Learning Pareto Stationary Fronts via Single-Pass Backpropagation

Elina Rojin Celik
Department of Computer Science
Saarland University, Germany
celik@cs.uni-saarland.de

Marcos Medeiros Raimundo
Instituto de Computação
Universidade Estadual de Campinas, Brazil
mraimundo@ic.unicamp.br

Isabel Valera
Department of Computer Science
Saarland University, Germany
ivalera@cs.uni-saarland.de

Abstract

We propose MOSEL (Multi-Objective Stackelberg Efficient Learning), a framework for a posteriori multi-objective optimization (MOO) in deep neural networks that recovers a full front of Pareto stationary solutions at the computational cost of standard single-objective training. MOSEL reformulates the problem as a bilevel optimization problem that leverages network modularity to decouple representation learning from objective-preference alignment. Casting the bilevel problem as a Stackelberg game enables solving the original a posteriori MOO problem in a single forward–backward pass. As a result, MOSEL matches the time and memory efficiency of standard single-objective training while enabling scalable Pareto stationary front learning. Empirically, MOSEL uncovers diverse and optimal Pareto frontiers in strongly conflicting settings (e.g., fairness–accuracy). Remarkably, even in weakly conflicting regimes such as multi-task learning, it consistently converges to solutions closer to the utopia point, outperforming both standard single-objective training and specialized multi-task learning methods. These results highlight the broader potential of a posteriori MOO learning as a pathway to efficiently learn more diverse and robust representations, ultimately improving generalization.

1 Introduction

Multi-objective optimization (MOO) is ubiquitous across deep learning applications, arising whenever models must balance multiple objectives (i.e., losses during training). Many such objectives stem from competing stakeholder demands, leading to *strongly conflicting objectives*. For example, in *fairness*, individual and group fairness must be traded-off against predictive performance [48]; and, in *LLM alignment*, objectives such as helpfulness, harmlessness, and honesty can conflict [21, 31, 51]. Similar conflicts emerge from the distributed or sequential nature of learning: in *federated learning* each client optimizes a local objective that may conflict with those of other clients [65]; and in *reinforcement learning*, agents must balance multiple reward signals simultaneously [19, 36, 70]. Other MOO applications lead to *weak conflicts* among objectives where, while the objectives may not be directly in conflict, the local nature of neural network optimization may lead to suboptimal solutions. For example, in *multi-task learning* (MTL) joint training can improve performance on individual tasks compared to training them independently (positive transfer); yet, task-specific losses may conflict during optimization, degrading overall performance [34, 56, 67]. Similar weak conflicts arise in *multimodal learning* [25, 63], and in *domain generalization* [5], often leading to suboptimal neural networks that fail to capture diversity across modalities and domains.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

In practice, multiple objectives are often aggregated into a single scalar loss, a strategy that simplifies training but obscures trade-offs and can lead to suboptimal solutions. In particular, collapsing objectives into a single loss: (i) in strongly conflicting regimes, fails to capture the full spectrum of Pareto optimal solutions, thereby masking the trade-offs among objectives [19, 52]; or, (ii) in weakly conflicting regimes, requires manual weighting that is problem-dependent and sensitive to factors such as loss magnitudes [6]. Importantly, such *a priori* scalarization tends to bias optimization toward dominant objectives, e.g., prioritizing performance over fairness [43, 68] or favoring certain tasks over others [56, 67]. Consequently, it can lead to misleading interpretations of the underlying trade-offs and, more importantly, hinder the optimizer from reaching solutions close to the utopia point, thereby limiting overall performance. A principled solution to these issues is *a posteriori* MOO, and in particular, Pareto front learning (PFL). However, to the best of our knowledge, *there is currently no PFL approach that is suitable for training deep neural networks at scale, while providing Pareto stationarity guarantees*.

In this paper, we propose MOSEL (Multi-Objective Stackelberg Efficient Learning), the first framework for PFL in deep neural networks that recovers Pareto stationary solutions at scale. To that end, we exploit the modular structure of neural networks to decouple representation learning from objective preference alignment, enabling efficient MOO training without sacrificing scalability. Specifically, we introduce a framework for PFL that reformulates the problem as a bilevel optimization, which we then cast into a Stackelberg game, allowing the original problem to be solved efficiently with a single forward–backward pass of backpropagation. As a result, *MOSEL achieves Pareto stationary front learning with time and memory complexity comparable to conventional single-objective training*.

Beyond efficiency, we demonstrate that explicitly modeling multiple objectives allows MOSEL to: (i) recover diverse and high-quality Pareto frontiers in strongly conflicting settings; and, (ii) converge to solutions closer to the utopia solution in weakly conflicting regimes such as MTL, outperforming both single-objective baselines and specialized multi-task methods. These findings suggest that *efficient multi-objective learning can serve as a foundation for learning richer, more diverse representations, potentially improving robustness and generalization in deep neural networks.*

2 Background and Related Work

In this section, we introduce the key concepts from multi-objective and bilevel optimization that underpin our approach, and provide an overview of the related literature.

2.1 Multi-objective optimization.

In deep learning, MOO naturally arises when training a model with multiple loss functions. Let $\boldsymbol{\theta} \in \Theta \subset \mathbb{R}^n$ denote the parameters of a neural network, and consider m non-negative, differentiable loss functions $\mathcal{L}_j : \Theta \to \mathbb{R}_+$, which may correspond to task-specific losses in multi-task learning or stakeholder-defined losses (e.g., fairness objectives). The training problem can then be formulated as a MOO problem [15, 44]:

$$\min_{\boldsymbol{\theta}\in\Theta} \mathcal{L}(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}\in\Theta} (\mathcal{L}_1(\boldsymbol{\theta}), \dots, \mathcal{L}_m(\boldsymbol{\theta})). \tag{1}$$

In MOO, the objectives are typically conflicting, meaning that no single parameter configuration simultaneously minimizes all losses. To formalize optimal trade-offs, [44] defines a dominance relation: a solution $\boldsymbol{\theta}_1 \in \Theta$ is said to be dominated by $\boldsymbol{\theta}_2 \in \Theta$, denoted by $\mathcal{L}(\boldsymbol{\theta}_2) \prec \mathcal{L}(\boldsymbol{\theta}_1)$, if $\boldsymbol{\theta}_2$ improves at least one objective without worsening any other. A point $\boldsymbol{\theta}^\star \in \Theta$ is called Pareto optimal (PO) if no other point in Θ dominates it. Given the non-convexity of neural network loss landscapes, we focus on local notions of optimality.

Optimality. The set of local Pareto optimal solutions [44] is defined as

$$\mathcal{P} := \{\boldsymbol{\theta}^\star \in \Theta \mid \exists \epsilon > 0 : \forall \boldsymbol{\theta} \in \mathcal{N}_\epsilon(\boldsymbol{\theta}^\star), \mathcal{L}(\boldsymbol{\theta}) \not\prec \mathcal{L}(\boldsymbol{\theta}^\star)\} \tag{2}$$

where $\mathcal{N}_\epsilon(\boldsymbol{\theta}^\star) = \{\boldsymbol{\theta} \in \Theta \mid \|\boldsymbol{\theta} - \boldsymbol{\theta}^\star\| < \epsilon\}$. The image set $\mathcal{L}(\mathcal{P})$ is referred to as the local Pareto front (PF), which may consist of multiple or even infinitely many solutions.

In differentiable settings, a point $\boldsymbol{\theta}^\star \in \Theta$ is Pareto stationary if there exists a convex combination of the gradients of the losses that vanishes. Formally, $\boldsymbol{\theta}^\star$ is Pareto stationary if

$$\exists\, \boldsymbol{r} \in \mathbb{R}^m, r_j \geq 0, \sum r_j = 1 : \sum_{j=1}^{m} r_j \nabla \mathcal{L}_j(\boldsymbol{\theta}^\star) = \boldsymbol{J}_{\mathcal{L}}(\boldsymbol{\theta}^\star)^T \boldsymbol{r} = \mathbf{0}, \tag{3}$$

where $\boldsymbol{J}_{\mathcal{L}}(\boldsymbol{\theta}^\star)$ denotes the Jacobian of the vector-valued objective. Pareto stationarity is a necessary condition for local Pareto optimality, and becomes sufficient under convexity assumptions [13, 17].

Scalarization. A natural approach to MOO is to reduce the vector-valued problem in Eq. (1) to a single surrogate objective via *scalarization*, which encodes a preference over the objectives through a vector $\boldsymbol{r}$ [44]. Two common scalarization functions that we will explore in this work include linear scalarization (LS) [44], which translates to a weighted sum of the objectives as $\sum_{i=1}^{m} r_i \mathcal{L}_i(\boldsymbol{\theta})$ and smooth Chebychev scalarization (CS) [33], which is defined as $\frac{1}{\tau} \log \sum_{i=1}^{m} \exp\left(\tau \cdot r_i \mid \mathcal{L}_i(\boldsymbol{\theta}) - u_i^\star \mid\right)$ with smoothing parameter τ and *utopia point* $\boldsymbol{u}^\star$, where each $u_i^\star$ is the ideal objective value of $\mathcal{L}_i(\boldsymbol{\theta})$ attainable in isolation. In the presence of conflicting objectives, this point is generally infeasible. While linear scalarization is simple and efficient, it can only recover solutions on convex parts of the Pareto front, Chebychev scalarization on the other hand does not suffer from this limitation [44].

Related works. MOO methods can be categorized based on when preferences are specified [38, 44]. *No-preference* approaches resolve competing gradients without user input [13, 56]. *A priori* methods compute a single Pareto stationary solution for a fixed preference, typically via scalarization [33] or preference-aware gradient balancing [41], but require retraining for new preferences. *A posteriori* methods instead generate a set of trade-off solutions, e.g., by partitioning the objective space [32], tracing Pareto sets [40], or sampling preferences for coverage [4]. *Interactive* methods iteratively refine solutions with user feedback [44], but remain costly in deep learning, although approaches such as RLHF [7] and LLM alignment [47] can be seen as instances of this paradigm.

A key limitation of both *a priori* and finite-solution *a posteriori* methods is their inability to generalize to unseen preferences without retraining. *Pareto Front Learning* (PFL) addresses this by learning a mapping from preferences to Pareto optimal solutions, aiming for (i) *optimality* (i.e., solutions lying on the Pareto front) and (ii) *diversity* (i.e., coverage of the full front) [9, 22]. Pareto Hypernetworks [45] achieve this but incur high computational and parameter costs due to per-objective gradients and auxiliary networks. COSMOS [53] improves scalability with a single training run and minimal overhead, but lacks guarantees of Pareto stationarity, as shown in Appendix A.

In contrast to prior work, *MOSEL combines the strengths of* a posteriori *methods and Pareto front learning while avoiding their key limitations, achieving Pareto stationary solutions without incurring significant computational or parameter overhead.*

2.2 Bilevel optimization

A bilevel optimization problem introduces a hierarchy between two connected optimization problems:

$$\min_{\boldsymbol{\phi} \in \Phi} \quad g(\boldsymbol{\phi}, \boldsymbol{\theta}) \quad \text{s.t.} \quad \boldsymbol{\theta} \in \arg\min_{\boldsymbol{\theta}' \in \Theta} f(\boldsymbol{\phi}, \boldsymbol{\theta}') \tag{4}$$

where $\Phi \subseteq \mathbb{R}^p$ and $\Theta \subseteq \mathbb{R}^q$ denote the feasible decision spaces for the upper and lower level problems; and g and f denote the upper and lower level objectives respectively [10, 12]. The bilevel formulation provides a more natural way to describe an objective that inherently depends on the outcome of a second objective [58], allowing for more flexibility in the optimization process by enabling separate optimization for each level.

Related works. Previous works [4, 66] have explored bilevel formulations of MOO. Rather than fixing preference vectors a priori, [4] optimizes a distribution over a finite set of preferences in the upper level with respect to a quality metric (e.g., hypervolume) computed in the lower level. While this yields well-distributed solutions, the number of Pareto points remains fixed and limited by the upper-level design. In contrast, [66] propose a bilevel formulation with accuracy as the upper-level and fairness as the lower level objective. However, scaling this approach beyond two objectives is non-trivial, as each additional objective requires introducing additional hierarchical levels.

In contrast to prior work, *MOSEL leverages bilevel optimization to efficiently learn Pareto stationary fronts without restricting the number of solutions or objectives.*

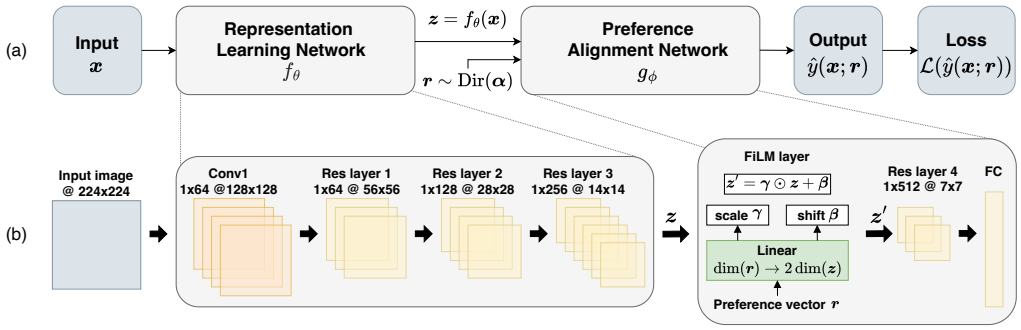

Figure 1: **Overview of MOSEL.** (a) It partitions the neural network into a representation learning network, steering $\boldsymbol{z}$ toward Pareto optimal solutions, and a preference alignment network conditioning on $\boldsymbol{r} \sim \mathrm{Dir}(\boldsymbol{\alpha})$ to cover diverse trade-offs. (b) Illustration on ResNet-34 [20]: $\boldsymbol{z}$ is extracted at the network split point and modulated via a FiLM layer [50], which maps $\boldsymbol{r}$ to a scaling and shift factor per feature channel, steering the final predictions $\hat{y}(\boldsymbol{x}; \boldsymbol{r})$ toward the desired objective trade-off.

3 From *a posteriori* MOO to bilevel optimization of neural networks

In this section, we cast the *a posteriori* MOO problem, or more specifically, the Pareto stationary front learning of deep neural networks into a bilevel optimization problem. To that end, we first assume a **scalarization function**, of the form $s : \mathbb{R}^m \times [0,1]^m \mapsto \mathbb{R}$, to aggregate the vector of loss functions (i.e., objectives) $\mathcal{L} = (\mathcal{L}_1, \ldots, \mathcal{L}_m)$ into a single scalar value, according to a given preference vector $\boldsymbol{r} \in [0,1]^m$. In our case, we consider linear scalarization $\sum_{i=1}^m r_i \mathcal{L}_i(\boldsymbol{\theta})$ [44] and smooth Tchebycheff scalarization $\frac{1}{\tau} \log \sum_{i=1}^m \exp\left(\tau \cdot r_i \mid \mathcal{L}_i(\boldsymbol{\theta}) - u_i^\star \mid\right)$ [33].

Second, we leverage the modularity of neural networks to decouples the learning of optimal (i.e., Pareto stationary) from diverse (i.e., different trade-offs among objectives) solutions, as illustrated in Figure 1 (a). To this end, we partition the network into: (i) a **representation learning network** f_θ, which seeks representations steering the Pareto front toward the utopia point, effectively guiding solutions toward *optimality*; and (ii) a **preference alignment network** g_ϕ, which focuses on maintaining *diversity* of the solutions across the Pareto front. By separating optimality and diversity within the neural network, MOSEL mitigates adversarial interactions between competing goals, yielding a more stable optimization process. In Section 4, we formalize this intuition by showing that stochastic preference sampling implicitly regularizes the representation network toward shared features, exponentially suppressing objective-specific ones. Importantly, this decomposition comes at almost no additional memory or computational cost, as it simply requires the incorporation of a preference vector as input to the preference alignment network and partitioning the neural network computational graph during training.

Figure 1 (b) illustrates this architectural design using the ResNet-34 architecture [20]. The network is divided into two parts: a representation learning network, comprising the convolutional stem and the first three residual layers, and a preference alignment network consisting of the fourth residual layer followed by the fully connected head. We leverage the intermediate representation after the third residual layer, denoted by $\boldsymbol{z}$, as the conditioning point for preference injection. As highlighted in Figure 1 (b), the only modification to the original architecture is a Feature-wise Linear Modulation (FiLM) layer [50] inserted at the split point, which modulates $\boldsymbol{z}$ channel-wise based on the preference vector $\boldsymbol{r}$, steering the shared representation towards the desired trade-off while preserving its spatial structure. The resulting conditioned representation $\boldsymbol{z}'$ is subsequently passed through the remaining network. This design introduces only negligible overhead relative to the base network.

Third, we can now reformulate the original *a posteriori* MOO problem from Eq. (1) as the following **bilevel optimization problem**:

$$\begin{aligned} &\text{(leader)} && \arg\min_{\boldsymbol{\phi}} \; s(\mathcal{L}(\boldsymbol{\theta}^\star, \boldsymbol{\phi}(\boldsymbol{r})); \boldsymbol{r}) \\ &\text{(follower)} && s.t. \;\; \boldsymbol{\theta}^\star \in \arg\min_{\boldsymbol{\theta}} \; s(\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}(\boldsymbol{r})); \boldsymbol{r}) \end{aligned} \quad , \quad \forall \boldsymbol{r} \sim Dir(\boldsymbol{\alpha}), \tag{5}$$

where the preference vectors $\boldsymbol{r}$ are sampled from a Dirichlet distribution with hyperparameter $\boldsymbol{\alpha}$ and s denotes the scalarization function.

In the context of Pareto front learning, the bilevel perspective provides a natural decomposition between optimality and diversity: the lower level ensures Pareto stationary solutions, while the upper level governs preference alignment and coverage of the front. We interpret the lower level optimization problem as the *follower*, which optimizes the parameters of the representation network $\boldsymbol{\theta}$, and the upper level problem as the *leader*, which optimizes the parameters of the preference alignment network $\boldsymbol{\phi}$. We adopt this leader-follower perspective in Section 4, where we cast the bilevel problem as a Stackelberg game. By combining this interpretation with the modular structure of neural networks, we enable efficient single-pass training that recovers diverse, Pareto stationary solutions without restricting the number of objectives or attainable trade-offs.

Before introducing the Stackelberg interpretation, we first show that any solution to Eq. (5) is a Pareto stationary solution of the original *a posteriori* MOO problem in Eq. (1). We assume that the scalarization function s satisfies the following properties:

(i) $s(\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}(\boldsymbol{r})); \boldsymbol{r})$ is continuously differentiable in $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ and $s(\cdot, \boldsymbol{r})$ in $\mathcal{L}$; and
(ii) $s(\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}(\boldsymbol{r})); \boldsymbol{r})$ is strictly increasing in each objective $\mathcal{L}_i$ with $\partial s / \partial \mathcal{L}_i > 0$ for all $i = 1, \ldots, m$.
Note that both linear scalarization and smooth Tchebycheff scalarization satisfy these conditions, since their partial derivatives w.r.t. each objective are strictly positive whenever $r_i > 0$.

Theorem 1 (Pareto stationarity)**.** *Let s be a scalarization function satisfying properties (i) and (ii), and let $(\boldsymbol{\phi}^\star, \boldsymbol{\theta}^\star)$ be a solution to the bilevel problem in Eq.* (5)*. Then $(\boldsymbol{\phi}^\star, \boldsymbol{\theta}^\star)$ is a Pareto stationary solution of the MOO problem in Eq.* (1)*.*

Proof. Let $\boldsymbol{\phi}^\star$ and $\boldsymbol{\theta}^\star$ denote solutions to the upper and lower level optimization problems, respectively. Define weights $\omega_i(\boldsymbol{r}) := \partial s / \partial \mathcal{L}_i$ with partial derivatives evaluated at $\mathcal{L}(\boldsymbol{\theta}^\star, \boldsymbol{\phi}^\star(\boldsymbol{r}))$ and normalize such that $\sum_{i=1}^m \omega_i(\boldsymbol{r}) = 1$. By property (ii), $\omega_i(\boldsymbol{r}) > 0$ so it lies in the probability simplex.

Since $\boldsymbol{\theta}^\star$ is an unconstrained solution to the lower level problem, the first-order optimality condition together with property (i) gives

$$\nabla_{\boldsymbol{\theta}}\, s(\mathcal{L}(\boldsymbol{\theta}^\star, \boldsymbol{\phi}^\star(\boldsymbol{r})); \boldsymbol{r}) = \left(\sum_{j=1}^m \frac{\partial s}{\partial \mathcal{L}_j} \right) \sum_{i=1}^m \omega_i(\boldsymbol{r})\, \nabla_{\boldsymbol{\theta}} \mathcal{L}_i(\boldsymbol{\theta}^\star, \boldsymbol{\phi}^\star(\boldsymbol{r})) = \mathbf{0}. \tag{6}$$

The upper level objective depends on ϕ both directly and through the followers best response $\boldsymbol{\theta}^\star(\phi)$. Using the first-order optimality condition and Eq. (6) its total derivative is

$$\begin{aligned} \frac{d\, s(\mathcal{L}(\boldsymbol{\theta}^\star, \boldsymbol{\phi}^\star(\boldsymbol{r})); \boldsymbol{r})}{d\boldsymbol{\phi}} &= \nabla_{\boldsymbol{\phi}} s(\mathcal{L}(\boldsymbol{\theta}^\star, \boldsymbol{\phi}^\star(\boldsymbol{r})); \boldsymbol{r}) + \left(\frac{\partial \boldsymbol{\theta}^\star}{\partial \phi} \right)^\top \nabla_{\boldsymbol{\theta}} s(\mathcal{L}(\boldsymbol{\theta}^\star, \boldsymbol{\phi}^\star(\boldsymbol{r})); \boldsymbol{r}) \\ &= \left(\sum_{j=1}^m \frac{\partial s}{\partial \mathcal{L}_j} \right) \sum_{i=1}^m \omega_i(\boldsymbol{r})\, \nabla_{\boldsymbol{\phi}} \mathcal{L}_i(\boldsymbol{\theta}^\star, \boldsymbol{\phi}^\star(\boldsymbol{r})) = \mathbf{0}. \end{aligned} \tag{7}$$

Since $\sum_j \partial s / \partial \mathcal{L}_i$ is strictly positive, we can divide it out of both Eq. (6) and (7). Since both equations contain the same weights we can combine them to expresses $\mathbf{0}$ as a convex combination of the per-objective gradients at $(\boldsymbol{\phi}^\star, \boldsymbol{\theta}^\star)$, which is precisely the Pareto stationarity condition in Eq. (3). Thus, any solution $(\boldsymbol{\phi}^\star, \boldsymbol{\theta}^\star)$ to the bilevel optimization problem in Eq. (5) is Pareto stationary. ■

4 MOSEL: Multi-Objective Stackelberg Efficient Learning

In this section, we introduce Multi-Objective Stackelberg Efficient Learning (MOSEL), a framework for *a posteriori* multi-objective optimization in deep neural networks that recovers a full front of Pareto stationary solutions at the computational cost of standard single-objective training. To that end, we build on recent works that leverage Stackelberg game theory for deep learning optimization—such as gradient homogenization in multi-task learning [24] and fairness–accuracy trade-offs [66]—to cast the bilevel optimization problem in Eq. (5) to a Stackelberg game. Following the notion of a differentiable Stackelberg equilibrium, introduced in [16], we train both the representation learning and preference alignment networks via simultaneous gradient updates—in other words, via a single forward-backward pass.

Algorithm 1 MOSEL algorithm

Input: Dataset $\mathcal{D} = (x, y)$, Dirichlet hyperparameter $\boldsymbol{\alpha}$,
learning rates η_ϕ and η_θ ▷ $\eta_\phi < \eta_\theta$
for e epochs **do**
  for every batch $(x, y) \sim p_{\mathcal{D}}$ **do**
    Sample $\boldsymbol{r} \sim \mathrm{Dir}(\boldsymbol{\alpha})$ ▷ 1 preference vector per batch
    (*Forward Pass*)
    $\hat{y} \leftarrow g_{\boldsymbol{\phi}}(f_{\boldsymbol{\theta}}(x),\, \boldsymbol{r})$ ▷ Compute neural network outputs
    $\ell := s(\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}(\boldsymbol{r})); \boldsymbol{r})$ ▷ Evaluate losses on $(\hat{y}, y)$ and scalarize
    (*Backward Pass*)
    $\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} - \eta_\phi \nabla_{\boldsymbol{\phi}} \ell$ ▷ Update preference alignment network
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta_\theta \nabla_{\boldsymbol{\theta}} \ell$ ▷ Update representation learning network
  end for
end for
return $\phi, \boldsymbol{\theta}$

A general **Stackelberg game** is a two-player game consisting of a leader and follower, where the leader plays first and the follower reacts optimally to this move, representing the hierarchical structure of bilevel optimization problems. As stated by Fiez et al. [16], given the objectives of leader and follower $g, f : \Phi \times \Theta \to \mathbb{R}$, a strategy $(\boldsymbol{\phi}^\star, \boldsymbol{\theta}^\star)$ is a Stackelberg equilibrium, if it minimizes

$$\underset{\boldsymbol{\phi} \in \Phi}{\arg\min} \quad g(\boldsymbol{\phi}, \boldsymbol{\theta}) \quad \text{s.t.} \quad \boldsymbol{\theta} \in \underset{\boldsymbol{\theta}' \in \Theta}{\arg\min}\, f(\boldsymbol{\phi}, \boldsymbol{\theta}'). \tag{8}$$

As we operate in non-convex settings, we consider the corresponding local formulation, where the strategy $\boldsymbol{\phi}^\star \in U_\phi$ is a local Stackelberg solution for the leader if, for all $\boldsymbol{\phi} \in U_\phi$,

$$\sup_{\boldsymbol{\theta} \in R_{U_\theta}(\boldsymbol{\phi}^\star)} g(\boldsymbol{\phi}^\star, \boldsymbol{\theta}) \;\leq\; \sup_{\boldsymbol{\theta} \in R_{U_\theta}(\boldsymbol{\phi})} g(\boldsymbol{\phi}, \boldsymbol{\theta}), \tag{9}$$

where $R_{U_\theta}(\boldsymbol{\phi}) = \left\{\boldsymbol{\theta} \in U_\theta \mid f(\boldsymbol{\phi}, \boldsymbol{\theta}) \leq f(\boldsymbol{\phi}, \boldsymbol{\theta}'),\, \forall \boldsymbol{\theta}' \in U_\theta\right\}$ is the follower's local best-response. Moreover, for any $\boldsymbol{\theta}^\star \in R_{U_\theta}(\boldsymbol{\phi}^\star)$, $(\boldsymbol{\phi}^\star, \boldsymbol{\theta}^\star)$ is a local Stackelberg equilibrium on $U_\phi \times U_\theta$. However, the exact computation of $R_{U_\theta}(\boldsymbol{\phi})$ is intractable in neural networks.

To overcome this limitation, Fiez et al. [16] introduce the notion of **differentiable Stackelberg equilibrium**, where $(\boldsymbol{\phi}^\star, \boldsymbol{\theta}^\star) = (\boldsymbol{\phi}^\star, b(\boldsymbol{\phi}^\star))$ is a differentiable Stackelberg equilibrium, if

$$\nabla_{\boldsymbol{\phi}}\, g(\boldsymbol{\phi}^\star, b(\boldsymbol{\phi}^\star)) = 0 \quad \text{and} \quad \nabla^2_{\boldsymbol{\phi}}\, g(\boldsymbol{\phi}^\star, b(\boldsymbol{\phi}^\star)) \succ 0, \tag{10}$$

where $\boldsymbol{\theta}^\star = b(\boldsymbol{\phi}^\star)$ is the follower's best response that is implicitly defined by $\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\phi}^\star, \boldsymbol{\theta}^\star) = 0$. The concepts of local Stackelberg equilibrium and differentiable Stackelberg equilibrium coincide except on a set of measure zero [16], making the differentiable Stackelberg equilibrium a natural target for gradient-based optimization. Fiez et al. [16] further show, that simultaneous gradient updates with a timescale separation, that is, a smaller learning rate for the leader than for the follower, is sufficient to ensure local convergence to a differentiable Stackelberg equilibrium. Intuitively, the faster follower updates keep $\boldsymbol{\theta}$ close to its best response at every leader step, approximating the hierarchical structure of the Stackelberg game without requiring exact best-response computation.

These guarantees directly connect to the Pareto stationarity condition in Eq. (3), as any differentiable Stackelberg equilibrium $(\boldsymbol{\phi}^\star, \boldsymbol{\theta}^\star)$ satisfy their first-order stationarity conditions, which coincide with the condition in Eq. (3). Thus, convergence to a differentiable Stackelberg equilibrium implies convergence to a Pareto stationary solution, providing the theoretical foundation for the **MOSEL algorithm** in Alg. 1. We highlight that the output of Alg. 1 correspond to local optima of the neural network parameters $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$, which in turn yield a Pareto stationary solution of the bilevel optimization problem in Eq. (5), and consequently of the original MOO problem in Eq. (1).

Beyond Pareto stationarity, the stochastic preference sampling in Alg. 1 implicitly regularizes the representation network toward shared features. Under the power-iteration characterization of early-stage gradient descent [69], we show in Theorem 3 in App. A.2 that objective-specific features are exponentially suppressed relative to shared ones, with strength controlled by the Dirichlet concentration $\boldsymbol{\alpha}$. The following proposition summarizes the main results.

Proposition 1 (Implicit regularization toward shared features)**.** *Consider the bilevel problem in Eq.* (5) *with two objectives A and B, solved via the Stackelberg formulation of Alg. 1. Let $\mathcal{X}_A, \mathcal{X}_B \subseteq \mathcal{X}$ denote the feature directions in f_θ relevant to each objective and $\mathcal{X}_{AB} = \mathcal{X}_A \cap \mathcal{X}_B$ the shared features. In the weakly conflicting regime, i.e., when $\mathcal{X}_{AB} \neq \emptyset$, let $\rho_{k,m}$ denote the relative learning ratio, i.e., the fraction of representational energy allocated to feature k over feature m, under the stochastic preference sampling of Alg. 1. Then:*

$$\forall\, k \in \mathcal{X}_{AB},\ m \in \mathcal{X} \setminus \mathcal{X}_{AB}\ :\ \exists\, \boldsymbol{\alpha} : \rho_{k,m} > 1.$$

Thus, there exists a Dirichlet parameter $\boldsymbol{\alpha}$, such that the network preferentially learns shared features over objective-specific ones, even when the latter has a larger average importance as shown in Remark 2. The full proof, assumptions, and generalization to K objectives can be found in App. A.2.

Computational time and memory complexity. Note that MOSEL shares the same asymptotic complexity as COSMOS [53], with $\mathcal{O}(e \cdot m \cdot p)$ time and $\mathcal{O}(e \cdot p)$ memory, where e denotes the number of epochs and p the number of network parameters. This efficiency arises because both methods perform a single forward and backward pass over m objectives per gradient step. Importantly, the proposed split architecture introduces no additional overhead: the timescale separation approximation reduces the nested second-order derivatives of exact bilevel optimization to standard gradient descent, thereby achieving Pareto stationarity at no extra computational cost.

5 Experiments

5.1 Strongly conflicting setting: Fairness-Performance trade-off

Fairness in machine learning is inherently a multi-objective problem, as metrics such as Difference in Equality of Opportunity (DEO) directly conflict with predictive performance [48, 62]. Existing approaches typically treat fairness as a constraint or penalty, reducing the problem to a single trade-off and limiting the information available to decision makers [37]. This is particularly problematic in high-stakes domains such as criminal justice, credit scoring, and healthcare [43, 62]. We leverage this use case in the following experiments to empirically validate that MOSEL recovers diverse and high-quality Pareto frontiers in strongly conflicting settings.

Datasets. We evaluate on Adult [2], COMPAS [1], and Credit [23], three widely used tabular fairness benchmarks with `'sex'` as the sensitive attribute, and on the image dataset FairFace [27], predicting `'gender'` under sensitive attribute `'race'`. Training objectives are binary cross-entropy (BCE) as a proxy for accuracy and a differentiable hyperbolic tangent relaxation of DEO [48] as a fairness surrogate; at test time we report misclassification error and DEO.

Baselines and metrics. We compare MOSEL against COSMOS [53], evaluating both on 25 uniformly distributed test rays. We additionally compare against the post-processing baseline of Hardt et al. [18], which operates on an unconstrained classifier (using the same neural network architecture) by sweeping a shared target TPR across per-group ROC convex hulls to derive group-specific decision thresholds, tracing out the Pareto front. Further details on hyperparameters, architectures, and training are provided in Appendix B. We report the relative hypervolume [71], defined as the achieved hypervolume divided by the maximum possible hypervolume between the utopia point (ideal, generally unattainable) and nadir point (worst per-objective value among observed solutions) as shown in Appendix B.2; alongside the number of non-dominated solutions per method, where dominated solutions are removed in post-processing.

Results. Table 1 summarizes the results on both training and test objectives. MOSEL-LS and MOSEL-CS consistently match or outperform COSMOS across datasets in terms of both hypervolume and the number of Pareto optimal solutions, providing strong empirical evidence that MOSEL recovers diverse and high-quality Pareto frontiers in strongly conflicting settings. This advantage is particularly pronounced in Figure 2, where on FairFace, COSMOS collapses to a near-degenerate front, while both MOSEL variants maintain high hypervolume and a substantially larger set of Pareto optimal solutions. The post-hoc fairness baseline [18] remains competitive in hypervolume in some cases (e.g., Default and FairFace test error), but generally yields fewer non-dominated solutions. Moreover, on datasets such as Adult, its solutions are concentrated in a narrow region of low unfairness but high error, limiting its practical utility.

Table 1: **Fairness-Performance Trade-offs.** Relative hypervolume and number of Pareto optimal solutions on the fairness benchmarks, computed on training loss and test error. Mean ± std over five runs; best entries per column shaded.

Dataset	Method	HV Loss ↑	HV Error ↑	# PO sol. (Loss) ↑	# PO sol. (Error) ↑
Adult	MOSEL-LS	0.939 ± 0.007	0.849 ± 0.012	22.8 ± 1.2	21.2 ± 1.5
	MOSEL-CS	0.819 ± 0.017	0.776 ± 0.015	23.8 ± 0.7	22.0 ± 1.4
	COSMOS	0.926 ± 0.014	0.830 ± 0.010	20.2 ± 0.7	17.4 ± 1.9
	Classifier	0.906 ± 0.025	0.618 ± 0.016	8.0 ± 0.6	6.2 ± 0.7
COMPAS	MOSEL-LS	0.920 ± 0.019	0.568 ± 0.008	15.6 ± 1.4	14.6 ± 1.4
	MOSEL-CS	0.930 ± 0.007	0.560 ± 0.005	15.2 ± 1.6	14.2 ± 1.6
	COSMOS	0.928 ± 0.002	0.567 ± 0.019	14.6 ± 1.0	13.6 ± 1.0
	Classifier	0.881 ± 0.026	0.645 ± 0.037	9.2 ± 1.6	6.2 ± 1.5
Default	MOSEL-LS	0.951 ± 0.011	0.922 ± 0.014	17.0 ± 1.7	15.4 ± 0.8
	MOSEL-CS	0.902 ± 0.029	0.897 ± 0.024	13.6 ± 2.1	10.8 ± 2.3
	COSMOS	0.938 ± 0.009	0.905 ± 0.016	14.6 ± 3.9	11.8 ± 2.5
	Classifier	0.941 ± 0.023	0.925 ± 0.016	4.8 ± 1.3	4.0 ± 1.1
FairFace	MOSEL-LS	0.987 ± 0.013	0.701 ± 0.047	13.6 ± 1.0	10.6 ± 2.3
	MOSEL-CS	0.998 ± 0.003	0.734 ± 0.042	13.2 ± 2.3	11.4 ± 2.4
	COSMOS	0.193 ± 0.038	0.176 ± 0.037	3.2 ± 1.0	2.0 ± 0.9
	Classifier	0.808 ± 0.009	0.708 ± 0.030	16.4 ± 1.4	12.2 ± 1.6

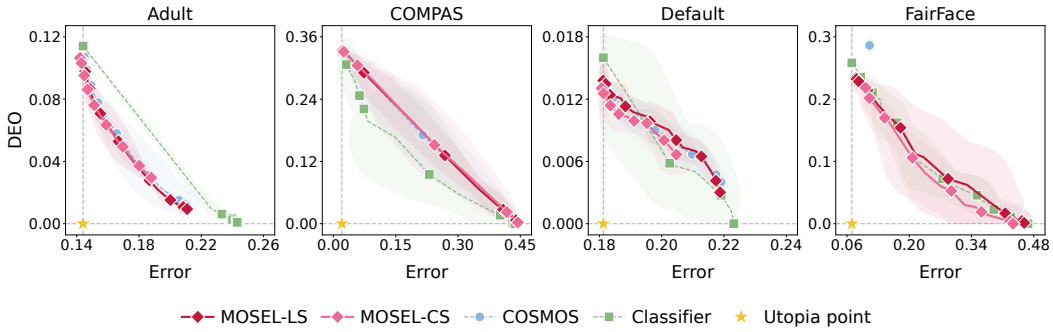

Figure 2: **Fairness-Performance Trade-offs.** Test-time Pareto fronts, plotting classification error against fairness violation. The grey dotted lines indicate zero unfairness and the best accuracy achievable by an unconstrained classifier; their intersection defines the utopia point—an ideal, yet unattainable solution toward which all methods aim to converge.

5.2 Weakly conflicting setting: Multi-Task Learning

In multi-task learning, joint training can improve task performance via positive transfer, but task-specific losses may also induce conflicting gradients, degrading performance [56]. However, such conflicts are not guaranteed as they depend on diverse factors like the data, training dynamics, etc., placing MTL in a weakly conflicting regime. As a result, methods designed to resolve gradient conflicts—such as GradNorm [6], FAMO [35], NashMTL [46], and PCGrad [67] —can even harm performance [34, 57].

We use the MTL setting to demonstrate that MOSEL adapts to both regimes without prior knowledge of the conflict structure. First, in a simple two-task setting, we show that MOSEL converges to solutions closer to the utopia point by effectively leveraging positive transfer, outperforming both standard single-objective training and specialized multi-task learning methods without additional computational cost. In a second experiment, we demonstrate that MOSEL efficiently enables positive transfer in settings with many tasks (objectives).

Datasets. We evaluate on three synthetic and one real-world benchmark. Multi-MNIST [54], Multi-Fashion and Multi-Fashion+MNIST [32] are synthetic multi-task benchmarks that are composed of two digits or clothing items based on the MNIST [30] and Fashion-MNIST [64] dataset,where the goal is to simultaneously classify both objects (top-left and bottom-right), using per-task cross-entropy as objectives. As real-world benchmark we consider the CelebA dataset [39], containing over 200,000 images of celebrities, annotated with 40 binary attributes.

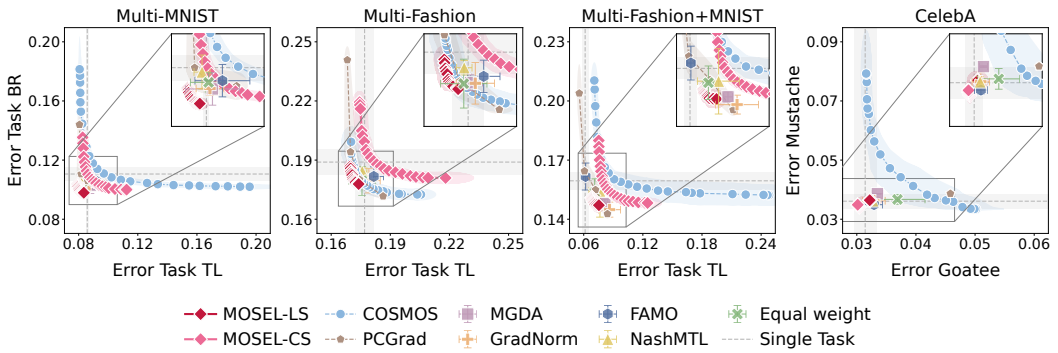

Figure 3: **Two-task setting.** Test-time Pareto fronts showing the trade-offs between task-specific errors. Grey dotted lines indicate the best achievable single-task performance on each task; their intersection marks the single-task baseline. In this weakly conflicting setting, positive transfer allows methods to surpass this baseline, with MOSEL achieving the largest improvements.

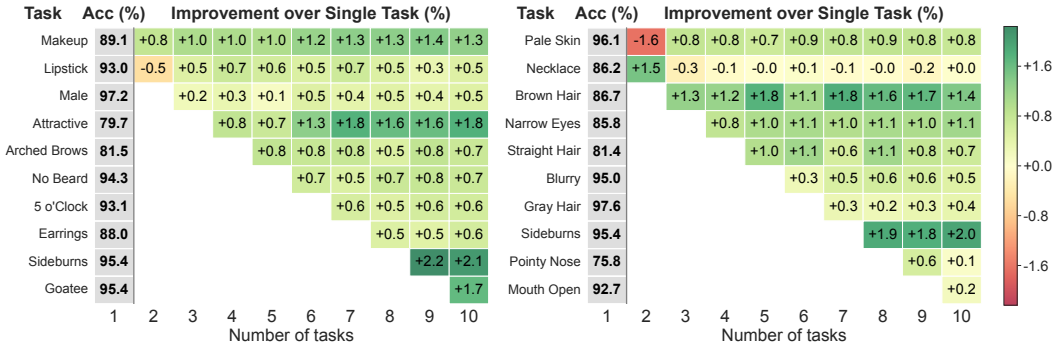

Figure 4: **Scalability to number of objectives on CelebA**. Per-task accuracy improvement over the single-task baseline (gray column) for the most correlated (left) and least correlated (right) task sets, as the number of jointly optimized tasks increases. Results are averaged over 5 seeds and evaluated on the Pareto stationary solution closest to the zero vector.

Baselines and metrics. We compare against COSMOS [53], MGDA [56], PCGrad [67], GradNorm [6], FAMO [35], NashMTL [46], equal weighting and single task performances. While standard PCGrad uses equal weighting, we weigh the objectives based on these preferences to obtain diverse solutions, retraining and evaluating on 5 preferences. As other baselines do not support preference conditioning, we report the single obtained solution. Details on hyperparameters, architectures, and training configurations for all experiments are provided in Appendix B. We train all methods using (binary) cross-entropy; and report misclassification error at test time.

Results on two-task setting. As shown in Figure 3, MOSEL recovers solutions closest to the utopia point on all synthetic benchmarks, consistently outperforming COSMOS, most of the MTL methods, and the equal weight baseline. Notably, even in these weakly conflicting settings, where gradient-based MTL methods are specifically designed to help, MOSEL matches or surpasses their performance, demonstrating its benefits regardless of the degree of gradient conflict.

Method	Δ Param.	Time (×)	CPU (×)
Equal weight	17,495,136	248.4 ± 45.7 min	11.15 ± 0.25 MB
MOSEL-LS	+672	$\times 1.0 \pm 0.1$	$\times 0.97 \pm 0.02$
MOSEL-CS	+672	$\times 1.0 \pm 0.2$	$\times 0.98 \pm 0.02$
COSMOS	+1,072	$\times 1.1 \pm 0.1$	$\times 0.99 \pm 0.02$
PCGrad	+0	$\times 9.8 \pm 2.2$	$\times 1.06 \pm 0.01$
MGDA	+0	$\times 3.0 \pm 0.5$	$\times 0.99 \pm 0.03$
GradNorm	+0	$\times 3.0 \pm 0.0$	$\times 0.95 \pm 0.02$
FAMO	+0	$\times 1.2 \pm 0.1$	$\times 0.96 \pm 0.03$
NashMTL	+0	$\times 2.4 \pm 0.2$	$\times 1.28 \pm 0.01$
Single Task	+0	$\times 2.0 \pm 0.3$	$\times 1.01 \pm 0.03$

Importantly, beyond solution quality, MOSEL also excels in efficiency. The table on the left shows the computational overhead on CelebA relative to equal weight training (top); # Param. are absolute counts; time and CPU are multiplicative factors; and, means and standard deviations are computed over five runs. As reported, MOSEL introduces negligible computational overhead, adding only 672 parameters and matching the training time of equal weight training, while PCGrad incurs a nearly $10\times$ slowdown due to per-preference retraining, and single-task

training scales linearly with the number of tasks. Together, these results confirm that MOSEL recovers diverse and high-quality Pareto fronts at the cost of standard single-objective training.

Results on scalability to number of objectives. To further assess scalability, we conduct an ablation on CelebA comparing correlated and uncorrelated task sets as the number of jointly optimized objectives grows (Figure 4), with tasks added incrementally and performance measured relative to single-task baselines. Task correlation is measured by the co-occurrence frequency of attributes in images. For *correlated tasks* (left panel), shared representations enable strong positive transfer, with accuracy improvements over the single-task baseline growing consistently as more objectives are added. For *uncorrelated tasks* (right panel), positive transfer is weaker, yet MOSEL still outperforms single-task training on the majority of objectives. The occasional degradations can be attributed to two factors: tasks with high single-task accuracy have little room for improvement, and by choosing the Pareto optimal solution closest to the zero vector for evaluation, high-loss tasks dominate the distance computation, while low-loss tasks (high single-task accuracy) contribute minimally. Nonetheless, the overall performance gains come at no additional cost: computational overhead remains negligible as the number of objectives grows, Figure 8 shows that runtime stays approximately constant, with only minor fluctuations; while memory increases only marginally, highlighting the scalability of MOSEL.

6 Conclusion

We propose an efficient *a posteriori* multi-objective framework for deep neural networks that achieves Pareto stationary solutions at the cost of standard single-objective training via a bilevel Stackelberg formulation enabling single-pass updates. Empirically, it recovers diverse Pareto frontiers in strongly conflicting settings and converges closer to the utopia point in weakly conflicting regimes such as multi-task learning, demonstrating that efficient multi-objective learning scales and improves performance by learning more diverse representations to capture all objectives.

Limitations and future work. Despite its advantages, our approach has several limitations that open avenues for future research. First, our results suggest that efficient multi-objective learning can yield richer, more robust representations, especially in weakly conflicting regimes, motivating further study on transfer [49] and alignment [21]. Moreover, our fairness–performance experiments show that our method achieves better trade-offs while providing diverse solutions for informed deployment; extending this to other high-stakes domains such as clinical risk prediction [42], recidivism assessment [37] or credit scoring [29] is an important direction for future work. Second, the method relies on a modular decomposition of neural networks, which may not be straightforward to design or optimal for all architectures; exploring adaptive or learned decompositions is therefore another important avenue. Third, while the computational overhead of MOSEL for recovering a Pareto stationary front is minimal, deploying the model requires selecting a preference vector—and thus a specific trade-off. This decision becomes increasingly challenging as the number of objectives grows and remains an open problem [26]; addressing it via human-in-the-loop methods for multi-objective evaluation and selection is a key direction for future work.

Overall, our results highlight the promise of our approach and, together with these challenges, motivate a broader research agenda at the intersection of scalable multi-objective learning, adaptive model design, and decision-aware deployment of deep neural models.

Acknowledgments and Disclosure of Funding

We would like to thank Nektarios Kalampalikis for the insightful discussions and continuous support throughout this project. This work has been supported by the project "Society-Aware Machine Learning: The paradigm shift demanded by society to trust machine learning," funded by the European Union and led by IV (ERC-2021-STG, SAML, 101040177). Views and opinions expressed are, however, those of the author(s) only and do not necessarily reflect those of the aforementioned funding agencies. Neither of the aforementioned parties can be held responsible for them.

References

[1] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. How we analyzed the COMPAS recidivism algorithm, 2016. URL `https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm`.

[2] Barry Becker and Ronny Kohavi. Adult, 1996. URL `https://doi.org/10.24432/C5XW20`.

[3] Gerard Ben Arous, Reza Gheissari, Jiaoyang Huang, and Aukosh Jagannath. High-dimensional sgd aligns with emerging outlier eigenspaces. In *International Conference on Learning Representations*, volume 2024, pages 47732–47778, 2024.

[4] Weiyu Chen and James Kwok. Multi-objective deep learning with adaptive reference vectors. *Advances in Neural Information Processing Systems*, 35:32723–32735, 2022.

[5] Yongqiang Chen, Kaiwen Zhou, Yatao Bian, Binghui Xie, Bingzhe Wu, Yonggang Zhang, MA KAILI, Han Yang, Peilin Zhao, Bo Han, and James Cheng. Pareto invariant risk minimization: Towards mitigating the optimization dilemma in out-of-distribution generalization. In *The Eleventh International Conference on Learning Representations*, 2023.

[6] 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.

[7] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

[8] cjadams, Daniel Borkan, inversion, Jeffrey Sorensen, Lucas Dixon, Lucy Vasserman, and nithum. Jigsaw unintended bias in toxicity classification. `https://kaggle.com/competitions/jigsaw-unintended-bias-in-toxicity-classification`, 2019. Kaggle.

[9] Carlos A Coello Coello, Gary B Lamont, and David A Van Veldhuizen. *Evolutionary algorithms for solving multi-objective problems*. Springer, 2007.

[10] Benoît Colson, Patrice Marcotte, and Gilles Savard. An overview of bilevel optimization. *Annals of operations research*, 153(1):235–256, 2007.

[11] Yatin Dandi, Matteo Vilucchio, Luca Arnaboldi, Hugo Tabanelli, and Florent Krzakala. Deep learning as neural low-degree filtering: A spectral theory of hierarchical feature learning. *arXiv preprint arXiv:2605.13612*, 2026.

[12] Stephan Dempe. *Foundations of bilevel programming*. Springer, 2002.

[13] Jean-Antoine Désidéri. Multiple-gradient descent algorithm (mgda) for multiobjective optimization. *Comptes Rendus. Mathématique*, 350(5-6):313–318, 2012.

[14] Alexey Dosovitskiy and Josip Djolonga. You only train once: Loss-conditional training of deep networks. In *International conference on learning representations*, 2019.

[15] Matthias Ehrgott. *Multicriteria optimization*. Springer, 2005.

[16] Tanner Fiez, Benjamin Chasnov, and Lillian Ratliff. Implicit learning dynamics in stackelberg games: Equilibria characterization, convergence analysis, and empirical study. In *Proceedings of the 37th International Conference on Machine Learning*, pages 3133–3144. PMLR, 2020.

[17] Jörg Fliege and Benar Fux Svaiter. Steepest descent methods for multicriteria optimization. *Mathematical methods of operations research*, 51(3):479–494, 2000.

[18] Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29, 2016.

[19] Conor F Hayes, Roxana Rădulescu, Eugenio Bargiacchi, Johan Källström, Matthew Macfarlane, Mathieu Reymond, Timothy Verstraeten, Luisa M Zintgraf, Richard Dazeley, Fredrik Heintz, et al. A practical guide to multi-objective reinforcement learning and planning: Cf hayes et al. *Autonomous Agents and Multi-Agent Systems*, 36(1):26, 2022.

[20] Kaiming He, Xiangyu Zhang, Shaoquing Ren, and Jian Sun. Deep residual learning for image recognition. 2016 ieee conf. In *Comput. Vis. Pattern Recognit*, pages 770–778, 2015.

[21] Qiang He and Setareh Maghsudi. Pareto multi-objective alignment for language models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 257–272. Springer, 2025.

[22] Long P Hoang, Dung D Le, Tran Anh Tuan, and Tran Ngoc Thang. Improving pareto front learning via multi-sample hypernetworks. *arXiv e-prints*, pages arXiv–2212, 2022.

[23] Hans Hofmann. Statlog (German Credit Data), 1994. URL `https://archive.ics.uci.edu/dataset/144/statlog+german+credit+data`.

[24] Adrián Javaloy and Isabel Valera. Rotograd: Gradient homogenization in multitask learning. In *International Conference on Learning Representations*, 2022.

[25] Adrian Javaloy, Maryam Meghdadi, and Isabel Valera. Mitigating modality collapse in multimodal VAEs via impartial optimization. In *Proceedings of the 39th International Conference on Machine Learning*, pages 9938–9964. PMLR, 2022.

[26] Adrián Javaloy, Antonio Vergari, and Isabel Valera. Copa: Comparing the incomparable in multi-objective model evaluation. *arXiv preprint arXiv:2503.14321*, 2025.

[27] Kimmo Karkkainen and Jungseock Joo. Fairface: Face attribute dataset for balanced race, gender, and age for bias measurement and mitigation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1548–1558, 2021.

[28] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[29] Nikita Kozodoi, Johannes Jacob, and Stefan Lessmann. Fairness in credit scoring: Assessment, implementation and profit implications. *European Journal of Operational Research*, 297(3): 1083–1094, 2022.

[30] Yann LeCun, Patrick Haffner, Léon Bottou, and Yoshua Bengio. Object recognition with gradient-based learning. In *Shape, contour and grouping in computer vision*, pages 319–345. Springer, 1999.

[31] Chengao Li, Hanyu Zhang, Yunkun Xu, Hongyan Xue, Xiang Ao, and Qing He. Gradient-adaptive policy optimization: Towards multi-objective alignment of large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11214–11232, 2025.

[32] Xi Lin, Hui-Ling Zhen, Zhenhua Li, Qing-Fu Zhang, and Sam Kwong. Pareto multi-task learning. *Advances in neural information processing systems*, 32, 2019.

[33] Xi Lin, Xiaoyuan Zhang, Zhiyuan Yang, Fei Liu, Zhenkun Wang, and Qingfu Zhang. Smooth tchebycheff scalarization for multi-objective optimization. In *International Conference on Machine Learning*, pages 30479–30509. PMLR, 2024.

[34] 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.

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

[36] Erlong Liu, Yu-Chang Wu, Xiaobin Huang, Chengrui Gao, Ren-Jian Wang, Ke Xue, and Chao Qian. Pareto set learning for multi-objective reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 18789–18797, 2025.

[37] Suyun Liu and Luis Nunes Vicente. Accuracy and fairness trade-offs in machine learning: a stochastic multi-objective approach: S. liu, ln vicente. *Computational Management Science*, 19 (3):513–537, 2022.

[38] Suyun Liu and Luis Nunes Vicente. The stochastic multi-gradient algorithm for multi-objective optimization and its application to supervised machine learning. *Annals of Operations Research*, 339(3):1119–1148, 2024.

[39] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[40] Pingchuan Ma, Tao Du, and Wojciech Matusik. Efficient continuous pareto exploration in multi-task learning. In *Proceedings of the 37th International Conference on Machine Learning*, pages 6522–6531. PMLR, 2020.

[41] Debabrata Mahapatra and Vaibhav Rajan. Multi-task learning with user preferences: Gradient descent with controlled ascent in pareto optimization. In *International Conference on Machine Learning*, pages 6597–6607. PMLR, 2020.

[42] N Martinez and M Bertran. Fairness with minimal harm: A pareto-optimal approach for healthcare. *NeurIPS ML4H: Machine Learning for Health*, 2019.

[43] Natalia Martinez, Martin Bertran, and Guillermo Sapiro. Minimax pareto fairness: A multi objective perspective. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 6755–6764. PMLR, 13–18 Jul 2020.

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

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

[46] Aviv Navon, Aviv Shamsian, Idan Achituve, Haggai Maron, Kenji Kawaguchi, Gal Chechik, and Ethan Fetaya. Multi-task learning as a bargaining game. In *International Conference on Machine Learning*, pages 16428–16446. PMLR, 2022.

[47] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[48] Kirtan Padh, Diego Antognini, Emma Lejal-Glaude, Boi Faltings, and Claudiu Musat. Addressing fairness in classification with a model-agnostic multi-objective algorithm. In *Uncertainty in artificial intelligence*, pages 600–609. PMLR, 2021.

[49] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[50] E Perez, F Strub, H De Vries, and V Dumoulin. Visual reasoning with a general conditioning layer, courville. In *In Proceedings of the AAAI Conference on Artificial Intelligence*, 2017.

[51] Yinuo Ren, Tesi Xiao, Michael Shavlovsky, Lexing Ying, and Holakou Rahmanian. Cos-dpo: Conditioned one-shot multi-objective fine-tuning framework. In *Proceedings of the Forty-first Conference on Uncertainty in Artificial Intelligence*, volume 286 of *Proceedings of Machine Learning Research*, pages 3525–3551. PMLR, 21–25 Jul 2025.

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

[53] Michael Ruchte and Josif Grabocka. Scalable pareto front approximation for deep multi-objective learning. In *2021 IEEE international conference on data mining (ICDM)*, pages 1306–1311. IEEE, 2021.

[54] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. *Advances in neural information processing systems*, 30, 2017.

[55] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *NeurIPS EMC^2 Workshop*, 2019.

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

[57] Guangyuan SHI, Qimai Li, Wenlong Zhang, Jiaxin Chen, and Xiao-Ming Wu. Recon: Reducing conflicting gradients from the root for multi-task learning. In *The Eleventh International Conference on Learning Representations*, 2023.

[58] Ankur Sinha, Pekka Malo, and Kalyanmoy Deb. A review on bilevel optimization: From classical to evolutionary approaches and applications. *IEEE transactions on evolutionary computation*, 22(2):276–295, 2017.

[59] Samuel L. Smith and Quoc V. Le. A bayesian perspective on generalization and stochastic gradient descent. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=BJij4yg0Z`.

[60] Samuel L. Smith, Pieter-Jan Kindermans, and Quoc V. Le. Don't decay the learning rate, increase the batch size. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=B1Yy1BxCZ`.

[61] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 09–15 Jun 2019.

[62] Susan Wei and Marc Niethammer. The fairness-accuracy pareto front. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 15(3):287–302, 2022.

[63] Yake Wei and Di Hu. Mmpareto: boosting multimodal learning with innocent unimodal assistance. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24, 2024.

[64] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[65] Haibo Yang, Zhuqing Liu, Jia Liu, Chaosheng Dong, and Michinari Momma. Federated multi-objective learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[66] Mehdi Yazdani-Jahromi, Ali K Yalabadi, AmirArsalan Rajabi, Aida Tayebi, Ivan Garibay, and Ozlem O Garibay. Fair bilevel neural network (fairbinn): on balancing fairness and accuracy via stackelberg equilibrium. *Advances in Neural Information Processing Systems*, 37: 105780–105818, 2024.

[67] 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.

[68] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rogriguez, and Krishna P Gummadi. Fairness constraints: Mechanisms for fair classification. In *Artificial intelligence and statistics*, pages 962–970. PMLR, 2017.

[69] Bohan Zhang, Zihao Wang, Hengyu Fu, and Jason D. Lee. Neural networks learn generic multi-index models near information-theoretic limit. In *The Fourteenth International Conference on Learning Representations*, 2026.

[70] Tianchen Zhou, Fnu Hairi, Haibo Yang, Jia Liu, Tian Tong, Fan Yang, Michinari Momma, and Yan Gao. Finite-time convergence and sample complexity of actor-critic multi-objective reinforcement learning. In *Proceedings of the 41st International Conference on Machine Learning*, pages 61913–61933. PMLR, 2024.

[71] Eckart Zitzler, Dimo Brockhoff, and Lothar Thiele. The hypervolume indicator revisited: On the design of pareto-compliant indicators via weighted integration. In *International conference on evolutionary multi-criterion optimization*, pages 862–876. Springer, 2007.

A Proofs

A.1 Non-stationarity of COSMOS solutions

Theorem 2. *Let G be the objective function of the COSMOS algorithm, defined as:*

$$G = \boldsymbol{r}^T \mathcal{L} - \lambda \frac{\boldsymbol{r}^T \mathcal{L}}{\|\boldsymbol{r}\| \|\mathcal{L}\|} \tag{11}$$

where $\mathcal{L} \in \mathbb{R}^m_{\geq 0}$ is the loss vector, $\boldsymbol{r} \in \mathbb{R}^m_{\geq 0}$ is the target preference vector on the probability simplex, i.e., $\sum r_i = 1$, and $\lambda > 0$ is a scalar hyperparameter. Assume that at any stationary point, the objective gradients exhibit non-degenerate conflict such that rank$(J_{\mathcal{L}}) = m - 1$. As $r_i \to 1$, if the hyperparameter violates the bound $\lambda \leq \frac{\|\mathcal{L}\|^2}{\|\mathcal{L}\| - \mathcal{L}_i}$, the stationary points of G are strictly not Pareto stationary.

Proof. To evaluate the Pareto stationarity of G, we first find its gradient with respect to parameters θ:

$$\nabla_\theta G = J_{\mathcal{L}}^T \boldsymbol{r} - \lambda J_{\mathcal{L}}^T \left[\frac{1}{\|\boldsymbol{r}\| \|\mathcal{L}\|} \left(\boldsymbol{r} - \frac{\boldsymbol{r}^T \mathcal{L}}{\|\mathcal{L}\|^2} \mathcal{L} \right) \right] \tag{12}$$

We can rewrite this by factoring out the Jacobian transpose $J_{\mathcal{L}}^T$ to isolate the effective weighting vector $\boldsymbol{r}'$:

$$\nabla_\theta G = J_{\mathcal{L}}^T \underbrace{\left[\left(1 - \frac{\lambda}{\|\boldsymbol{r}\| \|\mathcal{L}\|} \right) \boldsymbol{r} + \left(\frac{\lambda (\boldsymbol{r}^T \mathcal{L})}{\|\boldsymbol{r}\| \|\mathcal{L}\|^3} \right) \mathcal{L} \right]}_{\boldsymbol{r}'} \tag{13}$$

At a stationary point of the COSMOS algorithm, $\nabla_\theta G = 0$, which implies $J_{\mathcal{L}}^T \boldsymbol{r}' = 0$. By our non-degenerate conflict assumption, the left null space of $J_{\mathcal{L}}$ is 1-dimensional. Therefore, any valid Pareto preference vector $\boldsymbol{r}$ satisfying $J_{\mathcal{L}}^T \boldsymbol{r} = 0$ must be a scalar multiple of $\boldsymbol{r}'$, i.e., $\boldsymbol{r} = c \cdot \boldsymbol{r}'$ for some $c \neq 0$. Consequently, for the point to be Pareto stationary, $\boldsymbol{r}'$ must be either strictly non-negative or strictly non-positive. Since $\boldsymbol{r}^T \mathcal{L} > 0$ and $\mathcal{L} \geq 0$, the vector $\boldsymbol{r}'$ cannot be strictly non-positive, and therefore Pareto stationarity strictly requires $\boldsymbol{r}' \geq 0$.

Enforcing $\boldsymbol{r}' \geq 0$ yields:

$$\mathcal{L} \geq -\left(\|\boldsymbol{r}\| \|\mathcal{L}\| - \lambda \right) \left(\frac{\|\mathcal{L}\|^2}{\lambda (\boldsymbol{r}^T \mathcal{L})} \right) \boldsymbol{r} \, . \tag{14}$$

Rearranging the terms, we get:

$$\mathcal{L} \geq \left(\frac{\lambda - \|\boldsymbol{r}\| \|\mathcal{L}\|}{\lambda} \right) \left(\frac{\|\mathcal{L}\|^2}{\boldsymbol{r}^T \mathcal{L}} \right) \boldsymbol{r} \, . \tag{15}$$

By the Cauchy–Schwarz inequality, $\boldsymbol{r}^T \mathcal{L} \leq \|\boldsymbol{r}\| \|\mathcal{L}\|$. Because $\boldsymbol{r}^T \mathcal{L} > 0$, we invert this to establish a strict lower bound for the fractional term:

$$\frac{1}{\boldsymbol{r}^T \mathcal{L}} \geq \frac{1}{\|\boldsymbol{r}\| \|\mathcal{L}\|} \implies \frac{\|\mathcal{L}\|^2}{\boldsymbol{r}^T \mathcal{L}} \geq \frac{\|\mathcal{L}\|}{\|\boldsymbol{r}\|} \, . \tag{16}$$

Substituting this lower bound back into our inequality, we establish the necessary condition for Pareto stationarity for any component i:

$$\mathcal{L}_i \geq \left(1 - \frac{\|\boldsymbol{r}\| \|\mathcal{L}\|}{\lambda} \right) \frac{\|\mathcal{L}\|}{\|\boldsymbol{r}\|} r_i \, . \tag{17}$$

To simplify the analysis, let $\Delta_i = \|\mathcal{L}\| - \mathcal{L}_i \geq 0$ be the objective deficit. Substituting $\mathcal{L}_i = \|\mathcal{L}\| - \Delta_i$ yields:

$$\|\mathcal{L}\| - \Delta_i \geq \left(1 - \frac{\|\boldsymbol{r}\| \|\mathcal{L}\|}{\lambda} \right) \frac{\|\mathcal{L}\|}{\|\boldsymbol{r}\|} r_i \, . \tag{18}$$

Expanding the right side and moving all terms to the left forms a quadratic inequality with respect to $\|\mathcal{L}\|$:

$$\frac{r_i}{\lambda}\|\mathcal{L}\|^2 + \left(1 - \frac{r_i}{\|\boldsymbol{r}\|}\right)\|\mathcal{L}\| - \Delta_i \geq 0\,. \tag{19}$$

Solving this quadratic inequality for its non-negative root gives the exact necessary lower bound for the total loss norm $\|\mathcal{L}\|$:

$$\|\mathcal{L}\| \geq \frac{\lambda}{2r_i}\left[-\left(1 - \frac{r_i}{\|\boldsymbol{r}\|}\right) + \sqrt{\left(1 - \frac{r_i}{\|\boldsymbol{r}\|}\right)^2 + \frac{4r_i\Delta_i}{\lambda}}\right]\,. \tag{20}$$

We now analyze the behavior of this necessary condition as the preference vector concentrates on objective i, meaning $r_i \to 1$. Because $\boldsymbol{r}$ is on the probability simplex ($\sum r_j = 1$ and $r_j \geq 0$), $r_i \to 1$ implies that all other components $r_j \to 0$, which consequently means $\|\boldsymbol{r}\| \to 1$. Applying this limit to the exact bound:

$$\begin{aligned}
&\lim_{r_i \to 1} \frac{\lambda}{2r_i}\left[-\left(1 - \frac{r_i}{\|\boldsymbol{r}\|}\right) + \sqrt{\left(1 - \frac{r_i}{\|\boldsymbol{r}\|}\right)^2 + \frac{4r_i\Delta_i}{\lambda}}\right] \\
&= \frac{\lambda}{2}\sqrt{\frac{4\Delta_i}{\lambda}} = \frac{\lambda}{2}\cdot\frac{2\sqrt{\Delta_i}}{\sqrt{\lambda}} \\
&= \sqrt{\lambda\Delta_i}\,.
\end{aligned} \tag{21}$$

Therefore, as $r_i \to 1$, the necessary condition for Pareto stationarity simplifies to:

$$\|\mathcal{L}\| \geq \sqrt{\lambda\Delta_i}\,. \tag{22}$$

Since both $\|\mathcal{L}\|$ and $\sqrt{\lambda\Delta_i}$ are non-negative, we can square both sides and isolate the hyperparameter λ:

$$\|\mathcal{L}\|^2 \geq \lambda\Delta_i \implies \lambda \leq \frac{\|\mathcal{L}\|^2}{\Delta_i}\,. \tag{23}$$

Substituting the objective deficit $\Delta_i = \|\mathcal{L}\| - \mathcal{L}_i$ back into the inequality:

$$\lambda \leq \frac{\|\mathcal{L}\|^2}{\|\mathcal{L}\| - \mathcal{L}_i}\,. \tag{24}$$

If this bound is violated, i.e., $\lambda > \frac{\|\mathcal{L}\|^2}{\|\mathcal{L}\| - \mathcal{L}_i}$, no value of $\|\mathcal{L}\|$ can maintain $\boldsymbol{r}' \geq 0$, forcing the effective weighting vector $\boldsymbol{r}'$ to contain negative components. Since our assumption guarantees that $\boldsymbol{r}'$ is the unique vector satisfying $J_{\mathcal{L}}^T\boldsymbol{r}' = 0$, the presence of negative components implies that no non-negative preference vector $\boldsymbol{r} \geq 0$ can exist. Thus, when $r_i \to 1$ and $\lambda > \frac{\|\mathcal{L}\|^2}{\|\mathcal{L}\| - \mathcal{L}_i}$, the stationary point is strictly not Pareto stationary. □

A.2 Stochastic weighting as implicit regularization

Zhang et al. [69] show that, under standard assumptions on the data and architecture, a two-layer neural network trained via layer-wise gradient descent on a Gaussian multi-index model $f(x) = g(Ux)$ learns the hidden subspace U through a power method dynamic. Specifically, they consider

- inputs $x \sim \mathcal{N}(0, I_d)$
- a two-layer network $h(x) = \sum_j a_j\sigma(w_j^\top x)$ with a locally quadratic activation ($\sigma'(0) = 0$, $\sigma''(0) = 1$, bounded third derivative)
- small isotropic initialization $w_j^{(0)} \sim \mathcal{N}(0, \sigma_w^2 I_d)$ with symmetric pairing of neurons such that the network output is zero at initialization and
- fixed outer weights $a_j \in \{\pm 1\}$ during the first training stage.

Near initialization the network output stays close to zero and $\sigma'(w_j^\top x) \approx \sigma''(0) w_j^\top x$, so the gradient w.r.t. w_j is $a_j \sigma''(0) \hat{\Sigma}_l w_j$ where

$$\hat{\Sigma}_l = \frac{1}{n} \sum_{i=1}^{n} l'(0, y_i) x_i x_i^\top \tag{25}$$

is the *empirical discovery matrix*. Under these conditions, gradient descent with learning rate η amd weight decay $1/\eta$ cancels the identity part of the update and yields a power iteration on $\hat{\Sigma}_l$ (using $\sigma''(0) = 1$):

$$w_j^{(t+1)} \approx -\eta a_j \hat{\Sigma}_l w_j^{(t)} \quad \text{so that} \quad w_j^{(T)} \approx (-\eta a_j)^T \hat{\Sigma}_l^T w_j^{(0)}. \tag{26}$$

This implies that after T steps, the expected fraction of a neuron's energy aligned with feature direction u_k scales as $\lambda_k^{2T} / (\sum_i \lambda_i^{2T})$, where λ_k is the corresponding eigenvalue of $\hat{\Sigma}_l$. Features with larger eigenvalues exponentially dominate, and finite-sample noise is filtered out through the spectral gap. We measure how much neuron j has aligned with u_k after T steps by its *feature energy* $\mathcal{E}_k(T) := \langle w_j^{(T)}, u_k \rangle^2$. Since $\mathcal{E}_k(T)$ is a product of T random factors, we summarize it by its geometric mean $Signal_k(T) = \exp\big(\mathbb{E}[\log \mathcal{E}_k(T)]\big)$.

Leveraging the results from Zhang et al. [69], we can show that in MTL stochastic task weights regularize against domination of one task, where the regularization strength is controlled by the variance of the Dirichlet distribution we sample from.

Theorem 3. *Consider two tasks A and B together with the following assumptions:*

- *The discovery matrices Σ_A, Σ_B share the same orthonormal eigenbasis U with task-specific eigenvalues $\lambda_{k,A}$ and $\lambda_{k,B}$ for each feature direction u_k.*
- *At each gradient descent step t, the tasks are combined with stochastic weights $r_A^{(t)}, r_B^{(t)}$ satisfying $r_A^{(t)} + r_B^{(t)} = 1$, drawn i.i.d. from a distribution with expectations $\bar{r}_A, \bar{r}_B$ and shared variance $V_\alpha(r) := Var(r_A) = Var(r_B)$.*
- *Assuming $\lambda_{k,A}, \lambda_{k,B} \geq 0$ with $\bar{\lambda}_k > 0$; let $\bar{\lambda}_k := \bar{r}_A \lambda_{k,A} + \bar{r}_B \lambda_{k,B}$ denote the expected eigenvalue and $\Delta_k := (\lambda_{k,A} - \lambda_{k,B})/\bar{\lambda}_k$ denote the relative task disagreement.*
- *Initialization $w_j^{(0)}$ is isotropic and independent of the task weights.*
- *The outer weights $a_j \in \{\pm 1\}$ are shared by both tasks and fixed.*

Then, under the power iteration regime of Zhang et al. [69], the relative learning ratio between features k and m after T steps satisfies

$$\rho_{k,m}(T) = \left(\frac{\bar{\lambda}_k}{\bar{\lambda}_m} \right)^{2T} \exp\big(-2T \left[\phi_k - \phi_m\right]\big), \tag{27}$$

with ϕ_k as defined in (36)*. If, in addition, $V_\alpha(r) \Delta_k^2 \ll 1$ (small fluctuations), then*

$$\begin{aligned} \rho_{k,m}(T) := \frac{Signal_k(T)}{Signal_m(T)} &= \frac{\exp\left(\mathbb{E}\big[\log |\langle w_j^{(T)}, u_k \rangle|^2\big]\right)}{\exp\left(\mathbb{E}\big[\log |\langle w_j^{(T)}, u_m \rangle|^2\big]\right)} \\ &\approx \left(\frac{\bar{\lambda}_k}{\bar{\lambda}_m} \right)^{2T} \exp\Big(-T \cdot V_\alpha(r) \cdot \big[\Delta_k^2 - \Delta_m^2\big]\Big). \end{aligned} \tag{28}$$

Proof. Under stochastic task weighting, the multi-task discovery matrix at step t is

$$\Sigma_{\text{MT}}^{(t)} = r_A^{(t)} \Sigma_A + r_B^{(t)} \Sigma_B \tag{29}$$

where Σ_A, Σ_B are the empirical discovery matrices for task A and B respectively. Since both tasks share the eigenbasis U, the operator acts as a scalar on each eigendirection

$$\lambda_k^{(t)} = r_A^{(t)} \lambda_{k,A} + r_B^{(t)} \lambda_{k,B} \, . \tag{30}$$

Writing the weight fluctuation as $\delta_t := r_A^{(t)} - \bar{r}_A$ (so that $r_B^{(t)} - \bar{r}_B = -\delta_t$), we obtain

$$\lambda_k^{(t)} = \bar{\lambda}_k + \delta_t(\lambda_{k,A} - \lambda_{k,B}) \,. \tag{31}$$

Note that $\lambda_k^{(t)}$ is a convex combination of $\lambda_{k,A}, \lambda_{k,B} \geq 0$, so $\lambda_k^{(t)} > 0$ almost surely and its logarithm below is well defined.

Since u_k is an eigenvector of both Σ_A and Σ_B, it is an eigenvector of $\Sigma_{\text{MT}}^{(t)}$ for every realization of the task weights. Projecting the power iteration from Eq. (26), using $\Sigma_{\text{MT}}^{(t)}$ in place of $\hat{\Sigma}_l$, onto u_k gives the following recursion

$$\langle w_j^{(t)}, u_k \rangle = -\eta a_j \lambda_k^{(t)} \langle w_j^{(t-1)}, u_k \rangle \tag{32}$$

for the t-th update. Iterating from $t = 1, \ldots, T$, the projection of the weight vector onto u_k after T steps factors as

$$\langle w_j^{(T)}, u_k \rangle = (-\eta a_j)^T \gamma_k \prod_{t=1}^{T} \lambda_k^{(t)}, \tag{33}$$

where $\gamma_k := \langle w_j^{(0)}, u_k \rangle$ is the initial projection. The feature energy aligned with feature k is the square of this projection and since $|a_j| = 1$:

$$\mathcal{E}_k(T) = |\langle w_j^{(T)}, u_k \rangle|^2 = \eta^{2T} \gamma_k^2 \prod_{t=1}^{T} \big(\lambda_k^{(t)}\big)^2. \tag{34}$$

Moving to log-space, we obtain:

$$\log \mathcal{E}_k(T) = 2T \log \eta + \log \gamma_k^2 + 2 \sum_{t=1}^{T} \log \lambda_k^{(t)}. \tag{35}$$

Taking expectations, $\mathbb{E}[\log \lambda_k^{(t)}]$ is the same for every t since the task weights are identically distributed. We define the *stochastic penalty* of feature k as

$$\phi_k := \log \bar{\lambda}_k - \mathbb{E}[\log \lambda_k^{(t)}] \geq 0 \tag{36}$$

which is non-negative by Jensen's inequality, with equality if and only if $\lambda_k^{(t)}$ is almost surely constant, e.g. for a shared feature with $\lambda_{k,A} = \lambda_{k,B}$. Substituting $\mathbb{E}[\log \lambda_k^{(t)}]$ according to Eq. (36) and exponentiating both sides gives the geometric mean signal for feature k:

$$Signal_k(T) = \exp\big(\mathbb{E}[\log \mathcal{E}_k(T)]\big) = \exp\big(\mathbb{E}[\log \gamma_k^2]\big) \cdot (\eta \bar{\lambda}_k)^{2T} \cdot \exp\left(-2T\phi_k\right). \tag{37}$$

Forming the ratio $\rho_{k,m}(T) := Signal_k(T)/Signal_m(T)$, the learning rate η^{2T} cancels identically, and the initialization terms cancel since $\mathbb{E}[\log \gamma_k^2] = \mathbb{E}[\log \gamma_m^2]$. Combining the exponential terms via $e^A/e^B = e^{A-B}$ yields

$$\rho_{k,m}(T) = \left(\frac{\bar{\lambda}_k}{\bar{\lambda}_m}\right)^{2T} \exp\Big(-2T\big[\phi_k - \phi_m\big]\Big). \tag{38}$$

Thus it remains to evaluate ϕ_k and ϕ_m. Substituting $\lambda_k^{(t)} = \bar{\lambda}_k + \delta_t(\lambda_{k,A} - \lambda_{k,B})$ and applying the second-order Taylor expansion

$$\log(\bar{\lambda}_k + x) \approx \log \bar{\lambda}_k + \frac{x}{\bar{\lambda}_k} - \frac{x^2}{2\bar{\lambda}_k^2} \tag{39}$$

with $x = \delta_t(\lambda_{k,A} - \lambda_{k,B})$, each summand becomes:

$$\log \lambda_k^{(t)} \approx \log \bar{\lambda}_k + \delta_t \, \frac{\lambda_{k,A} - \lambda_{k,B}}{\bar{\lambda}_k} - \frac{\delta_t^2}{2} \left(\frac{\lambda_{k,A} - \lambda_{k,B}}{\bar{\lambda}_k}\right)^2. \tag{40}$$

Taking expectations term by term, the zero-order term contributes $\log \bar{\lambda}_k$; the first-order term vanishes since $\mathbb{E}[\delta_t] = 0$; the second-order term contributes $-\frac{1}{2} V_\alpha(r) \Delta_k^2$, using $\mathbb{E}[\delta_t^2] = V_\alpha(r)$ and the definition $\Delta_k := (\lambda_{k,A} - \lambda_{k,B})/\bar{\lambda}_k$. Thus

$$\phi_k \approx \frac{1}{2} V_\alpha(r) \Delta_k^2 \quad \text{and} \quad \phi_m \approx \frac{1}{2} V_\alpha(r) \Delta_m^2 \tag{41}$$

with $V_\alpha(r)\Delta_k^2, V_\alpha(r)\Delta_m^2 \ll 1$ under the small fluctuation assumption.

Subsituting Eq. (41) in Eq. (38) yields Eq. (28). □

Remark 1. Equation (28) decomposes the competition between any two features into two multiplicative forces. The first factor, $(\bar{\lambda}_k/\bar{\lambda}_m)^{2T}$, is a ***deterministic race***: it exponentially favors whichever feature has the larger average eigenvalue across tasks, and it is exactly the dynamics that would arise under fixed (non-stochastic) task weighting. The second factor, $\exp\big(-T \cdot V_\alpha(r)[\Delta_k^2 - \Delta_m^2]\big)$, is a *stochastic penalty*: it exponentially suppresses whichever feature has higher relative task disagreement Δ_k. This penalty arises because multiplicative noise always reduces geometric growth rates, the same mechanism by which SGD's minibatch noise implicitly regularizes in the single-task setting [59]. Crucially, shared features ($\Delta_k \approx 0$) are unaffected by the penalty, while task-specific features ($|\Delta_k| \gg 0$) are actively suppressed, preventing the dominant task from hijacking the network's representational capacity. The sampling variance $V_\alpha(r)$ controls the strength of this regularization, playing a role analogous to the noise scale η/B in SGD implicit regularization [60].

Remark 2 (Regularization overcoming task domination.). Suppose feature k has a larger average eigenvalue than feature m ($\bar{\lambda}_k > \bar{\lambda}_m$), but feature k is task-specific while feature m is shared ($|\Delta_k| > |\Delta_m| \approx 0$). The two factors in (28) pull in opposite directions. Feature m wins the competition (i.e., $\rho_{k,m}(T) < 1$) when the stochastic penalty dominates. Since both factors scale linearly in T in the exponent, the step count cancels and the critical variance at which the transition occurs is:

$$V_\alpha^\star(r) = \frac{2\log(\bar{\lambda}_k/\bar{\lambda}_m)}{\Delta_k^2 - \Delta_m^2}. \tag{42}$$

For $V_\alpha(r) > V_\alpha^\star(r)$, the network aligns more strongly with the weaker but shared feature than with the dominant but task-specific one. For $V_\alpha(r) < V_\alpha^\star(r)$, the deterministic advantage prevails and the task-specific feature dominates, leading to negative transfer. When sampling task weights from a symmetric Dirichlet distribution $\mathrm{Dir}(\alpha, \alpha)$, $V_\alpha(r) = 1/(8\alpha + 4)$ for the two-task case, so lower concentration α increases the regularization strength.

The threshold (42) relies on the second-order approximation (28) and therefore inherits the small-fluctuation assumption $V_\alpha(r)\Delta_k^2 \ll 1$ of Theorem 3. Within this regime, $V_\alpha(r) > V_\alpha^\star(r)$ requires

$$2\log(\bar{\lambda}_k/\bar{\lambda}_m) < V_\alpha(r)\,(\Delta_k^2 - \Delta_m^2) \ll 1, \tag{43}$$

so (42) only describes reversals between features whose average eigenvalues are close, $\bar{\lambda}_k/\bar{\lambda}_m \approx 1$.

To overturn a large eigenvalue gap, the weights must fluctuate strongly, i.e. the variance must be large, which lies outside the validity of the Taylor expansion. In that case we use the exact identity (38), which requires no small-fluctuation assumption and according to which $\rho_{k,m}(T) < 1$ if and only if

$$\log(\bar{\lambda}_k/\bar{\lambda}_m) < \phi_k - \phi_m, \tag{44}$$

again independently of T. For example, let feature k be specific to task A ($\lambda_{k,B} = 0$), let feature m be shared ($\lambda_{m,A} = \lambda_{m,B}$) and let $(r_A^{(t)}, r_B^{(t)}) \sim \mathrm{Dir}(\alpha_A, \alpha_B)$ with $\alpha_0 := \alpha_A + \alpha_B$. Then $\lambda_k^{(t)} = r_A^{(t)}\lambda_{k,A}$ and $\lambda_m^{(t)} = \bar{\lambda}_m$, so that $\phi_m = 0$ and

$$\phi_k = \log \bar{r}_A - \mathbb{E}\big[\log r_A^{(t)}\big]. \tag{45}$$

Here $\bar{r}_A = \alpha_A/\alpha_0$ and $\mathbb{E}[\log r_A^{(t)}] = \psi(\alpha_A) - \psi(\alpha_0)$, where ψ is the digamma function, so the penalty has the closed form

$$\phi_k = \log\frac{\alpha_A}{\alpha_0} - \psi(\alpha_A) + \psi(\alpha_0). \tag{46}$$

The shared feature therefore wins if and only if $\bar{\lambda}_k/\bar{\lambda}_m < \exp(\phi_k)$. In the symmetric case $\mathrm{Dir}(\alpha, \alpha)$, the penalty reduces to $\phi_k = \psi(2\alpha) - \psi(\alpha) - \log 2$. It decreases monotonically in α, diverges as $\alpha \to 0$, and approaches 1 as $\alpha \to \infty$, where it agrees with (42). If feature k is only approximately task-specific or feature m only approximately shared, these expressions hold approximately. Consequently, for any eigenvalue gap $\bar{\lambda}_k/\bar{\lambda}_m$, a sufficiently small concentration lets the shared feature win. However, strong regularization comes at a cost: as $\alpha \to 0$ the weights concentrate on the vertices of the simplex, training degenerates into alternating single-task updates, and every feature with $\phi_k > 0$ is slowed relative to fixed weighting.

Remark 3 (No regularization effect.). If both features exhibit similar levels of task disagreement ($\Delta_k \approx \Delta_m$), then $\Delta_k^2 - \Delta_m^2 \approx 0$ and the stochastic penalty cancels out of the ratio regardless of $V_\alpha(r)$. The competition reduces to the pure deterministic race $\rho_{k,m}(T) \approx (\bar{\lambda}_k/\bar{\lambda}_m)^{2T}$, and the stronger feature dominates exponentially. This is expected: the variance penalty targets *differential* disagreement between features, not disagreement per se. When both features are equally task-specific (or equally shared), the stochastic weighting provides no selective pressure.

Proposition 2 (Worst-case generalization to J tasks)**.** *Consider J tasks that share the same orthonormal eigenbasis U for their discovery matrices, with task-specific eigenvalues $\lambda_{k,1}, \ldots, \lambda_{k,J} \geq 0$ for each feature direction u_k. At each gradient descent step t, the tasks are combined with stochastic weights $r_1^{(t)}, \ldots, r_J^{(t)}$ satisfying $\sum_{i=1}^J r_i^{(t)} = 1$, drawn i.i.d. from a distribution with expectations $\bar{r}_i$ and shared variance $V_\alpha(r) := Var(r_1)$. Let $\bar{\lambda}_k := \sum_{i=1}^J r_i^{(t)} \lambda_{k,i}$ denote the expected eigenvalue and let*

$$\lambda_k^{max} := \max_i \lambda_{k,i} \qquad \lambda_k^{min} := \min_i \lambda_{k,i} \qquad \Delta_k := \frac{\lambda_k^{max} - \lambda_k^{min}}{\bar{\lambda}_k}$$

denote the eigenvalue of most and least favored task for feature k and the worst-case relative disagreement between them. Assuming $\Delta_k, \Delta_m \ll 1$, the relative learning ratio is bounded by

$$\left(\frac{\bar{\lambda}_k}{\bar{\lambda}_m}\right)^{2T} \exp\left(-\frac{T}{4}\Delta_k^2\right) \lesssim \rho_{k,m}(T) \lesssim \left(\frac{\bar{\lambda}_k}{\bar{\lambda}_m}\right)^{2T} \exp\left(\frac{T}{4}\Delta_m^2\right) \tag{47}$$

where $\lesssim$ denotes inequality up to the second-order approximation.

Proof. Since $\lambda_k^{(t)}$ is a convex combination of $\{\lambda_{k,i}\}_i$ it takes values in $[\lambda_k^{min}, \lambda_k^{max}]$ almost surely and $\mathbb{E}[\lambda_k^{(t)}] = \bar{\lambda}_k$. By Popoviciu's inequality,

$$\text{Var}\left(\lambda_k^{(t)}\right) \leq \frac{(\lambda_k^{max} - \lambda_k^{min})^2}{4} = \frac{\Delta_k^2 \bar{\lambda}_k^2}{4}\,. \tag{48}$$

The proof of Theorem 3 up to Eq. (38) uses only the shared eigenbasis, positivity of $\lambda_k^{(t)}$ and identically distributed weights and thus applies to J tasks in the same manner. Hence we can use Eq. (38) and bound ϕ_k from both sides. From below, $\phi_k \geq 0$ by Jensen's inequality. From above, we can repeat the Taylor expansion of Theorem 3 with $\lambda_k^{(t)} - \bar{\lambda}_k$ in place of $\delta_t(\lambda_{k,A} - \lambda_{k,B})$ gives

$$\phi_k \approx \frac{1}{2\bar{\lambda}_k^2} \text{Var}\left(\lambda_k^{(t)}\right) \leq \frac{\Delta_k^2}{8} \tag{49}$$

using Popoviciu's bound in (48). Thus $0 \leq \phi_k \lesssim \Delta_k^2/8$, plugging this to Eq. (37) yields

$$\exp\left(\mathbb{E}[\log \gamma_k^2]\right) \cdot (\eta\bar{\lambda}_k)^{2T} \cdot \exp\left(-\frac{T}{4}\Delta_k^2\right) \lesssim Signal_k(T) \leq \exp\left(\mathbb{E}[\log \gamma_k^2]\right) \cdot (\eta\bar{\lambda}_k)^{2T} \tag{50}$$

The same bounds hold for feature m. To bound the ratio $\rho_{k,m}(T) = Signal_k(T)/Signal_m(T)$, we combine the upper bound on one signal with the lower bound on the other. The prefactors $\exp\left(\mathbb{E}[\log \gamma_k^2]\right)$ and $\exp\left(\mathbb{E}[\log \gamma_m^2]\right)$ cancel because $\mathbb{E}[\log \gamma_k^2] = \mathbb{E}[\log \gamma_m^2]$ by the isotropic initialization, and η^{2T} cancels identically. For the upper bound, dividing the upper bound on $Signal_k(T)$ by the lower bound on $Signal_m(T)$ gives

$$\rho_{k,m}(T) \lesssim \frac{(\eta\bar{\lambda}_k)^{2T}}{(\eta\bar{\lambda}_m)^{2T} \exp\left(-\frac{T}{4}\Delta_m^2\right)} = \left(\frac{\bar{\lambda}_k}{\bar{\lambda}_m}\right)^{2T} \exp\left(\frac{T}{4}\Delta_m^2\right). \tag{51}$$

For the lower bound, dividing the lower bound on $Signal_k(T)$ by the upper bound on $Signal_m(T)$ gives

$$\rho_{k,m}(T) \gtrsim \frac{(\eta\bar{\lambda}_k)^{2T} \exp\left(-\frac{T}{4}\Delta_k^2\right)}{(\eta\bar{\lambda}_m)^{2T}} = \left(\frac{\bar{\lambda}_k}{\bar{\lambda}_m}\right)^{2T} \exp\left(-\frac{T}{4}\Delta_k^2\right). \tag{52}$$

Together, these two inequalities yield (47). □

Remark 4 (Extension and Scope). The theory presented in Theorem 3 extends to broader architectures and settings through a single key structural property: that early-stage gradient descent drives each neuron's parameters via a *power iteration* on a label-weighted second-order operator, such that the projection onto any joint feature direction u_k factors multiplicatively as

$$\langle w_j^{(T)}, u_k \rangle = \eta^T \gamma_k \prod_{t=1}^{T} \lambda_k^{(t)}.$$

We discuss the conditions required by Theorem 3 and how they generalize below.

Fixed outer weights. MOSEL trains the representation network f_θ and the preference alignment network g_ϕ with learning rates satisfying $\eta_\phi \ll \eta_\theta$, following the Stackelberg timescale separation of Fiez et al. [16]. Thus g_ϕ is approximately stationary over the timescale on which θ updates, corresponding to fixed outer weights during the feature learning phase while f_θ executes the power iteration.

Power iteration in representation space. Under the layer-wise decoupling of Dandi et al. [11], which holds for smooth activations with $\sigma''(0) \neq 0$ and vanishing initialization scaling, each layer ℓ independently performs a power iteration on a label-weighted moment operator

$$\hat{C}^{(\ell)} = \tfrac{1}{n} \sum_{\mu=1}^{n} y_\mu z_{\ell-1}(x_\mu) z_{\ell-1}(x_\mu)^\top.$$

Linear scalarization under stochastic preference sampling $r^{(t)} \sim \mathrm{Dir}(\alpha)$ produces at each layer and step an effective operator $\hat{C}^{(\ell,t)} = \sum_i r_i^{(t)} \hat{C}_i^{(\ell)}$, a stochastic convex combination of the per-objective operators. This layer-wise analysis extends naturally to convolutional and other structured architectures, where the spectral operators can be equivalently defined over the channel space.

Shared eigenbasis. Requiring a common eigenbasis is equivalent to requiring that the per-task discovery matrices Σ_A and Σ_B commute. Shared inputs alone do not guarantee this. It holds when the tasks depend on a common set of orthonormal feature directions and each task's discovery matrix is diagonal in that set. This includes tasks whose relevant subspaces coincide, tasks whose subspaces are mutually orthogonal, and tasks that draw on different subsets of a shared feature dictionary. It generally fails when the task subspaces overlap at oblique angles. The assumption is therefore most natural in multi-task learning, where all tasks are defined on the same inputs and differ only in their targets. In fairness settings it is more restrictive and does not hold in general. We further require the tasks to agree in sign on each feature's eigenvalue. If they do not, the combined eigenvalue $\lambda_k^{(t)}$ can reach zero, and the corresponding direction is erased rather than merely slowed down.

Loss function. The power-iteration mechanism requires only that early training iterates some label-dependent operator; the loss function affects the dynamics solely through the form of this operator. Under general losses, the relevant spectral object generalizes to a transformed correlation operator $\hat{\Sigma}_g = \frac{1}{n} \sum_{i=1}^{n} g(y_i) x_i x_i^\top$, where $g(y)$ is determined by the loss gradient [11]. The core stochastic mitigation of task domination is therefore decoupled from the squared loss. While an explicit multiplicative formula for classification losses remains an open challenge, the high-dimensional analysis of Ben Arous et al. [3] provides rigorous support: SGD trajectories concentrate layer by layer on the low-rank outlier eigenspaces of the empirical Hessian and gradient matrices, confirming that an eigenvalue-ordered selection process governs representations under classification losses and suggesting our analysis extends to classification regimes in future work.

B Experimental details

B.1 Experimental setup

General settings. All datasets use a 70/10/20 train/validation/test split. We use early-stopping based on validation hypervolume across all methods. For a fair comparison, all methods share the same architecture. MOSEL additionally requires splitting the network into a representation learning and a preference alignment component, as described in the respective subsections below. For COSMOS, we use all hyperparameters as reported in the original paper.

Unless stated otherwise, all methods share the same training hyperparameters, including the Dirichlet sampling parameter $\boldsymbol{\alpha}$, number of epochs, and batch size, which mostly follow the setup in [53]. All methods use Adam [28] as the optimizer. For MOSEL, we use separate Adam optimizers for the leader and follower. To ensure convergence to an equilibrium point, the leader learning rate must be smaller than the follower's, we found that one order of magnitude difference works best. Based on that, we select the learning rate combination yielding the best validation hypervolume. Learning rates are decayed by a factor of 0.1 at dataset-specific milestones. For all other methods, hyperparameters are selected via grid search over validation hypervolume. The smooth Chebychev scalarization [33] additionally requires a smoothing parameter τ, set to 0.01 across all datasets, and the utopia point $\boldsymbol{u}^\star$. Since $\boldsymbol{u}^\star$ is generally unknown, we approximate it with a running minimum over objectives observed during training.

All experiments were conducted on an internal computing cluster with an automatic job assignment system, thus we cannot ensure the specific CPU and GPU used for each particular experiment. However, all jobs were submitted with defined resource requirements: multi-dataset experiments requested 1 GPU and 1 CPU, FairFace and CelebA experiments requested 1 GPU and 4 CPUs, and tabular experiments requested no GPU and 2 CPUs. The required memory for each experiment is reported in the respective tables.

Our implementation can be found at `https://github.com/ElinaCelik/mosel`.

B.1.1 Tabular datasets

Model. For the fairness use case, we consider three tabular datasets: Adult, COMPAS and Default. Following prior work [53], we use an MLP with two hidden layers and hidden dimensions 60 and 25: `Linear(`d`, 60)` $\rightarrow$ `ReLU` $\rightarrow$ `Linear(60, 25)` $\rightarrow$ `ReLU` $\rightarrow$ `Linear(25, 1)`.

Hyperparameters. The model is trained for 50 epochs with a batch size of 256, using $\boldsymbol{\alpha} = [0.5, 0.5]$ for Adult and COMPAS and $\boldsymbol{\alpha} = [0.1, 0.5]$ for Default, following the values of [53]. The COSMOS learning rate is 10^{-3} for all tabular datasets without a learning rate scheduler. For MOSEL, we use leader and follower learning rates of $[10^{-3}, 10^{-2}]$ for Adult and Default, and $[10^{-2}, 10^{-1}]$ for COMPAS. The learning rate is decayed by a factor of 0.1 at epochs 15 and 40 for Adult, epochs 30 and 40 for COMPAS, and epochs 30 and 45 for Default.

Splitting the architecture. This architecture provides two natural split points, namely before the second or third linear layer. Empirically, splitting before the last linear layer yields insufficient capacity in the preference alignment network to meaningfully incorporate the preference vector. Thus we split before the second linear layer across all datasets.

B.1.2 Multi-Image classification

Model. For the Multi-MNIST, Multi-Fashion and Multi-Fashion+MNIST datasets we adopt the standard LeNet architecture [30].

Hyperparameters. Following the values of [53], the model is trained for 100 epochs with a batch size of 256 and $\boldsymbol{\alpha} = [1.2, 1.2]$. The COSMOS learning rate is 10^{-3}, decayed by a factor of 0.1 at epochs 20, 40, 80, and 90. For MOSEL, we use leader and follower learning rates of $[10^{-4}, 10^{-3}]$ with the same scheduler milestones. A weight decay of 10^{-3} is applied to the leader to keep the impact of leader and follower balanced and improve training stability.

Splitting the architecture. We split the network immediately after flattening, using the convolutional trunk for representation learning which produces a 720-dimensional embedding z, that is then passed to the subsequent shared linear layer and two task-specific heads that form the preference alignment network.

B.1.3 CelebA

Model. For CelebA, we use the pretrained EfficientNet-B4 [61].

Hyperparameters. Again, we follow the values of [53], thus the model is finetuned for 30 epochs with a batch size of 32 and $\boldsymbol{\alpha} = [1, 1]$. The COSMOS learning rate is 5×10^{-4}, for MOSEL, we use leader and follower learning rates of $[5 \times 10^{-5}, 5 \times 10^{-4}]$.

Splitting the architecture. The modular block structure allows for many candidate split points. We select the split parameter by cross-validation on validation hypervolume. The optimal split was found after the 16th MBConv block, such that the convolutional stem and first 16 MBConv blocks form the representation learning network. The remaining MBConv blocks together with the convolutional head, pooling, and two-layer MLP task heads form the preference alignment network.

B.1.4 FairFace

Model. For FairFace we use a pretrained ResNet-34 [20] following [27].

Hyperparameters. Since FairFace was not used in [53], all hyperparameters are selected via cross-validation on validation hypervolume. The model is finetuned for 30 epochs with a batch size of 128 and $\boldsymbol{\alpha} = [0.5, 0.5]$. The COSMOS hyperparameters are $\lambda = 0.1$ and learning rate of 10^{-4}, decayed by a factor of 0.1 at epochs 20. For MOSELwe use leader and follower learning rates of $[10^{-5}, 10^{-4}]$ with the same scheduler milestones.

Splitting the architecture. Possible split points are restricted to boundaries between residual stages (layer1 - layer4), which gives a total of four possible split poitns. Again, we cross-validate this parameter on validation hypervolume. The optimal split was found after layer3, thus the convolutional stem together with the first three residual stages form the representation learning network, and layer4 together with the two-layer MLP task heads form the preference alignment network.

B.2 Hypervolume computation

The hypervolume indicator measures the volume of the objective space dominated by a set of solutions and bounded by a reference point [71]. As illustrated in Fig. 5, we compute it stepwise as a sum of rectangles, where each solution on the approximated Pareto front contributes a rectangular region bounded by its objective values and the nadir point. To enable comparison across datasets and methods, we report the *relative hypervolume*, defined as the ratio of the achieved hypervolume (shown in blue) to the maximum possible hypervolume enclosed between the utopia and nadir points (shown in gray). The nadir point is set per dataset as the largest objective value observed across all methods plus a margin of 0.05, ensuring all solutions are strictly dominated. The utopia point is task-dependent: for multi-task learning it is set to (0,0), representing zero loss or zero error on all tasks, while for fairness it is set to (min error/min loss, 0), reflecting the best achievable error/loss at zero discrimination. A relative hypervolume of 1 thus corresponds to a solution set that perfectly covers the entire trade-off region between the utopia and nadir points. Note that the utopia point is in general not achievable in multi-objective optimization, as it would minimize all objectives simultaneously.

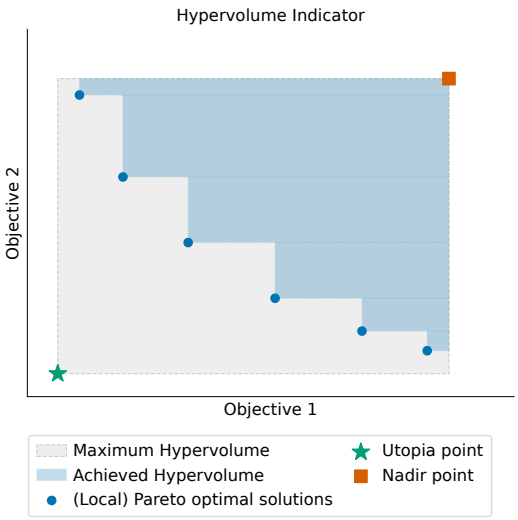

Figure 5: Hypervolume computation

B.3 Conditioning on the preference vector

For the tabular datasets, we concatenate the preference vector $\boldsymbol{r}$ with the learned representation z and pass both directly to the preference alignment network, which is sufficient when no spatial structure is present. For image data on the other hand, we condition on the preference vector $\boldsymbol{r}$ using a Feature-wise Linear Modulation (FiLM) layer [50], which has been shown to enable effective preference-dependent representation modulation in MOO settings [4, 14]. Concretely, a linear layer projects the preference vector $\boldsymbol{r}$ to channel-wise scale and shift parameters $(\boldsymbol{\gamma}, \boldsymbol{\beta})$, which modulate the representation $\boldsymbol{z}$ channel-wise as $\boldsymbol{z}' := \boldsymbol{\gamma}(\boldsymbol{r}) \odot \boldsymbol{z} + \boldsymbol{\beta}(\boldsymbol{r})$. This requires adding a single linear layer $\texttt{Linear}(dim(\boldsymbol{r}),\ 2 \cdot dim(\boldsymbol{z}))$, introducing only $2 \cdot dim(\boldsymbol{z}) \cdot (dim(\boldsymbol{r}) + 1)$ additional parameters, which are a negligible overhead relative to the full network.

C Additional experimental results

C.1 Fairness use case

Figure 6 shows the Pareto fronts for all fairness datasets on the training objectives, complementing the main paper results. Table 2 reports the computational overhead of each method relative to the Classifier baseline in terms of parameter count, training time, CPU memory, and GPU memory. Both MOSEL variants add substantially fewer parameters than COSMOS across all datasets (+50 vs. +120 for tabular, +1,536 vs. +6,993 for FairFace), while incurring comparable or lower training time and memory overhead. GPU memory is reported for FairFace, tabular datasets were trained on CPU only.

Table 2: Computational overhead on fairness datasets relative to classifier (bottom); # Param. are absolute counts, time and CPU are multiplicative factors. Mean $\pm$ std over five runs.

Dataset	Method	Δ#Param.	Time (rel.)	CPU Mem (rel.)	GPU Mem (rel.)
Adult	MOSEL-LS	+50	1.2 ± 0.2	1.10 ± 0.07	–
	MOSEL-CS	+50	1.2 ± 0.1	1.06 ± 0.06	–
	COSMOS	+120	1.2 ± 0.2	1.09 ± 0.03	–
	Classifier	6,891	44.4 ± 4.4 s	10.30 ± 0.34 MB	–
COMPAS	MOSEL-LS	+50	1.5 ± 0.7	1.00 ± 0.03	–
	MOSEL-CS	+50	1.5 ± 0.7	0.96 ± 0.03	–
	COSMOS	+120	1.1 ± 0.0	0.99 ± 0.03	–
	Classifier	2,811	11.5 ± 1.0 s	11.43 ± 0.46 MB	–
Default	MOSEL-LS	+50	1.2 ± 0.3	0.91 ± 0.05	–
	MOSEL-CS	+50	1.0 ± 0.1	0.92 ± 0.02	–
	COSMOS	+120	1.0 ± 0.1	0.91 ± 0.01	–
	Classifier	7,011	33.2 ± 5.0 s	12.22 ± 0.48 MB	–
FairFace	MOSEL-LS	+1,536	0.7 ± 0.0	0.88 ± 0.04	0.50 ± 0.00
	MOSEL-CS	+1,536	1.2 ± 0.6	0.93 ± 0.09	0.56 ± 0.07
	COSMOS	+6,993	1.6 ± 0.6	1.05 ± 0.11	0.58 ± 0.07
	Classifier	21,293,393	2927.9 ± 1425.5 s	8.89 ± 0.04 MB	8929.68 ± 536.47 MB

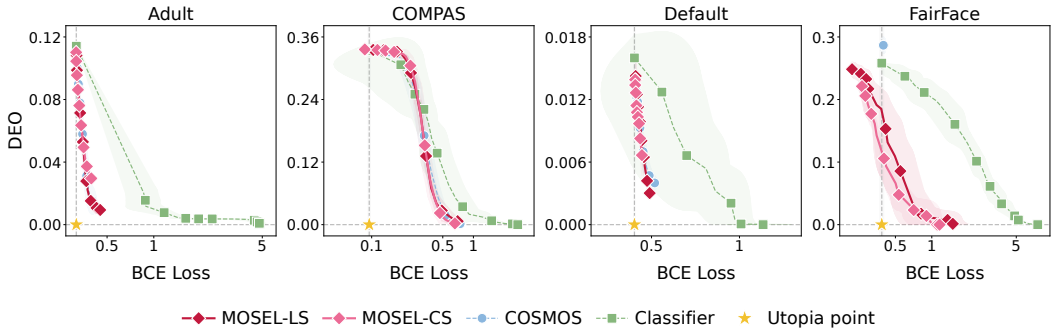

Figure 6: Test-time Pareto fronts on the fairness use case, showing training objectives (BCE loss vs DEO). The grey dotted lines mark zero unfairness and the best BCE loss achievable by an unconstrained classifier; their intersection defines the utopia point, the ideal but infeasible solution, toward which all methods aim to converge.

C.2 MTL use case

Figure 7 shows the Pareto fronts for all MTL datasets on the training objectives, complementing the main paper results. Table 3 reports per-task accuracy at the solution closest to the utopia point, and Table 5 reports hypervolume and number of Pareto optimal solutions for both loss and error metrics. Both MOSEL variants consistently achieve higher hypervolume than COSMOS and the other

baselines across most datasets, with MOSEL-LS performing particularly strongly on the synthetic datasets and MOSEL-CS on CelebA.

While COSMOS tends to produce more Pareto optimal solutions, MOSEL variants achieve better overall front quality as measured by hypervolume. Table 4 reports computational overhead relative to the Equal weight baseline. Both MOSEL variants add more parameters than COSMOS (+4,320 vs. +708), which is attributable to the FiLM conditioning layer applied to the 720-dimensional representation z in the LeNet architecture. However, this additional parameter count is solely due to preference conditioning and does not translate into meaningful increases in training time or memory consumption, with both remaining comparable to COSMOS and the Equal weight baseline.

Table 3: Per-task accuracy at the solution closest to the utopia point on the test Pareto front, across all MTL benchmarks. As joint training enables positive transfer, we take $(0, 0)$ as the utopia point and select the solution minimizing Euclidean distance to it, which can lie beyond the single-task baseline. TL and BR denote top-left and bottom-right tasks on the synthetic benchmarks. Mean $\pm$ std over five independent runs; best results per column shaded.

	Multi-MNIST		Multi-Fashion		Multi-Fashion+MNIST		CelebA	
Method	Acc TL	Acc BR	Acc TL	Acc BR	Acc TL	Acc BR	Acc Goatee	Acc Mustache
MOSEL-LS	0.919 ± 0.001	0.901 ± 0.002	0.829 ± 0.003	0.821 ± 0.001	0.930 ± 0.001	0.851 ± 0.003	0.968 ± 0.002	0.964 ± 0.001
MOSEL-CS	0.910 ± 0.002	0.895 ± 0.002	0.817 ± 0.004	0.812 ± 0.003	0.916 ± 0.003	0.845 ± 0.003	0.970 ± 0.001	0.966 ± 0.001
COSMOS	0.907 ± 0.003	0.886 ± 0.002	0.824 ± 0.004	0.821 ± 0.004	0.915 ± 0.004	0.837 ± 0.005	0.960 ± 0.005	0.959 ± 0.004
PCGrad	0.914 ± 0.006	0.894 ± 0.004	0.814 ± 0.004	0.828 ± 0.001	0.928 ± 0.003	0.845 ± 0.007	0.954 ± 0.000	0.961 ± 0.000
MGDA	0.912 ± 0.009	0.897 ± 0.006	0.822 ± 0.004	0.820 ± 0.002	0.919 ± 0.002	0.852 ± 0.002	0.967 ± 0.001	0.961 ± 0.000
GradNorm	0.913 ± 0.005	0.897 ± 0.002	0.821 ± 0.006	0.820 ± 0.006	0.914 ± 0.011	0.855 ± 0.004	0.967 ± 0.002	0.964 ± 0.001
FAMO	0.909 ± 0.010	0.894 ± 0.006	0.818 ± 0.005	0.818 ± 0.005	0.938 ± 0.002	0.838 ± 0.007	0.967 ± 0.001	0.965 ± 0.001
NashMTL	0.916 ± 0.003	0.891 ± 0.006	0.824 ± 0.002	0.816 ± 0.002	0.924 ± 0.011	0.846 ± 0.013	0.967 ± 0.001	0.964 ± 0.001
Equal weight	0.913 ± 0.006	0.895 ± 0.003	0.824 ± 0.007	0.820 ± 0.007	0.929 ± 0.003	0.846 ± 0.007	0.963 ± 0.005	0.963 ± 0.002
Single Task	0.914 ± 0.001	0.889 ± 0.004	0.823 ± 0.005	0.811 ± 0.006	0.939 ± 0.003	0.841 ± 0.004	0.969 ± 0.002	0.964 ± 0.002

Table 4: Computational overhead on MTL datasets relative to equal weight training (bottom); # Param. are absolute counts, time and CPU are multiplicative factors. Mean $\pm$ std over five runs.

Dataset	Method	Δ#Param.	Time (rel.)	CPU Mem (rel.)	GPU Mem (rel.)
Multi-MNIST	MOSEL-LS	+4,320	1.1 ± 0.2	1.15 ± 0.05	1.01 ± 0.00
	MOSEL-CS	+4,320	1.2 ± 0.2	1.09 ± 0.09	1.00 ± 0.00
	COSMOS	+708	1.3 ± 0.2	1.07 ± 0.02	1.09 ± 0.03
	PCGrad*	+0	6.2 ± 0.6	1.22 ± 0.04	1.00 ± 0.00
	MGDA	+0	2.0 ± 0.4	1.04 ± 0.06	1.12 ± 0.02
	GradNorm	+0	1.9 ± 0.2	1.07 ± 0.05	1.00 ± 0.00
	FAMO	+0	1.2 ± 0.1	1.04 ± 0.04	1.00 ± 0.00
	NashMTL	+0	6.7 ± 1.0	0.76 ± 0.02	1.01 ± 0.00
	Single Task*	+0	1.6 ± 0.1	1.14 ± 0.04	1.00 ± 0.00
	Equal weight	42,350	428.8 ± 75.1 s	14.65 ± 1.18 MB	63.97 ± 0.00 MB
Multi-Fashion	MOSEL-LS	+4,320	1.1 ± 0.3	1.13 ± 0.04	0.99 ± 0.04
	MOSEL-CS	+4,320	1.1 ± 0.1	1.06 ± 0.06	0.96 ± 0.09
	COSMOS	+708	1.5 ± 0.2	1.07 ± 0.08	1.11 ± 0.04
	PCGrad*	+0	7.2 ± 1.5	1.25 ± 0.05	1.00 ± 0.00
	MGDA	+0	2.1 ± 0.4	1.01 ± 0.04	1.12 ± 0.02
	GradNorm	+0	2.0 ± 0.2	1.09 ± 0.09	1.00 ± 0.00
	FAMO	+0	1.2 ± 0.1	1.10 ± 0.08	1.00 ± 0.00
	NashMTL	+0	7.1 ± 1.1	0.76 ± 0.02	1.01 ± 0.00
	Single Task*	+0	1.7 ± 0.1	1.14 ± 0.04	1.00 ± 0.00
	Equal weight	42,350	401.7 ± 45.8 s	14.66 ± 0.90 MB	63.97 ± 0.01 MB
Multi-Fashion+MNIST	MOSEL-LS	+4,320	1.4 ± 0.1	1.01 ± 0.02	1.01 ± 0.00
	MOSEL-CS	+4,320	1.4 ± 0.1	0.96 ± 0.07	1.00 ± 0.00
	COSMOS	+708	1.5 ± 0.2	0.97 ± 0.05	1.09 ± 0.03
	PCGrad*	+0	7.4 ± 1.0	1.15 ± 0.03	1.00 ± 0.00
	MGDA	+0	1.9 ± 0.6	0.93 ± 0.07	1.13 ± 0.01
	GradNorm	+0	2.1 ± 0.2	0.96 ± 0.05	1.00 ± 0.00
	FAMO	+0	1.3 ± 0.1	0.98 ± 0.06	1.00 ± 0.00
	NashMTL	+0	7.7 ± 1.3	0.71 ± 0.01	1.01 ± 0.00
	Single Task*	+0	1.9 ± 0.0	1.01 ± 0.05	1.00 ± 0.00
	Equal weight	42,350	377.7 ± 55.1 s	15.97 ± 0.82 MB	63.97 ± 0.00 MB

*PCGrad requires an independent run for each of the five preference vectors, SingleTask requires an independent run for each task

Table 5: Relative hypervolume and number of Pareto stationary solutions on the MTL benchmarks, computed on training loss and test error. Mean ± std over five runs; best entries per column shaded.

Dataset	Method	HV Loss ↑	HV Error ↑	# PS sol. (Loss) ↑	# PS sol. (Error) ↑
Multi-MNIST	MOSEL-LS	0.330 ± 0.010	0.441 ± 0.008	10.6 ± 0.8	9.0 ± 0.6
	MOSEL-CS	0.318 ± 0.008	0.426 ± 0.009	17.8 ± 0.4	16.8 ± 1.2
	COSMOS	0.303 ± 0.008	0.418 ± 0.007	20.0 ± 1.1	19.0 ± 3.0
	PCGrad	0.274 ± 0.017	0.425 ± 0.010	2.2 ± 0.7	2.8 ± 0.4
	MGDA	0.267 ± 0.026	0.409 ± 0.034	1.0 ± 0.0	1.0 ± 0.0
	GradNorm	0.281 ± 0.010	0.410 ± 0.016	1.0 ± 0.0	1.0 ± 0.0
	FAMO	0.265 ± 0.029	0.394 ± 0.033	1.0 ± 0.0	1.0 ± 0.0
	NashMTL	0.271 ± 0.015	0.401 ± 0.021	1.0 ± 0.0	1.0 ± 0.0
	Equal weight	0.265 ± 0.008	0.406 ± 0.022	1.0 ± 0.0	1.0 ± 0.0
	Single Task	0.264 ± 0.011	0.393 ± 0.010	1.0 ± 0.0	1.0 ± 0.0
Multi-Fashion	MOSEL-LS	0.094 ± 0.004	0.171 ± 0.004	10.8 ± 1.0	8.8 ± 2.9
	MOSEL-CS	0.086 ± 0.004	0.158 ± 0.007	17.4 ± 0.8	16.8 ± 1.3
	COSMOS	0.097 ± 0.006	0.177 ± 0.010	17.8 ± 0.7	16.4 ± 1.9
	PCGrad	0.070 ± 0.007	0.179 ± 0.005	2.2 ± 0.7	3.6 ± 0.5
	MGDA	0.073 ± 0.004	0.158 ± 0.007	1.0 ± 0.0	1.0 ± 0.0
	GradNorm	0.073 ± 0.002	0.155 ± 0.004	1.0 ± 0.0	1.0 ± 0.0
	FAMO	0.071 ± 0.004	0.150 ± 0.005	1.0 ± 0.0	1.0 ± 0.0
	NashMTL	0.069 ± 0.004	0.154 ± 0.005	1.0 ± 0.0	1.0 ± 0.0
	Equal weight	0.072 ± 0.009	0.161 ± 0.019	1.0 ± 0.0	1.0 ± 0.0
	Single Task	0.072 ± 0.008	0.146 ± 0.013	1.0 ± 0.0	1.0 ± 0.0
Multi-Fashion+MNIST	MOSEL-LS	0.295 ± 0.012	0.376 ± 0.012	10.4 ± 0.8	8.6 ± 5.0
	MOSEL-CS	0.281 ± 0.010	0.360 ± 0.009	17.4 ± 0.5	19.4 ± 1.9
	COSMOS	0.270 ± 0.016	0.350 ± 0.016	22.0 ± 1.3	19.0 ± 1.1
	PCGrad	0.263 ± 0.010	0.401 ± 0.008	2.4 ± 0.8	3.8 ± 0.4
	MGDA	0.237 ± 0.009	0.354 ± 0.008	1.0 ± 0.0	1.0 ± 0.0
	GradNorm	0.256 ± 0.011	0.355 ± 0.008	1.0 ± 0.0	1.0 ± 0.0
	FAMO	0.244 ± 0.016	0.344 ± 0.019	1.0 ± 0.0	1.0 ± 0.0
	NashMTL	0.238 ± 0.040	0.346 ± 0.049	1.0 ± 0.0	1.0 ± 0.0
	Equal weight	0.245 ± 0.015	0.354 ± 0.019	1.0 ± 0.0	1.0 ± 0.0
	Single Task	0.254 ± 0.018	0.351 ± 0.010	1.0 ± 0.0	1.0 ± 0.0
CelebA	MOSEL-LS	0.555 ± 0.013	0.531 ± 0.019	22.8 ± 3.0	3.2 ± 1.2
	MOSEL-CS	0.605 ± 0.004	0.556 ± 0.007	8.0 ± 6.7	4.6 ± 1.7
	COSMOS	0.565 ± 0.010	0.536 ± 0.008	25.0 ± 0.0	13.0 ± 2.4
	PCGrad	0.214 ± 0.000	0.427 ± 0.000	1.8 ± 0.7	1.0 ± 0.0
	MGDA	0.248 ± 0.089	0.510 ± 0.006	1.0 ± 0.0	1.0 ± 0.0
	GradNorm	0.508 ± 0.012	0.525 ± 0.016	1.0 ± 0.0	1.0 ± 0.0
	FAMO	0.506 ± 0.006	0.532 ± 0.012	1.0 ± 0.0	1.0 ± 0.0
	NashMTL	0.285 ± 0.057	0.527 ± 0.010	1.0 ± 0.0	1.0 ± 0.0
	Equal weight	0.420 ± 0.120	0.496 ± 0.036	1.0 ± 0.0	1.0 ± 0.0
	Single Task	0.491 ± 0.096	0.536 ± 0.021	1.0 ± 0.0	1.0 ± 0.0

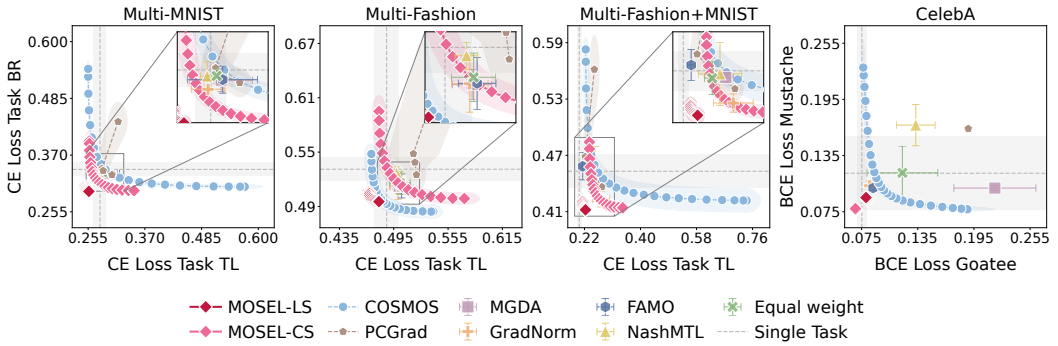

Figure 7: Test-time Pareto fronts on the MTL use case, showing the trade-offs between task-specific (Binary) Cross-Entropy losses. Grey dotted lines indicate best achievable single-task performance on each task; their intersection marks the single-task baseline. In this weakly conflicting setting, positive transfer allows methods to surpass this baseline, with MOSEL achieving the largest improvements.

D Ablations

D.1 Dirichlet parameter α

To isolate the effect of the Dirichlet parameter α on the quality and diversity of the recovered Pareto front, we conduct an ablation study by varying α across four datasets, two from each use case, under both linear scalarization (LS) and Chebyshev scalarization (CS). Results are reported in Table 6.

The sensitivity to α varies across datasets and scalarization functions. In strongly conflicting settings, performance degrades substantially as α increases, particularly under LS: on Adult, HV drops from 0.939 at $\alpha = 0.5$ to 0.816 at $\alpha = 1.2$, while on COMPAS the decline is even more dramatic (from 0.920 to 0.164), indicating a near-total failure to discover diverse trade-off solutions when preference samples concentrate near the simplex center. Under CS, Adult follows a similar trend (HV from 0.819 to 0.524), whereas COMPAS remains remarkably stable with HV above 0.920 across all α values. In contrast, the two MTL datasets are largely insensitive to α: on Multi-Fashion, HV and HV MCR remain effectively constant, and on Multi-Fashion+MNIST, only a mild upward trend is observed as α increases.

Taken together, these results highlight two key findings. First, the impact of α is most pronounced on datasets where the objectives are strongly conflicting, and where the scalarization function is sensitive to the geometry of the preference simplex (LS more so than CS). Second, even on tasks where hypervolume remains stable, increasing α systematically contracts the extreme solutions, reducing the diversity of the recovered front—an effect that aggregate metrics can obscure. For the MTL setting, this behavior aligns with the theoretical analysis in Appendix A.2, which shows that smaller α values (below 1) increase the variance of stochastic preference sampling, strengthening the implicit bias toward learning features shared across objectives and thereby promoting greater Pareto-front diversity. On the strongly conflicting benchmarks, the effect of α is empirically even more pronounced, yet falls outside the assumptions of our current theoretical framework—extending the analysis to such settings remains an interesting direction for future work. Overall, $\alpha = 0.5$ serves as a robust default: it achieves the best or near-best hypervolume across all dataset–scalarization combinations and consistently recovers the widest spread of extreme solutions.

Table 6: **Ablation on Dirichlet parameter α.** Relative hypervolume on two strongly and two weakly conflicting datasets computed on training loss (HV) and test error (HV MCR). Mean $\pm$ std over five runs. Mean extreme solutions (Obj_1-best, Obj_2-best) are reported for varying α and scalarization.

Dataset	Scalarization	α	HV $\uparrow$	HV MCR $\uparrow$	Obj_1-best (mean)	Obj_2-best (mean)
Adult	LS	0.5	0.939 ± 0.007	0.849 ± 0.012	(0.143,0.106)	(0.211,0.009)
		0.8	0.912 ± 0.011	0.838 ± 0.019	(0.144,0.105)	(0.202,0.014)
		1.2	0.816 ± 0.015	0.774 ± 0.009	(0.143,0.105)	(0.186,0.030)
	CS	0.5	0.819 ± 0.017	0.776 ± 0.015	(0.142,0.106)	(0.188,0.030)
		0.8	0.702 ± 0.019	0.683 ± 0.022	(0.142,0.105)	(0.172,0.049)
		1.2	0.524 ± 0.023	0.525 ± 0.025	(0.142,0.106)	(0.153,0.079)
COMPAS	LS	0.5	0.920 ± 0.019	0.568 ± 0.008	(0.021,0.334)	(0.439,0.007)
		0.8	0.541 ± 0.193	0.434 ± 0.117	(0.021,0.334)	(0.218,0.169)
		1.2	0.164 ± 0.051	0.161 ± 0.049	(0.020,0.335)	(0.033,0.324)
	CS	0.5	0.930 ± 0.007	0.560 ± 0.005	(0.021,0.335)	(0.444,0.002)
		0.8	0.927 ± 0.005	0.563 ± 0.009	(0.021,0.335)	(0.441,0.005)
		1.2	0.921 ± 0.007	0.558 ± 0.014	(0.021,0.334)	(0.436,0.008)
Multi-Fashion	LS	0.5	0.269 ± 0.003	0.683 ± 0.003	(0.169,0.192)	(0.180,0.178)
		1.0	0.267 ± 0.006	0.680 ± 0.004	(0.171,0.185)	(0.176,0.180)
		1.2	0.268 ± 0.004	0.683 ± 0.003	(0.170,0.187)	(0.174,0.178)
	CS	0.5	0.261 ± 0.005	0.678 ± 0.006	(0.172,0.261)	(0.258,0.180)
		1.0	0.260 ± 0.005	0.677 ± 0.003	(0.174,0.221)	(0.228,0.180)
		1.2	0.260 ± 0.005	0.676 ± 0.004	(0.174,0.219)	(0.218,0.181)
Multi-F+MNIST	LS	0.5	0.450 ± 0.011	0.789 ± 0.004	(0.068,0.165)	(0.081,0.153)
		1.0	0.454 ± 0.010	0.791 ± 0.005	(0.069,0.156)	(0.075,0.151)
		1.2	0.462 ± 0.010	0.795 ± 0.005	(0.068,0.151)	(0.075,0.147)
	CS	0.5	0.439 ± 0.012	0.782 ± 0.005	(0.075,0.212)	(0.225,0.154)
		1.0	0.444 ± 0.011	0.785 ± 0.006	(0.074,0.185)	(0.138,0.152)
		1.2	0.447 ± 0.009	0.787 ± 0.004	(0.075,0.180)	(0.125,0.148)

Table 7: **Ablation on split point.** The split is represented by the number of allocated blocks to representation learning (RL) and preference alignment (PA) network respectively. We report obtained HV in training and evaluation space as well as extreme points.

Dataset	RL / PA	HV ↑	HV MCR ↑	Obj_1-best	Obj_2-best
CelebA	3 / 35	0.213	0.396	(0.075,0.080)	(0.076,0.079)
	7 / 31	0.238	0.406	(0.070,0.077)	(0.071,0.076)
	11 / 27	0.232	0.401	(0.070,0.079)	(0.071,0.078)
	17 / 21	0.242	0.415	(0.068,0.079)	(0.071,0.076)
	23 / 15	0.242	0.415	(0.068,0.077)	(0.069,0.077)
	31 / 7	0.238	0.422	(0.069,0.078)	(0.071,0.076)
	33 / 5	0.222	0.402	(0.072,0.086)	(0.077,0.079)
FairFace	5 / 6	0.595	0.562	(0.264,0.239)	(0.720,0.016)
	6 / 5	0.616	0.561	(0.243,0.236)	(0.686,0.008)
	7 / 4	0.627	0.541	(0.235,0.235)	(0.728,0.000)
	8 / 3	0.627	0.587	(0.202,0.255)	(0.481,0.027)

Table 8: **Ablation on learning rate ratio.** Sensitivity of MOSEL to the follower/leader learning rate ratio on Adult. The leader learning rate is fixed at $\eta_{\text{leader}} = 0.001$. The default ratio is marked in gray; best values per column are in bold. MCR-best and DEO-best report the values of the solution that is best on the respective objective.

Ratio	η_{leader}	η_{follower}	HV ↑	HV MCR ↑	MCR-best	Obj_2DEO-best
2×	0.001	0.002	0.933	0.849	(0.143, 0.109)	(0.209, 0.011)
5×	0.001	0.005	**0.939**	**0.853**	(0.143, 0.108)	(0.211, 0.010)
10×	0.001	0.010	0.935	0.845	(0.143, 0.106)	(0.211, 0.010)
20×	0.001	0.020	**0.939**	**0.853**	(0.143, 0.107)	(0.211, 0.009)
50×	0.001	0.050	0.876	0.807	(0.145, 0.104)	(0.196, 0.020)
100×	0.001	0.100	0.755	0.715	(0.146, 0.098)	(0.179, 0.040)

D.2 Split point

To further analyze the impact of the split point on performance, we conducted ablations on ResNet, EfficientNet, and DistilBERT (see App. E). The results for ResNet and EfficientNet for FairFace and CelebA respectively, are shown in Table 7.

On CelebA (EfficientNet-B4, 38 total blocks), performance varies primarily at the two extremes of the allocation spectrum. Assigning nearly all capacity to the preference alignment network (3 / 35) yields the lowest hypervolume in both training and evaluation, as the representation learning network lacks sufficient depth to learn transferable representations.

Conversely, restricting the preference alignment network to only 5 blocks (33 / 5) also degrades performance, as the preference alignment network can no longer adequately specialize to diverse preference vectors. Between these extremes, performance remains stable: splits ranging from 7 / 31 to 31 / 7 all achieve comparable hypervolume, indicating broad robustness to the precise split location. On FairFace (ResNet-34, 11 total blocks), we observe a similar pattern. Hypervolume increases modestly from 0.595 at split 5 / 6 to 0.627 at 7 / 4 and 8 / 3, the performance differences across all four tested splits remain small.

Across all architectures, performance is robust once the preference alignment network has sufficient capacity, indicating that the split point is not a critical hyperparameter. Based on these results, we recommend allocating approximately the final 20–25% of the blocks to the preference alignment network, while assigning the remaining blocks to the representation learning network. This simple guideline consistently performs well across CNNs and Transformers, substantially reducing the need for extensive split point tuning.

Table 9: Results on DistilBERT text-classification. Results show HV and extreme points for the front for different split points (splits between transformer blocks)

			Fairness extreme		Accuracy extreme	
RL / PA	HV ↑	HV MCR ↑	Acc@MinDEO(%) ↑	MinDEO ↓	DEO@MaxAcc ↓	MaxAcc(%)↑
5 / 1	0.423	0.479	88.68	0.0000	0.037	92.22
4 / 2	0.419	0.477	88.69	0.0001	0.035	92.17
3 / 3	0.419	0.475	88.70	0.0002	0.036	92.12

D.3 Learning rate ratio

To analyze the impact of the learning rate ratio, we keep the leader (preference alignment) network at the standard learning rate and vary the follower/leader ratio on the example of the Adult dataset. Table 8 shows that MOSEL is robust for ratios between 2× and 20×, with degradation only at 50×+, where an overly aggressive follower destabilizes training. Our default 10× ratio lies comfortably within this robust regime and consistently performs well without additional tuning.

D.4 Number of objectives

To validate the scalability of our method to many objectives, we conduct an ablation study on CelebA as presented in Figure 4. To this end, we identify the most correlated (`'Heavy Makeup'` and `'Wearing Lipstick'`) and least correlated (`'Pale Skin'` and `'Wearing Necklace'`) task pairs from the full set of 40 CelebA attributes. Starting from each pair, we iteratively add the next most correlated or least correlated task, scaling up to 10 objectives per set. Task correlation is determined via Pearson correlation coefficients ρ computed over the attribute labels, greedily selecting tasks that maximize (i.e., $|\rho| \approx 1$) or minimize (i.e., $|\rho| \approx 0$) the mean absolute correlation with the already selected set. All configurations are trained following the same pipeline as the main CelebA experiments. Since hypervolume computation scales exponentially with the number of objectives, we restrict evaluation to the closest point to utopia, which is generally the solution of interest in MTL. Mean accuracy and improvement over the single task baseline over 5 seeds are reported in Figure 4. Note that for improved readability, some attribute names in Figure 4 have been abbreviated without altering their meaning.

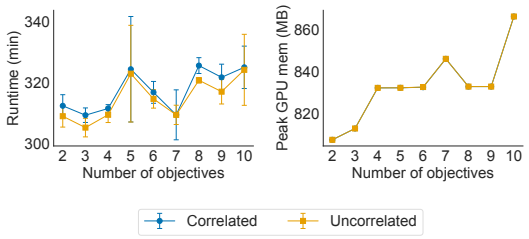

Figure 8: Ablation study on CelebA: runtime and peak memory for increasing numbers of objectives, for correlated and uncorrelated features.

Figure 8 reports runtime and peak memory for correlated and uncorrelated features. We can see that memory is identical for both settings and only slightly increases with the numbers of objectives. While small differences in runtime exist between correlated and uncorrelated settings, the overall runtime remains roughly constant across the number of objectives.

E Extension to transformer-based architectures

To explore the applicability of our method beyond MLPs and CNNs, we additionally conducted experiments on transformer-based architectures. To this end, we evaluate MOSEL on fairness-aware text classification using DistilBERT [55] on a 100k-sample subset of CivilComments [8], optimizing toxicity prediction (BCE) and fairness (DEO) under linear scalarization with Dirichlet parameter $\alpha = 0.5$. DistilBERT has six Transformer blocks, and we evaluate three candidate split points.

MOSEL naturally extends to this Transformer architecture without any architectural modifications, achieving consistent Pareto-front quality across three different split points (HV 0.419–0.423, standard deviation ≤ 0.011). These results emphasize, that MOSEL does not rely on architecture-specific decompositions, but instead leverages the modular structure common to modern neural networks by partitioning them at their natural block boundaries (e.g., residual stages or Transformer blocks). Extending this empirical validation to foundation vision-language models and LLMs is an exciting direction for future work.

F Convergence analysis

To verify that preference conditioning does not introduce optimization instability, we compare the convergence behavior of MOSEL-LS against an equal weight baseline on Multi-MNIST. Both methods use an identical optimizer and learning rate schedule (standard Adam). For each epoch we record the maximum loss across all sampled preference vectors, capturing worst-case optimization behavior. Results are reported in Table 10 as means and standard deviations over 5 seeds.

Both methods exhibit nearly identical convergence profiles: a rapid descent during the first 10 epochs followed by a stable plateau, with no signs of divergence or increasing instability. After the initial transient, MOSEL maintains a small, constant gap in maximum loss relative to the baseline ($\Delta \approx +0.09$ from epoch 10 onward). This gap is expected, as MOSEL repeatedly optimizes under extreme preference vectors (encouraged by our choice of $\alpha < 1$) in order to recover a diverse Pareto front, rather than a single fixed weighting. Despite this more challenging optimization setting, MOSEL achieves tighter cross-seed variance than the baseline (± 0.02 vs. ± 0.04 at convergence), and the constant magnitude of Δ after epoch 10 confirms that the additional complexity introduced by preference conditioning does not compound over training. These results demonstrate that MOSEL preserves the optimization stability of conventional single-objective training while successfully learning to cover the Pareto front.

Table 10: Per-epoch maximum objective loss (mean $\pm$ std over 5 seeds). Δ is defined as difference between MOSEL and Equal Weight baseline; negative values indicate lower (better) worst-case loss for MOSEL. After an initial transient, MOSEL maintains a small, constant gap with tighter cross-seed variance, confirming stable optimization throughout training.

Epoch	MOSEL-LS Max	Equal Weight Max	Δ Max
0	5.54 ± 0.86	15.89 ± 5.91	-10.35
10	0.52 ± 0.03	0.42 ± 0.04	+0.09
30	0.38 ± 0.02	0.30 ± 0.05	+0.08
50	0.36 ± 0.02	0.28 ± 0.05	+0.08
70	0.37 ± 0.01	0.28 ± 0.05	+0.09
99	0.36 ± 0.02	0.27 ± 0.04	+0.09

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction assert that MOSEL obtains Pareto stationary solutions at the cost of single objective training. We provide a formal proof for the claim of Pareto stationarity in section 3 and validate the empirical performance in section 5.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: In Section 6, we explicitly discuss the limitations and venues for future work of our paper.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: All assumptions are detailed in the main text, which contain summaries of the proofs, which are derived in detail in Appendix A.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: All details are fully specified in the main part and Appendix B.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: All datasets that are used in this paper are publicly available and the code to reproduce the experiments is available at `https://github.com/ElinaCelik/mosel`.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: A summary is provided in the main text, and all the details in Appendix B.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: As detailed in the main text and Appendix B, all experiments are repeated across 5 independent seeds, with results reported as mean and standard deviation throughout.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: All information can be found in Appendix B.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We follow the Code of Ethics in every aspect.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: This paper advances the field of *a posterior* multi-objective optimization in deep learning, thus enabling for more optimal and transparent trade-offs between competing objectives. As demonstrated in our fairness–performance experiments, our method not only achieves better trade-offs than existing approaches, but also provides practitioners with a diverse set of solutions for informed, context-dependent deployment decisions. Extending and evaluating our approach in other high-stakes domains remains an important direction for future work. Beyond these considerations, we do not foresee negative societal consequences that are unique to this work, beyond those generally associated with the development and deployment of machine learning systems.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: We do not provide any of the above.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Our code builds on a repository released under the MIT License (https://github.com/ruchtem/cosmos). The original copyright notice and MIT License text are retained in all modified and extended files, in compliance with the license terms.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The modified and extended codebase introduced in this paper is documented alongside the submission. It is released under the MIT License (inherited from the original codebase). Documentation covering usage, dependencies, and how to reproduce the experiments is provided in the repository README.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: Not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: Not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: We do not use LLM as any important, original, or non-standard component. We only use LLMs as support for rewriting.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.