BALANCING AVERAGE AND WORST-CASE ACCURACY IN MULTITASK LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

When training and evaluating machine learning models on a large number of tasks, it is important to not only look at average task accuracy—which may be biased by easy or redundant tasks—but also worst-case accuracy (i.e. the performance on the task with the lowest accuracy). In this work, we show how to use techniques from the distributionally robust optimization (DRO) literature to improve worst-case performance in multitask learning. We highlight several failure cases of DRO when applied off-the-shelf and present an improved method, Lookahead-DRO (L-DRO), which mitigates these issues. The core idea of L-DRO is to anticipate the interaction between tasks during training in order to choose a dynamic re-weighting of the various task losses, which will (i) lead to minimal worst-case loss and (ii) train on as many tasks as possible. After demonstrating the efficacy of L-DRO on a small controlled synthetic setting, we evaluate it on two realistic benchmarks: a multitask version of the CIFAR-100 image classification dataset and a large-scale multilingual language modeling experiment. Our empirical results show that L-DRO achieves a better trade-off between average and worst-case accuracy with little computational overhead compared to several strong baselines.

1 INTRODUCTION

Multitask learning—the process by which a single model is trained to perform a variety of different tasks—has become a subject of increasing interest with many successful applications in a variety of domains (Ruder, 2017; Yu et al., 2020; Wang et al., 2021). By and large, multitask learning models are evaluated by reporting their average performance on all tasks (McCann et al., 2018; Wang et al., 2019). However, average accuracy does not paint the full picture of a multitask model's performance. Ensuring comparable accuracy across tasks, including on tasks from under-represented training data distributions, is an important issue from both practical and fairness perspectives. For example, in natural language processing, multilingual models often performs worse on languages with smaller amounts of resources on the web (e.g., Wikipedia) due to the resulting scarcity of relevant training data (Hu et al., 2020). In computer vision, facial recognition models have been shown to perform worse on racial groups that are underrepresented in the training data (Buolamwini & Gebru, 2018).

In this paper, we argue that multitask models should also be evaluated by their worst-case accuracy. Yet commonly used task re-weighting based objectives are inadequate to ensure good worst-case performance. Rather, a more natural choice is to use a type of min-max objective from the distributionally robust optimization (DRO) literature (Rahimian & Mehrotra, 2019; Sagawa et al., 2020): $\min_{\boldsymbol{\theta}} \max_i \ell_i(\boldsymbol{\theta})$, where ℓ_i is the loss of the i-th task and $\boldsymbol{\theta}$ are the model parameters. Training with this objective guarantees at least a certain level of performance on all tasks. However, we observe that DRO does not always work well in practice. Since the min-max objective optimizes the worst loss, it tends to do so by sacrificing performance on other tasks, which hurts average accuracy (§2.3).

Motivated by this observation, we propose a novel optimization algorithm called Lookahead-DRO (L-DRO). The intuition behind L-DRO is to anticipate the evolution of the loss of each task depending on which task we train on. Using this information, L-DRO chooses a weight allocation that (i) leads to minimal worst-case loss and (ii) trains on the maximal number of tasks possible (§2.4).

We evaluate L-DRO on a synthetic task, an image classification task, and a language modeling task. In experiments on a multitask version of the CIFAR-100 dataset, we show that L-DRO consistently allows us to train models that achieve a good trade-off of worst and average task accuracy. On a

realistic large-scale multilingual language modeling dataset, we find that L-DRO is able to improve performance on the worst-performing languages at very little cost to average performance.

Our main contributions are as follows: (i) we show how distributionally robust optimization (DRO) can be applied to multitask learning to maximize worst-case accuracy; (ii) we highlight failure cases of the standard DRO formulation in the multitask setting; (iii) we propose a new algorithm, L-DRO that addresses DRO's deficiencies; and (iv) we demonstrate that L-DRO achieves a better trade-off between average and worst-case accuracy than strong baselines in a synthetic and two realistic experiments from two different modalities.

2 Training Multitask Models

Consider a machine learning model parameterized by $\boldsymbol{\theta} \in \mathbb{R}^{d_{\text{model}}}$, which is trained to perform N tasks. In general, let a task refer to the combination of examples drawn from a data distribution and an objective function. For simplicity, we define a task solely by its objective function $\ell_{i=1\ldots N}$. As a running example, we consider a multilingual language model with dozens of tasks where each task corresponds to a language (Conneau et al., 2020; Xue et al., 2021).

2.1 Average Task Loss

The most common objective for multitask learning is to minimize the average of all the task losses:

$$\min_{\boldsymbol{\theta}} \frac{1}{N} \sum_{i=1}^{N} \ell_i(\boldsymbol{\theta})$$

A downside of this objective is that all tasks are treated equally. In real-world scenarios, we often encounter a scenario where task difficulty is not uniform (i.e., the magnitude of ℓ_i varies wildly; Hessel et al. 2019) or we want to pay more attention to certain tasks. If the distribution of all tasks is not homogeneous, training with $\mathcal{L}_{\text{avg}}$ results in a model with performance that is skewed towards certain clusters of tasks. In our multilingual example, if a large number of languages come from the same language family such as Indo-European languages, $\mathcal{L}_{\text{avg}}$ will choose models which perform better on these languages at the expense of typologically distinct languages such as Mandarin or Finnish.

2.2 Static mixing

Rather than simple averaging, prior work in multitask learning often uses arbitrary weightings $\mathbf{w} = \{w_1, \ldots, w_N\}$ in the probablility simplex $\Delta_N = \{\mathbf{w} \mid w_i > 0, \sum_i w_i = 1\}$ to modulate the losses of individual tasks:

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} w_i \, \ell_i(\boldsymbol{\theta}).$$

The weights are typically decided based on some statistics of the data. In proportional sampling (Sanh et al., 2019), the weight of a task i is proportional to the size of its data $w_i \propto |D_i|$. This biases the model towards tasks that are over-represented in the data. To counter-balance this, recent approaches use an inverse temperature α where $w_i \propto |D_i|^{\alpha}$ with α typically in the range of $[0.2, 0.3]$ (Aharoni et al., 2019; Conneau et al., 2020; Xue et al., 2021).

2.3 Worst Task Loss

A third option that considers the task on which the model performs worst is the min-max objective. We refer to this objective as DRO because of its prominence in the distributional robustness literature (Sagawa et al., 2020):

$$\min_{\boldsymbol{\theta}} \max_{i} \ell_i(\theta) = \min_{\boldsymbol{\theta}} \underbrace{\max_{\mathbf{w} \in \Delta_N} \sum_{i=1}^{N} w_i \ell_i(\theta)}_{\mathcal{L}_{\text{DRO}}(\boldsymbol{\theta})} .$$

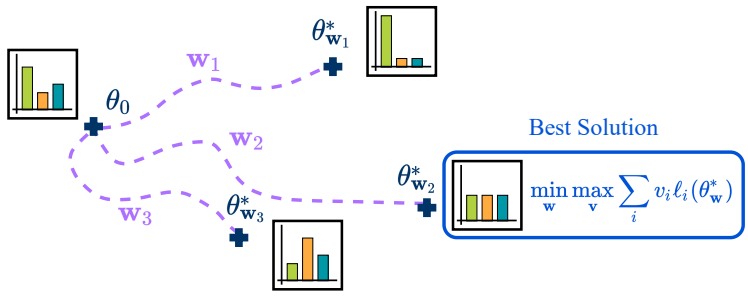


Figure 1: Overview of the Lookahead-DRO procedure. Starting from parameters $\boldsymbol{\theta}_0$, the optimal training weights $\mathbf{w}$ are chosen such that the resulting parameters $\boldsymbol{\theta}^*_{\mathbf{w}}$ minimize the worst case loss.

The above objective minimizes the worst-case loss of the model at every step. Alternatively, we can view the objective similar to the previous ones as minimizing the loss under a certain task weighting $\mathbf{w}$. In this case, the weights $\mathbf{w}$ are not static, but rather chosen dynamically to maximize the weighted loss by allocating the maximum weight(s) to the task(s) on which the model performs worst.

The main advantage of minimizing $\mathcal{L}_{\text{DRO}}$ is that it ensures a certain level of performance on all tasks. However, it suffers from a number of drawbacks. First, it is sensitive to task difficulty: if one task ℓ_{hard} is much more difficult than the others (i.e., $\min_{\boldsymbol{\theta}} \ell_{\text{hard}}(\boldsymbol{\theta}) > \min_{\boldsymbol{\theta}} \ell_j(\boldsymbol{\theta})$ for all other tasks j), then $\mathcal{L}_{\text{DRO}}$ is likely to get "stuck" trying to minimize ℓ_{hard} at the detriment of other tasks.

In addition, DRO is agnostic to positive (or negative) interaction between tasks. In some cases, training on one task ℓ_i may have a positive effect on another task ℓ_j because they share some similarities—for example if they represent related languages. In such cases it may be desirable to train on both tasks rather than on one or the other (as it may reduce the variance of the gradient estimator), even if one has a higher loss than the other. On the other hand, when two tasks are at odds with each other (i.e., training on one task decreases performance on the other), the $\max$ objective in $\mathcal{L}_{\text{DRO}}$ can lead to oscillatory behavior where the weights $\mathbf{w}$ alternate between two competing tasks.

2.4 Lookahead DRO

The core limitation of $\mathcal{L}_{\text{DRO}}$ underpinning the aforementioned issues is the fact that it does not take the optimization procedure into account (i.e., it will bluntly try to minimize the worst loss). This is suboptimal because in certain cases, taking gradient steps on the worst task might hurt other tasks disproportionately. If the worst-performing task has converged already, this weight allocation will also miss out on optimizing other tasks, which can attain lower losses and are still not converged.

Formally, for a given value of the model parameters $\boldsymbol{\theta}_0$, $\mathcal{L}_{\text{DRO}}$ chooses weights maximizing the model's loss $\mathbf{w}^*_0 := \arg\max_{\mathbf{w}} \sum_i w_i \ell_i(\boldsymbol{\theta}_0)$. After training with weights $\mathbf{w}^*_0$, we obtain parameters $\boldsymbol{\theta}_1$ with loss $\mathcal{L}_{\text{DRO}}(\boldsymbol{\theta}_1) = \max_{\mathbf{w}} \sum_i w_i \ell_i(\boldsymbol{\theta}_1)$. However, note that for $\boldsymbol{\theta}_1$, the worst-case weight allocation $\mathbf{w}^*_1 := \arg\max_{\mathbf{w}} \sum_i w_i \ell_i(\boldsymbol{\theta}_1)$ might be different than $\mathbf{w}^*_0$, so we have no guarantees that $\mathcal{L}_{\text{DRO}}(\boldsymbol{\theta}_1) \leq \mathcal{L}_{\text{DRO}}(\boldsymbol{\theta}_0)$. In other words, training with the worst-case weights $\mathbf{w}^*_0$ does not guarantee that $\mathcal{L}_{\text{DRO}}$ will decrease.

This observation suggests an alternative strategy: suppose we have a fixed—and, for simplicity, deterministic—learning algorithm $\mathbf{w}, \boldsymbol{\theta}_0 \rightarrow \boldsymbol{\theta}_{\mathbf{w},\boldsymbol{\theta}_0}$ for minimizing the $\mathbf{w}$-weighted loss $\sum_i w_i \ell_i$ starting from parameters $\boldsymbol{\theta}_0$. Rather than choosing $\mathbf{w}$ to be the worst-case weights corresponding to $\boldsymbol{\theta}_0$, we should pick weights that minimize the worst-case loss of $\boldsymbol{\theta}^*_{\mathbf{w},\boldsymbol{\theta}_0}$, *i.e. after training*. This can be formalized as solving the min-max problem:

$$\min_{\mathbf{w} \in \Delta_N} \max_{\mathbf{v} \in \Delta_N} \sum_i v_i \ell_i(\boldsymbol{\theta}_{\mathbf{w},\boldsymbol{\theta}_0}) \tag{1}$$

Intuitively, decoupling the weights $\mathbf{w}$ used for training from the weights $\mathbf{v}$ used for computing $\mathcal{L}_{\text{DRO}}(\boldsymbol{\theta}^*_{\mathbf{w},\boldsymbol{\theta}_0})$ removes the restriction of just training on the worst-performing tasks and enables us to find a task weighting which is able to continuously minimize the worst-case loss. Note that in this case there might be multiple solution $\mathbf{w}$: for instance, it might be that one or more tasks have the same effect on the resulting worst-case task, and as such any permutation of their weights is an optimal solution. In such scenarios, we do not want to miss out on training on tasks on which performance

can still be improved (as long as worst-case performance decreases). This can be formulated as a maximum entropy principle (Jaynes, 1957): *if multiple optimal training weights exist, we should pick the one with the highest entropy*.

Because finding the optimal $\mathbf{w}$ necessitates "looking ahead" in the optimization trajectory from $\boldsymbol{\theta}_0$ to identify the DRO loss $\max_{\mathbf{v}} \sum_i v_i \ell_i(\boldsymbol{\theta}_{\mathbf{w},\boldsymbol{\theta}_0})$, we call this weight selection principle **Lookahead-DRO** (L-DRO). We provide a graphical illustration in Figure 1.

2.5 One-step Lookahead DRO

In practice we cannot explicitly solve the min-max game in Eq. 1 because estimating the payoff $\max_{\mathbf{v}} \sum_i v_i \ell_i(\boldsymbol{\theta}_{\mathbf{w},\boldsymbol{\theta}_0})$ would require unrolling the full optimization process to compute $\boldsymbol{\theta}^*_{\mathbf{w},\boldsymbol{\theta}_0}$. Instead, we simplify and take only one gradient step with a small learning rate λ, *i.e.* replacing $\boldsymbol{\theta}^*_{\mathbf{w},\boldsymbol{\theta}_0}$ with $\boldsymbol{\theta}_0 - \lambda \sum_i w_i \nabla \ell_i(\boldsymbol{\theta}_0)$. Such one-step approximations are often used in the meta-learning literature (Finn et al., 2017; Liu et al., 2018). Using the first order Taylor extension of ℓ_i enables us to rewrite the individual lookahead task losses as:

$$\ell_i(\boldsymbol{\theta}_0 - \lambda[\sum_{j=1}^{N} w_j \nabla \ell_j(\boldsymbol{\theta}_0)]) \approx \ell_i(\boldsymbol{\theta}_0) - \lambda \sum_{j=1}^{N} w_j \nabla \ell_i(\boldsymbol{\theta}_0)^\intercal \nabla \ell_j(\boldsymbol{\theta}_0)$$

The resulting payoffs are now linear in $\mathbf{w}$, which reduces the lookahead optimization problem in Eq. 1 into a bilinear matrix game:

$$\min_{\mathbf{w}} \max_{\mathbf{v}} \mathbf{v}^\intercal \, \mathbf{L} \, \mathbf{w} \tag{2}$$

with a payoff matrix $\mathbf{L}$, called the "lookahead matrix", which can be decomposed as follows:

$$L_{ij} = \underbrace{\ell_i(\boldsymbol{\theta}_0)}_{\text{Loss of task } i} - \lambda \underbrace{\nabla \ell_i(\boldsymbol{\theta}_0)^\intercal \nabla \ell_j(\boldsymbol{\theta}_0)}_{\substack{\text{"interaction"} \\ \text{between tasks } i \text{ and } j}}$$

We can find a solution (Nash equilibrium) of this game using many available solvers such as fictitious play (Brown, 1951) or linear programming (Lemke, 1965). In such cases where multiple equilibria exist, we choose the one with the highest entropy as dictated by our maximum entropy principle. The lookahead learning rate λ is treated as a hyperparameter. This results in a training algorithm which we outline in Algorithm 1.

Algorithm 1: Lookahead DRO

Input: Lookahead learning rate λ, initial parameters $\boldsymbol{\theta}_0$
$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}_0$
while *training is not over* **do**
  $L \leftarrow [\ell_i(\boldsymbol{\theta}) - \lambda \nabla \ell_i(\boldsymbol{\theta})^\intercal \nabla \ell_j(\boldsymbol{\theta})]_{ij}$ `// Compute lookahead matrix`
  $\mathbf{v}^*, \mathbf{w}^* \leftarrow$ `maxent_nash_solver(`L`)` `// Find Nash equilibrium of` L
  $\boldsymbol{\theta} \leftarrow$ `optimizer_step(`$\sum_i w_i^* \ell_i$`)` `// Optimizer step on` $\mathbf{w}^*$`-weighted loss`
end

To build more intuitions, we consider several edge cases and discuss how L-DRO handles them:

Independent tasks. We have $\nabla \ell_i^\intercal \nabla \ell_j = \mathbb{1}_{i=j}$. In this case, for small enough values of the lookahead learning rate, the optimal strategy for the row player ($\mathbf{v}$) is to pick the task $i^* = \arg\max_i \ell_i$ with the highest loss, and the only possible best response for the column player ($\mathbf{w}$) is to match this selection (since $L_{i^*i} = \ell_i^* - \lambda < L_{ij} = \ell_i$ for all $j \neq i$). In other words, we recover the original DRO objective of training against the worst-case loss. This can be interpreted as an "implicit assumption" of DRO that training on one task does not affect the others.

Redundant tasks. We have $\nabla \ell_i^\intercal \nabla \ell_j = 1$. In this case, the gradients of the tasks are perfectly aligned. In practice, this means that the payoff is independent of the value of $\mathbf{w}$, so any choice of the training weights is valid. By the maximum entropy principle, we should pick uniform weights, thus recovering the average task loss baseline.

The highest loss task has already converged. Consider the case where ℓ_0 is maximal, but the model has already converged on task 0 (in other words $\nabla \ell_0 = 0$). In this scenario—assuming there

is no strong negative interaction among the remaining tasks—the highest loss will always be ℓ_0 irrespective of the training weights $\mathbf{w}$. Again, by the maximum entropy principle, L-DRO advocates for picking uniform weights. This makes it possible to continue training on the remaining tasks even though ℓ_0 has converged, bypassing the core limitations of distributionally robust optimization (DRO) identified in §2.3.

3 Lookahead DRO in Practice

3.1 Gradient and Loss Noises

In most realistic machine learning scenarios we do not have access to the full losses $\ell_i(\boldsymbol{\theta})$ and their gradients. Rather, we compute stochastic estimates on minibatches $\hat{\ell}_i(\boldsymbol{\theta})$. In preliminary experiments, we find that this often leads to instability in the training algorithm, since even little noise in the lookahead matrix can lead to dramatically different Nash equilibria, which causes high fluctuations of the training weights $\mathbf{w}^*$.

To mitigate this issue, we adopt an online weight update procedure inspired by the online-DRO algorithm proposed by Sagawa et al. (2020). Instead of computing the optimal weights $\mathbf{w}^*$ at each step based on noisy losses and gradients, we keep a running set of weights $\mathbf{w}$, which we update jointly with model parameters $\boldsymbol{\theta}$. Specifically, we interleave gradient updates on $\boldsymbol{\theta}$ with exponentiated gradient steps (Kivinen & Warmuth, 1997) on $\mathbf{w}$ to minimize $\max_{\mathbf{v}} \mathbf{v}^\intercal L\, \mathbf{w}$. We show the weight update rule in Algorithm 2.

Algorithm 2: Lookahead DRO online weight update

Function `update_weights` (*Lookahead matrix* $\mathbf{L}$*, weights* $\mathbf{w}$*, weight update rate* η) **:**

$\mathbf{v}^* \leftarrow \arg\max_{\mathbf{v}} \mathbf{v}^\intercal \mathbf{L}\, \mathbf{w}$ `// Compute best response to` $\mathbf{w}$

$\tilde{w}_i \leftarrow w_i e^{-\eta [\mathbf{v}^{*\intercal} \mathbf{L}]_i}$ `// Update weights`

$w_i \leftarrow \frac{\tilde{w}_i}{\sum_j \tilde{w}_j}$ `// Re-normalize weights`

return $\mathbf{w}$

In practice, we find that this online weight update alleviates the instability issues mentioned above. Note that the choice of exponentiated gradient is not arbitrary. It corresponds to a mirror descent step (Nemirovskij & Yudin, 1983) using the Kullback-Leibler divergence (Kullback & Leibler, 1951) as a distance function. Choosing weight updates that locally minimize the change in relative entropy plays into our maximum entropy guiding principle (Jumarie, 1990; Kapur & Kesavan, 1992).

3.2 Lookahead Matrix Computation

The main computational bottleneck of L-DRO lies in the computation of the lookahead matrix $\mathbf{L}$. Recall that L_{ij} can be decomposed in two terms: the loss ℓ_i and an "interaction" term $\nabla \ell_i(\boldsymbol{\theta})^\intercal \nabla \ell_j(\boldsymbol{\theta})$. While the former can be estimated essentially "for free" during the forward pass of back-propagation, computing the pairwise task gradients dot products is much more computationally demanding. Indeed, it necessitates computing (and holding in memory) n model gradients, which can be prohibitive for larger models. To reduce the computational overhead, we only update this second term (which we call the "interaction matrix" $\mathbf{A} = [\nabla \ell_i(\boldsymbol{\theta})^\intercal \nabla \ell_j(\boldsymbol{\theta})]_{i,j=1\dots N}$) periodically, typically every 100 or 1000 training steps. In preliminary experiments, we also find that Lookahead-DRO works more reliably when the lookahead gradients of each task are re-normalized to one. This means that the interaction matrix is actually computed to be $A_{ij} = \nabla \ell_i(\boldsymbol{\theta})^\intercal \left(\nabla \ell_j(\boldsymbol{\theta}) / \|\nabla \ell_j(\boldsymbol{\theta})\|\right)$. We show the final algorithm for online L-DRO in Algorithm 3.

4 Synthetic Experiments as a Proof of Concept

In this section, we demonstrate the efficacy of L-DRO on a synthetic multitask setting. We design a toy experiment such that neither DRO nor static mixing is able to reach an optimal solution (Figure 2). We identify each task i with a point y_i on the two dimensional plane $\mathbb{R}^2$ and define their respective loss functions as the squared Euclidean distance of the model prediction $f(\boldsymbol{\theta})$ to y_i, $\ell_i(\boldsymbol{\theta}) = \|y_i - f(\boldsymbol{\theta})\|_2^2$. We distribute the tasks such that most points are clustered around $(-1, 0)$, to simulate a grouping of similar tasks (*e.g.* Indo-Europoean languages in our multilingual example). We then place a single point y_{outlier} at $(1, 0)$ to emulate the presence of an outlier task.

Algorithm 3: Online lookahead DRO training loop

Input: Lookahead learning rate λ, weight update rate η, interaction matrix update interval k

$A \leftarrow 0$ `// Initialize interaction matrix at 0`

$\mathbf{w} \leftarrow \left(\frac{1}{N} \cdots \frac{1}{N}\right)$ `// Initialize` $\mathbf{w}$ `as uniform weights`

while *training is not over* **do**

  if *i% k == 0* **then**

    $A \leftarrow \left[\nabla \ell_i(\boldsymbol{\theta})^\intercal \frac{\nabla \ell_j(\boldsymbol{\theta})}{\|\nabla \ell_j(\boldsymbol{\theta})\|}\right]_{1 \leq i,j \leq n}$ `// Update interaction matrix`

  $L \leftarrow \ell(\boldsymbol{\theta}) - \lambda A$ `// Compute lookahead matrix`

  $\mathbf{w} \leftarrow$ `update_weights`$(\mathbf{w}, L)$ `// Update weights using Algorithm 2`

  $\boldsymbol{\theta} \leftarrow$ `optimizer_step(`$\sum_i w_i \ell_i$`)` `// Optimizer step on` $\mathbf{w}$`-weighted loss`

We simulate the increased difficulty of the outlier task by adopting a parameterization $f(\boldsymbol{\theta})$ that prevents the model from reaching any position closer than a fixed value R from y_{outlier}. Specifically we adopt a polar parameterization $\boldsymbol{\theta} := (r, \phi)$ and set

$$f(\boldsymbol{\theta}) = y_{\text{outlier}} + (R + r^2) \begin{bmatrix} \cos \phi \\ \sin \phi \end{bmatrix}.$$

This essentially creates a "forbidden region" of radius R around y_{outlier} of the space that the model cannot penetrate, thus lower-bounding ℓ_{outlier} to R. In experiments, we set $R = 1.25$.

Figure 2a depicts the optimization trajectories when the model is trained either with the average objective or with DRO. As expected, minimizing the average loss leads to a solution that performs well on the majority of tasks but poorly on the outlier. On the other hand, the DRO trajectory moves towards the latter but ends up stuck when it reaches a point where ℓ_{outlier} cannot be improved despite being the highest loss among all tasks.

On the other hand, while the L-DRO trajectory shown in Figure 2b follows a similar trajectory as DRO at first, it is able to reach a solution that still achieves the best possible ℓ_{outlier} while achieving lower loss on the other tasks. Also notice that the online variant follows a similar trajectory, despite the fact that it does not compute a Nash equilibrium at every step.

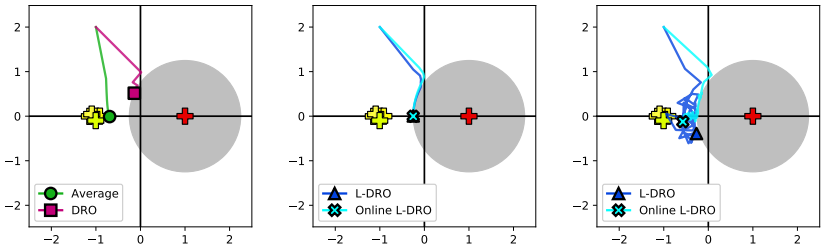
Finally, Figure 2c highlights the importance of the online variant in the presence of noise. We emulate the stochasticity of SGD by adding a small amount of Gaussian noise to both losses and gradients. Under these conditions, lookahead DRO becomes very unstable, an issue that is partially mitigated by the online update.

5 Experiments

5.1 CIFAR100

As a first step towards a more realistic setting, we experiment with a multitask version of the CIFAR-100 dataset (Krizhevsky et al., 2009), in which the 20 coarse labels are treated as separate 5-way classification tasks (Rosenbaum et al., 2018; Yu et al., 2020). In this setting, we train a small CNN model with four convolution layers with 32 3×3 filters each followed by a ReLU activation and a 2×2 max-pooling layer. The resulting representation is averaged and fed through a 3-layer fully connected feed-forward network with hidden dimension 128 and ReLU activations. Predictions are made using a shared softmax layer across all tasks (*i.e.* across all 100 classes).

In addition to L-DRO, we report results for three baselines. First **static-mixing**, where all tasks are weighted equally (§2.2). Another natural baseline is **DRO** (§2.3) using the online algorithm proposed by Sagawa et al. (2020). Finally we compare to **PCGrad** (Yu et al., 2020), a multitask learning approach, which uses gradient level projections to mitigate interferences between tasks during learning. We train using regular stochastic gradient descent with a batch size of 128, a learning rate of 0.025 and with Nesterov accelerated gradients with a momentum rate of 0.9. For DRO, we sweep over 5 values of the hyper-parameter which controls the online weight update rate $\{0, 0.0001, 0.001, 0.01, 0.1\}$. For L-DRO, we sweep over 3 values of the weight update rate $\{0.01, 0.001, 0.0001\}$ and 3 values of the lookahead learning rate $\{0.001, 0.01, 0.1\}$. For Static mixing and PCGrad, there are no additional parameters to sweep over.



(a) DRO vs. Average loss (b) L-DRO (c) L-DRO (with noise)

Figure 2: Synthetic multitask experiment. The outlier task is shown in red while the others are in yellow. The grayed out area represents the region in space that cannot be attained by the model with our chosen parameterization

To gain a better understanding of the various trade-offs of each approach in terms of average and worst accuracy, we do not report a single number. Rather, we periodically evaluate the trained models on the test set and report the Pareto frontier of all checkpoints (across epochs and hyper-parameter configurations) of each method in the two dimensional plane of worst and average accuracy.

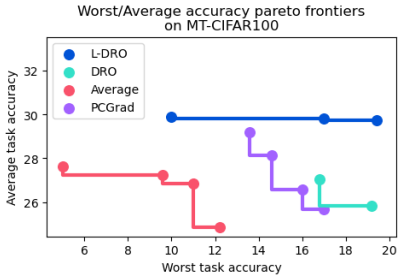


Figure 3: Average/worst task accuracy Pareto frontiers on multitask CIFAR-100

We show the results in Figure 2. We can see that L-DRO (in blue) consistently achieves a good trade-off between the two metrics, as its Pareto frontier is situated in the upper right corner of the space (high average and worst-case accuracy). In particular it is able to reach similar worst-case performance as DRO, without incurring a hit in terms of average performance. We find that this trend is consistent across different random seeds (see Appendix A.1).

An important difference in our experimental setting with Yu et al. (2020) is that the model we train is comparatively much smaller. In fact, as we increase the size of the model, we observe that the difference between individual method grows thinner (detailed results are depicted in Appendix A.2). This finding suggests that L-DRO is primarily useful when the model's capacity is a limiting factor compared to the number of tasks and the difficulty of individual tasks, which is not necessarily the case for the simple CIFAR-100 benchmark.

5.2 Multilingual Language Modeling

To evaluate L-DRO in a real-world scenario, we turn to large-scale multilingual language modeling. As NLP models are increasingly applied to new settings, pretraining on many languages is becoming a common scenario (Conneau et al., 2020; Xue et al., 2021). Improved pretraining performance has generally been found to correlate with better downstream performance (Raffel et al., 2020).

Furthermore, multilingual language modeling is a particulary interesting testbed because it exhibits various properties of real world multitask learning scenarios. For example, there is a large data imbalance across tasks: the highest resource language (English) contains $55\times$ more data than the lowest resource language (Yorùbá) in our training corpus. Additionally, there is a difference in difficulty between the various languages. Formally, for a parametric language model $p_{\boldsymbol{\theta}}$, its loss on the i-th language is the cross-entropy $\mathbb{H}(p_i, p_{\boldsymbol{\theta}})$ with the underlying distribution of the language, p_i. This can be re-written as $\mathbb{H}(p_i) + D_{\mathrm{KL}}(p_i || p_{\boldsymbol{\theta}})$. Since the KL divergence is non-negative, the loss of each language is lower-bounded by the language-specific entropy term $\mathbb{H}(p_i)$, irrespective of the capacity or the fitness of the model. In other words, languages with higher entropy will be more difficult to learn (Mielke et al., 2019).

5.2.1 Experimental Setting

We train a large autoregressive transformer model (Vaswani et al., 2017) on the multilingual C4 dataset (mC4; Xue et al., 2021). We limit ourselves to the 40 languages that are present in the

XTREME benchmark (Hu et al., 2020), which belong to 12 different language families and are written in 12 different scripts. Out of all languages, 40% are Indo-European and 42.5% are written in the Latin script.[1] The mC4 dataset comes with a canonical "training" and "evaluation" split. In order to distinguish between validation and test set, we modify the evaluation split as follows: for each language we split the evaluation set into a validation and test set containing up to 5M tokens each (using mT5 subword tokenization; Kudo 2018), or half the entire evaluation set, whichever is largest. We limit the size of the validation and test sets to facilitate running validation of our models.

We train each model for $300,000$ steps with a batch size of 256 elements of 512 tokens using the Adam optimizer (Kingma & Ba, 2014) and a cosine annealed learning rate schedule (Vaswani et al., 2017). For each run we select the best model in terms of average validation loss on all languages. For evaluation, we set the sequence length to 1024.

In this experiment we compare against three baselines:[2]

- **Static mixing** (§2.2). We try 3 different values for the inverse temperature hyper-parameter α: 0 (uniform sampling), 0.3 (a common value in the literature; Xue et al. 2021) and 1 (purely proportional sampling)
- **DRO** (§2.3), which minimizes the worst-case loss among all languages using the online algorithm from Sagawa et al. (2020). We sweep over the weight learning rate $\eta \in \{0.01, 0.001, 0.0001\}$.
- **Baselined-DRO**, a variation on DRO proposed by Oren et al. (2019) where the losses of each language i is adjusted with a "baseline" loss to account for the entropy term which biases DRO towards selecting languages that are inherently more difficult. In practice, we estimate $\mathbb{H}(p_i)$ by training a separate *monolingual* model on each language, $p_{\boldsymbol{\theta}_i^*}$ and using its cross-entropy $\mathbb{H}(p_i, p_{\boldsymbol{\theta}_i^*}) = \min_{\boldsymbol{\theta}} \mathbb{H}(p_i, p_{\boldsymbol{\theta}})$ as a proxy. We then train to minimize $\max_i \ell_i(\boldsymbol{\theta}) - \mathbb{H}(p_i, p_{\boldsymbol{\theta}_i^*})$ using the same algorithm as DRO, sweeping over the same values of the weight learning rate. Note that this particular variant necessitates training an additional model for each language.

5.2.2 Results

We report results in Table 1 in terms of perplexity (Jelinek et al., 1977), the standard metric for language modeling. Perplexity is the exponential of the (token-wise) cross-entropy. As a result, perplexities of different languages are not directly comparable. Languages with inherently higher entropy will tend to contribute more to the average (or worst) perplexity. Therefore, we also report average and worst-case perplexity *relative* to a monolingual model. Specifically, the relative perplexity of a model $\boldsymbol{\theta}$ on language i is defined as:

$$\text{Relative perplexity}(\boldsymbol{\theta}, i) = \frac{\text{Perplexity}(\boldsymbol{\theta}, i) - \text{Perplexity}(\boldsymbol{\theta}_i^*, i)}{\text{Perplexity}(\boldsymbol{\theta}_i^*, i)} \tag{3}$$

where $\boldsymbol{\theta}_i^*$ is a monolingual model trained on language i. In other words, relative perplexity of a given model on a language quantifies how much worse the multilingual model is compared to a monolingual model trained on the same language (with the same data).

Our results show that L-DRO reaches the best average score both in terms of perplexity and relative perplexity. While DRO is by far the best alternative in terms of worst perplexity, it ranks lowest in terms of relative perplexity. This is because its focus on languages with high perplexity comes at the detriment of low-entropy languages. Unsurprisingly, proportional mixing based methods ($\alpha = 0.3$ or 1) perform very well on high resource languages (*e.g.* English or French) but poorly on almost all other languages, leading to low performance along almost all metrics.

On the other hand, L-DRO performs much better in terms of relative perplexity: it outperforms both static mixing and DRO in terms of average and worst-case performance. Note that Baselined-DRO, which takes the entropy of individual languages into account when computing the maximum, reaches the lowest worst-case relative perplexity. However, it is a much more computationally demanding approach: it requires training 39 additional monolingual models to estimate the language entropies. In comparison, L-DRO comes in as a close second in terms of worst-case relative perplexity, with a comparatively much smaller computational overhead (periodic computation of $\mathbf{L}$), which makes it a much more appealing alternative.

[1] In experiments, we omit the Tagalog language, which is not included in mC4.

[2] We do not compare against PCGrad due to its underwhelming performance in the CIFAR-100 experiments and the prohibitive cost of its gradient projection step in this setting.

	absolute perplexity		relative perplexity	
	average	worst	average	worst
Static Mixing ($\alpha = 1$)	48.7	135.5	358.3	1170.7
Static Mixing ($\alpha = 0.3$)	24.0	41.2	130.6	213.7
Static Mixing ($\alpha = 0$)	23.9	48.7	128.2	198.1
DRO	24.5	**35.1**	142.7	286.7
Baselined-DRO	23.9	50.7	127.3	**168.5**
L-DRO	**23.6**	55.5	**126.0**	191.7
Monolingual	10.6	25.3	0	0

Table 1: Results on the multilingual language modeling experiment. We report results both in terms of absolute and relative perplexity.

6 Related Work

Much work in multi-task learning has focused on improving the optimization process when learning many tasks. Some prior work (Sener & Koltun, 2018; Lin et al., 2019; Ma et al., 2020) casts multi-task learning as multi-objective optimization, with the goal of finding a Pareto-optimal solution. Other work proposes to directly modify the gradients. In order to minimize gradient interference, Yu et al. (2020) project conflicting gradients on the normal plane of the other. Wang et al. (2021) alter both the direction and magnitude of gradients to achieve a certain gradient similarity.

Many other methods have been proposed to weight the task losses according to different criteria such as uncertainty (Kendall et al., 2018), reward magnitude (Hessel et al., 2019), and learning speed (Chen et al., 2018; Liu et al., 2019; Zheng et al., 2019). Most of the latter methods increase a task's loss weight when the learning speed for that task is low. Similar to learning speed, a few approaches (Guo et al., 2018; Jean et al., 2019) weight task losses based on performance, focusing on optimizing tasks with poor performance. Among these, our proposed method is most similar in spirit to performance weighting-based methods. Such methods, however, do not consider the optimization procedure and are not aware of task similarity. Loss weighting can be seen as a continuous relaxation of the task scheduling and sampling so that most task scheduling methods can be easily adapted to loss weighting methods, and vice versa (Ruder, 2019; Crawshaw, 2020). Kiperwasser & Ballesteros (2018) proposed different predefined sampling schedules while Sanh et al. (2019) proposed a proportional sampling strategy. In multilingual language modeling and machine translation (Aharoni et al., 2019; Conneau et al., 2020; Xue et al., 2021), proportional sampling with a temperature is typically employed.

The distributional robustness literature focuses on training models that perform well on various domains (Ben-Tal et al., 2009; Rahimian & Mehrotra, 2019). This is generally performed by minimizing the worst-case expected risk over a pre-determined family called the "uncertainty" set. There has been substantial work on formulating appropriate uncertainty sets (Hu & Hong, 2013; Gao & Kleywegt, 2016; Levy et al., 2020). Most relevant to our multitask setting are group-structured uncertainty sets, where the maximimum is taken over a mixture of sub-populations (Oren et al., 2019; Sagawa et al., 2020; Zhou et al., 2021).

7 Conclusion

In this paper, we argued for the importance of looking at worst-case performance when training and evaluating multi-task models, especially in cases where the distribution of tasks is not homogeneous (e.g., when some tasks are more difficult than others) or when there is an imbalance towards certain types of tasks. We proposed L-DRO, which enables efficient optimization of the worst-case loss without the pitfalls of existing DRO-based max-loss minimization approaches. We demonstrated the benefits of L-DRO in a synthetic scenario as well as in realistic image classification and language modeling experiments. Overall, we encourage the community to evaluate not only on homogeneous task collections (e.g., CIFAR-100) but also "heterogeneous" multitask learning scenarios (e.g., multilingual language modeling) when assessing the performance of a multitask model.

ETHICS STATEMENT

Measuring models only based on their average performance obfuscates their limitations on under-represented subpopulations and contributes to the marginalization of said groups. Developing algorithms that work well not just for the average user but for every type of user is thus important from an ethical perspective.

REPRODUCIBILITY STATEMENT

We take several steps to ensure reproducibility of our work. Pseudo-code describing the final version of our algorithm is given in Algorithm 3. Moreover, we detail experimental settings and model hyper-parameters in §5. Finally, all our experiments are performed on datasets that are openly available: CIFAR-100[3] (Krizhevsky et al., 2009) and mC4[4] (Xue et al., 2021).

REFERENCES

Roee Aharoni, Melvin Johnson, and Orhan Firat. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pp. 3874–3884, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1388. URL https://aclanthology.org/N19-1388.

Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust optimization*, volume 28. Princeton University Press, 2009.

George W Brown. Iterative solution of games by fictitious play. *Activity analysis of production and allocation*, 13(1):374–376, 1951.

Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*, pp. 77–91. PMLR, 2018.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International Conference on Machine Learning*, pp. 794–803. PMLR, 2018.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 8th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 8440–8451, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.747. URL https://aclanthology.org/2020.acl-main.747.

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

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pp. 1126–1135. PMLR, 2017.

Rui Gao and Anton J Kleywegt. Distributionally robust stochastic optimization with wasserstein distance. *arXiv preprint arXiv:1604.02199*, 2016.

Michelle Guo, Albert Haque, De-An Huang, Serena Yeung, and Li Fei-Fei. Dynamic task prioritization for multitask learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 270–287, 2018.

[3] https://www.cs.toronto.edu/~kriz/cifar.html

[4] https://www.tensorflow.org/datasets/catalog/c4#c4multilingual

Matteo Hessel, Hubert Soyer, Lasse Espeholt, Wojciech Czarnecki, Simon Schmitt, and Hado van Hasselt. Multi-task deep reinforcement learning with popart. In *Proceedings of the 33rd Meeting of the Association for Advancement of Artificial Intelligence (AAAI)*, pp. 3796–3803, 2019.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. XTREME: A Massively Multilingual Multi-task Benchmark for Evaluating Cross-lingual Generalization. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, 2020.

Zhaolin Hu and L Jeff Hong. Kullback-leibler divergence constrained distributionally robust optimization. *Available at Optimization Online*, 2013.

Edwin T Jaynes. Information theory and statistical mechanics. *Physical review*, 106(4):620, 1957.

Sébastien Jean, Orhan Firat, and Melvin Johnson. Adaptive scheduling for multi-task learning. *arXiv preprint arXiv:1909.06434*, 2019.

Fred Jelinek, Robert L Mercer, Lalit R Bahl, and James K Baker. Perplexity—a measure of the difficulty of speech recognition tasks. *The Journal of the Acoustical Society of America*, 62(S1): S63–S63, 1977.

Guy Jumarie. Relative information—what for? In *Relative Information*, pp. 1–11. Springer, 1990.

Jagat Narain Kapur and Hiremaglur K Kesavan. Entropy optimization principles and their applications. In *Entropy and energy dissipation in water resources*, pp. 3–20. Springer, 1992.

Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the 31st IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7482–7491, 2018.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014.

Eliyahu Kiperwasser and Miguel Ballesteros. Scheduled multi-task learning: From syntax to translation. *Transactions of the Association for Computational Linguistics*, 6:225–240, 2018.

Jyrki Kivinen and Manfred K Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *information and computation*, 132(1):1–63, 1997.

Alex Krizhevsky et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 66–75, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1007. URL https://aclanthology.org/P18-1007.

Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

Carlton E Lemke. Bimatrix equilibrium points and mathematical programming. *Management science*, 11(7):681–689, 1965.

Daniel Levy, Yair Carmon, John C Duchi, and Aaron Sidford. Large-scale methods for distributionally robust optimization. *Proceedings of the 34th Annual Conference on Neural Information Processing Systems (NeurIPS)*, 33, 2020.

Xi Lin, Hui-Ling Zhen, Zhenhua Li, Qing-Fu Zhang, and Sam Kwong. Pareto multi-task learning. *Advances in neural information processing systems*, 32:12060–12070, 2019.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

Shengchao Liu, Yingyu Liang, and Anthony Gitter. Loss-balanced task weighting to reduce negative transfer in multi-task learning. In *Proceedings of the 33rd Meeting of the Association for Advancement of Artificial Intelligence (AAAI)*, pp. 9977–9978, 2019.

Pingchuan Ma, Tao Du, and Wojciech Matusik. Effcient Continuous Pareto Exploration in Multi-Task Learning. In *Proceedings of ICML 2020*, 2020.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730*, 2018.

Sabrina J. Mielke, Ryan Cotterell, Kyle Gorman, Brian Roark, and Jason Eisner. What kind of language is hard to language-model? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4975–4989, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1491. URL https://aclanthology.org/P19-1491.

Arkadij Semenovič Nemirovskij and David Borisovich Yudin. Problem complexity and method efficiency in optimization. *Wiley-Interscience Series in Discrete Mathematics*, 1983.

Yonatan Oren, Shiori Sagawa, Tatsunori Hashimoto, and Percy Liang. Distributionally robust language modeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4227–4237, 2019. URL https://www.aclweb.org/anthology/D19-1432.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21, 2020. URL http://arxiv.org/abs/1910.10683.

Hamed Rahimian and Sanjay Mehrotra. Distributionally Robust Optimization: A Review. *arXiv preprint arXiv:1908.05659*, 2019.

Clemens Rosenbaum, Tim Klinger, and Matthew Riemer. Routing networks: Adaptive selection of non-linear functions for multi-task learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

Sebastian Ruder. An Overview of Multi-Task Learning in Deep Neural Networks. In *arXiv preprint arXiv:1706.05098*, 2017.

Sebastian Ruder. *Neural transfer learning for natural language processing*. PhD thesis, NUI Galway, 2019.

Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020. URL https://arxiv.org/pdf/1911.08731.pdf.

Victor Sanh, Thomas Wolf, and Sebastian Ruder. A hierarchical multi-task approach for learning embeddings from semantic tasks. In *Proceedings of the 33rd Meeting of the Association for Advancement of Artificial Intelligence (AAAI)*, pp. 6949–6956, 2019.

Ozan Sener and Vladlen Koltun. Multi-Task Learning as Multi-Objective Optimization. In *Proceedings of NIPS 2018*, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st Annual Conference on Neural Information Processing Systems (NIPS)*, pp. 5998–6008, 2017.

Alex Wang, Amapreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue benchmark leaderboard. https://gluebenchmark.com/leaderboard, 2019. URL https://gluebenchmark.com/leaderboard. Accessed: 2021-09-24.

Zirui Wang, Yulia Tsvetkov, Orhan Firat, and Yuan Cao. Gradient Vaccine: Investigating and Improving Multi-task Optimization in Massively Multilingual Models. In *Proceedings of ICLR 2021*, 2021.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pp. 483–498, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.41. URL https://aclanthology.org/2021.naacl-main.41.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Proceedings of NeurIPS 2020*, 2020.

Feng Zheng, Cheng Deng, Xing Sun, Xinyang Jiang, Xiaowei Guo, Zongqiao Yu, Feiyue Huang, and Rongrong Ji. Pyramidal person re-identification via multi-loss dynamic training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8514–8522, 2019.

Chunting Zhou, Xuezhe Ma, Paul Michel, and Graham Neubig. Examining and combating spurious features under distribution shift. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, 2021.

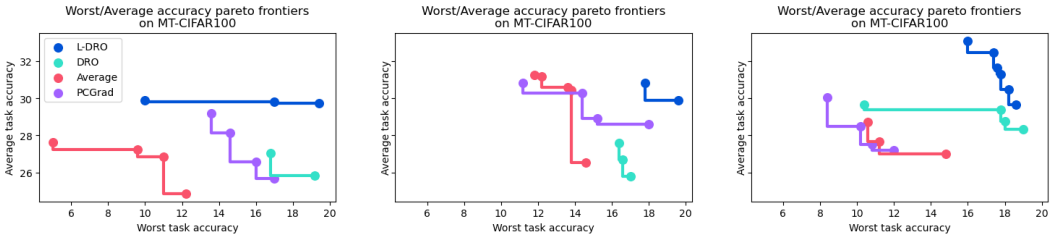


Figure 4: Average/worst task accuracy Pareto frontiers across 3 random seeds on multitask CIFAR-100

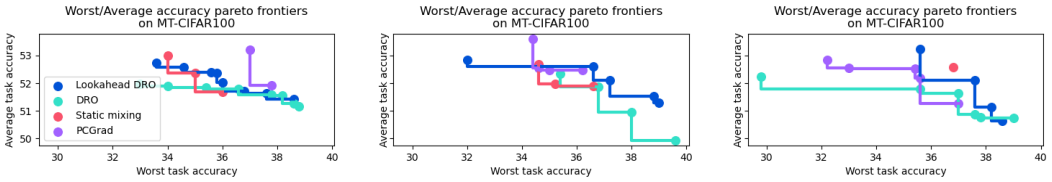


Figure 5: Average/worst task accuracy Pareto frontiers across 3 random seeds on multitask CIFAR-100

A Additional CIFAR-100 Results

A.1 Multiple Random Seeds

Figure 4 depicts results on the CIFAR-100 dataset for 3 different random seeds. We find that L-DRO consistently outperforms baselines.

A.2 Experiments with larger CNN

In Figure 5, we report CIFAR-100 results across 3 random seeds using a larger CNN model: 3 layers of 160 3×3 filters each followed by a ReLU activation and a 2×2 max-pooling layer, followed by a 2 layer multilayer perceptron with hidden dimension 320 and ReLU activations leading into a final fully connected layer mapping to the 100 classes. We find that the difference between the various method vanishes. Although L-DRO tends to achieve similar robust accuracy as DRO at higher average accuracy, the difference is small but not consistent across the different runs.