios.

Dealing with Large Number of Objectives. Some real-world problems involve handling a large number of objectives, which poses significant challenges for current Pareto set learning algorithms in Section 5. As the number of objectives grows, the preference vector space expands exponentially, making it challenging for existing random sampling-based techniques to effectively learn the mapping between preference vectors and solutions. Replacing random sampling with systematic preference exploration, such as methods used in MORL (Röpke et al., 2025), can enable more efficient coverage of the solution set. Additionally, exploring methods to reduce or merge objectives based on their properties can be a promising approach to minimize the total number of objectives. Finally, adopting a utility-based approach (Roijers et al., 2013; Hayes et al., 2022) is promising; by utilizing priors on user utility, learning can be restricted to the significantly smaller subset.

Distributed Training. Current MOO algorithms in deep learning are limited to single GPUs or machines, but scaling them to multi-GPU and distributed environments is increasingly important as models and datasets grow. This scaling introduces unique challenges compared to single-objective optimization. Gradient-balancing algorithms require efficient gradient distribution and communication across GPUs. Learning a set of Pareto-optimal solutions involves synchronizing multiple models across nodes with minimal overhead. Additionally, when data for different objectives is distributed across devices and cannot be shared (e.g., due to privacy constraints), collaborative computation of solutions or Pareto sets without direct data sharing becomes an important challenge.

Advancements in Large Language Models (LLMs). As discussed in Section 7.4, current research in MOO for LLMs largely focuses on the RLHF stage. Expanding MOO techniques to other stages in the LLM lifecycle, such as pre-training and inference, is a valuable direction for future research. Addressing these challenges can lead to models that are better aligned with user needs across their entire development processes. Additionally, user preferences are currently modeled as a preference vector. However, this representation may oversimplify more complex user preferences on LLMs. Exploring advanced methods to capture and represent these intricate preferences can enhance the effectiveness and personalization of LLMs.

Application in More Deep Learning Scenarios. While MOO has already been utilized in various deep learning scenarios, as highlighted in Section 7, there remain numerous unexplored areas within this field. Multi-objective characteristics are inherently present in most deep learning problems, as models are typically developed or evaluated based on multiple criteria. Consequently, trade-offs often arise naturally. Leveraging MOO methods to effectively address these trade-offs presents an opportunity for further exploration.

10 Conclusion

In this paper, we provide the first comprehensive review of gradient-based multi-objective deep learning, a field of growing importance as models are required to balance multiple, often conflicting, objectives. We have systematically surveyed the spectrum of algorithms, covering methods for finding a single balanced model, a finite set of models, and an entire infinite Pareto set of models.

The review also delves into the theoretical foundations of these methods, summarizing key results in convergence while highlighting the need for a deeper understanding of generalization. The practical significance of MOO is demonstrated across diverse applications, including its emerging role in Large Language Models (LLMs). By unifying these approaches and identifying key challenges, this paper serves as a foundational resource to guide and inspire future advancements in this critical and rapidly evolving domain.

References

Baris Askin, Pranay Sharma, Gauri Joshi, and Carlee Joe-Wong. Federated communication-efficient multi-objective optimization. *arXiv preprint arXiv:2410.16398*, 2024.

Pranjal Awasthi, Nika Haghtalab, and Eric Zhao. Open problem: The sample complexity of multi-distribution learning for vc classes. In *Annual Conference on Learning Theory*, 2023.

Hao Ban and Kaiyi Ji. Fair resource allocation in multi-task learning. In *International Conference on Machine Learning*, 2024.

Nicola Beume, Carlos M Fonseca, Manuel Lopez-Ibanez, Luis Paquete, and Jan Vahrenhold. On the complexity of computing the hypervolume indicator. *IEEE Transactions on Evolutionary Computation*, 13(5): 1075–1082, 2009.

V Joseph Bowman Jr. On the relationship of the tchebycheff norm and the efficient frontier of multiple-criteria objectives. In *Multiple Criteria Decision Making: Proceedings of a Conference Jouy-en-Josas*, 1976.

Han Cai, Ligeng Zhu, and Song Han. ProxylessNAS: Direct neural architecture search on target task and hardware. In *International Conference on Learning Representations*, 2019.

Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. A survey on mixture of experts. *arXiv preprint arXiv:2407.06204*, 2024.

Jinbiao Chen, Jiahai Wang, Zizhen Zhang, Zhiguang Cao, Te Ye, and Siyuan Chen. Efficient meta neural heuristic for multi-objective combinatorial optimization. In *Conference on Neural Information Processing Systems*, 2023a.

Jinbiao Chen, Zizhen Zhang, Zhiguang Cao, Yaoxin Wu, Yining Ma, Te Ye, and Jiahai Wang. Neural multi-objective combinatorial optimization with diversity enhancement. In *Conference on Neural Information Processing Systems*, 2023b.

Lisha Chen, Heshan Fernando, Yiming Ying, and Tianyi Chen. Three-way trade-off in multi-objective learning: Optimization, generalization and conflict-avoidance. In *Conference on Neural Information Processing Systems*, 2023c.

Lisha Chen, AFM Saif, Yanning Shen, and Tianyi Chen. FERERO: A flexible framework for preference-guided multi-objective learning. In *Conference on Neural Information Processing Systems*, 2024a.

Shijie Chen, Yu Zhang, and Qiang Yang. Multi-task learning in natural language processing: An overview. *ACM Computing Surveys*, 56(12):1–32, 2024b.

Weiyu Chen and James Kwok. Multi-objective deep learning with adaptive reference vectors. In *Conference on Neural Information Processing Systems*, 2022.

Weiyu Chen and James Kwok. Efficient Pareto manifold learning with low-rank structure. In *International Conference on Machine Learning*, 2024a.

Weiyu Chen and James Kwok. You only merge once: Learning the Pareto set of preference-aware model merging. *arXiv preprint arXiv:2408.12105*, 2024b.

Weiyu Chen, Hansi Yang, Yunhao GOU, Han Shi, En-Liang Hu, Zhenguo Li, and James Kwok. Multi-objective one-shot pruning for large language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International Conference on Machine Learning*, 2018.

Zhao Chen, Jiquan Ngiam, Yanping Huang, Thang Luong, Henrik Kretzschmar, Yuning Chai, and Dragomir Anguelov. Just pick a sign: Optimizing deep multitask models with gradient sign dropout. In *Conference on Neural Information Processing Systems*, 2020.

Ran Cheng, Yaochu Jin, Markus Olhofer, et al. Test problems for large-scale multiobjective and many-objective optimization. *IEEE Transactions on Cybernetics*, 47(12):4108–4121, 2016.

Eng Ung Choo and Derek R Atkins. Proper efficiency in nonconvex multicriteria programming. *Mathematics of Operations Research*, 8(3):467–470, 1983.

Carlos A Coello Coello, Gregorio Toscano Pulido, and Maximino Salazar Lechuga. Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 8(3):256–279, 2004.

Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016.

Corinna Cortes, Mehryar Mohri, Javier Gonzalvo, and Dmitry Storcheus. Agnostic learning with multiple objectives. In *Conference on Neural Information Processing Systems*, 2020.

Michael Crawshaw. Multi-task learning with deep neural networks: A survey. *arXiv preprint arXiv:2009.09796*, 2020.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. Ultrafeedback: Boosting language models with high-quality feedback. *arXiv preprint arXiv:2310.01377*, 2023.

Kalyanmoy Deb. Multi-objective optimisation using evolutionary algorithms: an introduction. In *Multi-objective Evolutionary Optimisation for Product Design and Manufacturing*, pp. 3–34. Springer, 2011.

Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

Timo M Deist, Stefanus C Maree, Tanja Alderliesten, and Peter AN Bosman. Multi-objective optimization by uncrowded hypervolume gradient ascent. In *International Conference on Parallel Problem Solving From Nature*, 2020.

Timo M Deist, Monika Grewal, Frank JWM Dankers, Tanja Alderliesten, and Peter AN Bosman. Multi-objective learning to predict Pareto fronts using hypervolume maximization. *arXiv preprint arXiv:2102.04523*, 2021.

Jean-Antoine Désidéri. Multiple-gradient descent algorithm (MGDA) for multiobjective optimization. *Comptes Rendus Mathematique*, 350(5-6):313–318, 2012.

Tommaso Di Noia, Jessica Rosati, Paolo Tomeo, and Eugenio Di Sciascio. Adaptive multi-attribute diversity for recommender systems. *Information Sciences*, 382:234–253, 2017.

Nikolaos Dimitriadis, Pascal Frossard, and François Fleuret. Pareto manifold learning: Tackling multiple tasks via ensembles of single-task models. In *International Conference on Machine Learning*, 2023.

Nikolaos Dimitriadis, Pascal Frossard, and Francois Fleuret. Pareto low-rank adapters: Efficient multi-task learning with preferences. In *International Conference on Learning Representations*, 2025.

Lisang Ding, Ziang Chen, Xinshang Wang, and Wotao Yin. Efficient algorithms for sum-of-minimum optimization. In *International Conference on Machine Learning*, 2024.

Alexey Dosovitskiy and Josip Djolonga. You only train once: Loss-conditional training of deep networks. In *International Conference on Learning Representations*, 2020.

Matthias Ehrgott. *Multicriteria optimization*. Springer, 2005.

Gabriele Eichfelder. Twenty years of continuous multiobjective optimization in the twenty-first century. *EURO Journal on Computational Optimization*, 9:100014, 2021.

Michael Emmerich, André Deutz, and Nicola Beume. Gradient-based/evolutionary relay hybrid for computing Pareto front approximations maximizing the s-metric. In *Hybrid Metaheuristics: 4th International Workshop*, 2007.

Florian Felten, El-Ghazali Talbi, and Grégoire Danoy. Multi-objective reinforcement learning based on decomposition: A taxonomy and framework. *Journal of Artificial Intelligence Research*, 79:679–723, 2024.

Heshan Fernando, Han Shen, Miao Liu, Subhajit Chaudhury, Keerthiram Murugesan, and Tianyi Chen. Mitigating gradient bias in multi-objective learning: A provably convergent approach. In *International Conference on Learning Representations*, 2023.

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

Jörg Fliege, A Ismael F Vaz, and Luís Nunes Vicente. Complexity of gradient descent for multiobjective optimization. *Optimization Methods and Software*, 34(5):949–959, 2019.

Yingqiang Ge, Xiaoting Zhao, Lucia Yu, Saurabh Paul, Diane Hu, Chu-Cheng Hsieh, and Yongfeng Zhang. Toward Pareto efficient fairness-utility trade-off in recommendation through reinforcement learning. In *ACM International Conference on Web Search and Data Mining*, 2022.

Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

Hyojun Go, Yunsung Lee, Seunghyun Lee, Shinhyeok Oh, Hyeongdon Moon, and Seungtaek Choi. Addressing negative transfer in diffusion models. In *Conference on Neural Information Processing Systems*, 2023.

Zi Gong, Hang Yu, Cong Liao, Bingchang Liu, Chaoyu Chen, and Jianguo Li. CoBa: Convergence balancer for multitask finetuning of large language models. In *Conference on Empirical Methods in Natural Language Processing*, 2024.

Andreia P Guerreiro, Carlos M Fonseca, Michael TM Emmerich, et al. A fast dimension-sweep algorithm for the hypervolume indicator in four dimensions. In *CCCG*, 2012.

Yiju Guo, Ganqu Cui, Lifan Yuan, Ning Ding, Zexu Sun, Bowen Sun, Huimin Chen, Ruobing Xie, Jie Zhou, Yankai Lin, et al. Controllable preference optimization: Toward controllable multi-objective alignment. In *Conference on Empirical Methods in Natural Language Processing*, 2024.

Soumyajit Gupta, Gurpreet Singh, Raghu Bollapragada, and Matthew Lease. Learning a neural Pareto manifold extractor with constraints. In *Conference on Uncertainty in Artificial Intelligence*, 2022.

David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. *arXiv preprint arXiv:1609.09106*, 2016.

Nika Haghtalab, Michael Jordan, and Eric Zhao. On-demand sampling: Learning optimally from multiple distributions. In *Conference on Neural Information Processing Systems*, 2022.

Tiankai Hang, Shuyang Gu, Chen Li, Jianmin Bao, Dong Chen, Han Hu, Xin Geng, and Baining Guo. Efficient diffusion training via min-snr weighting strategy. In *International Conference on Computer Vision*, 2023.

Conor F Hayes, Roxana Rădulescu, Eugenio Bargiacchi, Johan Källström, Matthew Macfarlane, Mathieu Reymond, Timothy Verstraeten, Luisa M Zintgraf, Richard Dazeley, Fredrik Heintz, et al. A practical guide to multi-objective reinforcement learning and planning. *Autonomous Agents and Multi-Agent Systems*, 36: 26, 2022.

Long P Hoang, Dung D Le, Tran Anh Tuan, and Tran Ngoc Thang. Improving Pareto front learning via multi-sample hypernetworks. In *Annual AAAI Conference on Artificial Intelligence*, 2023.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *International Conference on Machine Learning*, 2020.

Zeou Hu, Kiarash Shaloudegi, Guojun Zhang, and Yaoliang Yu. Federated learning meets multi-objective optimization. *IEEE Transactions on Network Science and Engineering*, 9(4):2039–2051, 2022.

Youngsik Hwang and Dong-Young Lim. Dual cone gradient descent for training physics-informed neural networks. In *Conference on Neural Information Processing Systems*, 2024.

Keishi Ishihara, Anssi Kanervisto, Jun Miura, and Ville Hautamaki. Multi-task learning with attention for end-to-end autonomous driving. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Martin Jaggi. Revisiting frank-wolfe: Projection-free sparse convex optimization. In *International Conference on Machine Learning*, 2013.

Moksh Jain, Sharath Chandra Raparthy, Alex Hernández-Garcıa, Jarrid Rector-Brooks, Yoshua Bengio, Santiago Miret, and Emmanuel Bengio. Multi-objective GFlowNets. In *International Conference on Machine Learning*, 2023.

Joel Jang, Seungone Kim, Bill Yuchen Lin, Yizhong Wang, Jack Hessel, Luke Zettlemoyer, Hannaneh Hajishirzi, Yejin Choi, and Prithviraj Ammanabrolu. Personalized soups: Personalized large language model alignment via post-hoc parameter merging. *arXiv preprint arXiv:2310.11564*, 2023.

Jiaming Ji, Donghai Hong, Borong Zhang, Boyuan Chen, Josef Dai, Boren Zheng, Tianyi Qiu, Boxun Li, and Yaodong Yang. PKU-SafeRLHF: Towards multi-level safety alignment for LLMs with human preference. *arXiv preprint arXiv:2406.15513*, 2024.

Siwei Jiang, Yew-Soon Ong, Jie Zhang, and Liang Feng. Consistencies and contradictions of performance metrics in multiobjective optimization. *IEEE Transactions on Cybernetics*, 44(12):2391–2404, 2014.

Yan Kang, Hanlin Gu, Xingxing Tang, Yuanqin He, Yuzhu Zhang, Jinnan He, Yuxing Han, Lixin Fan, Kai Chen, and Qiang Yang. Optimizing privacy, utility, and efficiency in a constrained multi-objective federated learning framework. *ACM Transactions on Intelligent Systems and Technology*, 15(6), 2024.

Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Technical Report*, 2009.

Sunggu Kyung, Jongjun Won, Seongyong Pak, Sunwoo Kim, Sangyoon Lee, Kanggil Park, Gil-Sun Hong, and Namkug Kim. Generative adversarial network with robust discriminator through multi-task learning for low-dose ct denoising. *IEEE Transactions on Medical Imaging*, 2024.

Anisio Lacerda. Multi-objective ranked bandits for recommender systems. *Neurocomputing*, 246:12–24, 2017.

Song Lai, Zhe Zhao, Fei Zhu, Xi Lin, Qingfu Zhang, and Gaofeng Meng. Pareto continual learning: Preference-conditioned learning and adaption for dynamic stability-plasticity trade-off. In *Annual AAAI Conference on Artificial Intelligence*, 2025.

Haochuan Li, Jian Qian, Yi Tian, Alexander Rakhlin, and Ali Jadbabaie. Convex and non-convex optimization under generalized smoothness. In *Conference on Neural Information Processing Systems*, 2023.

Kaiwen Li, Tao Zhang, and Rui Wang. Deep reinforcement learning for multiobjective optimization. *IEEE Transactions on Cybernetics*, 51(6):3103–3114, 2020.

Wei-Hong Li, Xialei Liu, and Hakan Bilen. Universal representations: A unified look at multiple task and domain learning. *International Journal of Computer Vision*, 132(5):1521–1545, 2024a.

Weiqi Li, Fan Lyu, Fanhua Shang, Liang Wan, and Wei Feng. Long-tailed learning as multi-objective optimization. In *Annual AAAI Conference on Artificial Intelligence*, 2024b.

Ziyue Li, Tian Li, Virginia Smith, Jeff Bilmes, and Tianyi Zhou. Many-objective multi-solution transport. In *International Conference on Learning Representations*, 2025.

Xiaodan Liang, Xiwen Liang, and Hang Xu. Multi-task perception for autonomous driving. In *Autonomous Driving Perception: Fundamentals and Applications*. 2023.

Baijiong Lin and Yu Zhang. LibMTL: A Python library for deep multi-task learning. *Journal of Machine Learning Research*, 24(209):1–7, 2023.

Baijiong Lin, Feiyang Ye, Yu Zhang, and Ivor Tsang. Reasonable effectiveness of random weighting: A litmus test for multi-task learning. *Transactions on Machine Learning Research*, 2022a.

Baijiong Lin, Weisen Jiang, Pengguang Chen, Yu Zhang, Shu Liu, and Ying-Cong Chen. MTMamba: Enhancing multi-task dense scene understanding by mamba-based decoders. In *European Conference on Computer Vision*, 2024a.

Baijiong Lin, Weisen Jiang, Pengguang Chen, Shu Liu, and Ying-Cong Chen. MTMamba++: Enhancing multi-task dense scene understanding via mamba-based decoders. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(11):10633–10645, 2025a.

Baijiong Lin, Weisen Jiang, Yuancheng Xu, Hao Chen, and Ying-Cong Chen. PARM: Multi-objective test-time alignment via preference-aware autoregressive reward model. In *International Conference on Machine Learning*, 2025b.

Baijiong Lin, Weisen Jiang, Feiyang Ye, Yu Zhang, Pengguang Chen, Ying-Cong Chen, Shu Liu, Ivor Tsang, and James T. Kwok. Dual-balancing for multi-task learning. *Neural Networks*, 195:108317, 2026. ISSN 0893-6080.

Xi Lin, Hui-Ling Zhen, Zhenhua Li, Qing-Fu Zhang, and Sam Kwong. Pareto multi-task learning. In *Conference on Neural Information Processing Systems*, 2019a.

Xi Lin, Zhiyuan Yang, Qingfu Zhang, and Sam Kwong. Controllable Pareto multi-task learning. *arXiv preprint arXiv:2010.06313*, 2020.

Xi Lin, Zhiyuan Yang, and Qingfu Zhang. Pareto set learning for neural multi-objective combinatorial optimization. In *International Conference on Learning Representations*, 2022b.

Xi Lin, Xiaoyuan Zhang, Zhiyuan Yang, Fei Liu, Zhenkun Wang, and Qingfu Zhang. Smooth tchebycheff scalarization for multi-objective optimization. In *International Conference on Machine Learning*, 2024b.

Xi Lin, Yilu Liu, Xiaoyuan Zhang, Fei Liu, Zhenkun Wang, and Qingfu Zhang. Few for many: Tchebycheff set scalarization for many-objective optimization. In *International Conference on Learning Representations*, 2025c.

Xiao Lin, Hongjie Chen, Changhua Pei, Fei Sun, Xuanji Xiao, Hanxiao Sun, Yongfeng Zhang, Wenwu Ou, and Peng Jiang. A Pareto-efficient algorithm for multiple objective optimization in e-commerce recommendation. In *ACM Conference on Recommender Systems*, 2019b.

Bingchang Liu, Chaoyu Chen, Zi Gong, Cong Liao, Huan Wang, Zhichao Lei, Ming Liang, Dajun Chen, Min Shen, Hailian Zhou, Wei Jiang, Hang Yu, and Jianguo Li. MFTCoder: Boosting code LLMs with multitask fine-tuning. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024a.

Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multi-task learning. In *Conference on Neural Information Processing Systems*, 2021a.

Bo Liu, Yihao Feng, Peter Stone, and Qiang Liu. FAMO: Fast adaptive multitask optimization. In *Conference on Neural Information Processing Systems*, 2023.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: differentiable architecture search. In *International Conference on Learning Representations*, 2019a.

Liyang Liu, Yi Li, Zhanghui Kuang, Jing-Hao Xue, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. Towards impartial multi-task learning. In *International Conference on Learning Representations*, 2021b.

Meitong Liu, Xiaoyuan Zhang, Chulin Xie, Kate Donahue, and Han Zhao. Online mirror descent for tchebycheff scalarization in multi-objective optimization. *arXiv preprint arXiv:2410.21764*, 2024b.

Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. In *Conference on Neural Information Processing Systems*, 2016.

Qiang Liu, Mengyu Chu, and Nils Thuerey. ConFIG: Towards conflict-free training of physics informed neural networks. In *International Conference on Learning Representations*, 2025.

Risheng Liu, Xuan Liu, Xiaoming Yuan, Shangzhi Zeng, and Jin Zhang. A value-function-based interior-point method for non-convex bi-level optimization. In *International Conference on Machine Learning*, 2021c.

Shikun Liu, Edward Johns, and Andrew J. Davison. End-to-end multi-task learning with attention. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019b.

Shikun Liu, Stephen James, Andrew Davison, and Edward Johns. Auto-Lambda: Disentangling dynamic task relationships. *Transactions on Machine Learning Research*, 2022.

Suyun Liu and Luis Nunes Vicente. The stochastic multi-gradient algorithm for multi-objective optimization and its application to supervised machine learning. *Annals of Operations Research*, pp. 1–30, 2021.

Xingchao Liu, Xin Tong, and Qiang Liu. Profiling Pareto front with multi-objective stein variational gradient descent. In *Conference on Neural Information Processing Systems*, 2021d.

Yilu Liu, Chengyu Lu, Xi Lin, and Qingfu Zhang. Many-objective cover problem: Discovering few solutions to cover many objectives. In *International Conference on Parallel Problem Solving From Nature*, 2024c.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *International Conference on Computer Vision*, 2015.

Pingchuan Ma, Tao Du, and Wojciech Matusik. Efficient continuous Pareto exploration in multi-task learning. In *International Conference on Machine Learning*, 2020.

Debabrata Mahapatra and Vaibhav Rajan. Multi-task learning with user preferences: Gradient descent with controlled ascent in Pareto optimization. In *International Conference on Machine Learning*, 2020.

Debabrata Mahapatra and Vaibhav Rajan. Exact Pareto optimal search for multi-task learning and multi-criteria decision-making. *arXiv preprint arXiv:2108.00597*, 2021.

R Timothy Marler and Jasbir S Arora. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26(6):369–395, 2004.

Natalia Martinez, Martin Bertran, and Guillermo Sapiro. Minimax pareto fairness: A multi objective perspective. In *International Conference on Machine Learning*, pp. 6755–6764. PMLR, 2020.

Kaisa Miettinen. *Nonlinear multiobjective optimization*, volume 12. Springer Science & Business Media, 1999.

Nikola Milojkovic, Diego Antognini, Giancarlo Bergamin, Boi Faltings, and Claudiu Musat. Multi-gradient descent for multi-objective recommender systems. *arXiv preprint arXiv:2001.00846*, 2019.

Blagoj Mitrevski, Milena Filipovic, Diego Antognini, Emma Lejal Glaude, Boi Faltings, and Claudiu Musat. Momentum-based gradient methods in multi-objective recommendation. *arXiv preprint arXiv:2009.04695*, 2020.

Michinari Momma, Chaosheng Dong, and Jia Liu. A multi-objective/multi-task learning framework induced by Pareto stationarity. In *International Conference on Machine Learning*, 2022.

Hiroaki Mukai. Algorithms for multicriterion optimization. *IEEE Transactions on Automatic Control*, 25 (2):177–186, 1980.

John Nash. Two-person cooperative games. *Econometrica: Journal of the Econometric Society*, pp. 128–140, 1953.

Aviv Navon, Aviv Shamsian, Ethan Fetaya, and Gal Chechik. Learning the Pareto front with hypernetworks. In *International Conference on Learning Representations*, 2021.

Aviv Navon, Aviv Shamsian, Idan Achituve, Haggai Maron, Kenji Kawaguchi, Gal Chechik, and Ethan Fetaya. Multi-task learning as a bargaining game. In *International Conference on Machine Learning*, 2022.

Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, et al. PyTorch: An imperative style, high-performance deep learning library. In *Conference on Neural Information Processing Systems*, 2019.

Shubham Patil, Debopriyo Banerjee, and Shamik Sural. A graph theoretic approach for multi-objective budget constrained capsule wardrobe recommendation. *ACM Transactions on Information Systems*, 40 (1):1–33, 2021.

Sebastian Peitz and Sedjro Salomon Hotegni. Multi-objective deep learning: Taxonomy and survey of the state of the art. *arXiv preprint arXiv:2412.01566*, 2024.

Binghui Peng. The sample complexity of multi-distribution learning. In *Annual Conference on Learning Theory*, 2024.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *Annual AAAI Conference on Artificial Intelligence*, 2018.

Pierre Quinton and Valérian Rey. Jacobian descent for multi-objective optimization. *arXiv preprint arXiv:2406.16232*, 2024.

Roxana Rădulescu, Patrick Mannion, Diederik M Roijers, and Ann Nowé. Multi-objective multi-agent decision making: a utility-based analysis and survey. *Autonomous Agents and Multi-Agent Systems*, 34 (1):10, 2020.

Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole Von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1(1):1–7, 2014.

Alexandre Rame, Guillaume Couairon, Corentin Dancette, Jean-Baptiste Gaya, Mustafa Shukor, Laure Soulier, and Matthieu Cord. Rewarded soups: towards Pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards. In *Conference on Neural Information Processing Systems*, 2023.

Dripta S Raychaudhuri, Yumin Suh, Samuel Schulter, Xiang Yu, Masoud Faraki, Amit K Roy-Chowdhury, and Manmohan Chandraker. Controllable dynamic multi-task architectures. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Xiaojiang Chen, and Xin Wang. A comprehensive survey of neural architecture search: Challenges and solutions. *ACM Computing Surveys*, 54(4):1–34, 2021.

Diederik M Roijers, Peter Vamplew, Shimon Whiteson, and Richard Dazeley. A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research*, 48:67–113, 2013.

Willem Röpke, Mathieu Reymond, Patrick Mannion, Diederik M Roijers, Ann Nowé, and Roxana Rădulescu. Divide and conquer: Provably unveiling the pareto front with multi-objective reinforcement learning. In *Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems*, pp. 1774–1783, 2025.

Michael Ruchte and Josif Grabocka. Scalable Pareto front approximation for deep multi-objective learning. In *IEEE International Conference on Data Mining*, 2021.

S Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017.

Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *European Conference on Computer Vision*, 2010.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Conference on Neural Information Processing Systems*, 2018.

Dmitry Senushkin, Nikolay Patakin, Arseny Kuznetsov, and Anton Konushin. Independent component alignment for multi-task learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Ke Shang, Hisao Ishibuchi, Linjun He, and Lie Meng Pang. A survey on the hypervolume indicator in evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 25(1):1–20, 2020.

Ruizhe Shi, Yifang Chen, Yushi Hu, ALisa Liu, Hannaneh Hajishirzi, Noah A Smith, and Simon S Du. Decoding-time language model alignment with multiple objectives. In *Conference on Neural Information Processing Systems*, 2024.

Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from RGBD images. In *European Conference on Computer Vision*, 2012.

Víctor Adrián Sosa Hernández, Oliver Schütze, Hao Wang, André Deutz, and Michael Emmerich. The set-based hypervolume newton method for bi-objective optimization. *IEEE Transactions on Cybernetics*, 50 (5):2186–2196, 2020.

Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Which tasks should be learned together in multi-task learning? In *International Conference on Machine Learning*, 2020.

Ralph E Steuer and Eng-Ung Choo. An interactive weighted tchebycheff procedure for multiple objective programming. *Mathematical Programming*, 26:326–344, 1983.

Ralph E Steuer and Paul Na. Multiple criteria decision making combined with finance: A categorized bibliographic study. *European Journal of Operational Research*, 150(3):496–515, 2003.

Peter Súkeník and Christoph Lampert. Generalization in multi-objective machine learning. *Neural Computing and Applications*, pp. 1–15, 2024.

Rhea Sanjay Sukthanker, Arber Zela, Benedikt Staffler, Samuel Dooley, Josif Grabocka, and Frank Hutter. Multi-objective differentiable neural architecture search. In *International Conference on Learning Representations*, 2025.

Anke Tang, Li Shen, Yong Luo, Shiwei Liu, Han Hu, and Bo Du. Towards efficient Pareto set approximation via mixture of experts based model fusion. *arXiv preprint arXiv:2406.09770*, 2024.

William Thomson. Cooperative models of bargaining. *Handbook of Game Theory with Economic Applications*, 2:1237–1284, 1994.

Tran Anh Tuan, Nguyen Viet Dung, and Tran Ngoc Thang. A hyper-transformer model for controllable Pareto front learning with split feasibility constraints. *arXiv preprint arXiv:2402.05955*, 2024.

Gwo-Hshiung Tzeng, Cheng-Wei Lin, and Serafim Opricovic. Multi-criteria analysis of alternative-fuel buses for public transportation. *Energy Policy*, 33(11):1373–1383, 2005.

David A Van Veldhuizen and Gary B Lamont. Multiobjective evolutionary algorithm test suites. In *Proceedings of the 1999 ACM symposium on Applied computing*, pp. 351–357, 1999.

Simon Vandenhende, Stamatios Georgoulis, Wouter Van Gansbeke, Marc Proesmans, Dengxin Dai, and Luc Van Gool. Multi-task learning for dense prediction tasks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7):3614–3633, 2021.

Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017.

Dilin Wang, Meng Li, Chengyue Gong, and Vikas Chandra. AttentiveNAS: Improving neural architecture search via attentive sampling. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021a.

Hao Wang, André Deutz, Thomas Bäck, and Michael Emmerich. Hypervolume indicator gradient ascent multi-objective optimization. In *International Conference on Evolutionary Multi-Criterion Optimization*, 2017.

Haoxiang Wang, Han Zhao, and Bo Li. Bridging multi-task learning and meta-learning: Towards efficient training and effective adaptation. In *International Conference on Machine Learning*, 2021b.

Haoxiang Wang, Yong Lin, Wei Xiong, Rui Yang, Shizhe Diao, Shuang Qiu, Han Zhao, and Tong Zhang. Arithmetic control of LLMs for diverse user preferences: Directional preference alignment with multi-objective rewards. In *Annual Meeting of the Association for Computational Linguistics*, 2024a.

Yufei Wang, Wanjun Zhong, Liangyou Li, Fei Mi, Xingshan Zeng, Wenyong Huang, Lifeng Shang, Xin Jiang, and Qun Liu. Aligning large language models with human: A survey. *arXiv preprint arXiv:2307.12966*, 2023a.

Zhenkun Wang, Shunyu Yao, Genghui Li, and Qingfu Zhang. Multiobjective combinatorial optimization using a single deep reinforcement learning model. *IEEE Transactions on Cybernetics*, 2023b.

Zhenyu Wang, Yali Li, Hengshuang Zhao, and Shengjin Wang. One for all: Multi-domain joint training for point cloud based 3d object detection. In *Conference on Neural Information Processing Systems*, 2024b.

Zhilin Wang, Yi Dong, Olivier Delalleau, Jiaqi Zeng, Gerald Shen, Daniel Egert, Jimmy J Zhang, Makesh Narsimhan Sreedhar, and Oleksii Kuchaiev. Helpsteer2: Open-source dataset for training top-performing reward models. In *Conference on Neural Information Processing Systems*, 2024c.

Zirui Wang, Yulia Tsvetkov, Orhan Firat, and Yuan Cao. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. In *International Conference on Learning Representations*, 2021c.

Tingyang Wei, Shibin Wang, Jinghui Zhong, Dong Liu, and Jun Zhang. A review on evolutionary multitask optimization: Trends and challenges. *IEEE Transactions on Evolutionary Computation*, 26(5):941–960, 2021.

Timo Wilm, Philipp Normann, and Felix Stepprath. Pareto front approximation for multi-objective session-based recommender systems. In *ACM Conference on Recommender Systems*, 2024.

Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. FBNet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

Yan Wu, Zhiwu Huang, Suryansh Kumar, Rhea Sanjay Sukthanker, Radu Timofte, and Luc Van Gool. Trilevel neural architecture search for efficient single image super-resolution. *arXiv preprint arXiv:2101.06658*, 2021.

Zeqiu Wu, Yushi Hu, Weijia Shi, Nouha Dziri, Alane Suhr, Prithviraj Ammanabrolu, Noah A Smith, Mari Ostendorf, and Hannaneh Hajishirzi. Fine-grained human feedback gives better rewards for language model training. In *Conference on Neural Information Processing Systems*, 2023.

Peiyao Xiao, Hao Ban, and Kaiyi Ji. Direction-oriented multi-objective learning: Simple and provable stochastic algorithms. In *Conference on Neural Information Processing Systems*, 2023.

Ruobing Xie, Yanlei Liu, Shaoliang Zhang, Rui Wang, Feng Xia, and Leyu Lin. Personalized approximate Pareto-efficient recommendation. In *International World Wide Web Conference*, 2021.

Tao Xie, Shiguang Wang, Ke Wang, Linqi Yang, Zhiqiang Jiang, Xingcheng Zhang, Kun Dai, Ruifeng Li, and Jian Cheng. Poly-PC: A polyhedral network for multiple point cloud tasks at once. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Tao Xie, Kun Dai, Qihao Sun, Zhiqiang Jiang, Chuqing Cao, Lijun Zhao, Ke Wang, and Ruifeng Li. CO-Net++: A cohesive network for multiple point cloud tasks at once with two-stage feature rectification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

Mingjing Xu, Peizhong Ju, Jia Liu, and Haibo Yang. PSMGD: Periodic stochastic multi-gradient descent for fast multi-objective optimization. In *Annual AAAI Conference on Artificial Intelligence*, 2025a.

Xiaogang Xu, Hengshuang Zhao, Vibhav Vineet, Ser-Nam Lim, and Antonio Torralba. MTFormer: Multi-task learning via transformer and cross-task reasoning. In *European Conference on Computer Vision*, 2022.

Zhiyuan Xu, Yinhe Chen, Huan-ang Gao, Weiyan Zhao, Guiyu Zhang, and Hao Zhao. Diffusion-based visual anagram as multi-task learning. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, 2025b.

Liangwei Yang, Zhiwei Liu, Jianguo Zhang, Rithesh Murthy, Shelby Heinecke, Huan Wang, Caiming Xiong, and Philip S Yu. Personalized multi-task training for recommender system. *arXiv preprint arXiv:2407.21364*, 2024a.

Rui Yang, Xiaoman Pan, Feng Luo, Shuang Qiu, Han Zhong, Dong Yu, and Jianshu Chen. Rewards-in-context: Multi-objective alignment of foundation models with dynamic preference adjustment. In *International Conference on Machine Learning*, 2024b.

Yinghua Yao, Yuangang Pan, Jing Li, Ivor Tsang, and Xin Yao. PROUD: PaRetO-gUided diffusion model for multi-objective generation. *Machine Learning*, 113(9):6511–6538, 2024.

Dongqiangzi Ye, Yufei Xie, Weijia Chen, Zixiang Zhou, Lingting Ge, and Hassan Foroosh. LPFormer: Lidar pose estimation transformer with multi-task network. In *IEEE International Conference on Robotics & Automation*, 2024a.

Feiyang Ye, Baijiong Lin, Zhixiong Yue, Pengxin Guo, Qiao Xiao, and Yu Zhang. Multi-objective meta learning. In *Conference on Neural Information Processing Systems*, 2021.

Feiyang Ye, Baijiong Lin, Xiaofeng Cao, Yu Zhang, and Ivor Tsang. A first-order multi-gradient algorithm for multi-objective bi-level optimization. In *European Conference on Artificial Intelligence*, 2024b.

Feiyang Ye, Baijiong Lin, Zhixiong Yue, Yu Zhang, and Ivor Tsang. Multi-objective meta-learning. *Artificial Intelligence*, 335:104184, 2024c.

Hanrong Ye and Dan Xu. Inverted pyramid multi-task transformer for dense scene understanding. In *European Conference on Computer Vision*, 2022.

Jun Yu, Yutong Dai, Xiaokang Liu, Jin Huang, Yishan Shen, Ke Zhang, Rong Zhou, Eashan Adhikarla, Wenxuan Ye, Yixin Liu, et al. Unleashing the power of multi-task learning: A comprehensive survey spanning traditional, deep, and pretrained foundation model eras. *arXiv preprint arXiv:2404.18961*, 2024.

Runsheng Yu, Weiyu Chen, Xinrun Wang, and James Kwok. Enhancing meta learning via multi-objective soft improvement functions. In *International Conference on Learning Representations*, 2023.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Conference on Neural Information Processing Systems*, 2020.

Zhixiong Yue, Baijiong Lin, Yu Zhang, and Christy Liang. Effective, efficient and robust neural architecture search. In *International Joint Conference on Neural Networks*, 2022.

Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

Mingzhu Zhang, Ruiping Yin, Zhen Yang, Yipeng Wang, and Kan Li. Advances and challenges of multi-task learning method in recommender system: a survey. *arXiv preprint arXiv:2305.13843*, 2023a.

Qi Zhang, Peiyao Xiao, Kaiyi Ji, and Shaofeng Zou. MGDA converges under generalized smoothness, provably. In *International Conference on Learning Representations*, 2025.

Qingfu Zhang and Hui Li. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007.

Xiaoyuan Zhang, Xi Lin, Bo Xue, Yifan Chen, and Qingfu Zhang. Hypervolume maximization: A geometric view of Pareto set learning. In *Conference on Neural Information Processing Systems*, 2023b.

Xiaoyuan Zhang, Genghui Li, Xi Lin, Yichi Zhang, Yifan Chen, and Qingfu Zhang. Gliding over the Pareto front with uniform designs. In *Conference on Neural Information Processing Systems*, 2024a.

Xiaoyuan Zhang, Xi Lin, and Qingfu Zhang. PMGDA: A preference-based multiple gradient descent algorithm. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2024b.

Xiaoyuan Zhang, Liang Zhao, Yingying Yu, Xi Lin, Yifan Chen, Han Zhao, and Qingfu Zhang. LibMOON: A gradient-based multiobjective optimization library in PyTorch. In *Conference on Neural Information Processing Systems*, 2024c.

Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 34(12):5586–5609, 2022.

Zihan Zhang, Wenhao Zhan, Yuxin Chen, Simon S Du, and Jason D Lee. Optimal multi-distribution learning. In *Annual Conference on Learning Theory*, 2024d.

Zhe Zhao, Pengkun Wang, HaiBin Wen, Wei Xu, Song Lai, Qingfu Zhang, and Yang Wang. Two fists, one heart: Multi-objective optimization based strategy fusion for long-tailed learning. In *International Conference on Machine Learning*, 2024.

Yong Zheng and David Xuejun Wang. A survey of recommender systems with multi-objective optimization. *Neurocomputing*, 474:141–153, 2022.

Yifan Zhong, Chengdong Ma, Xiaoyuan Zhang, Ziran Yang, Qingfu Zhang, Siyuan Qi, and Yaodong Yang. Panacea: Pareto alignment via preference adaptation for LLMs. In *Conference on Neural Information Processing Systems*, 2024.

Shiji Zhou, Wenpeng Zhang, Jiyan Jiang, Wenliang Zhong, Jinjie Gu, and Wenwu Zhu. On the convergence of stochastic multi-objective gradient manipulation and beyond. In *Conference on Neural Information Processing Systems*, 2022.

Zhanhui Zhou, Jie Liu, Jing Shao, Xiangyu Yue, Chao Yang, Wanli Ouyang, and Yu Qiao. Beyond one-preference-fits-all alignment: Multi-objective direct preference optimization. In *Findings of Annual Meeting of the Association for Computational Linguistics*, 2024a.

Zhipeng Zhou, Liu Liu, Peilin Zhao, and Wei Gong. Pareto deep long-tailed recognition: A conflict-averse solution. In *International Conference on Learning Representations*, 2024b.

Yiheng Zhu, Jialu Wu, Chaowen Hu, Jiahuan Yan, Tingjun Hou, and Jian Wu. Sample-efficient multi-objective molecular optimization with GFlowNets. In *Conference on Neural Information Processing Systems*, 2023.

Eckart Zitzler and Lothar Thiele. Multiobjective optimization using evolutionary algorithms—a comparative case study. In *International Conference on Parallel Problem Solving From Nature*, 1998.

Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M Fonseca, and Viviane Grunert Da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on evolutionary computation*, 7(2):117–132, 2003.