ERCo: Evaluation-Guided Response Coordination for Multi-Task Learning

Anonymous authors
Paper under double-blind review

Abstract

Multi-task learning (MTL) exploits shared representations to learn multiple tasks jointly, but most MTL optimizers coordinate tasks through training losses or their gradients rather than the criteria used for final evaluation. Since losses and evaluation criteria can emphasize different error structures, improving or balancing task losses does not necessarily yield favorable evaluation trade-offs. To bridge this gap, we propose ERCo, an *Evaluation-Guided Response Coordination* framework that separates evaluation guidance from loss-based learning. ERCo aggregates normalized differentiable evaluation surrogates at the upper level while retaining dynamically weighted standard task losses at the lower level. At each iteration, an evaluation-guided primary model and a loss-only response model are updated under the same task weights. Their task-wise loss differences provide feedback for adapting task priorities without explicitly storing per-task parameter gradients. Extensive experiments demonstrate strong aggregate performance across diverse MTL benchmarks, with measured training efficiency comparable to simple loss-weighting baselines.

1 Introduction

Multi-task learning (MTL) jointly learns multiple related tasks within a shared model and has been widely adopted in dense visual prediction and scene understanding (Kokkinos, 2017; Kendall et al., 2018; Liu et al., 2019a), autonomous-driving perception (Teichmann et al., 2018), natural-language understanding (Liu et al., 2019b), and large-scale recommendation systems (Ma et al., 2018). By exploiting shared representations, MTL can improve data efficiency, facilitate knowledge transfer, and reduce the cost of training and deploying separate task-specific models. However, jointly optimizing heterogeneous tasks remains challenging because different tasks may have different loss scales, learning speeds, and preferred update directions, causing competition for shared model capacity and imbalanced final performance (Sener & Koltun, 2018; Yu et al., 2020a; Standley et al., 2020).

Most existing MTL optimizers address this challenge by adapting task-loss scalarization or modifying per-task gradients. Recent methods further characterize task imbalance through fair resource allocation, group-level alignment, performance variance, and hierarchical coordination (Ban & Ji, 2024; Shen et al., 2024; Qin et al., 2025b;a; Xiao et al., 2026). However, task coordination is still driven predominantly by training losses or gradients, whereas final models are assessed using task-specific evaluation criteria.

Specifically, training losses and evaluation criteria are two distinct measures. They may differ in how they rank predictions and in how much importance they assign to different errors. For example, for two samples with ground-truth depths 1 and 10, absolute-error vectors $(1, 0)$ and $(0, 1)$ both have mean absolute error 0.5, but their mean relative errors are 0.5 and 0.05, respectively. Thus, even within one task, equal loss values can conceal substantially different evaluation outcomes. Single-task studies address this loss–metric mismatch through optimization of non-decomposable measures, metric-specific surrogates, or validation-driven loss adaptation (Narasimhan et al., 2015; Eban et al., 2017; Berman et al., 2018; Huang et al., 2019; Li et al., 2021). Deep multi-objective learning also shows that variables controlling a lower-level scalarization can be optimized according to an upper-level measure of solution quality (Chen & Kwok, 2022). These lines of work motivate evaluation-aware task weighting, but leave open how to coordinate evaluation information with the standard losses needed for scalable multi-task training.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

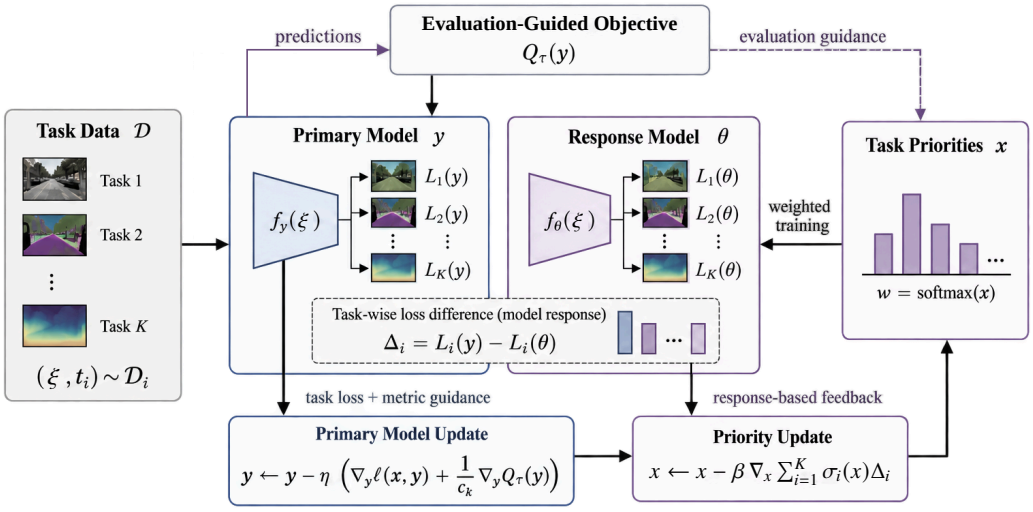


Figure 1: Overview of ERCo. Differentiable evaluation surrogates guide the primary model, while the response model follows the current weighted task losses without direct evaluation guidance. Their task-wise loss differences update the task priorities and determine subsequent loss-based training.

To address this problem, we propose **ERCo**, an *Evaluation-Guided Response Coordination* framework. ERCo formulates task weighting as a bilevel problem: the upper level evaluates the model using normalized differentiable surrogates of the reported criteria, while the lower level trains the model using dynamically weighted standard task losses. The upper objective uses a log-sum-exp smooth maximum, related to smooth Tchebycheff scalarization (Lin et al., 2024), to emphasize current evaluation bottlenecks while retaining all selected criteria. The remaining challenge is to translate this model-level guidance into task-wise weight updates. As shown in Figure 1, ERCo pairs an evaluation-guided primary model with a loss-only response model under the same task weights. Their task-wise loss differences provide function-value feedback for adapting the priorities, thereby coupling evaluation guidance to subsequent loss-based learning. The procedure uses aggregated first-order backward passes and avoids explicitly materializing the full set of per-task parameter gradients. Extensive experiments demonstrate strong aggregate performance and favorable computational efficiency.

Our contributions are summarized as follows:

- **Evaluation-guided MTL formulation.** We separate the reported evaluation criteria from the standard losses used for parameter learning. Differentiable metric surrogates define a unified upper-level objective, while the lower level retains dynamically weighted task-loss training. The same guidance term can also augment existing loss-based weighting objectives.
- **Response-mediated task coordination.** We pair an evaluation-guided primary model with a loss-only response model and use their task-wise function-value differences to adapt task priorities through aggregated first-order updates, without explicitly storing or manipulating per-task parameter gradients.
- **Extensive empirical evaluation.** Extensive experiments across diverse multi-task benchmarks demonstrate strong aggregate performance and measured training efficiency comparable to simple loss-weighting baselines.

2 Related Work

Multi-task learning. Existing MTL optimization methods mainly coordinate tasks using either loss values or task gradients. Loss-oriented approaches adapt task-loss scalarization through uncertainty weighting (Kendall et al., 2018), GradNorm (Chen et al., 2018), DWA (Liu et al., 2019a), RLW (Lin et al., 2022), FAMO (Liu et al., 2023), or task-group alignment (Shen et al., 2024). Gradient-oriented

approaches instead explicitly manipulate individual task gradients, including MGDA (Sener & Koltun, 2018), PCGrad (Yu et al., 2020a), GradDrop (Chen et al., 2020), CAGrad (Liu et al., 2021a), IMTL-G (Liu et al., 2021b), MoCo (Fernando et al., 2023), Nash-MTL (Navon et al., 2022), and Aligned-MTL (Senushkin et al., 2023).

Loss–metric mismatch. Standard losses are often chosen for decomposability and stable optimization, whereas final models are selected using task-specific or non-decomposable evaluation criteria. This gap has motivated scalable optimization of F-measures and related objectives (Narasimhan et al., 2015; Eban et al., 2017), the Lovász–Softmax surrogate for IoU (Berman et al., 2018), adaptive loss alignment using validation metric feedback (Huang et al., 2019), and automatic search for segmentation metric surrogates (Li et al., 2021). In MTL, structured hinge surrogates have also been combined with task-relatedness regularization to optimize evaluation measures (Ke et al., 2018). These studies show that evaluation criteria can shape learning through direct metric optimization, differentiable surrogates, or metric-driven loss adaptation, although most focus on a single task or incorporate metric information directly into the training objective.

Performance-aware coordination. Dynamic Task Prioritization (Guo et al., 2018) adapts task priorities according to performance-based difficulty, Auto-λ (Liu et al., 2022) learns task relationships using a higher-level meta-objective, and PIVRG (Qin et al., 2025b) uses performance variance to reduce task imbalance. In multi-objective learning, adaptive reference vectors (Chen & Kwok, 2022) and smooth Tchebycheff scalarization (Lin et al., 2024) use preference or solution-quality information to control scalarization. These methods illustrate that task priorities and scalarization variables can be adapted from performance feedback or explicit preferences, rather than only from instantaneous task-loss magnitudes.

Bilevel optimization for MTL. Bilevel optimization provides a hierarchical formulation for separating task coordination from model learning. FairGrad (Ban & Ji, 2024) formulates fair resource allocation through bilevel optimization, ConsMTL (Qin et al., 2025a) coordinates shared and task-specific learning hierarchically, and WC-MHGD (Zhang et al., 2026) studies multi-objective bilevel optimization. LDC-MTL (Xiao et al., 2026) uses an upper-level loss-discrepancy objective and a lower-level dynamically weighted training problem. General bilevel problems are commonly addressed through unrolling, implicit differentiation, or first-order reformulations; Moreau-envelope methods (Liu et al., 2024) provide a single-loop alternative by tracking an auxiliary lower-level response.

3 Evaluation-Guided Response Coordination for MTL

3.1 Learning with Losses and Coordinating with Evaluation Criteria

Consider a multi-task model with parameters y and task losses $\{\mathcal{L}_i(y)\}_{i=1}^K$. Following the multi-objective optimization (MOO) view of MTL (Sener & Koltun, 2018), the learning problem is

$$\min_y \ \boldsymbol{\mathcal{L}}(y) := \big(\mathcal{L}_1(y), \ldots, \mathcal{L}_K(y)\big).$$

A common training strategy uses a positive weighted scalarization,

$$\ell(x, y) := \sum_{i=1}^K \sigma_i(x)\mathcal{L}_i(y), \qquad \sigma_i(x) := \frac{\exp(x_i)}{\sum_{j=1}^K \exp(x_j)}, \tag{1}$$

where $x \in \mathbb{R}^K$ denotes the task-priority logits and $\sigma(x) \in \Delta_K$. The priorities determine how strongly each task contributes to joint parameter learning.

Standard task losses provide stable optimization signals, but they need not reflect the criteria used to select the final model. Conversely, differentiability alone does not ensure that evaluation surrogates are effective standalone training objectives. We therefore retain the weighted objective in (1) for parameter learning and introduce a separate differentiable objective for evaluation-aware coordination.

Evaluation-guidance objective. Let $\mathcal{E}_r$ be an evaluation criterion and $\widehat{\mathcal{E}}_r(y)$ a differentiable surrogate. We orient every criterion as a smaller-is-better badness,

$$\mathcal{B}_r(y) := \begin{cases} 1 - \widehat{\mathcal{E}}_r(y), & \mathcal{E}_r \text{ is higher-is-better,} \\ \widehat{\mathcal{E}}_r(y), & \mathcal{E}_r \text{ is lower-is-better.} \end{cases} \tag{2}$$

A task may contribute one or more criteria, so the number of criteria R can differ from the number of tasks K. To account for heterogeneous scales, we normalize each badness by a dataset-specific calibration value and denote the result by $z_r(y)$; the dataset-specific surrogates and calibration procedure are given in Appendix A.

Under a uniform preference over the selected criteria, we use log-sum-exp as a smooth approximation of their largest normalized badness,

$$Q_\tau(y) := \tau \log \left(\frac{1}{R} \sum_{r=1}^{R} \exp\left(\frac{z_r(y)}{\tau} \right) \right), \qquad \tau > 0. \tag{3}$$

This form is closely related to smooth Tchebycheff scalarization in MOO (Lin et al., 2024). Its gradient is

$$\nabla_y Q_\tau(y) = \sum_{r=1}^{R} \pi_r(y) \nabla_y z_r(y), \qquad \pi_r(y) = \frac{\exp(z_r(y)/\tau)}{\sum_{q=1}^{R} \exp(z_q(y)/\tau)}.$$

Hence, criteria with larger normalized badness receive greater emphasis, while all selected criteria retain a nonzero contribution.

Combining learned task scalarization with evaluation guidance gives

$$\min_{x \in \mathbb{R}^K,\, y} \; Q_\tau(y) \quad \text{s.t.} \quad y \in \arg\min_{\widetilde{y}} \ell(x, \widetilde{y}). \tag{4}$$

While Eq. (4) provides the bilevel modeling principle, solving the lower-level problem to convergence at every update of x would be computationally prohibitive. We therefore use a single-loop first-order procedure based on paired model responses. At each iteration, the primary model y is updated using both the weighted task loss and the evaluation guidance $Q_\tau(y)$, while a loss-only branch provides a response reference. The remaining question is how to translate this evaluation guidance into a task-wise signal for updating x within the same training loop.

3.2 Loss-Only Response Tracking

A local weighting principle. Let s denote a common model state and define the weighted-loss direction

$$g_w(s; x) := \nabla_s \ell(x, s) = \sum_{i=1}^{K} \sigma_i(x) \nabla_s \mathcal{L}_i(s).$$

After one small weighted-loss step, a first-order expansion gives

$$Q_\tau\big(s - \eta g_w(s; x)\big) = Q_\tau(s) - \eta \sum_{i=1}^{K} \sigma_i(x) \left\langle \nabla_s Q_\tau(s), \nabla_s \mathcal{L}_i(s) \right\rangle + O(\eta^2). \tag{5}$$

Thus, locally, a useful scalarization assigns greater weight to task-loss directions that support the evaluation-guidance direction. Computing the inner products in (5) from explicitly formed task gradients entails K per-task gradient computations.

We instead estimate this interaction from paired model responses. Starting from the same anchor s, consider a loss-only response and an evaluation-guided response,

$$\theta^+ = s - \eta g_w(s; x), \qquad y^+ = s - \eta\big(g_w(s; x) + \rho \nabla_s Q_\tau(s)\big),$$

where $\rho > 0$ controls the evaluation-guidance strength. For task i, define the response discrepancy $\Delta_i := \mathcal{L}_i(y^+) - \mathcal{L}_i(\theta^+)$. A first-order expansion around s yields

$$\Delta_i = -\eta\rho \left\langle \nabla_s \mathcal{L}_i(s), \nabla_s Q_\tau(s) \right\rangle + O(\eta^2). \tag{6}$$

A negative discrepancy means that task i has lower loss after the evaluation-guided step than after the loss-only step; a positive discrepancy indicates local tension between the two signals. Importantly, these quantities are obtained from task-wise function values at two model states rather than from K stored parameter-gradient vectors. Equation (6) provides a local matched-step interpretation that motivates the response comparison below.

3.3 RESPONSE-MEDIATED TASK COORDINATION

ERCo maintains a primary model y, which receives both task losses and evaluation guidance, and a response model θ, which receives only the current weighted task losses. We periodically synchronize the response parameters with the current primary parameters, $\theta^n \leftarrow y^n$. Both branches are updated at each iteration using the same mini-batch and task weights. Synchronization details are given in Appendix B.2.

Loss-only response. For the current priorities x^n, the response branch is updated by

$$\theta^{n+1} = \theta^n - \eta_n \nabla_\theta \ell(x^n, \theta^n). \tag{7}$$

This branch records how the present weighted task losses train the model without directly optimizing Q_τ.

Evaluation-guided primary model. The primary-model direction and update are

$$\begin{aligned} d_y^n &= \rho_n \nabla_y Q_\tau(y^n) + \nabla_y \ell(x^n, y^n), \\ y^{n+1} &= y^n - \eta_n d_y^n, \end{aligned} \tag{8}$$

where $\rho_n := 1/c_n > 0$ controls the relative strength of evaluation guidance. The two branches use the same task weights but differ in whether they receive the evaluation-guidance signal.

Priority feedback. Once both branches have been updated, we evaluate their task losses on the same mini-batch and form the task-wise response discrepancies,

$$\Delta_i^n := \mathcal{L}_i(y^{n+1}) - \mathcal{L}_i(\theta^{n+1}), \qquad i = 1, \dots, K.$$

These discrepancies compare how evaluation-guided and loss-only learning affect each task. With the updated model states held fixed, we then take a descent step on the weighted response discrepancy to update the priorities,

$$\begin{aligned} d_x^n &:= \nabla_x \sum_{i=1}^K \sigma_i(x^n) \Delta_i^n = \sum_{i=1}^K \nabla_x \sigma_i(x^n) \Delta_i^n, \\ x^{n+1} &= x^n - \beta_n d_x^n. \end{aligned} \tag{9}$$

For the softmax parameterization,

$$[d_x^n]_j = \sigma_j(x^n) \left(\Delta_j^n - \overline{\Delta}^n \right), \qquad \overline{\Delta}^n := \sum_{i=1}^K \sigma_i(x^n) \Delta_i^n. \tag{10}$$

The descent step increases logits for tasks whose discrepancies lie below the current weighted average and decreases those above it. The resulting task weights then guide the next model update.

3.4 TRAINING PROCEDURE AND COMPLEXITY

Algorithm 1 summarizes the complete procedure. At each iteration, ERCo computes both responses and uses their task-wise loss differences to update the priorities for the next iteration.

The resulting loop is $Q_\tau \to y \to \Delta \to x \to \sigma(x) \to \ell$: evaluation criteria shape the primary model, paired responses expose task-wise compatibility, and the updated priorities determine subsequent loss-based learning.

Relation to MEHA. At the level of update structure, ERCo is related to the auxiliary-response strategy of MEHA (Liu et al., 2024), which is derived from a Moreau-envelope reformulation with proximal coupling. Our formulation instead pairs loss-only and evaluation-guided responses with periodic synchronization, and updates task priorities from their task-wise function-value contrast. The resulting algorithm is specialized to metric-guided MTL rather than presented as a new general-purpose bilevel solver.

Algorithm 1 Evaluation-Guided Response Coordination for MTL (ERCo)

Initialize task logits $x^0 = \mathbf{0}$, primary parameters y^0, step sizes η_n, β_n, and guidance schedule ρ_n.
for $n = 0$ **to** $N - 1$ **do**
  If scheduled, synchronize the response anchor: $\theta^n \leftarrow y^n$.
  1: Update the loss-only response θ^{n+1} via (7).
  2: Update the evaluation-guided primary model y^{n+1} via (8).
  3: Compute $\{\Delta_i^n\}_{i=1}^K$ and update x^{n+1} via (9).
end for

Computation and state. ERCo uses a constant number of aggregated first-order backward passes with respect to the number of tasks. It maintains K task logits and one auxiliary model with its optimizer state. The task-dependent coordination signal consists of K scalar response discrepancies, avoiding explicit storage and manipulation of the full set of per-task parameter gradients.

4 Experiments

4.1 Experimental Setup

We evaluate ERCo in two multi-task learning settings. For *multi-task supervised learning*, we consider four benchmarks: CelebA (Liu et al., 2015), Cityscapes (Cordts et al., 2016), NYU-v2 (Silberman et al., 2012), and QM9 (Ramakrishnan et al., 2014), covering many-task image classification, dense scene understanding, and molecular property regression. For *multi-task reinforcement learning*, we evaluate on Meta-World MT10 (Yu et al., 2020b), which contains ten robotic manipulation tasks. For the supervised benchmarks, the upper objective includes surrogates of all reported task-level evaluation criteria, while MT10 uses a learned distance-based surrogate of task success. All experiments are implemented in PyTorch 2.0 on a server with an Intel Xeon Gold 5218R CPU, 256GB memory, and an NVIDIA A100 GPU with 40GB memory. Detailed architectures, task losses, evaluation surrogates, and training schedules are provided in the appendix.

Baselines. We compare with two families of MTL optimizers. *Gradient-oriented methods* include MGDA (Sener & Koltun, 2018), PCGrad (Yu et al., 2020a), GradDrop (Chen et al., 2020), CAGrad (Liu et al., 2021a), IMTL-G (Liu et al., 2021b), MoCo (Fernando et al., 2023), Nash-MTL (Navon et al., 2022), Aligned-MTL (Senushkin et al., 2023), FairGrad (Ban & Ji, 2024), ConsMTL (Qin et al., 2025a), and PIVRG (Qin et al., 2025b). *Loss-oriented methods* include Single-task learning (STL), Linear Scalarization (LS), Scale-Invariant (SI), RLW (Lin et al., 2022), DWA (Liu et al., 2019a), UW (Kendall et al., 2018), FAMO (Liu et al., 2023), GO4Align (Shen et al., 2024), and LDC-MTL (Xiao et al., 2026).

Metrics and protocol. Following prior MTL work (Navon et al., 2022; Ban & Ji, 2024; Qin et al., 2025a), we report two aggregate measures. (1) $\Delta m\%$, the average relative performance drop with respect to the STL baseline, is computed as $\Delta m\% = \frac{1}{S}\sum_{s=1}^{S}(-1)^{\delta_s}(\mathcal{M}_s - \mathcal{B}_s)/\mathcal{B}_s \times 100\%$, where S is the number of reported metrics and $\mathcal{M}_s$ and $\mathcal{B}_s$ denote the s-th metric of an MTL method and its STL baseline, respectively, and $\delta_s = 1$ for higher-is-better metrics and 0 otherwise. Lower $\Delta m\%$ indicates better overall performance. (2) Mean Rank (MR) averages the rank of each method across all reported metrics, with lower values indicating better performance. A method ranked first on every reported metric would therefore achieve an MR of 1.

4.2 Multi-Task Supervised Learning

CelebA: 40-task attribute classification. CelebA consists of 40 binary facial-attribute prediction tasks and provides a representative setting for evaluating task coordination at scale. As shown in Table 1, ERCo achieves the best $\Delta m\%$ of -4.78, improving on ConsMTL, the strongest competing method on this measure, by 3.36 percentage points. Its MR of 4.40 remains close to the best value of 4.18 from our reproduced LDC-MTL. The larger F1-based improvement with a near-best MR highlights the value of evaluation-aware coordination when all 40 tasks use binary cross-entropy (BCE) losses.

Table 1: Aggregate performance on CelebA, Cityscapes, and QM9, averaged over three seeds. Daggered CelebA results are reproduced by us. Best in **bold**, second best underlined.

	CelebA		Cityscapes		QM9	
Method	MR ↓	$\Delta m\%$ ↓	MR ↓	$\Delta m\%$ ↓	MR ↓	$\Delta m\%$ ↓
MGDA	16.00	14.85	15.00	44.14	12.55	120.5
PCGrad	11.10	3.18	10.75	18.29	10.36	125.7
CAGrad	10.65	2.48	9.50	11.64	11.55	112.8
IMTL-G	8.85	0.84	8.00	11.10	10.09	77.2
Nash-MTL	9.00	2.84	5.25	6.82	7.27	62.0
FairGrad	9.43	0.37	4.25	5.18	7.73	57.9
ConsMTL	5.48	-1.42	5.50	-0.59	**2.82**	**23.2**
PIVRG	5.38	-0.96	**2.75**	-0.54	5.36	33.6
LS	10.53	4.15	10.50	22.60	12.27	177.6
SI	12.60	7.20	14.25	14.11	8.64	77.8
RLW	9.55	1.46	13.00	24.38	13.82	203.8
DWA	11.43	3.20	10.25	21.45	12.00	175.3
UW	9.68	3.23	11.00	5.89	10.91	108.0
FAMO	8.85	1.21	10.75	8.13	8.82	58.5
GO4Align	5.93†	2.10†	12.00	8.11	7.82	52.7
LDC-MTL	**4.18**†	-1.08†	7.00	-0.57	5.91	45.8
ERCo	4.40	**-4.78**	**2.75**	**-8.51**	4.73	32.9

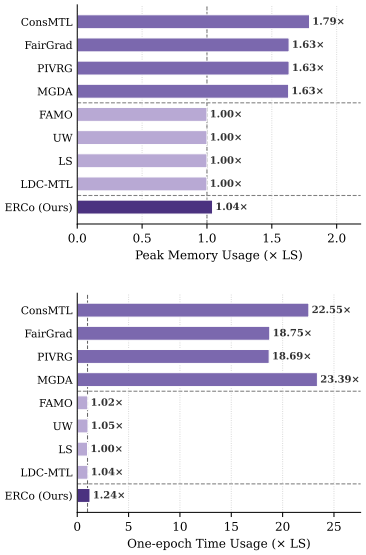


Figure 2: Peak memory usage and one-epoch runtime on CelebA (40 tasks), normalized by Linear Scalarization (LS).

Computational Efficiency on CelebA. Figure 2 compares peak memory and one-epoch runtime after normalization by LS. ERCo maintains memory usage and runtime comparable to LS and the measured loss-oriented methods, despite the additional paired response update. In contrast, the measured gradient-oriented methods incur substantially higher computational costs. Absolute epoch times and the measurement protocol are reported in Appendix D.5.

Dense prediction. We evaluate ERCo on NYU-v2 and Cityscapes, covering semantic segmentation, depth estimation, and surface-normal prediction. On NYU-v2, ERCo achieves the best aggregate performance and leads six of the nine reported criteria, with gains spanning both semantic and geometric predictions. On Cityscapes, ERCo also obtains the best aggregate performance, with its strongest improvement in relative depth error: ERCo achieves a Rel Err of 17.79, compared with 26.41 for ConsMTL, the strongest competing result on this metric. This 8.62 absolute reduction highlights the benefit of incorporating an evaluation preference that is not explicitly encoded by the absolute-error training loss. Complete Cityscapes results and a target-distribution analysis that helps interpret the RelErr gain are provided in Appendix D.1.

QM9: molecular property regression. QM9 provides a complementary setting in which property-wise MAE is closely related to the regression losses and the 11 targets have substantially different scales. ERCo nevertheless obtains the second-best aggregate performance; complete property-wise results are provided in Appendix Table 6.

4.3 Multi-Task Reinforcement Learning

Meta-World MT10 (Yu et al., 2020b) comprises ten robotic manipulation tasks and evaluates whether a single policy can learn multiple manipulation behaviors. Following prior multi-task reinforcement learning studies (Ban & Ji, 2024; Qin et al., 2025b), we use Soft Actor-Critic (SAC) (Haarnoja et al., 2018) as the underlying learner and use average episode success across the ten tasks as the evaluation metric.

In addition to the MTL optimizers introduced in Section 4.1, we compare with MTL SAC, MTL SAC with a task encoder (MTL SAC + TE), and multi-headed SAC (MH SAC) (Yu et al., 2020b). ERCo uses a distance-margin surrogate learned from training replay, while the paired response follows the standard weighted SAC actor losses. The surrogate construction is detailed in Appendix A.6.

Table 2: Performance comparison on NYU-v2 (3-task dense prediction). Results are averaged over three runs with different random seeds. Best in **bold**, second best underlined.

Method	Segmentation		Depth		Surface Normal					MR ↓	$\Delta m\%$ ↓
					Angle Distance ↓		Within t° ↑				
	mIoU ↑	Pix Acc ↑	Abs Err ↓	Rel Err ↓	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		
MGDA	30.47	59.90	0.6070	0.2555	24.88	19.45	29.18	56.88	69.36	12.56	1.38
PCGrad	38.06	64.64	0.5550	0.2325	27.41	22.80	23.86	49.83	63.14	15.78	3.97
GradDrop	39.39	65.12	0.5455	0.2279	27.48	22.96	23.38	49.44	62.87	14.44	3.58
CAGrad	39.79	65.49	0.5486	0.2250	26.31	21.58	25.61	52.36	65.58	11.00	0.20
IMTL-G	39.35	65.60	0.5426	0.2256	26.02	21.19	26.20	53.13	66.24	10.33	-0.60
MoCo	40.30	66.07	0.5575	**0.2135**	26.67	21.83	25.61	51.78	64.85	10.11	0.16
Nash-MTL	40.13	65.93	**0.5261**	0.2171	25.26	20.08	28.40	55.47	68.15	7.33	-4.04
Aligned-MTL	40.82	66.33	0.5300	0.2200	25.19	19.71	28.88	56.23	68.54	6.33	-4.93
FairGrad	39.74	66.01	0.5377	0.2236	24.84	19.60	29.26	56.58	69.16	6.78	-4.66
ConsMTL	40.33	65.32	0.5491	0.2151	24.35	18.80	31.06	58.28	70.31	4.56	-6.72
PIVRG	39.90	65.74	0.5365	0.2243	**24.30**	18.80	30.95	58.26	70.38	4.22	-6.50
LS	39.29	65.33	0.5493	0.2263	28.15	23.96	22.09	47.50	61.08	16.33	5.59
SI	38.45	64.27	0.5354	0.2201	27.60	23.37	22.53	48.57	62.32	14.56	4.39
RLW	37.17	63.77	0.5759	0.2410	28.27	24.18	22.26	47.05	60.62	19.22	7.78
DWA	39.11	65.31	0.5510	0.2285	27.61	23.18	24.17	50.18	62.39	15.44	3.57
UW	36.87	63.17	0.5446	0.2260	27.04	22.61	23.54	49.05	63.65	14.89	4.05
FAMO	38.88	64.90	0.5474	0.2194	25.06	19.57	29.21	56.61	68.98	9.00	-4.10
GO4Align	40.42	65.37	0.5492	0.2167	24.76	18.94	30.54	57.87	69.84	5.44	-6.08
LDC-MTL	38.04	65.92	0.5402	0.2278	24.70	19.19	29.97	57.44	69.69	7.44	-4.75
ERCo	**41.10**	**67.01**	0.5500	0.2249	24.76	**17.57**	**33.76**	**60.62**	**71.34**	**4.00**	**-8.97**

Table 3 reports the mean success rate and standard error over ten random seeds. Performance is evaluated every 10,000 environment steps, and we report the highest mean success rate achieved during training. ERCo matches the highest mean success in this comparison. The result extends response-based coordination beyond supervised prediction to policies guided by a surrogate learned from replay.

Overall, ERCo achieves the best $\Delta m\%$ on the three visual benchmarks, the second-best aggregate performance on QM9, and joint-best mean success on MT10. These results support evaluation information as a coordination signal for both supervised and reinforcement learning, while retaining standard task losses for parameter learning.

Table 3: MT10 success rate over ten runs (mean $\pm$ standard error). Best mean in **bold**.

Method	Success rate ↑
STL	0.90 ± 0.03
MTL SAC	0.49 ± 0.07
MTL SAC + TE	0.54 ± 0.05
MH SAC	0.61 ± 0.04
PCGrad	0.72 ± 0.02
CAGrad	0.83 ± 0.05
MoCo	0.75 ± 0.05
Nash-MTL	0.91 ± 0.03
FAMO	0.83 ± 0.05
FairGrad	0.84 ± 0.07
PIVRG	$\mathbf{0.96} \pm 0.02$
ERCo	$\mathbf{0.96} \pm 0.02$

4.4 Ablating Evaluation Guidance and Response Feedback

The local interpretation in (6) motivates using paired response differences to adapt task priorities. We further examine the empirical contribution of the two components involved in ERCo: direct evaluation guidance and response-based task coordination.

We consider three controlled variants. *Q-only* optimizes the evaluation objective without standard task losses. We further combine the same Q_τ with two weighted-loss schemes: *Q+LS* uses uniform weights, while *Q+FAMO* adopts the adaptive weights of FAMO (Liu et al., 2023):

$$\mathcal{J}_n(y) = \sum_{i=1}^{K} w_i^n \mathcal{L}_i(y) + \rho_n Q_\tau(y). \tag{11}$$

These variants test whether ERCo's gains can be explained by direct evaluation guidance with conventional loss weighting, without response-based coordination.

Table 4: Ablation results on CelebA and NYU-v2.

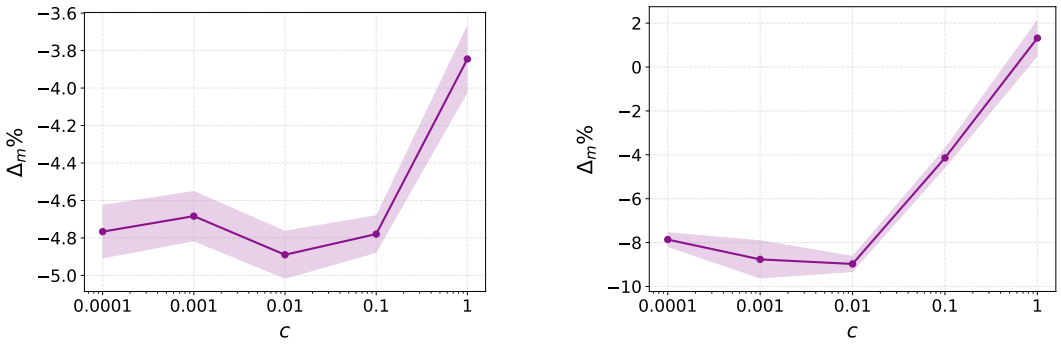
Method	Standard losses	Task coordination	CelebA $\Delta m\% \downarrow$	NYU-v2 $\Delta m\% \downarrow$
Q-only	No	–	-1.58	-0.76
Q+LS	Yes	Uniform	-1.66	-6.62
Q+FAMO	Yes	FAMO	-2.24	-7.64
ERCo	Yes	Response	**-4.78**	**-8.97**

As shown in Table 4, evaluation guidance is more effective when combined with standard task-loss learning than when used alone. Adding the same Q_τ to both fixed LS and adaptive FAMO weighting yields clear improvements, showing that the benefit of evaluation guidance is not specific to a particular weighting scheme. ERCo performs best overall, indicating that response-based coordination provides additional benefit beyond directly augmenting conventional weighted-loss training with evaluation guidance. Complete results are provided in Appendix Table 10.

4.5 Parameter Sensitivity

We examine the base coefficient c in $c_n = c\log(n+2)$, where $\rho_n = 1/c_n$ controls the strength of evaluation guidance in (8). Following the increasing-penalty motivation of MEHA (Liu et al., 2024), c_n grows gradually during training, so the relative contribution of Q_τ decreases over time while remaining active throughout optimization.

We vary c on CelebA and NYU-v2, spanning four orders of magnitude while keeping all other settings fixed. We use $c = 0.01$ for NYU-v2 and QM9, and $c = 0.1$ for CelebA and Cityscapes. CelebA is relatively insensitive over the range 10^{-4}–10^{-1}. Although $c = 0.01$ yields the lowest $\Delta m\%$ on CelebA, we select $c = 0.1$ because it gives a lower MR while still outperforming competing methods in $\Delta m\%$. These values are selected from the coarse grid above; finer tuning may further improve performance but is not explored here.



(a) CelebA (b) NYU-v2

Figure 3: Sensitivity to the base coefficient c on CelebA and NYU-v2. Results are averaged over three runs with different random seeds; shaded regions denote standard error.

5 Conclusion

We introduced ERCo, an evaluation-guided MTL framework that aggregates normalized differentiable evaluation surrogates at the upper level while retaining dynamically weighted standard task losses at the lower level. Task-wise loss differences between an evaluation-guided primary model and a loss-only response model provide feedback for adapting task priorities through first-order updates. Extensive experiments across supervised MTL and multi-task reinforcement learning demonstrate strong aggregate performance with favorable computational efficiency. Ablation results further support the benefit of response-based coordination. Future work will explore more general upper-level objectives that reduce criterion-specific design and handle heterogeneous scales more robustly.

AI USE STATEMENT

We made limited use of large language models (LLMs) during manuscript preparation. Specifically, LLMs were used to assist with writing and language polishing, check the logical flow and consistency of the manuscript, support the retrieval and discovery of related work, and help identify implementation or coding issues. All such use was conducted under author supervision, and all outputs were reviewed and verified by the authors. The technical content, mathematical formulations, method design, experimental design and execution, result interpretation, and conceptual contributions presented in this work were developed by the authors, who remain solely responsible for the content and conclusions of the paper.

ETHICS STATEMENT

This work raises no specific ethical concerns. Our study does not involve human-subject experiments, the collection of sensitive personal data, or real-world deployment. All experiments are conducted using established research benchmarks and standard experimental protocols. We have conducted this work in accordance with the ICLR Code of Ethics and relevant research standards.

REPRODUCIBILITY STATEMENT

To support reproducibility, we provide the source code in the supplementary materials. Detailed descriptions of our method, experimental settings, and additional results are included in the appendix. These materials are intended to facilitate independent verification and reproduction of our results.

REFERENCES

Hao Ban and Kaiyi Ji. Fair resource allocation in multi-task learning. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*. PMLR, 2024.

Maxim Berman, Amal Rannen Triki, and Matthew B. Blaschko. The Lovasz-Softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4413–4421, 2018.

Weiyu Chen and James T. Kwok. Multi-objective deep learning with adaptive reference vectors. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, 2022.

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*, pp. 794–803, Stockholm, Sweden, 2018. PMLR.

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.

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 (CVPR)*, pp. 3213–3223. IEEE, 2016.

Elad Eban, Mariano Schain, Alan Mackey, Ariel Gordon, Ryan Rifkin, and Gal Elidan. Scalable learning of non-decomposable objectives. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 832–840, 2017.

H. D. Fernando, H. Shen, M. Liu, S. Chaudhury, K. Murugesan, and T. Chen. Mitigating gradient bias in multi-objective learning: A provably convergent approach. In *The Eleventh International Conference on Learning Representations*, Kigali, Rwanda, 2023. OpenReview.net.

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)*, pp. 270–287, Munich, Germany, 2018. Springer.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870. PMLR, 2018.

Chen Huang, Shuangfei Zhai, Walter Talbott, Miguel Bautista Martin, Shih-Yu Sun, Carlos Guestrin, and Josh Susskind. Addressing the loss-metric mismatch with adaptive loss alignment. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2891–2900. PMLR, 2019.

Ge-Yang Ke, Yan Pan, Jian Yin, and Chang-Qin Huang. Optimizing evaluation metrics for multitask learning via the alternating direction method of multipliers. *IEEE Transactions on Cybernetics*, 48 (3):993–1006, 2018. doi: 10.1109/TCYB.2017.2670608.

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*, pp. 7482–7491, Salt Lake City, UT, USA, 2018. IEEE/CVF.

Iasonas Kokkinos. UberNet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6129–6138, 2017.

Hao Li, Chenxin Tao, Xizhou Zhu, Xiaogang Wang, Gao Huang, and Jifeng Dai. Auto Seg-Loss: Searching metric surrogates for semantic segmentation. In *International Conference on Learning Representations*, 2021.

Baijiong Lin, Feiyang Ye, Yu Zhang, and Ivor W. Tsang. Reasonable effectiveness of random weighting: A litmus test for multi-task learning. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856.

Xi Lin, Xiaoyuan Zhang, Zhiyuan Yang, Fei Liu, Zhenkun Wang, and Qingfu Zhang. Smooth tchebycheff scalarization for multi-objective optimization. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 30479–30509, 2024.

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, 2021a.

Bo Liu, Yihao Feng, Peter Stone, and Qiang Liu. Famo: Fast adaptive multitask optimization. *Advances in Neural Information Processing Systems*, 36:57226–57243, 2023.

Liyang Liu, Yi Li, Zhanghui Kuang, J. Xue, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. Towards impartial multi-task learning. In *International Conference on Learning Representations (ICLR)*, Virtual, 2021b. OpenReview.net.

Risheng Liu, Zhu Liu, Wei Yao, Shangzhi Zeng, and Jin Zhang. Moreau envelope for nonconvex bi-level optimization: A single-loop and hessian-free solution strategy. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 31566–31596, Vienna, Austria, 2024. PMLR.

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*, pp. 1871–1880, Long Beach, CA, USA, 2019a. IEEE/CVF.

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

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4487–4496, 2019b. doi: 10.18653/v1/P19-1441.

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 (ICCV)*, pp. 3730–3738. IEEE, 2015.

Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H. Chi. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1930–1939, 2018. doi: 10.1145/3219819.3220007.

Harikrishna Narasimhan, Purushottam Kar, and Prateek Jain. Optimizing non-decomposable performance measures: A tale of two classes. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 199–208, 2015.

Aviv Navon, Aviv Shamsian, Idan Achituve, Haggai Maron, Kenji Kawaguchi, Gal Chechik, and Ethan Fetaya. Multi-task learning as a bargaining game. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, volume 162, pp. 16428–16446. PMLR, 2022.

Xiaohan Qin, Xiaoxing Wang, and Junchi Yan. Towards consistent multi-task learning: Unlocking the potential of task-specific parameters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10067–10076, Nashville, TN, USA, 2025a. IEEE/CVF.

Xiaohan Qin, Xiaoxing Wang, and Junchi Yan. Revisiting fairness in multitask learning: A performance-driven approach for variance reduction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20492–20501, Nashville, TN, USA, 2025b. IEEE/CVF.

Raghunathan Ramakrishnan, Pavlo O. Dral, Matthias Rupp, and O. Anatole von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1:140022, 2014.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. *Advances in neural information processing systems*, 31, 2018.

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 (CVPR)*, pp. 20083–20093, 2023.

Jiayi Shen, Qi Wang, Zehao Xiao, Nanne Van Noord, and Marcel Worring. GO4Align: Group optimization for multi-task alignment. In *Advances in Neural Information Processing Systems*, volume 37, pp. 111382–111405, 2024.

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*, pp. 746–760, Florence, Italy, 2012. Springer.

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*, pp. 9120–9132, Virtual, 2020. PMLR.

Marvin Teichmann, Michael Weber, Marius Zoellner, Roberto Cipolla, and Raquel Urtasun. MultiNet: Real-time joint semantic reasoning for autonomous driving. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pp. 1013–1020, 2018. doi: 10.1109/IVS.2018.8500504.

Peiyao Xiao, Chaosheng Dong, Shaofeng Zou, and Kaiyi Ji. LDC-MTL: Balancing multi-task learning through scalable loss discrepancy control. In *Computer Vision – ECCV 2026*, volume 17031 of *Lecture Notes in Computer Science*, pp. 201–218. Springer, 2026. doi: 10.1007/978-3-032-37490-5_12.

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, 2020a.

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 *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 1094–1100. PMLR, 2020b.

Zhiyao Zhang, Zhuqing Liu, Xin Zhang, Wen-Yen Chen, Jiyan Yang, and Jia Liu. Multi-objective bilevel learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 28653–28661, 2026. doi: 10.1609/aaai.v40i34.40097.

A Differentiable Evaluation Surrogates

This section defines the reported evaluation criteria and their differentiable surrogates. Hard metrics follow the benchmark evaluation protocol, whereas surrogates are computed on the current mini-batch for training.

A.1 Common Construction

Each criterion is expressed as a smaller-is-better badness according to (2). We normalize each criterion by a positive calibration scale,

$$z_r(y) = \frac{\mathcal{B}_r(y)}{s_r}, \qquad r = 1, \dots, R, \tag{12}$$

and aggregate the normalized terms with Q_τ in (3), where R denotes the number of selected criteria.

For NYU-v2 and Cityscapes, the calibration scales s_r are computed from training data before optimization and then kept fixed. Specifically,

$$s_r = \frac{1}{|\mathcal{D}_{\mathrm{cal}}|} \sum_{\xi \in \mathcal{D}_{\mathrm{cal}}} \mathcal{B}_r(y^0; \xi), \tag{13}$$

where $\mathcal{D}_{\mathrm{cal}}$ denotes the training data used for calibration and y^0 is the initial model. For QM9, each scale is the training-target mean absolute deviation in the reported unit, computed before training and then kept fixed. For CelebA, all evaluation badness terms lie in $[0, 1]$, so we simply set $s_r = 1$. MT10 instead uses delayed calibration because its distance surrogate is learned online from replay; details are provided in Appendix A.6.

A.2 CelebA: Binary Attribute F1

CelebA contains $K = 40$ binary attribute-prediction tasks. For attribute i and example n, let a_{ni} be the model logit, $p_{ni} = \mathrm{sigmoid}(a_{ni})$ the predicted probability, and $t_{ni} \in \{0, 1\}$ the label. The hard prediction is $h_{ni} = \mathbf{1}[p_{ni} \geq 1/2]$. Over the evaluated examples, define

$$\mathrm{TP}_i = \sum_n h_{ni} t_{ni}, \qquad \mathrm{FP}_i = \sum_n h_{ni}(1 - t_{ni}), \qquad \mathrm{FN}_i = \sum_n (1 - h_{ni}) t_{ni}.$$

The reported F1 score is

$$\mathrm{F1}_i = \frac{2\mathrm{TP}_i}{2\mathrm{TP}_i + \mathrm{FP}_i + \mathrm{FN}_i},$$

with the benchmark convention applied when the denominator is zero.

To obtain a differentiable counterpart, we replace h_{ni} with p_{ni} and compute the soft counts on the current mini-batch:

$$\widetilde{\mathrm{TP}}_i = \sum_n p_{ni} t_{ni}, \qquad \widetilde{\mathrm{FP}}_i = \sum_n p_{ni}(1 - t_{ni}), \qquad \widetilde{\mathrm{FN}}_i = \sum_n (1 - p_{ni}) t_{ni}.$$

The soft F1 and its badness are

$$\widehat{\mathrm{F1}}_i = \frac{2\widetilde{\mathrm{TP}}_i}{2\widetilde{\mathrm{TP}}_i + \widetilde{\mathrm{FP}}_i + \widetilde{\mathrm{FN}}_i}, \qquad \mathcal{B}_i^{\mathrm{F1}} = 1 - \widehat{\mathrm{F1}}_i.$$

The soft counts reduce to the hard counts when predictions become binary. Each attribute contributes one upper-level term, giving $R = 40$. The lower level retains binary cross-entropy,

$$\mathcal{L}_i^{\mathrm{BCE}} = -\frac{1}{N} \sum_{n=1}^{N} \big[t_{ni} \log p_{ni} + (1 - t_{ni}) \log(1 - p_{ni}) \big],$$

where N is the mini-batch size. Unlike sample-wise BCE, soft F1 couples examples through attribute-level confusion statistics.

A.3 CITYSCAPES: SEGMENTATION AND DEPTH

Cityscapes reports mIoU and pixel accuracy for semantic segmentation, and absolute and relative error for depth estimation. ERCo includes all four criteria in its upper objective ($R = 4$).

Semantic segmentation. Let Ω be a nonempty set of valid pixels and $\mathcal{C}$ the class set. For pixel u, the class probability is $p_{u,c} = \exp(s_{u,c}) / \sum_{j \in \mathcal{C}} \exp(s_{u,j})$, where $s_{u,c}$ is the model logit. Let t_u be its target class, $y_{u,c} = \mathbf{1}[t_u = c]$ the one-hot label, $\widehat{t}_u = \arg\max_c p_{u,c}$ the hard prediction, and $\widehat{y}_{u,c} = \mathbf{1}[\widehat{t}_u = c]$. The class-wise counts are

$$\mathrm{TP}_c = \sum_{u \in \Omega} y_{u,c} \widehat{y}_{u,c}, \qquad \mathrm{FP}_c = \sum_{u \in \Omega} (1 - y_{u,c}) \widehat{y}_{u,c},$$
$$\mathrm{FN}_c = \sum_{u \in \Omega} y_{u,c} (1 - \widehat{y}_{u,c}).$$

The reported segmentation metrics are

$$\mathrm{IoU}_c = \frac{\mathrm{TP}_c}{\mathrm{TP}_c + \mathrm{FP}_c + \mathrm{FN}_c}, \qquad \mathrm{mIoU} = \frac{1}{|\mathcal{C}_{\mathrm{eval}}|} \sum_{c \in \mathcal{C}_{\mathrm{eval}}} \mathrm{IoU}_c,$$
$$\mathrm{PixAcc} = \frac{1}{|\Omega|} \sum_{u \in \Omega} \mathbf{1}[\widehat{t}_u = t_u],$$

where $\mathcal{C}_{\mathrm{eval}}$ denotes the classes included by the evaluation protocol. The discrete predictions prevent useful direct backpropagation through these metrics.

We replace $\widehat{y}_{u,c}$ with $p_{u,c}$ and define

$$I_c = \sum_{u \in \Omega} y_{u,c} p_{u,c}, \qquad P_c = \sum_{u \in \Omega} p_{u,c}, \qquad G_c = \sum_{u \in \Omega} y_{u,c}.$$

Here I_c is the soft intersection, P_c is the predicted probability mass, and G_c is the number of target pixels in class c. The soft false-positive and false-negative masses are $P_c - I_c$ and $G_c - I_c$, respectively, yielding union $U_c = P_c + G_c - I_c$. Thus the relaxation preserves the hard confusion-count structure. For $\mathcal{C}_b = \{c \in \mathcal{C} : U_c > 0\}$, we use

$$\widehat{\mathrm{mIoU}} = \frac{1}{|\mathcal{C}_b|} \sum_{c \in \mathcal{C}_b} \frac{I_c}{U_c}, \qquad \mathcal{B}_{\mathrm{mIoU}} = 1 - \widehat{\mathrm{mIoU}}.$$

The corresponding soft pixel accuracy is

$$\widehat{\mathrm{PixAcc}} = \frac{1}{|\Omega|} \sum_{u \in \Omega} p_{u,t_u}, \qquad \mathcal{B}_{\mathrm{PixAcc}} = 1 - \widehat{\mathrm{PixAcc}}.$$

The standard segmentation loss is $\mathcal{L}_{\mathrm{CE}} = -|\Omega|^{-1} \sum_{u \in \Omega} \log p_{u,t_u}$. Unlike pixel-wise CE, soft mIoU forms class-level intersection-over-union ratios, giving prediction-dependent feedback coupled across pixels.

Depth estimation. Let $\mathcal{V}_d$ be the valid depth pixels with nonzero target depth, $N_d = |\mathcal{V}_d| > 0$, and $\widehat{d}_u, d_u$ the predicted and target depths. The reported criteria are

$$\mathrm{AbsErr} = \frac{1}{N_d} \sum_{u \in \mathcal{V}_d} |\widehat{d}_u - d_u|, \tag{14}$$

$$\mathrm{RelErr} = \frac{1}{N_d} \sum_{u \in \mathcal{V}_d} \frac{|\widehat{d}_u - d_u|}{|d_u|}. \tag{15}$$

Both criteria are differentiable almost everywhere. Absolute error is used directly as $\mathcal{B}_{\mathrm{AbsErr}} = \mathrm{AbsErr}$. For Cityscapes, we limit the influence of extremely small inverse-depth targets through a denominator floor in the RelErr training surrogate:

$$\mathcal{B}^{\mathrm{clip}}_{\mathrm{RelErr}} = \frac{1}{N_d} \sum_{u \in \mathcal{V}_d} \frac{|\widehat{d}_u - d_u|}{\max(d_u, 0.005)}. \tag{16}$$

This changes only the upper-level surrogate; the standard task losses, evaluation mask, and reported RelErr remain unchanged. The label audit and its implications for the results are discussed in Appendix D.1. Together with $\mathcal{B}_{\mathrm{mIoU}}$ and $\mathcal{B}_{\mathrm{PixAcc}}$, these depth terms form the four-criterion Cityscapes upper objective.

A.4 NYU-v2: Dense-Prediction Criteria

NYU-v2 reports nine criteria across semantic segmentation, depth estimation, and surface-normal prediction. ERCo uses a differentiable badness term for each criterion, giving $R = 9$.

Segmentation and depth. The segmentation criteria and absolute depth error use the constructions in Appendix A.3. Relative depth error is used directly as $\mathcal{B}_{\mathrm{RelErr}} = \mathrm{RelErr}$ on NYU-v2, without the Cityscapes-specific denominator floor.

Surface-normal evaluation. Let $\mathcal{V}_n$ be the valid normal pixels, $N_n = |\mathcal{V}_n| > 0$, and $\widehat{\mathbf{n}}_u, \mathbf{n}_u \in \mathbb{R}^3$ the nonzero predicted and target normals. Define the cosine similarity and angular error in degrees as

$$c_u = \frac{\widehat{\mathbf{n}}_u^\top \mathbf{n}_u}{\|\widehat{\mathbf{n}}_u\|_2 \|\mathbf{n}_u\|_2}, \qquad \theta_u = \frac{180}{\pi} \arccos(c_u).$$

The reported mean and median errors are

$$\mathrm{Mean} = \frac{1}{N_n} \sum_{u \in \mathcal{V}_n} \theta_u, \qquad \mathrm{Median} = \mathrm{median}\{\theta_u : u \in \mathcal{V}_n\}.$$

For $t \in \{11.25, 22.5, 30\}$ degrees, the within-threshold accuracy is

$$A_t = \frac{1}{N_n} \sum_{u \in \mathcal{V}_n} \mathbf{1}[\theta_u < t].$$

The tables report $100 A_t$ as a percentage.

Differentiable normal proxies. For mean angular error, we use the mean angle in degrees:

$$\mathcal{B}_{\mathrm{Mean}} = \frac{1}{N_n} \sum_{u \in \mathcal{V}_n} \theta_u.$$

For the median criterion, Let $m = \mathrm{median}\{\theta_u : u \in \mathcal{V}_n\}$ be the batch median, treated as fixed during differentiation. We weight angles near this center and form the surrogate

$$a_u = \frac{\exp(-|\theta_u - m|/\gamma_{\mathrm{med}})}{\sum_{v \in \mathcal{V}_n} \exp(-|\theta_v - m|/\gamma_{\mathrm{med}})}, \qquad \mathcal{B}_{\mathrm{Median}} = \sum_{u \in \mathcal{V}_n} a_u \theta_u.$$

The bandwidth $\gamma_{\mathrm{med}} = 1.0$ degree controls concentration around the median.

For threshold accuracy, write $\kappa_t = \cos(\pi t/180)$ so that $\theta_u < t$ is equivalent to $c_u > \kappa_t$. We replace the indicator with a sigmoid:

$$\widehat{A}_t = \frac{1}{N_n} \sum_{u \in \mathcal{V}_n} \mathrm{sigmoid}\left(\frac{c_u - \kappa_t}{\gamma_t}\right), \qquad \mathcal{B}_t = 1 - \widehat{A}_t,$$

where $\gamma_t = 0.02$ is the transition width for each threshold. This relaxation approaches the hard decision as γ_t decreases, away from the boundary. These five normal proxies, together with the two segmentation and two depth terms, form the nine-criterion objective.

A.5 QM9: Molecular Property Regression

QM9 jointly predicts $K = 11$ molecular properties. Let $\widehat{q}_{ni}$ and q_{ni} be the prediction and target for property i of molecule n. The reported property-wise error is

$$\mathrm{MAE}_i = \frac{1}{N} \sum_{n=1}^{N} |\widehat{q}_{ni} - q_{ni}|,$$

where N is the number of evaluated molecules. Since MAE is differentiable almost everywhere and smaller is better, we use its mini-batch value directly as $\mathcal{B}_i^{\mathrm{QM9}} = \mathrm{MAE}_i$. The $R = 11$ terms are calibrated by (12) and aggregated by Q_τ; reported MAEs retain the benchmark units.

A.6 MT10: Episode Success and Distance Guidance

For task i, hard episode success is one if the environment reports success at any time during the rollout and zero otherwise. The reported metric is $\mathrm{SR} = K^{-1}\sum_{i=1}^{K}\mathrm{SR}_i$, with $K = 10$. This trajectory-level indicator is not differentiated. Instead, we learn a distance-based surrogate from training replay, without using evaluation rollouts to update the model. Since MT10 success conditions are task-specific but closely tied to geometric task progress, we use the corresponding task-specific distance as a continuous proxy for approaching success.

Lower-level policy losses. Let y parameterize the shared policy π_y, Q_{ϕ_1} and Q_{ϕ_2} denote the SAC reward critics, and $\mathcal{R}_i$ denote the task-i replay distribution. The lower-level losses in (1) are

$$\mathcal{L}_i^{\pi}(y) = \mathbb{E}_{s\sim\mathcal{R}_i,\, a\sim\pi_y(\cdot|s)}\left[\alpha\log\pi_y(a\mid s) - \min_{j=1,2}\mathsf{Q}_{\phi_j}(s,a)\right], \qquad \ell(x,y) = \sum_{i=1}^{10}\sigma_i(x)\mathcal{L}_i^{\pi}(y). \tag{17}$$

Here α is the learned entropy temperature. Reward-critic parameters and α are held fixed during the paired actor updates. The response model θ has the same policy architecture as y; no response reward critic is introduced.

Learning a distance surrogate. For a stored transition, let d_i be the environment-provided distance after the action, divided by the task's distance threshold δ_i, so that the distance boundary is one. This geometric proxy is distinct from the environment's full success condition. Two auxiliary networks $D_{\psi_j}(s,a,h,i) > 0$ provide a proxy for the future minimum normalized distance over the remaining h steps. They take state, action, remaining horizon and task identity as inputs and use softplus outputs. Writing $\mu_y(s)$ for the policy's deterministic squashed-mean action, their detached regression target is

$$\widehat{d}_i = \begin{cases} 0, & \text{success observed,} \\ d_i, & \text{terminal, without success,} \\ \min\Big\{d_i, \max_{j=1,2} D_{\bar{\psi}_j}(s', \mu_y(s'), (h-1)_+, i)\Big\}, & \text{otherwise,} \end{cases} \tag{18}$$

where $\bar{\psi}_j$ are target-network parameters and success includes both previously observed and newly reached success. This finite-horizon minimum-distance target has no reward discount. Both predictions are fitted with smooth-L_1 regression; transitions from episodes that had already succeeded are excluded from this regression because success history is not part of the policy observation.

Upper-level margin. Let $b \in \{0,1\}$ indicate whether the episode had already succeeded before the sampled action. We use the conservative distance estimate and badness

$$\begin{aligned} D_i(y;s,h,b) &= (1-b)\max_{j=1,2} D_{\psi_j}(s,\mu_y(s),h,i), \\ \mathcal{B}_i^{\mathrm{MT10}}(y) &= \mathbb{E}_{\mathcal{R}_i}\left[\kappa\log\left(1+\exp\left(\frac{D_i(y;s,h,b)-1}{\kappa}\right)\right)\right], \qquad \kappa = 0.25. \end{aligned} \tag{19}$$

The maximum is conservative because larger distances indicate worse performance. Already-successful episodes contribute a constant margin and no actor gradient. The proxy parameters are frozen when differentiating this badness, but gradients pass through the action to y. These networks are surrogate estimators, distinct from both the SAC reward critics and the loss-only response policy; their outputs are not calibrated success probabilities.

We use $z_i = \mathcal{B}_i^{\mathrm{MT10}}/s_i$ and aggregate the $R = 10$ terms by (3). The scales start at $s_i = 1$. Between vector steps 50,000 and 200,000, each scale is the running average of task-wise target margins obtained by applying the same softplus margin to $\widehat{d}_i$; it is frozen thereafter and detached throughout. This delayed calibration uses informative replay rather than an untrained proxy's initial predictions.

B EXPERIMENTAL DETAILS

We provide dataset-specific settings, training implementation details, and hyperparameter choices for the four supervised benchmarks and the MT10 reinforcement-learning benchmark. Evaluation metrics and surrogate definitions are given in Appendix A.

B.1 DATASET-SPECIFIC SETTINGS

CelebA. CelebA contains more than 200,000 face images with 40 binary attributes. Following the standard MTL setup used by prior work, we use a 9-layer CNN with a shared convolutional feature extractor and task-specific linear heads, one for each attribute. We follow the standard train/validation/test split and train for 15 epochs using Adam with a learning rate of 3×10^{-4} and batch size 256.

NYU-v2. NYU-v2 comprises semantic segmentation, depth estimation, and surface-normal prediction. We follow the standard MTL preprocessing protocol and use the SegNet-based MTAN architecture (Liu et al., 2019a), following the experimental setup of PIVRG (Qin et al., 2025b). MTAN augments the shared SegNet encoder–decoder with task-specific attention modules and separate prediction heads for the three tasks. Training runs for 200 epochs with batch size 2 and learning rate 10^{-4} for the first 100 epochs, followed by a factor-of-two reduction.

Cityscapes. Cityscapes comprises urban street scenes with semantic segmentation and depth estimation. We use the common 7-class segmentation protocol and the same SegNet-based MTAN architecture as on NYU-v2, with task-specific attention modules and separate segmentation and depth prediction heads. We train for 200 epochs with batch size 8 and learning rate 10^{-4}, halved after 100 epochs.

QM9. QM9 contains approximately 134,000 small organic molecules with 11 regression targets. We follow the standard multi-task QM9 setup used by ConsMTL (Qin et al., 2025a) and PIVRG (Qin et al., 2025b). The model consists of a shared graph neural network backbone and 11 property-specific regression heads. Training uses Adam, learning rate 10^{-3}, batch size 60, and 300 epochs. Each property contributes a calibrated MAE badness term to the upper objective, while reported MAEs use the conventional benchmark units.

MT10. Following prior MT10 evaluation settings (Ban & Ji, 2024; Qin et al., 2025b), we train using Adam with a learning rate of 3×10^{-4}, a batch size of 1280, 1,500 initial random-interaction vector steps, and a total budget of 2×10^{6} vector steps. Details of the distance-based evaluation surrogate are provided in Appendix A.6.

B.2 ERCO IMPLEMENTATION

ERCo maintains the primary model y, a same-architecture loss-only response model θ, and K task-priority logits x, with weights $w_i = \sigma_i(x)$. At the beginning of each training epoch, we synchronize the response parameters with the current primary parameters, $\theta^n \leftarrow y^n$. When Adam is used, the corresponding optimizer state of the primary model is also copied to the response optimizer, including the first- and second-moment estimates and the optimizer step counter. Thus, the two branches start each epoch from the same model parameters and adaptive optimizer state. Within each epoch, their parameters and optimizer states evolve separately.

Using the same mini-batch and weights $\sigma(x^n)$, we first update θ through weighted standard losses and then update y through those losses and evaluation guidance, following (7) and (8). Once both models have been updated, we evaluate their task losses on the same mini-batch and update x from the differences $\Delta_i^n = \mathcal{L}_i(y^{n+1}) - \mathcal{L}_i(\theta^{n+1})$ via (9). Weights are held fixed during model updates, and model states are held fixed during the priority update; no gradient is propagated through either training trajectory. The new weights are used in the next iteration. Only the primary model is retained for inference.

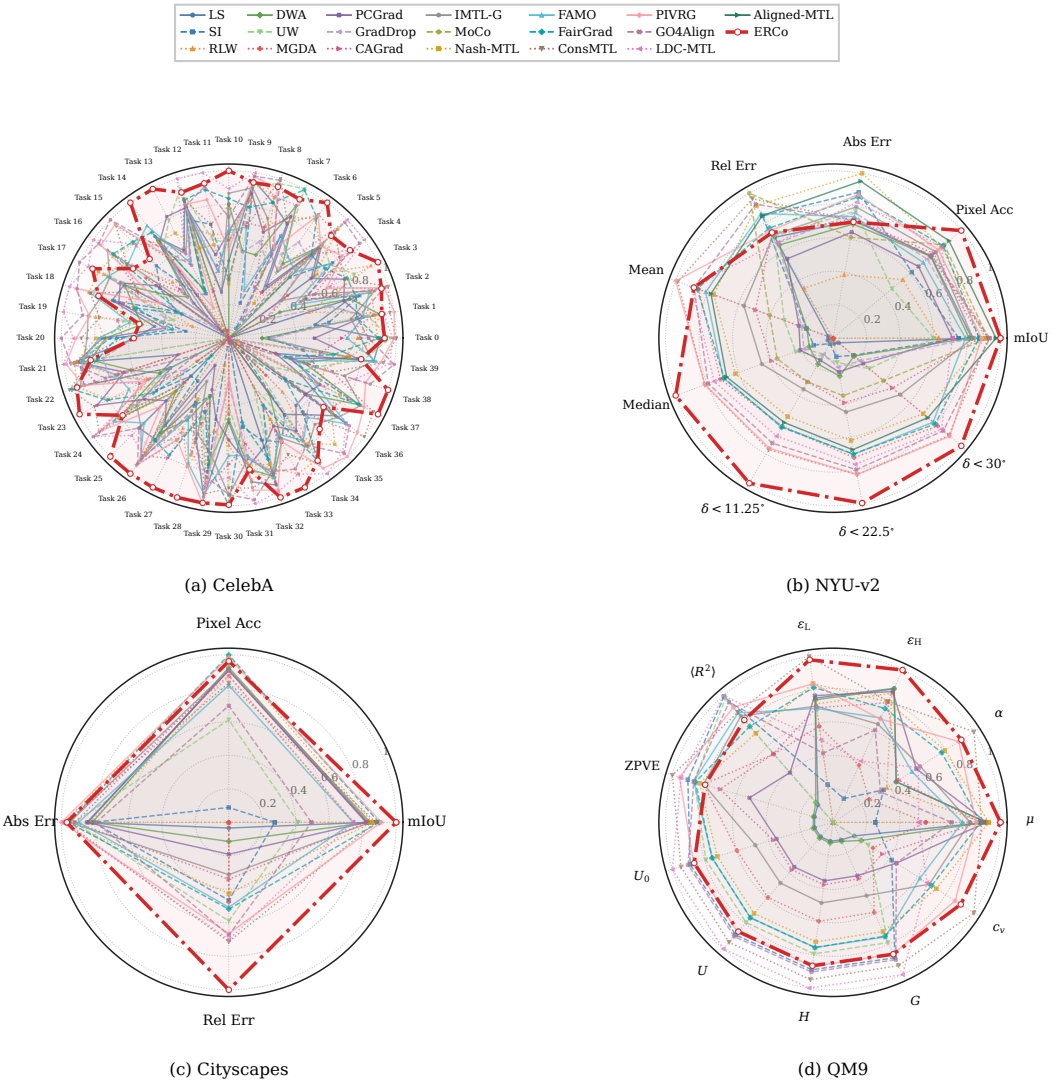
B.3 Hyperparameter Settings

Across all experiments, we set the priority step size equal to the model learning rate, i.e., $\beta_n = \eta_n$. Both updates use Adam with the same learning-rate schedule. In principle, β_n and η_n can be tuned independently.

We fix the aggregation temperature at $\tau = 0.5$ across benchmarks and use $c_n = c\log(n+2)$ with evaluation-guidance coefficient $\rho_n = 1/c_n$. The base coefficient is selected from $c \in \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1\}$. For applications requiring additional refinement, we recommend a finer search. The selected settings are $c = 0.01$ for NYU-v2 and QM9, $c = 0.1$ for CelebA and Cityscapes, and $c = 10^{-3}$ for MT10. The logarithmic schedule gradually reduces the explicit evaluation-guidance weight relative to standard task losses. Unlike the Moreau response-gap penalty in MEHA (Liu et al., 2024), it serves only as a guidance schedule here.

C Additional Performance Visualizations

Figure 4 compares the per-criterion performance on CelebA, NYU-v2, Cityscapes, and QM9. Each metric is normalized within the plotted comparison set, with lower-is-better criteria reversed so that larger radial values consistently indicate better performance.



(a) CelebA

(b) NYU-v2

(c) Cityscapes

(d) QM9

Figure 4: Radar profiles on CelebA, NYU-v2, Cityscapes, and QM9. Larger radial values indicate better performance.

D Additional Experimental Results

This section provides complete per-metric results on Cityscapes and QM9, a diagnostic analysis of the Cityscapes RelErr gain, and detailed component ablations. We also report ERCo's mean and standard error over multiple random seeds on the supervised benchmarks, together with absolute CelebA training times and the corresponding measurement protocol.

D.1 Detailed Cityscapes Results

Table 5: Performance comparison on Cityscapes (2-task dense prediction). Results are averaged over three runs with different random seeds. Best in **bold**, second best underlined.

	Segmentation		Depth			
Method	mIoU ↑	Pix Acc ↑	Abs Err ↓	Rel Err ↓	MR ↓	$\Delta m\%$ ↓
STL (Single-Task)	74.01	93.16	0.0125	27.77		
MGDA (Sener & Koltun, 2018)	68.84	91.54	0.0309	33.50	15.00	44.14
PCGrad (Yu et al., 2020a)	75.13	93.48	0.0154	42.07	10.75	18.29
CAGrad (Liu et al., 2021a)	75.16	93.48	0.0141	37.60	9.50	11.64
IMTL-G (Liu et al., 2021b)	75.33	93.49	0.0135	38.41	8.00	11.10
Nash-MTL (Navon et al., 2022)	75.41	93.66	0.0129	35.02	5.25	6.82
FairGrad (Ban & Ji, 2024)	75.72	**93.68**	0.0134	32.25	4.25	5.18
ConsMTL (Qin et al., 2025a)	75.57	93.32	0.0131	26.41	5.50	-0.59
PIVRG (Qin et al., 2025b)	75.82	93.65	**0.0126**	27.87	**2.75**	-0.54
LS (Linear Scalarization)	75.18	93.49	0.0155	46.77	10.50	22.60
SI (Scale-Invariant)	70.95	91.73	0.0161	33.83	14.25	14.11
RLW (Lin et al., 2022)	74.57	93.41	0.0158	47.79	13.00	24.38
DWA (Liu et al., 2019a)	75.24	93.52	0.0160	44.37	10.25	21.45
UW (Kendall et al., 2018)	72.02	92.85	0.0140	30.13	11.00	5.89
FAMO (Liu et al., 2023)	74.54	93.29	0.0145	32.59	10.75	8.13
GO4Align (Shen et al., 2024)	72.63	93.03	0.0164	27.58	12.00	8.11
LDC-MTL (Xiao et al., 2026)	74.53	93.42	0.0128	26.79	7.00	-0.57
ERCo	**76.51**	93.60	0.0132	**17.79**	**2.75**	**-8.51**

Interpreting the relative-error gain. The pronounced Cityscapes RelErr gain is consistent with the pixel weighting induced by its preprocessed inverse-depth targets. Our label audit finds that 2.60% of valid validation pixels take the isolated minimum value $d = 1/65535$. The next non-negligibly populated level is approximately $258/65535 \simeq 0.00394$, with an essentially empty interval between them. The minimum-value region accounts for 98.98% of the total coefficient mass $\sum_u 1/d_u$. Separately summing the unclipped per-pixel relative errors inside and outside this region shows that it contributes approximately 99% of the total relative-error sum for each of the three final-epoch checkpoints. In contrast, the lowest 2.5% of valid NYU-v2 depths account for only 6.33% of the corresponding coefficient mass. This concentration makes Cityscapes RelErr particularly sensitive to errors in a small pixel subset, whose residuals are not similarly amplified by absolute-error training.

We retain the shared targets and evaluation mask, and apply a denominator floor of 0.005, slightly above the next populated level, only to the Cityscapes training surrogate in (16). Applying the clipped coefficients to the same validation labels reduces the minimum-value region's coefficient share to 22.93%. This limits extreme weighting while retaining greater emphasis on these pixels than uniform absolute-error training. All reported RelErr values remain unclipped. These observations help explain why evaluation guidance can yield a pronounced RelErr gain without a comparable improvement in absolute error. The resulting aggregate gain should therefore be interpreted as an improvement under the reported evaluation criteria, rather than uniform gains across all depth ranges.

D.2 Detailed QM9 Results

Table 6: Performance comparison on QM9 (11-task regression). Results are averaged over three runs with different random seeds. Best in **bold**, second best underlined.

Method	μ	α	ϵ_{HOMO}	ϵ_{LUMO}	$\langle R^2 \rangle$	ZPVE	U_0	U	H	G	c_v	MR ↓	$\Delta m\%$ ↓
						MAE ↓							
STL (Single-Task)	0.067	0.181	60.57	53.91	0.502	4.53	58.8	64.2	63.8	66.2	0.072		
MGDA (Sener & Koltun, 2018)	0.217	0.368	126.8	104.6	3.22	5.69	88.37	89.4	89.32	88.01	0.120	12.55	120.5
PCGrad (Yu et al., 2020a)	0.106	0.293	75.85	88.33	3.94	9.15	116.36	116.8	117.2	114.5	0.110	10.36	125.7
CAGrad (Liu et al., 2021a)	0.118	0.321	83.51	94.81	3.21	6.93	113.99	114.3	114.5	112.3	0.116	11.55	112.8
IMTL-G (Liu et al., 2021b)	0.136	0.287	98.31	93.96	1.75	5.69	101.4	102.4	102.0	100.1	0.096	10.09	77.2
Nash-MTL (Navon et al., 2022)	0.102	0.248	82.95	81.89	2.42	5.38	74.5	75.02	75.10	74.16	0.093	7.27	62.0
FairGrad (Ban & Ji, 2024)	0.117	0.253	87.57	84.00	2.15	5.07	70.89	71.17	71.21	70.88	0.095	7.73	57.9
ConsMTL (Qin et al., 2025a)	0.115	**0.202**	82.69	**67.58**	1.61	**3.33**	48.84	49.04	49.07	49.63	**0.077**	**2.82**	**23.2**
PIVRG (Qin et al., 2025b)	0.125	0.226	94.80	81.98	1.41	3.87	57.79	57.90	58.09	57.86	0.085	5.36	33.6
LS (Linear Scalarization)	0.106	0.325	73.57	89.67	5.19	14.06	143.4	144.2	144.6	140.3	0.128	12.27	177.6
SI (Scale-Invariant)	0.309	0.345	149.8	135.7	1.00	4.50	55.3	55.75	55.82	55.27	0.112	8.64	77.8
RLW (Lin et al., 2022)	0.113	0.340	76.95	92.76	5.86	15.46	156.3	157.1	157.6	153.0	0.137	13.82	203.8
DWA (Liu et al., 2019a)	0.107	0.325	74.06	90.61	5.09	13.99	142.3	143.0	143.4	139.3	0.125	12.00	175.3
UW (Kendall et al., 2018)	0.386	0.425	166.2	155.8	1.06	4.99	66.4	66.78	66.80	66.24	0.122	10.91	108.0
FAMO (Liu et al., 2023)	0.150	0.300	94.0	95.2	1.63	4.95	70.82	71.2	71.2	70.3	0.100	8.82	58.5
GO4Align (Shen et al., 2024)	0.170	0.350	102.4	119.0	1.22	4.94	53.9	54.3	54.3	53.9	0.110	7.82	52.7
LDC-MTL (Xiao et al., 2026)	0.230	0.290	123.89	111.95	**0.97**	3.99	**42.73**	**43.10**	**43.20**	**43.10**	0.097	5.91	45.8
ERCo	**0.081**	0.222	**60.93**	69.28	1.90	5.81	58.19	58.68	58.63	57.84	0.083	4.73	32.9

D.3 Experimental Results with Standard Errors

Table 7: ERCo results on NYU-v2 over three random seeds. Mean and standard error are reported.

Method	Segmentation		Depth		Surface Normal					$\Delta m\%$ ↓
	mIoU ↑	Pix Acc ↑	Abs Err ↓	Rel Err ↓	Angle Distance ↓		Within t° ↑			
					Mean	Median	11.25	22.5	30	
ERCo (mean)	41.098	67.012	0.54998	0.22487	24.762	17.573	33.765	60.621	71.339	-8.974
ERCo (stderr)	±0.324	±0.352	±0.00981	±0.00413	±0.079	±0.148	±0.271	±0.361	±0.253	±0.368

Table 8: ERCo results on Cityscapes and CelebA over three random seeds. Mean and standard error are reported.

	Cityscapes					CelebA
Method	mIoU ↑	Pix Acc ↑	Abs Err ↓	Rel Err ↓	$\Delta m\%$ ↓	$\Delta m\%$ ↓
ERCo (mean)	76.508	93.600	0.013217	17.793	-8.510	-4.779
ERCo (stderr)	±0.274	±0.039	±0.000285	±0.885	±1.033	±0.101

Table 9: ERCo results on QM9 over three random seeds. Mean and standard error are reported.

Method	μ	α	ϵ_{HOMO}	ϵ_{LUMO}	$\langle R^2 \rangle$	ZPVE	U_0	U	H	G	c_v	$\Delta m\%$ ↓
						MAE ↓						
ERCo (mean)	0.081	0.222	60.93	69.28	1.90	5.81	58.19	58.68	58.63	57.84	0.083	32.9
ERCo (stderr)	±0.0007	±0.0024	±0.846	±0.406	±0.0291	±0.0638	±1.92	±1.94	±1.93	±1.80	±0.0009	±1.19

D.4 Detailed Ablation Results

Table 10 provides the complete NYU-v2 results for Q-only, Q+LS, Q+FAMO, and ERCo, complementing Table 4. Both Q+LS and Q+FAMO combine the same evaluation objective Q_τ with standard task-loss training, using fixed uniform weights and FAMO (Liu et al., 2023) weighting, respectively. For both variants, we apply the same hyperparameter grid search used for ERCo and report the best result obtained.

Table 10: Detailed ablation results on NYU-v2. Best in **bold**, second best underlined.

Method	Segmentation		Depth		Surface Normal					$\Delta m\% \downarrow$
	mIoU ↑	Pix Acc ↑	Abs Err ↓	Rel Err ↓	Angle Distance ↓		Within t° ↑			
					Mean	Median	11.25	22.5	30	
Q-only	24.12	60.75	0.6064	0.2510	25.38	**17.51**	**33.98**	**60.65**	70.94	-0.76
Q+LS	39.67	66.80	0.5547	0.2297	26.12	18.03	32.89	59.53	69.93	-6.62
Q+FAMO	40.63	66.50	**0.5386**	**0.2179**	**24.59**	18.38	31.76	59.10	70.50	-7.64
ERCo	**41.10**	**67.01**	0.5500	0.2249	24.76	17.57	33.76	60.62	**71.34**	**-8.97**

On NYU-v2, Q-only remains competitive on several surface-normal criteria but underperforms in segmentation under the evaluated configuration. One possible contributing factor is incomplete mini-batch class coverage: the least frequently observed class appears in only 74 of 795 training images (9.31%), and a uniformly sampled two-image batch without replacement is expected to cover 8.37 of the 13 classes. The class-level statistics used by soft-mIoU can therefore be sensitive to batch composition. In addition, the soft-pixel-accuracy proxy provides weaker logit-level correction than cross-entropy for pixels assigned very low true-class probability. These properties may limit proxy-only optimization in this setting. The segmentation recovery with Q+LS, Q+FAMO, and ERCo supports the complementary role of standard pixel-wise supervision. ERCo further achieves the best aggregate $\Delta m\%$ among these configurations, supporting the additional value of response-based coordination beyond fixed or adaptive weighted-loss training with the same evaluation guidance.

D.5 Detailed CelebA Runtime

To complement the normalized efficiency comparison in Figure 2, Table 11 reports absolute one-epoch training times on an NVIDIA A100. After one warm-up epoch, we measure the next three complete training epochs and report the mean ± sample standard deviation. Relative to LS, ERCo additionally performs one forward–backward pass for the response model and two no-gradient forward evaluations of the updated models per mini-batch. All of these operations are included in the reported runtime. Despite these additional computations, ERCo introduces only a modest increase in training time relative to LS on this benchmark, indicating that the proposed coordination mechanism remains computationally efficient in practice.

Table 11: One-epoch CelebA training time (seconds; mean ± standard deviation).

Method	Time (s/epoch)
LS	40.382 ± 0.721
UW	42.227 ± 0.493
FAMO	41.261 ± 0.611
LDC-MTL	41.869 ± 0.492
FairGrad	756.983 ± 0.823
ConsMTL	910.779 ± 0.124
MGDA	944.430 ± 5.843
PIVRG	754.683 ± 1.535
ERCo	50.042 ± 0.045