Revisiting Fairness in Multitask Learning: A Performance-Driven Approach for Variance Reduction

Xiaohan Qin **Xiaoxing Wang**[‡] **Junchi Yan**[‡]
School of Computer Science & School of Artificial Intelligience, Shanghai Jiao Tong University
{galaxy-1,figure1_wxx,yanjunchi}@sjtu.edu.cn
https://github.com/jianke0604/MTLlib

Abstract

Multi-task learning (MTL) can leverage shared knowledge across tasks to improve data efficiency and generalization performance, and has been applied in various scenarios. However, task imbalance remains a major challenge for existing MTL methods. While the prior works have attempted to mitigate inter-task unfairness through loss-based and gradient-based strategies, they still exhibit imbalanced performance across tasks on common benchmarks. This key observation motivates us to consider performance-level information as an explicit fairness indicator, which can precisely reflect the current optimization status of each task, and accordingly help to adjust the gradient aggregation process. Specifically, we utilize the performance variance among tasks as the fairness indicator and introduce a dynamic weighting strategy to gradually reduce the performance variance. Based on this, we propose PIVRG, a novel performance-informed variance reduction gradient aggregation approach. Extensive experiments show that PIVRG achieves SOTA performance across various benchmarks, spanning both supervised learning and reinforcement learning tasks with task numbers ranging from 2 to 40. Results from the ablation study also show that our approach can be integrated into existing methods, significantly enhancing their performance while reducing the performance variance among tasks, thus achieving fairer optimization.

1. Introduction

Multi-task learning (MTL) [7, 32, 42] is an approach where a single model is trained to solve multiple tasks simultaneously. This paradigm allows tasks to share information and representations, which can enhance the generalization capabilities of the model and improve performance across tasks [5, 29, 36]. MTL is especially beneficial in scenarios where computational resources are limited, as it reduces the need for separate models for each task. Its applications span a variety of domains, such as computer vision [1, 26, 43], natural language processing [8, 25, 31], and robotics [13, 37]. Despite its advantages, MTL faces one major significant challenge: task imbalance, which describes the phenomenon where some tasks dominate the learning process while others are under-optimized, leading to degraded overall performance. Overcoming this challenge requires careful design of optimization strategies to ensure that all tasks benefit equally from the shared model.

To address these issues, previous works have focused on two primary approaches: loss-based and gradient-based methods [24, 34]. Loss-based methods aggregate the losses of different tasks by adjusting the loss scales, then backpropagate the total loss to compute gradients for shared parameters. These methods attempt to reflect the optimization status of each task through their respective losses. However, since tasks often use different loss functions (e.g., cross-entropy for classification and L1 loss for regression), their scales can differ significantly, making it hard to compare or balance them directly. Gradient-based methods compute the gradient for each task separately and then aggregate these gradients using various algorithms to produce a final update. Although these methods ensure that the gradient of each task is considered, relying on the gradients to represent optimization fairness can be misleading at certain stages of the optimization process. For instance, the gradient approaches zero at a local minimum, while the optimization state may still be suboptimal. Additionally, these methods typically aim to equalize the task gradients at the shared layers [9, 24], but can not guarantee the balance in the training progress because the difficulty of tasks may differ. Easier tasks may converge quickly, while difficult tasks require more time to optimize [16]. As a result, considering only gradients can overlook differences in task difficulty, making it insufficient to ensure balance across all tasks.

A clear illustration of the issues above can be observed

[‡] Corresponding author. This work was partly supported by NSFC (62222607) and Shanghai Municipal Science and Technology Major Project under Grant 2021SHZDZX0102.

in the widely used NYUv2 benchmark [35], which involves three tasks: segmentation, depth estimation, and surface normal prediction. While recent MTL methods have demonstrated improvements over single-task learning (STL), as evidenced by negative average performance drops Δm [4, 23, 30], their experiments show that these methods outperform STL primarily on the segmentation and depth tasks. However, their performance on the surface normal task consistently lags behind STL, leading to a substantial variance in the performance drop across tasks. This variance contradicts the original goal of MTL, which aims to achieve balanced optimization across all tasks. This key observation prompts us to rethink: *Is information from loss-level and gradient-level metrics sufficient to represent fairness in multi-task optimization?* In our view, performance-level information should also be considered. A more interesting result is that, as shown in Fig. 1, the performance drop Δm of different MTL methods exhibits a striking positive correlation with cross-task performance variance. Methods with smaller performance variance tend to achieve better overall performance, which has been overlooked by most previous MTL approaches.

Building upon the above insight, we employ the performance variance across tasks as a fairness indicator and implement a dynamic weighting approach aimed at progressively decreasing this performance variance. This enhances the generalization of the shared representations, reducing excessive performance discrepancies between tasks. In summary, our contributions can be outlined as follows:

1). **We highlight the positive correlation between the performance and cross-task performance variance of MTL methods and propose PIVRG, a performance-informed variance reduction gradient aggregation approach.** Specifically, we utilize the performance variance among tasks as a fairness indicator and introduce a dynamic weighting strategy, which serves as a regularization mechanism balancing the performance drop across different tasks during MTL. Both theoretical analysis and experimental results demonstrate that our method not only converges to the Pareto stationary but also achieves superior performance.

2). **The proposed performance-informed dynamic weighting strategy is orthogonal to existing approaches, making it possible to integrate with these methods.** Experimental results demonstrate that incorporating our dynamic weighting strategy not only significantly improves the overall performance of these methods but also substantially reduces the performance variance across tasks, leading to a more balanced optimization. This further validates the potential of our approach.

3). **Extensive experiments demonstrate that PIVRG achieves state-of-the-art performance across various benchmarks.** The experimental scenarios span both supervised learning and reinforcement learning tasks, with task numbers ranging from 2 to 40. Notably, our method also achieves the smallest performance variance across various benchmarks, indicating that it not only achieves SOTA performance but also ensures fairer optimization.

2. Related Work

Loss-based MTL approaches. These methods reweight task-specific losses with loss-level information. A key advantage of loss-based methods is their efficiency, as they only require backpropagation on the aggregated loss, reducing computational overhead compared to handling each task individually [23]. These approaches include operations on the scale of the loss, such as Linear Scalarization (LS), which minimizes the sum of task losses, and Scale-Invariant (SI), which reduces the sum of logarithmic losses. Additionally, various approaches for handling task weights have been proposed, including using homoscedastic uncertainty weighting [19], dynamic weight averaging [26], self-paced learning [28], geometric loss [11], random loss weighting [21], impartial loss weighting [24] and fast adaptive optimization [23]. Although loss-oriented methods are more computationally efficient, they often underperform gradient-oriented ones in most multi-task benchmarks.

Gradient-based MTL approaches. These methods address the task-balancing problem by fully leveraging the gradient information of the shared network across different tasks. These approaches include Pareto optimal solutions [33], gradient normalization [9], projecting gradient conflicts [38], gradient sign dropout [10], impartial gradient weighting [24], conflict-averse gradients [22], independent gradient alignment [34], and Bayesian uncertainty gradients [2]. Recent works [30] and [4] employe Nash bargaining solution and fair resource allocation respectively to address the gradient aggregation. Their utility functions are primarily based on the first-order Taylor expansion of the loss, thereby incorporating loss-level information. Our proposed PIVRG method is also a gradient-based approach. However, it not only incorporates loss-level information but also introduces higher-order insights from performance-level.

Remarks. Note that although [16] and [41] also incorporate performance or accuracy to assess task priority or potential, their approaches use performance-level information to balance the loss, without incorporating gradient-level information. Additionally, they utilize the performance of each task to help design the weights of losses, which neglects the influence of cross-task weight constraints, and thus fails to ensure the reduction of performance variance.

3. Method

3.1. Preliminaries

Pareto Optimality. Optimization in MTL can be understood as an instance of multi-objective optimization

(MOO). For a set of objective functions $\ell_1, \cdots, \ell_k$, the quality of a solution x is determined by the vector of its corresponding objective values, i.e., $(\ell_1(x), \cdots, \ell_k(x))$. A key characteristic of MOO is the absence of a natural linear ordering for such vectors, meaning that solutions are not always directly comparable, and thus no single optimal solution exists.

We define a solution x as dominating another solution x' if it is strictly better in at least one objective while being no worse in all others. A solution that is not dominated by any other solution is termed Pareto optimal, and the set of all such solutions forms the Pareto front. In the case of non-convex problems, a solution is considered locally Pareto optimal if it is Pareto optimal within a neighborhood around it. Furthermore, a solution is called Pareto stationary when there exists a convex combination of the gradients at that point that equals zero, which is a necessary condition for Pareto optimality.

Multi-Task Optimization Objectives. One of the most crucial distinctions between different MTL methods lies in their choice of optimization objectives. A traditional approach is to minimize the average loss across all tasks:

$$\min_{\theta} \left\{ \mathcal{L}(\theta) := \frac{1}{k} \sum_{i=1}^{k} \ell_i(\theta) \right\}, \tag{1}$$

where $\theta \in \mathbb{R}^n$ is the parameter shared across tasks. Directly optimizing Eq. 1 can lead to significant under-optimization for certain tasks and this is often caused by the varying scales of the different loss functions as discussed in Sec. 1. Gradient-based methods typically propose an aggregation algorithm $\mathcal{A}$ (e.g., conflict projection [38], cosine similarity balancing [24]), which solves an optimization problem $\mathcal{A}(g_1, g_2, \cdots, g_k)$ to obtain the update direction d. Recent works [4, 30] represent updates at each iteration as $\theta_{t+1} = \theta_t - \eta d$, where η is the current step size and d is the computed update direction. Considering a first-order Taylor expansion $\ell_i(\theta_{t+1}) - \ell_i(\theta_t) \approx -\eta g_i^\top d$, they interpret $g_i^\top d$ as the utility of task i at the current step, thus taking loss-level information into account.

In this paper, we also consider $g_i^\top d$ as the utility of task i at the current step. However, unlike Nash-MTL [30], which aims to maximize the sum of the log-utilities, we propose to minimize the mean of the inverse utilities for each task:

$$\arg\min_{d} \frac{1}{k} \sum_{i=1}^{k} \frac{1}{g_i^\top d} \qquad \text{s.t.} \quad g_i^\top d > 0, \forall i. \tag{2}$$

It emphasizes tasks with lower utilities, thereby preventing tasks with high utilities from dominating the optimization process. In fact, from another perspective, $g_i^\top d$, as an approximation of the change in loss, can be viewed as the current optimization speed of task i, while $1/g_i^\top d$ can be interpreted as the number of steps required for unit improvement. The objective in (2) essentially minimizes the average number of steps needed for unit optimization across tasks.

Note that similar to MGDA [33] and Nash-MTL [30], the optimization objective in (2) can also be interpreted as a specific case of the FairGrad framework [4]. As a strong gradient-based baseline, it has been extensively compared with our approach in the experimental results.

3.2. Performance-Informed Weighting Strategy

To mitigate the task imbalance issue mentioned in Sec. 1, we propose incorporating performance-level information $\{\Delta m_i\}_{i=1}^k$ to account for the varying difficulties across tasks. Specifically, for each task i, we define the per-task performance drop Δm_i as:

$$\Delta m_i = \frac{1}{\mathcal{S}_i} \sum_{j=1}^{\mathcal{S}_i} (-1)^{\delta_j} (M_{m,j} - M_{b,j})/M_{b,j} \times 100, \tag{3}$$

where $\mathcal{S}_i$ is the number of metrics for task i, i.e. $\mathcal{S}_i \geq 1$. $M_{b,j}$ is the value of metric M_j obtained by the STL baseline and $M_{m,j}$ denotes the value from the compared MTL method. $\delta_j = 1$ if a higher value is better for the metric M_j and 0 otherwise. This ratio quantifies the relative degradation of performance when tasks are optimized jointly. Note that the $\{\Delta m_i\}_{i=1}^k$ defined by our method corresponds to the **per-task performance drop**, rather than the **per-metric performance drop** used to compare different MTL methods, which is specifically defined in Section 4.

Moreover, $\mathbb{E}[\Delta m_i] = \frac{1}{k} \sum_{i=1}^k \Delta m_i$ is a metric reflecting the overall performance of the MTL method across tasks. While reducing the average performance drop is the goal of all MTL methods, we utilize the performance variance $\text{Var}[\Delta m_i]$ among tasks as a fairness indicator and consider it as a potential optimization target for ensuring fairness. Due to Δm_i is not a random variable, we just use $\text{Var}[\cdot]$ as a formal notation to represent the variance of performance drop.

Since $\{\Delta m_i\}_{i=1}^k$ represents actual performance and lacks gradients for backpropagation, it cannot be directly optimized for variance reduction without further assumptions. Thus, we introduce dynamic weights $\boldsymbol{\omega} = (\omega_1, \omega_2, \ldots, \omega_k)^\top \in \mathbb{R}^k$ as regularizers to guide the optimization. We rewrite the original objective as:

$$\arg\min_{d} \frac{1}{k} \sum_{i=1}^{k} \frac{\omega_i}{g_i^\top d} \qquad \text{s.t.} \quad g_i^\top d > 0, \forall i. \tag{4}$$

In (4), the choice of $\boldsymbol{\omega}$ is crucial. We aim for $\boldsymbol{\omega}$ to reflect the current performance-level information of each task and to promote reducing $\text{Var}[\Delta m_i]$ during optimization. To this end, $\boldsymbol{\omega}$ should satisfy the following properties:

Property 1. *ω_i should be positively correlated with Δm_i.*

The objective in (2) minimizes the average number of steps required for unit optimization across tasks. However, as previously discussed, different tasks have varying difficulties, and more difficult tasks may require more steps at the same optimization step size. Without considering task difficulty, the objective in (2) might result in over-optimization of some tasks while others remain under-optimized, thus maintaining task imbalance. By modifying the weights ω_i to be positively correlated with Δm_i, the objective becomes aware of task difficulty.

Property 2. $\mathbf{1}^\top \boldsymbol{\omega} = k$.

This ensures alignment with the original objective without the weight. In the unweighted case, $\boldsymbol{\omega} = (1, 1, ..., 1)^\top$ satisfies $\mathbf{1}^\top \boldsymbol{\omega} = k$. Our weighting strategy dynamically adjusts this k from a fixed mean to a more flexible distribution, and adds correlation to the weights of different tasks. Furthermore, this property imposes an overall constraint on the performance of all tasks, such that the performance of each task not only affects its own weight but also influences the weights of other tasks. This serves as a key condition for achieving variance reduction.

Property 3. *$\boldsymbol{\omega}$ is bounded, i.e., $\omega_i \in [\underline{\omega}, \overline{\omega}]$.*

This constraint ensures that the weight does not become too extreme, for instance, preventing one task with poor performance from consuming all resources (especially during early training when $\mathrm{Var}[\Delta m_i]$ might be large). In practice, we typically choose $\underline{\omega} \in [0.5, 0.8]$ and $\overline{\omega} \in [1.2, 2]$.

Property 2 is essentially a special case of normalization, making the commonly used softmax a natural choice. Applying softmax to $\{\Delta m_i\}_{i=1}^k$ also ensures the positive correlation required by Property 1. However, we observe that direct normalization makes it challenging to enforce $\omega_i \in [\underline{\omega}, \overline{\omega}]$. To address this, a simple yet effective idea is to adopt a variant of softmax with a temperature parameter:

$$\omega_i = \frac{k \cdot \exp(\Delta m_i/\tau)}{\sum_{j=1}^k \exp(\Delta m_j/\tau)} \tag{5}$$

where τ is the temperature parameter controlling the smoothness of the softmax output. We assert that by choosing an appropriate τ, Property 3 can also be satisfied.

Proposition 1. *Let Δm^{max} be the maximum value of $\{\Delta m_i\}_{i=1}^k$, and Δm^{min} be the minimum value. Define $s = \min\left(\frac{1}{\underline{\omega}}, \overline{\omega}\right)$. Then, for $\tau > \frac{\Delta m^{max} - \Delta m^{min}}{\log s}$, Property 3 is satisfied.*

The proof can be found in the Appendix. We also demonstrate that the fairness indicator and potential optimization target, $\mathrm{Var}[\Delta m_i]$, can be approximated by the norm of $\boldsymbol{\omega}$, specifically $\boldsymbol{\omega}^\top \boldsymbol{\omega}$.

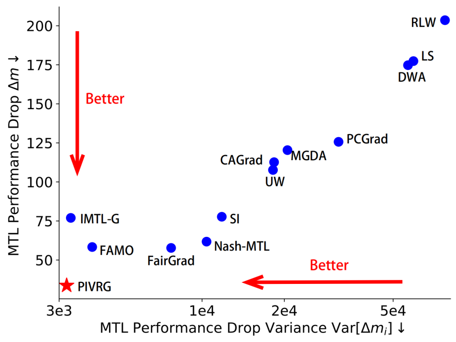

Figure 1. Experiment results about per-metric Δm and $\mathrm{Var}[\Delta m_i]$ on the QM9 dataset.

Proposition 2. *For $\boldsymbol{\omega}$ satisfying the three properties above, we have the following approximation:*

$$\mathrm{Var}[\Delta m_i] \approx \frac{\tau^2}{k} \boldsymbol{\omega}^\top \boldsymbol{\omega} - \tau^2.$$

This implies $\mathrm{Var}[\Delta m_i] \propto \boldsymbol{\omega}^\top \boldsymbol{\omega}$. The detailed proof can be found in the Appendix. The result in Fig. 1 shows that PIVRG outperforms other approaches and produces the lowest performance variance. Furthermore, the experimental results in Sec.7.3 in the Appendix show that both $\boldsymbol{\omega}^\top \boldsymbol{\omega}$ and $\mathrm{Var}[\Delta m_i]$ decrease progressively throughout the optimization process, further confirming the effectiveness of our dynamic weights which serve as regularizers. After refining the definition of $\boldsymbol{\omega}$, the problem reduces to solving for the optimal d in Eq. (4).

3.3. Deriving the Optimal Update Vector d

To search for the update vector d, we introduce an innovative approach by deriving the analytical solution for d using Lagrange multipliers as follows. First, we show that both the objective function and constraints are convex:
Convexity of the Objective Function: For each task i, the term $\omega_i / g_i^\top d$ is convex in d over the region where $g_i^\top d > 0$, as the function ω_i / x is convex for $x > 0$ and $\omega_i > 0$.
Convexity of the Constraints: The constraint $g_i^\top d > 0$ is linear in d, and the norm constraint $\|d\| \leq \epsilon$ is also convex.

Since (4) is a convex optimization problem, we define the Lagrangian for this problem as follows:

$$L(d, \lambda, \{\mu_i\}) = \frac{1}{k} \sum_{i=1}^k \frac{\omega_i}{g_i^\top d} + \lambda(\|d\|^2 - \epsilon^2) - \sum_{i=1}^k \mu_i (g_i^\top d),$$

where $\lambda \geq 0$ and $\mu_i \geq 0$ are Lagrange multipliers associated with the constraints. The following Karush-Kuhn-Tucker (KKT) conditions provide the necessary conditions for optimality:

1) Primal Feasibility: $g_i^\top d > 0,\ \|d\| \leq \epsilon,\ \forall i$. This ensures that the inner products are positive and the norm of d does not exceed ϵ.
2) Dual Feasibility: $\mu_i \geq 0,\ \lambda \geq 0$. This condition guarantees that the multipliers are non-negative, maintaining the validity of the constraints.
3) Complementary Slackness: $\mu_i(-g_i^\top d) = 0,\ \lambda(\|d\|^2 - \epsilon^2) = 0$. Since $g_i^\top d > 0$, it follows that $\mu_i = 0$ for all i.
4) Stationarity: The gradient of the Lagrangian with respect to d must be zero at the optimum: $\nabla_d L = -\frac{1}{k}\sum_{i=1}^k \frac{\omega_i}{(g_i^\top d)^2} g_i + 2\lambda d = 0$. This equation describes the balance between the gradient contributions from each task and the regularization term from the norm constraint.
Given that $\mu_i = 0$, the stationarity condition simplifies to:

$$\sum_{i=1}^k \frac{\omega_i}{(g_i^\top d)^2} g_i = 2k\lambda d. \tag{6}$$

Following previous works [4, 30], we similarly assume that the gradients of tasks are linearly independent otherwise it would imply reaching a Pareto stationary point. Hence, d can be represented as a linear combination of task gradients: $d = \sum_{i=1}^k \alpha_i g_i$. Ignoring the parameter $2k\lambda$ in Eq. 6, which can be adjusted by the step size η_t, we obtain $\alpha_i = \frac{\omega_i}{(g_i^\top d)^2}$, i.e., $(g_i^\top d)^2 = \frac{\omega_i}{\alpha_i}$.

Let $\boldsymbol{\mathcal{G}} = (g_1, g_2, \ldots, g_k) \in \mathbb{R}^{n\times k}$ denote the matrix of task gradients. Then, we can express this in matrix form as:

$$(\boldsymbol{\mathcal{G}}^\top \boldsymbol{\mathcal{G}} \boldsymbol{\alpha})^2 = \frac{\boldsymbol{\omega}}{\boldsymbol{\alpha}}, \tag{7}$$

where the square operation is element-wise. We treat Eq. 7 as a simple constrained nonlinear least squares problem, which can be efficiently solved using the `scipy` library. Our complete algorithmic procedure is summarized in Algorithm 1. Note that for certain benchmarks lacking a validation set, to ensure consistency with other methods on the dataset, we use $\{\Delta m_i\}_{i=1}^k$ from the training set to obtain and update the $\boldsymbol{\omega}$. To avoid incorporating test set information into the training process, we do not use $\{\Delta m_i\}_{i=1}^k$ from the test set, even though it may be computed every epoch in previous works.

3.4. Theoretical Analysis

In this section, we present a theoretical analysis of our method about its convergence to a Pareto stationary point, where a convex combination of task gradients becomes zero. As previously noted, we assume that task gradients remain linearly independent until the system reaches a Pareto stationary point. Formally, we adopt the following assumption, similarly used by [30] and [4].

Assumption 1. *For the output sequence $\{\theta_t\}_{t=1}^\infty$ produced by the proposed method, the gradients of the tasks $g_{1,t}, g_{2,t}, \cdots, g_{k,t}$ remain linearly independent as long as the system has not reached a Pareto stationary point.*

Algorithm 1 PIVRG for MTL

```
1: Input: Model parameters $\theta_0$; Initial $\{\Delta m_i\}_{i=1}^k = \{0\}$;
   Learning rate $\{\eta_t\}$; Train and Val set $D_t, D_v$.
2: for $t = 1$ to $T-1$ do
3:     Compute gradients $\boldsymbol{\mathcal{G}}(\theta_t) = [g_{1,t}, \cdots, g_{k,t}]$ on $D_t$
4:     Obtain weights $\boldsymbol{\omega_t}$ by Eq. 5 based on $\{\Delta m_i\}_{i=1}^k$
5:     Solve Eq. 7 to obtain $\boldsymbol{\alpha_t}$
6:     Compute $d_t = \boldsymbol{\mathcal{G}}(\theta_t)\boldsymbol{\alpha_t}$
7:     Update the parameters $\theta_{t+1} = \theta_t - \eta_t d_t$
8:     Evaluate and update $\{\Delta m_i\}_{i=1}^k$ on $D_v$
9: end for
```

In practice, this assumption generally holds during the optimization process, as the number of tasks k is often much smaller than the dimension n of the shared parameters θ. The following assumption imposes differentiability and Lipschitz continuity on the loss functions, as also adopted by previous works [4, 22, 30].

Assumption 2. *For each task, the loss function $\ell_i(\theta)$ is differentiable and L-smooth such that $\|\nabla\ell_i(\theta_1) - \nabla\ell_i(\theta_2)\| \leq L\|\theta_1 - \theta_2\|$ for any θ_1 and θ_2.*

Then, we can obtain the following convergence theorem:

Theorem 1. *Suppose Assumptions 1 and 2 hold. We set the stepsize $\eta_t = \frac{\sum_i \sqrt{\omega_{i,t}/\alpha_{i,t}}}{kL\sum_i \sqrt{\omega_{i,t}\alpha_{i,t}}}$. Then, the sequence $\{\theta_t\}_{t=1}^\infty$ has a subsequence that converges to a Pareto stationary point θ^*.*

The detailed proof can be found in Sec.6.3 in the Appendix. Our main idea is to show that the smallest singular value of $\boldsymbol{\mathcal{G}}^\top\boldsymbol{\mathcal{G}}$ gradually approaches zero as the number of optimization steps t increases, thereby leading to the eventual convergence to a Pareto stationary point, where the gradients become linearly dependent.

4. Experiments

4.1. Protocols

We evaluate the proposed PIVRG on a variety of multi-task learning problems under both supervised learning and reinforcement learning settings to demonstrate its effectiveness. For multi-task supervised learning, we validate on the Scene Understanding benchmarks NYUv2 [35] and CityScapes [12], regression tasks from QM9 [6], and image-level classification with the CelebA [27] dataset. For multi-task reinforcement learning, we conduct experiments on the MT10 environment from the Meta-World benchmark [40]. Additionally, the ablation study demonstrates performance-level

Table 1. Results on NYU-v2 (3-task) dataset. Each experiment is repeated 3 times with different random seeds and the average is reported. The detailed standard error is reported in the Appendix.

Method	Segmentation		Depth		Surface Normal					MR $\downarrow$	$\Delta m(\%)\downarrow$
	mIoU $\uparrow$	Pix Acc $\uparrow$	Abs Err $\downarrow$	Rel Err $\downarrow$	Angle Distance $\downarrow$		Within t° $\uparrow$				
					Mean	Median	11.25	22.5	30		
STL	38.30	63.76	0.6754	0.2780	25.01	19.21	30.14	57.20	69.15		
LS	39.29	65.33	0.5493	0.2263	28.15	23.96	22.09	47.50	61.08	11.78	5.59
SI	38.45	64.27	0.5354	0.2201	27.60	23.37	22.53	48.57	62.32	10.22	4.39
RLW [20]	37.17	63.77	0.5759	0.2410	28.27	24.18	22.26	47.05	60.62	14.22	7.78
DWA [26]	39.11	65.31	0.5510	0.2285	27.61	23.18	24.17	50.18	62.39	10.67	3.57
UW [18]	36.87	63.17	0.5446	0.2260	27.04	22.61	23.54	49.05	63.65	10.33	4.05
MGDA [33]	30.47	59.90	0.6070	0.2555	24.88	19.45	29.18	56.88	69.36	8.11	1.38
PCGrad [39]	38.06	64.64	0.5550	0.2325	27.41	22.80	23.86	49.83	63.14	10.89	3.97
GradDrop [10]	39.39	65.12	0.5455	0.2279	27.48	22.96	23.38	49.44	62.87	9.89	3.58
CAGrad [22]	39.79	65.49	0.5486	0.2250	26.31	21.58	25.61	52.36	65.58	6.89	0.20
IMTL-G [24]	39.35	65.60	0.5426	0.2256	26.02	21.19	26.20	53.13	66.24	6.11	-0.76
MoCo [14]	**40.30**	**66.07**	0.5575	**0.2135**	26.67	21.83	25.61	51.78	64.85	6.22	0.16
Nash-MTL [30]	40.13	65.93	**0.5261**	0.2171	25.26	20.08	28.40	55.47	68.15	3.67	-4.04
FAMO[23]	38.88	64.90	0.5474	0.2194	25.06	19.57	29.21	56.61	68.98	5.33	-4.10
FairGrad [4]	39.74	66.01	0.5377	0.2236	24.84	19.60	29.26	56.58	69.16	3.44	-4.66
PIVRG	39.90	65.74	0.5365	0.2243	**24.30**	**18.80**	**30.95**	**58.26**	**70.38**	**2.33**	**-6.50**

information in PIVRG can be integrated into existing methods to significantly improve their performance. Note that for the QM9 and CelebA benchmarks, which already have predefined validation sets, we use $\{\Delta m_i\}_{i=1}^k$ from the validation set to update $\boldsymbol{w}$. For the NYUv2 and CityScapes benchmarks, which lack validation sets, we use $\{\Delta m_i\}_{i=1}^k$ from the training set to update $\boldsymbol{w}$ to maintain consistency with other methods on the dataset. Moreover, we visualize the optimization process of PIVRG on a 2-task toy example [22] in Fig.2 in the Appendix.

Baselines: We compare our proposed PIVRG described in Section 3 with the following methods in our experiments: Single-task learning (STL), Linear Scalarization (LS), Scale-Invariant (SI), Dynamic Weight Average (DWA) [26], Uncertainty Weighting (UW) [18], MGDA [33], RLW [20], PCGrad [39], GradDrop [10], CAGrad [22], IMTL-G [24], Nash-MTL [30], FAMO [23] and FairGrad [4].

Evaluation Metrics: Given that MTL does not inherently have a single objective and that metrics can vary across tasks, we follow previous works and focus on two overall performance metrics: (1) Δm, the average **per-metric performance drop** of method m relative to the STL baseline, which differs form $\{\Delta m_i\}_{i=1}^k$ and is defined as

$$\Delta m = \frac{1}{\mathcal{K}}\sum_{i=1}^{\mathcal{K}}(-1)^{\delta_i}(M_{m,i} - M_{b,i})/M_{b,i} \times 100,$$

where $\mathcal{K}$ is the number of total metrics, i.e. $\mathcal{K} \geq k$. $M_{m,i}$, $M_{b,i}$ and δ_i are consistent with their previous definitions in Section 3.2. (2) Mean Rank (MR): The average rank of each method across tasks (lower is better). A method achieves the best MR of 1 if it ranks first in all tasks.

4.2. Multi-Task Supervised Learning

Scene Understanding. Following previous works [4, 23, 30], we evaluate PIVRG on the NYUv2 and CityScapes datasets. NYUv2 [35] contains 1449 densely annotated indoor images, with three pixel-level tasks: 13-class semantic segmentation, depth estimation, and surface normal prediction. CityScapes [12] is a similar dataset containing 5000 street-view images with two tasks: semantic segmentation and depth estimation. These scenarios test the effectiveness of MTL in complex, pixel-level predictions. We follow the setup in [23, 30] using MTAN [24], which adds task-specific attention modules on top of SegNet [3]. To align with previous works, the model is trained for 200 epochs with a learning rate of 10^{-4} for the first 100 epochs, decaying by half for the remaining epochs.

The results in Table 1 and Table 2 demonstrate the remarkable performance of our method. On the NYUv2 dataset, previous methods typically outperform the STL baseline on segmentation and depth estimation tasks but fail to surpass STL on the surface normal prediction task, indicating a task imbalance. In contrast, our method is the only one that consistently outperforms STL across all 3 tasks and 9 evaluation metrics, and achieves an impressive average rank of **2.33** and the best performance drop of **-6.50%**.

In the CityScapes dataset, prior methods often exhibit better optimization on the segmentation task while underperforming on the depth estimation task. In contrast, PIVRG achieves more balanced results and is the only

Table 2. Results on CityScapes (2-task). Each experiment is repeated 3 times by different random seeds with average reported. The detailed standard error is reported in Appendix.

Method	CityScapes					
	Segmentation		Depth		MR ↓	$\Delta m(\%)$ ↓
	mIoU ↑	Pix Acc ↑	Abs Err ↓	Rel Err ↓		
STL	74.01	93.16	0.0125	27.77		
LS	75.18	93.49	0.0155	46.77	8.75	22.60
SI	70.95	91.73	0.0161	33.83	11.25	14.11
RLW [20]	74.57	93.41	0.0158	47.79	11.25	24.38
DWA [26]	75.24	93.52	0.0160	44.37	8.50	21.45
UW [18]	72.02	92.85	0.0140	30.13	7.75	5.89
MGDA [33]	68.84	91.54	0.0309	33.50	11.75	44.14
PCGrad [39]	75.13	93.48	0.0154	42.07	9.00	18.29
GradDrop [10]	75.27	93.53	0.0157	47.54	8.00	23.73
CAGrad [22]	75.16	93.48	0.0141	37.60	7.75	11.64
IMTL-G [24]	75.33	93.49	0.0135	38.41	6.00	11.10
Nash-MTL [30]	75.41	93.66	0.0129	35.02	3.50	6.82
FAMO [23]	74.54	93.29	0.0145	32.59	8.25	8.13
FairGrad [4]	75.72	93.68	0.0134	32.25	2.25	5.18
PIVRG	**75.82**	93.65	**0.0126**	**27.87**	**1.50**	**-0.54**

Table 3. Results on CelebA (40-task) and QM9 (11-task) datasets. Each experiment is repeated 3 times with different random seeds and the average is reported. The detailed standard error is reported in the Appendix.

Method	CelebA		QM9	
	MR ↓	$\Delta m(\%)$ ↓	MR ↓	$\Delta m(\%)$ ↓
LS	7.20	4.15	9.09	177.6
SI	8.88	7.20	5.55	77.8
RLW [20]	6.35	1.46	10.64	203.8
DWA [26]	7.92	3.20	8.82	175.3
UW [18]	6.62	3.23	7.27	108.0
MGDA [33]	12.03	14.85	8.91	120.5
PCGrad [39]	7.65	3.17	7.27	125.7
CAGrad [22]	7.33	2.48	8.18	112.8
IMTL-G [24]	5.70	0.84	7.18	77.2
Nash-MTL [30]	6.05	2.84	4.36	62.0
FAMO [23]	5.75	1.21	5.73	58.5
FairGrad [4]	6.38	0.37	4.73	57.9
PIVRG	**3.15**	**-0.96**	**3.00**	**33.6**

method to achieve a negative Δm on this benchmark. This further highlights both the potential of MTL and the superiority of PIVRG. In Sec. 7.3, we also show that our method not only achieves SOTA performance on NYUv2 and CityScapes but also produces the lowest performance variance, indicating a fairer optimization.

Image-Level Classification. CelebA [27] is a large-scale facial attributes dataset containing over 200K images, annotated with 40 attributes such as smiling, wavy hair, and mustache. This scenario represents a 40-task MTL classification problem, where each task predicts a binary attribute. We follow the setup in [23] and use a 9-layer convolutional neural network (CNN) as the backbone, with task-specific linear layers. The method is trained for 15 epochs using the Adam optimizer with a batch size of 256. The results are shown in Table 2. On this benchmark with as many as 40 tasks, PIVRG also shows state-of-the-art performance, achieving a negative Δm for the first time, validating the superiority of our approach.

Multi-Task Regression. QM9 [6] is a commonly used benchmark in graph neural networks, containing over 130K organic molecules represented as graphs. Each task predicts one of 11 molecular properties, which vary in scale. This setting evaluates the ability of MTL methods to balance task variations. Predicting molecular properties in the QM9 dataset presents a major challenge for MTL methods due to the large number of tasks and the substantial variation in loss scales. In our experiments, we train each method for 300 epochs and employ a learning rate scheduler to adjust the learning rate.

The results are presented in Figure 1 and Table 3. PIVRG achieves the best performance in terms of both MR and Δm. On the QM9 benchmark, where task difficulty is highly imbalanced, prior methods have struggled to optimize all tasks effectively, leading to a large overall Δm. By incorporating performance-level information and employing dynamic weight allocation to control variance, PIVRG reduces the average Δm by over 20%. Meanwhile, the results in Figure 1 also show that PIVRG achieves the smallest performance variance while obtaining the optimal Δm, further validating the effectiveness of the performance-informed dynamic weight allocation strategy. This also underscores the potential of MTL approaches and the distinct advantages of PIVRG in addressing task imbalance and achieving superior optimization across tasks.

Table 4. Results on MT10. Average over 10 random seeds.

Method	Success rate (mean ± stderr)
STL	0.90 ± 0.03
MTL SAC [40]	0.49 ± 0.07
MTL SAC + TE [40]	0.54 ± 0.05
MH SAC[40]	0.61 ± 0.04
PCGrad[39]	0.72 ± 0.02
CAGrad [22]	0.83 ± 0.05
MoCo [14]	0.75 ± 0.05
Nash-MTL[30]	0.91 ± 0.03
FAMO [23]	0.83 ± 0.05
FairGrad[4]	0.84 ± 0.07
PIVRG	$\mathbf{0.96} \pm 0.02$

4.3. Multi-Task Reinforcement Learning

We further evaluate our method on the MT10 benchmark, which includes 10 robotic manipulation tasks from the MetaWorld environment [40], where the objective is to learn a single policy that generalizes across various tasks such as pick and place, and opening doors. We follow the methodologies outlined in [23, 30] and adopt Soft Actor-Critic (SAC) [17] as the underlying algorithm. Our implementation utilizes the MTRL codebase used in [4, 30] and trains the model for 2 million steps with a batch size of 1280. We compare our proposed PIVRG with Multi-task

Table 5. Results of integrating our performance-informed weighting strategy into existing methods on the NYU-v2 (3-task) dataset. Each experiment is repeated 3 times with different random seeds and the average is reported.

METHOD	SEGMENTATION		DEPTH		SURFACE NORMAL					$\Delta m(\%) \downarrow$	VAR$[\Delta m_i] \downarrow$
	mIoU ↑	PIX ACC ↑	ABS ERR ↓	REL ERR ↓	ANGLE DISTANCE ↓		WITHIN $t°$ ↑				
					MEAN	MEDIAN	11.25	22.5	30		
LS	39.29	65.33	0.5493	0.2263	28.15	23.96	22.09	47.50	61.08	5.59	259.1
PI-LS	40.59	66.24	0.5330	0.2191	26.66	21.80	25.19	51.94	65.05	**-0.06**	**173.2**
RLW	37.17	63.77	0.5759	0.2410	28.27	24.18	22.26	47.05	60.62	7.78	205.3
PI-RLW	39.86	64.86	0.5744	0.2410	27.38	22.84	22.75	49.58	63.26	**4.52**	**170.5**
DWA	39.11	65.31	0.5510	0.2285	27.61	23.18	24.17	50.18	62.39	3.57	191.9
PI-DWA	40.55	66.31	0.5480	0.2261	26.63	21.97	25.03	51.42	64.67	**0.78**	**158.8**
UW	36.87	63.17	0.5446	0.2260	27.04	22.61	23.54	49.05	63.65	4.05	190.7
PI-UW	40.23	65.84	0.5182	0.2147	26.13	21.14	26.25	53.09	66.09	**-1.71**	**158.7**
MGDA	30.47	59.90	0.6070	0.2555	24.88	19.45	29.18	56.88	69.36	1.38	68.6
PI-MGDA	35.45	63.04	0.6025	0.2364	24.32	18.59	31.06	58.73	70.62	**-3.45**	**36.6**
NASH-MTL	40.13	65.93	0.5261	0.2171	25.26	20.08	28.40	55.47	68.15	-4.04	108.0
PI-NASH-MTL	42.14	66.83	0.5317	0.2259	24.79	19.46	29.46	56.93	69.30	**-5.77**	**70.7**

SAC (MTL SAC) [40], Multi-task SAC with task encoder (MTL SAC + TE) [40], Multi-headed SAC (MH SAC) [40], PCGrad [39], CAGrad [22], MoCo [14], Nash-MTL [30], FAMO [23] and FairGrad [4].

The results are shown in Table 4. Each method is evaluated every 10,000 steps, and the best average success rate over 10 random seeds throughout the entire training period is reported. In this context, we directly utilize the success rate to update $\boldsymbol{\omega}$. The results indicate that PIVRG achieves state-of-the-art performance on the MT10 benchmark, with an access rate approaching 100%.

4.4. Integrating Performance-Informed Weighting

Previous loss-based and gradient-based methods have often overlooked performance-level information, leading to a lack of clarity regarding task difficulty during the training process. We propose to integrate our performance-informed weighting strategy into these methods to enhance fairness in optimization. Specifically, for loss-based approaches, we adjust the initial loss $\mathcal{L} = (\ell_1, \ell_2, \cdots, \ell_k)$ using weights $\boldsymbol{\omega}$ to reflect the current optimization progress of different tasks, replacing $\mathcal{L}$ with $\mathcal{L}' = \boldsymbol{\omega} \odot \mathcal{L}$.

For gradient-based methods, since the motivation behind the aggregation algorithms varies, it is necessary to analyze each method individually to incorporate $\boldsymbol{\omega}$ into the design of the aggregation process. For instance, Nash-MTL maximizes the sum of log utilities, we thus replace the original equal summation $(1, 1, \cdots, 1)$ with a weighted sum $(\omega_1, \omega_2, \cdots, \omega_k)$.

We apply the performance-informed weighting strategy to a series of MTL methods, including LS, RLW [20], DWA [26], UW [18], MGDA [33], and Nash-MTL [30], and evaluate their performance on the NYUv2 benchmark. Table 5 shows that incorporating performance-level information and integrating dynamic weighting can bring significant performance improvements for these methods. Var$[\Delta m_i]$ is also reduced, which indicates a notable alleviation of task imbalance.

5. Conclusion, Limitations and Future Work

In this paper, we propose PIVRG, a new performance-informed variance reduction gradient aggregation approach. Building on the observation that previous loss-based and gradient-based methods exhibit common task imbalance across standard benchmarks, we point out the necessity of incorporating performance-level information to better represent fairness across tasks during the optimization process. Specifically, we use performance variance across tasks as a fairness indicator and introduce a dynamic weighting strategy aimed at gradually reducing this variance. Extensive experiments show that PIVRG achieves state-of-the-art performance across various benchmarks. The experimental results also show that incorporating our dynamic weighting strategy into existing loss-based and gradient-based methods not only significantly improves overall performance but also reduces performance variance across tasks, leading to a more balanced optimization process.

Limitations and Future Work. In this work, we regard performance variance across tasks as a fairness indicator and design a dynamic weighting strategy to progressively reduce this variance. However, there are numerous ways to incorporate performance-level information, and we would like to explore more effective fairness indicators in our future work. Additionally, our underlying optimization objective is not fixed, and future work may explore alternative designs and approaches to further enhance fairness and efficiency in multi-task learning.

References

[1] Idan Achituve, Haggai Maron, and Gal Chechik. Self-supervised learning for domain adaptation on point clouds. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 123–133, 2021. 1

[2] Idan Achituve, Idit Diamant, Arnon Netzer, Gal Chechik, and Ethan Fetaya. Bayesian uncertainty for gradient aggregation in multi-task learning. *arXiv preprint arXiv:2402.04005*, 2024. 2

[3] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017. 6

[4] Hao Ban and Kaiyi Ji. Fair resource allocation in multi-task learning. *arXiv preprint arXiv:2402.15638*, 2024. 2, 3, 5, 6, 7, 8

[5] Jonathan Baxter. A model of inductive bias learning. *Journal of artificial intelligence research*, 12:149–198, 2000. 1

[6] Lorenz C Blum and Jean-Louis Reymond. 970 million druglike small molecules for virtual screening in the chemical universe database gdb-13. *Journal of the American Chemical Society*, 131(25):8732–8733, 2009. 5, 7

[7] Rich Caruana. Multitask learning. *Machine learning*, 28: 41–75, 1997. 1

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

[9] 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*, pages 794–803. PMLR, 2018. 1, 2

[10] 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. *Advances in Neural Information Processing Systems*, 33:2039–2050, 2020. 2, 6, 7, 5

[11] Sumanth Chennupati, Ganesh Sistu, Senthil Yogamani, and Samir A Rawashdeh. Multinet++: Multi-stream feature aggregation and geometric loss strategy for multi-task learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 0–0, 2019. 2

[12] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. 5, 6

[13] Coline Devin, Abhishek Gupta, Trevor Darrell, Pieter Abbeel, and Sergey Levine. Learning modular neural network policies for multi-task and multi-robot transfer. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 2169–2176. IEEE, 2017. 1

[14] Heshan Fernando, Han Shen, Miao Liu, Subhajit Chaudhury, Keerthiram Murugesan, and Tianyi Chen. Mitigating gradient bias in multi-objective learning: A provably convergent approach. International Conference on Learning Representations, 2023. 6, 7, 8, 5

[15] Chris Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. Efficiently identifying task groupings for multi-task learning. *Advances in Neural Information Processing Systems*, 34:27503–27516, 2021. 7

[16] Michelle Guo, Albert Haque, De-An Huang, Serena Yeung, and Li Fei-Fei. Dynamic task prioritization for multitask learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 270–287, 2018. 1, 2

[17] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018. 7

[18] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7482–7491, 2018. 6, 7, 8, 5

[19] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7482–7491, 2018. 2

[20] Baijiong Lin, Feiyang Ye, and Yu Zhang. A closer look at loss weighting in multi-task learning. 2021. 6, 7, 8, 5

[21] Baijiong Lin, Feiyang Ye, Yu Zhang, and Ivor W Tsang. Reasonable effectiveness of random weighting: A litmus test for multi-task learning. *arXiv preprint arXiv:2111.10603*, 2021. 2

[22] Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multi-task learning. *Advances in Neural Information Processing Systems*, 34:18878–18890, 2021. 2, 5, 6, 7, 8, 3

[23] Bo Liu, Yihao Feng, Peter Stone, and Qiang Liu. Famo: Fast adaptive multitask optimization. *Advances in Neural Information Processing Systems*, 36, 2024. 2, 6, 7, 8, 3, 5

[24] Liyang Liu, Yi Li, Zhanghui Kuang, J Xue, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. Towards impartial multi-task learning. iclr, 2021. 1, 2, 3, 6, 7, 5

[25] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Adversarial multi-task learning for text classification. *arXiv preprint arXiv:1704.05742*, 2017. 1

[26] Shikun Liu, Edward Johns, and Andrew J Davison. End-to-end multi-task learning with attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1871–1880, 2019. 1, 2, 6, 7, 8, 5

[27] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pages 3730–3738, 2015. 5, 7

[28] Keerthiram Murugesan and Jaime Carbonell. Self-paced multitask learning with shared knowledge. *arXiv preprint arXiv:1703.00977*, 2017. 2

[29] Aviv Navon, Idan Achituve, Haggai Maron, Gal Chechik, and Ethan Fetaya. Auxiliary learning by implicit differentiation. *arXiv preprint arXiv:2007.02693*, 2020. 1

[30] Aviv Navon, Aviv Shamsian, Idan Achituve, Haggai Maron, Kenji Kawaguchi, Gal Chechik, and Ethan Fetaya. Multi-task learning as a bargaining game. *arXiv preprint arXiv:2202.01017*, 2022. 2, 3, 5, 6, 7, 8

[31] Jonathan Pilault, Amine Elhattami, and Christopher Pal. Conditionally adaptive multi-task learning: Improving transfer learning in nlp using fewer parameters & less data. *arXiv preprint arXiv:2009.09139*, 2020. 1

[32] S Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017. 1

[33] Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. *Advances in neural information processing systems*, 31, 2018. 2, 3, 6, 7, 8, 5

[34] Dmitry Senushkin, Nikolay Patakin, Arseny Kuznetsov, and Anton Konushin. Independent component alignment for multi-task learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20083–20093, 2023. 1, 2, 3

[35] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part V 12*, pages 746–760. Springer, 2012. 2, 5, 6

[36] 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*, pages 9120–9132. PMLR, 2020. 1, 7

[37] Caiming Xiong, SHU Tianmin, and Richard Socher. Hierarchical and interpretable skill acquisition in multi-task reinforcement learning, 2023. US Patent 11,562,287. 1

[38] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33:5824–5836, 2020. 2, 3, 5

[39] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33:5824–5836, 2020. 6, 7, 8

[40] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pages 1094–1100. PMLR, 2020. 5, 7, 8

[41] Hayoung Yun and Hanjoo Cho. Achievement-based training progress balancing for multi-task learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16935–16944, 2023. 2

[42] Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE transactions on knowledge and data engineering*, 34 (12):5586–5609, 2021. 1

[43] Ce Zheng, Wenhan Wu, Chen Chen, Taojiannan Yang, Sijie Zhu, Ju Shen, Nasser Kehtarnavaz, and Mubarak Shah. Deep learning-based human pose estimation: A survey. *ACM Computing Surveys*, 56(1):1–37, 2023. 1