Gradient Surgery for Multi-Task Learning

Tianhe Yu [1] **Saurabh Kumar** [1] **Abhishek Gupta** [2] **Sergey Levine** [2] **Karol Hausman** [3] **Chelsea Finn** [1]

Abstract

While deep learning and deep reinforcement learning (RL) systems have demonstrated impressive results in domains such as image classification, game playing, and robotic control, data efficiency remains a major challenge. Multi-task learning has emerged as a promising approach for sharing structure across multiple tasks to enable more efficient learning. However, the multi-task setting presents a number of optimization challenges, making it difficult to realize large efficiency gains compared to learning tasks independently. The reasons why multi-task learning is so challenging compared to single-task learning are not fully understood. In this work, we identify a set of three conditions of the multi-task optimization landscape that cause detrimental gradient interference, and develop a simple yet general approach for avoiding such interference between task gradients. We propose a form of gradient surgery that projects a task's gradient onto the normal plane of the gradient of any other task that has a *conflicting* gradient. On a series of challenging multi-task supervised and multi-task RL problems, this approach leads to substantial gains in efficiency and performance. Further, it is model-agnostic and can be combined with previously-proposed multi-task architectures for enhanced performance.

1 Introduction

While deep learning and deep reinforcement learning (RL) have shown considerable promise in enabling systems to learn complex tasks, the data requirements of current methods make it difficult to learn a breadth of capabilities, particularly when all tasks are learned individually from scratch. A natural approach to such multi-task learning problems is to train a network on all tasks jointly, with the aim of discovering shared structure across the tasks in a way that achieves greater efficiency and performance than solving tasks indi-

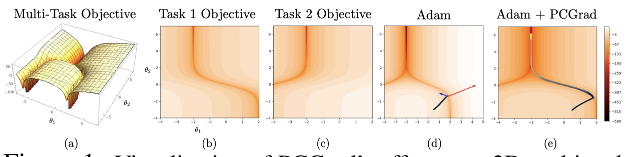

Figure 1: Visualization of PCGrad's effect on a 2D multi-task optimization problem. (a) A multi-task objective landscape. (b) & (c) Contour plots of the individual task objectives that comprise the multi-task objective. (d) Trajectory of gradient updates on the multi-task objective using the Adam optimizer. The gradient vectors of the two tasks at the end of the trajectory are indicated by blue and red arrows, where the relative lengths are on a log scale.(e) Trajectory of gradient updates on the multi-task objective using Adam with PCGrad. For (d) and (e), the optimization trajectory goes from black to yellow.

vidually. However, learning multiple tasks all at once results is a difficult optimization problem, sometimes leading to *worse* overall performance and data efficiency compared to learning tasks individually (Parisotto et al., 2015; Rusu et al., 2016a). These optimization challenges are so prevalent that multiple multi-task RL algorithms have considered using independent training as a subroutine of the algorithm before distilling the independent models into a multi-tasking model (Levine et al., 2016; Parisotto et al., 2015; Rusu et al., 2016a; Ghosh et al., 2017; Teh et al., 2017), producing a multi-task model but losing out on the efficiency gains over independent training. If we could tackle the optimization challenges of multi-task learning effectively, we may be able to actually realize the hypothesized benefits of multi-task learning without the cost in final performance.

While there has been a significant amount of research in multi-task learning (Caruana, 1997; Ruder, 2017), the optimization challenges are not well understood. Prior work has described varying learning speeds of different tasks (Chen et al., 2017; Hessel et al., 2019) and plateaus in the optimization landscape (Schaul et al., 2019) as potential causes, whereas a range of other works have focused on the model architecture (Misra et al., 2016b; Liu et al., 2018). In this work, we instead hypothesize that one of the main optimization issues in multi-task learning arises from gradients from different tasks conflicting with one another in a way that is detrimental to making progress. We define two gradients to be conflicting if they point away from one another, i.e., have a negative cosine similarity. We hypothesize that such conflict is detrimental when a) conflicting gradients coincide with b) high positive curvature and c) a large difference in

[1]Stanford University [2]UC Berkeley [3]Robotics at Google. Correspondence to: Tianhe Yu <tianheyu@cs.stanford.edu>.

Preliminary work. Under review.

gradient magnitudes.

As an illustrative example, consider the 2D optimization landscapes of two task objectives in Figure 1a-c. The optimization landscape of each task consists of a deep valley, a property that has been observed in neural network optimization landscapes (Goodfellow et al., 2014), and the bottom of each valley is characterized by high positive curvature and large differences in the task gradient magnitudes. Under such circumstances, the multi-task gradient is dominated by one task gradient, which comes at the cost of degrading the performance of the other task. Further, due to high curvature, the improvement in the dominating task may be overestimated, while the degradation in performance of the non-dominating task may be underestimated. As a result, the optimizer struggles to make progress on the optimization objective. In Figure 1d), the optimizer reaches the deep valley of task 1, but is unable to traverse the valley in a parameter setting where there are conflicting gradients, high curvature, and a large difference in gradient magnitudes (see gradients plotted in Fig. 1d). In Section 5.4, we find experimentally that this *tragic triad* also occurs in a higher-dimensional neural network multi-task learning problem.

The core contribution of this work is a method for mitigating gradient interference by altering the gradients directly, i.e. by performing "gradient surgery." If two gradients are conflicting, we alter the gradients by projecting each onto the normal plane of the other, preventing the interfering components of the gradient from being applied to the network. We refer to this particular form of gradient surgery as *projecting conflicting gradients* (PCGrad). PCGrad is model-agnostic, requiring only a single modification to the application of gradients. Hence, it is easy to apply to a range of problem settings, including multi-task supervised learning and multi-task reinforcement learning, and can also be readily combined with other multi-task learning approaches, such as those that modify the architecture. We theoretically prove the local conditions under which PCGrad improves upon standard multi-task gradient descent, and we empirically evaluate PCGrad on a variety of challenging problems, including multi-task CIFAR classification, multi-objective scene understanding, a challenging multi-task RL domain, and goal-conditioned RL. Across the board, we find PCGrad leads to substantial improvements in terms of data efficiency, optimization speed, and final performance compared to prior approaches, including a more than 30% absolute improvement in multi-task reinforcement learning problems. Further, on multi-task supervised learning tasks, PCGrad can be successfully combined with prior state-of-the-art methods for multi-task learning for even greater performance.

2 Multi-Task Learning with PCGrad

While the multi-task problem can in principle be solved by simply applying a standard single-task algorithm with a suitable task identifier provided to the model, or a simple multi-head or multi-output model, a number of prior works (Parisotto et al., 2015; Rusu et al., 2016a; Sener & Koltun, 2018) have found this learning problem to be difficult. In this section, we introduce notation, identify possible causes for the difficulty of multi-task optimization, propose a simple and general approach to mitigate it, and theoretically analyze the proposed approach.

2.1 Preliminaries: Problem and Notation

The goal of multi-task learning is to find parameters θ of a model f_θ that achieve high average performance across all the training tasks drawn from a distribution of tasks $p(\mathcal{T})$. More formally, we aim to solve the problem: $\min_\theta \mathbb{E}_{\mathcal{T}_i \sim p(\mathcal{T})}\left[\mathcal{L}_i(\theta)\right]$, where $\mathcal{L}_i$ is a loss function for the i-th task $\mathcal{T}_i$ that we want to minimize. For a set of tasks, $\{\mathcal{T}_i\}$, we denote the multi-task loss as $\mathcal{L}(\theta) = \sum_i \mathcal{L}_i(\theta)$, and the gradients of each task as $\mathbf{g}_i = \nabla \mathcal{L}_i(\theta)$ for a particular θ. (We drop the reliance on θ in the notation for brevity.) To obtain a model that solves a specific task from the task distribution $p(\mathcal{T})$, we define a task-conditioned model $f_\theta(y|x, z_i)$, with input x, output y, and encoding z_i for task $\mathcal{T}_i$, which could be provided as a one-hot vector or in any other form.

2.2 The Tragic Triad: Conflicting Gradients, Dominating Gradients, High Curvature

We hypothesize that a key optimization issue in multi-task learning arises from conflicting gradients, where gradients for different tasks point away from one another as measured by a negative inner product. However, conflicting gradients are not detrimental on their own. Indeed, simply averaging task gradients should provide the correct solution to descend the multi-task objective. However, there are conditions under which such conflicting gradients lead to significantly degraded performance. Consider a two-task optimization problem. If the gradient of one task is much larger in magnitude than the other, it will dominate the average gradient. If there is also high positive curvature along the directions of the task gradients, then the improvement in performance from the dominating task may be significantly overestimated, while the degradation in performance from the dominated task may be significantly underestimated. Hence, we can characterize the co-occurrence of three conditions as follows: (a) when gradients from multiple tasks are in conflict with one another (b) when the difference in gradient magnitudes is large, leading to some task gradients dominating others, and (c) when there is high curvature in the multi-task optimization landscape. We formally define the three conditions below.

Definition 1. *We define ϕ_{ij} as the angle between two task gradients $\mathbf{g}_i$ and $\mathbf{g}_j$. We define the gradients as* ***conflicting*** *when* $\cos\phi_{ij} < 0$.

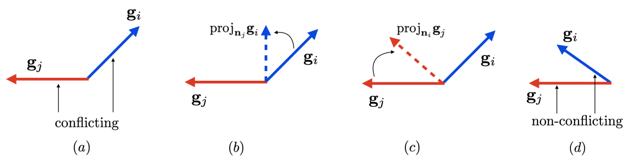

Figure 2: Conflicting gradients and PCGrad. In (a), tasks i and j have conflicting gradient directions, which can lead to destructive interference. In (b) and (c), we illustrate the PCGrad algorithm in the case where gradients are conflicting. PCGrad projects the gradient of task i onto the normal vector of task j's gradient, and vice versa. Gradients of tasks that are not conflicting (d) are not altered under PCGrad, allowing for constructive interaction.

Definition 2. *We define the **gradient magnitude similarity** between two gradients $\mathbf{g}_i$ and $\mathbf{g}_j$ as*

$$\Phi(\mathbf{g}_i, \mathbf{g}_j) = \frac{2\|\mathbf{g}_i\|_2\|\mathbf{g}_j\|_2}{\|\mathbf{g}_i\|_2^2 + \|\mathbf{g}_j\|_2^2}$$

When the magnitude of two gradients is the same, this value is equal to 1. As the gradient magnitudes become increasingly different, this value goes to zero.

Definition 3. *We define **multi-task curvature** as*

$$\mathbf{H}(\mathcal{L}; \theta, \theta') = \int_0^1 \nabla\mathcal{L}(\theta)^T \nabla^2\mathcal{L}(\theta + a(\theta' - \theta))\nabla\mathcal{L}(\theta)da$$

which is the averaged curvature of $\mathcal{L}$ between θ and θ' in the direction of the multi-task gradient $\nabla\mathcal{L}(\theta)$. When $\mathbf{H}(\mathcal{L}; \theta, \theta') > C$ for some large positive constant C for the model parameter θ and θ' at the current and next iteration, we characterize that the high curvature phenomenon exists in the optimization landscape.

We aim to study the tragic triad and observe the presence of the three conditions listed above through two examples. First, consider the two-dimensional optimization landscape illustrated in Fig. 1a, where the landscape for each task objective corresponds to a deep and curved valley with large curvatures (Fig. 1b and 1c). The optima of this multi-task objective correspond to where the two valleys meet. More details on the optimization landscape are in Appendix E. Particular points of this optimization landscape exhibit the three described conditions, and we observe that, the Adam (Kingma & Ba, 2014) optimizer stalls precisely at one of these points (see Fig. 1d), preventing it from reaching an optimum. This provides some empirical evidence for our hypothesis. Our experiments in Section 5.4 further suggest that this phenomenon occurs in multi-task learning with deep neural networks.

Motivated by these observations, we develop an algorithm that aims to alleviate the optimization challenges caused by conflicting gradients, dominating gradients, and high curvature, which we describe next.

Algorithm 1 PCGrad Update Rule

```
Require: Model parameters $\theta$, task minibatch $\mathcal{B} = \{\mathcal{T}_k\}$
 1: $\mathbf{g}_k \leftarrow \nabla_\theta \mathcal{L}_k(\theta)\ \forall k$
 2: for $\mathcal{T}_i \in \mathcal{B}$ do
 3:   for $\mathcal{T}_j \overset{\text{uniformly}}{\sim} \mathcal{B} \setminus \mathcal{T}_i$ in random order do
 4:     if $\mathbf{g}_i^T \mathbf{g}_j < 0$ then
 5:       // Subtract the projection of $\mathbf{g}_i$ onto $\mathbf{g}_j$
 6:       Set $\mathbf{g}_i = \mathbf{g}_i - \frac{\mathbf{g}_i \cdot \mathbf{g}_j}{\|\mathbf{g}_j\|^2}\mathbf{g}_j$
 7:     end if
 8:   end for
 9:   Store $\mathbf{g}_i^{\text{PC}} = \mathbf{g}_i$
10: end for
11: return update $\Delta\theta = \mathbf{g}^{\text{PC}} = \sum_i \mathbf{g}_i^{\text{PC}}$
```

2.3 PCGrad: Project Conflicting Gradients

Our goal is to break one condition of the tragic triad by directly altering the gradients themselves to prevent conflict. In this section, we outline our approach for altering the gradients. In the next section, we will theoretically show that de-conflicting gradients can benefit multi-task learning when dominating gradients and high curvatures are present.

To be maximally effective and widely applicable, we aim to alter the gradients in a way that allows for positive interactions between the task gradients and does not introduce assumptions on the form of the model. Hence, when gradients do not conflict, we do not change the gradients. When gradients do conflict, the goal of PCGrad is to modify the gradients for each task so as to minimize negative conflict with other task gradients, which will in turn mitigate under- and over-estimation problems arising from high curvature.

To deconflict gradients during optimization, PCGrad adopts a simple procedure: if the gradients between two tasks are in conflict, i.e. their cosine similarity is negative, we project the gradient of each task onto the normal plane of the gradient of the other task. This amounts to removing the conflicting component of the gradient for the task, thereby reducing the amount of destructive gradient interference between tasks. A pictorial description of this idea is shown in Fig. 2. Suppose the gradient for task $\mathcal{T}_i$ is $\mathbf{g}_i$, and the gradient for task $\mathcal{T}_j$ is $\mathbf{g}_j$. PCGrad proceeds as follows: (1) First, it determines whether $\mathbf{g}_i$ conflicts with $\mathbf{g}_j$ by computing the cosine similarity between vectors $\mathbf{g}_i$ and $\mathbf{g}_j$, where negative values indicate conflicting gradients. (2) If the cosine similarity is negative, we replace $\mathbf{g}_i$ by its projection onto the normal plane of $\mathbf{g}_j$: $\mathbf{g}_i = \mathbf{g}_i - \frac{\mathbf{g}_i \cdot \mathbf{g}_j}{\|\mathbf{g}_j\|^2}\mathbf{g}_j$. If the gradients are not in conflict, i.e. cosine similarity is non-negative, the original gradient $\mathbf{g}_i$ remains unaltered. (3) PCGrad repeats this process across all of the other tasks sampled in random order from the current batch $\mathcal{T}_j\ \forall\ j \neq i$, resulting in the gradient $\mathbf{g}_i^{\text{PC}}$ that is applied for task $\mathcal{T}_i$. We perform the same procedure for all tasks in the batch to obtain their respective gradients. The full update procedure is described in

Algorithm 1 and a discussion on using a random task order is included in Appendix H.

This procedure, while simple to implement, ensures that the gradients that we apply for each task per batch interfere minimally with the other tasks in the batch, mitigating the conflicting gradient problem, producing a variant on standard first-order gradient descent in the multi-objective setting. In practice, PCGrad can be combined with any gradient-based optimizer, including commonly used methods such as SGD with momentum and Adam (Kingma & Ba, 2014), by simply passing the computed update to the respective optimizer instead of the original gradient. Our experimental results verify the hypothesis that this procedure reduces the problem of conflicting gradients, and find that, as a result, learning progress is substantially improved.

2.4 Theoretical Analysis of PCGrad

In this section, we theoretically analyze the performance of PCGrad with two tasks:

Definition 4. *Consider two task loss functions* $\mathcal{L}_1 : \mathbb{R}^n \to \mathbb{R}$ *and* $\mathcal{L}_2 : \mathbb{R}^n \to \mathbb{R}$. *We define the two-task learning objective as* $\mathcal{L}(\theta) = \mathcal{L}_1(\theta) + \mathcal{L}_2(\theta)$ *for all* $\theta \in \mathbb{R}^n$, *where* $\mathbf{g}_1 = \nabla\mathcal{L}_1(\theta)$, $\mathbf{g}_2 = \nabla\mathcal{L}_2(\theta)$, *and* $\mathbf{g} = \mathbf{g}_1 + \mathbf{g}_2$.

We first aim to verify that the PCGrad update corresponds to a sensible optimization procedure under simplifying assumptions. We analyze convergence of PCGrad in the convex setting, under standard assumptions:

Theorem 1. *Assume* $\mathcal{L}_1$ *and* $\mathcal{L}_2$ *are convex and differentiable. Suppose the gradient of* $\mathcal{L}$ *is Lipschitz continuous with constant* $L > 0$. *Then, the PCGrad update rule with step size* $t \leq \frac{1}{L}$ *will converge to either (1) a location in the optimization landscape where* $\cos(\phi_{12}) = -1$ *or (2) the optimal value* $\mathcal{L}(\theta^*)$.

Proof. See Appendix A. □

Theorem 1 states that application of the PCGrad update in the two-task setting with a convex and Lipschitz multi-task loss function $\mathcal{L}$ leads to convergence to either the minimizer of $\mathcal{L}$ or a potentially sub-optimal objective value. A sub-optimal solution occurs when the cosine similarity between the gradients of the two tasks is exactly -1, i.e. the gradients directly conflict, leading to zero gradient after applying PCGrad. However, in practice, since we are using SGD, which is a noisy estimate of the true batch gradients, the cosine similarity between the gradients of two tasks in a minibatch is unlikely to be -1, thus avoiding this scenario.

Now that we have checked the sensibility of PCGrad, we aim to understand how PCGrad relates to the three conditions in the tragic triad. In particular, we derive sufficient conditions under which PCGrad achieves lower loss after one update. Here, we still analyze the two task setting, but no longer assume convexity of the loss functions.

Definition 5. *We define the* ***multi-task curvature bounding measure***

$$\xi(\mathbf{g}_1, \mathbf{g}_2) = (1 - \cos^2\phi_{12})\frac{\|\mathbf{g}_1 - \mathbf{g}_2\|_2^2}{\|\mathbf{g}_1 + \mathbf{g}_2\|_2^2}.$$

With the above definition, we present our next theorem:

Theorem 2. *Suppose* $\mathcal{L}$ *is differentiable and the gradient of* $\mathcal{L}$ *is Lipschitz continuous with constant* $L > 0$. *Let* θ^{MT} *and* θ^{PCGrad} *be the parameters after applying one update to* θ *with* $\mathbf{g}$ *and PCGrad-modified gradient* $\mathbf{g}^{PC}$ *respectively, with step size* $t > 0$. *Moreover, assume* $\mathbf{H}(\mathcal{L}; \theta, \theta^{MT}) \geq \ell\|\mathbf{g}\|_2^2$ *for some constant* $\ell \leq L$, *i.e. the multi-task curvature is lower-bounded. Then* $\mathcal{L}(\theta^{PCGrad}) \leq \mathcal{L}(\theta^{MT})$ *if*

(a) $\cos\phi_{12} \leq -\Phi(\mathbf{g}_1, \mathbf{g}_2)$,

(b) $\ell \geq \xi(\mathbf{g}_1, \mathbf{g}_2)L$, *and*

(c) $t \geq \frac{2}{\ell - \xi(\mathbf{g}_1, \mathbf{g}_2)L}$.

Proof. See Appendix B. □

Intuitively, Theorem 2 implies that PCGrad achieves lower loss value after a single gradient update compared to standard gradient descent in multi-task learning when (i) the angle between task gradients is not too small, i.e. the two tasks need to conflict sufficiently (condition (a)), (ii) the difference in magnitude needs to be sufficiently large (condition (a)), (iii) the curvature of the multi-task gradient should be large (condition (b)), (iv) and the learning rate should be big enough so that large curvature would lead to overestimation of performance improvement on the dominating task and underestimation of performance degradation on the dominated task (condition (c)). These first three points (i-iii) correspond to exactly the triad of conditions outlined in Section 2.2, while the latter condition (iv) is desirable as we hope to learn quickly. We empirically validate that the first three points, (i-iii), are frequently met in a neural network multi-task learning problem in Figure 5 in Section 5.4.

For additional analysis, including complete sufficient and necessary conditions for the PCGrad update to outperform the vanilla multi-task gradient, see Appendix C.

3 PCGrad in Practice

We use PCGrad in supervised learning and reinforcement learning problems with multiple tasks or goals. Here, we discuss the practical application of PCGrad to those settings.

In multi-task supervised learning, each task $\mathcal{T}_i \sim p(\mathcal{T})$ has a corresponding training dataset $\mathcal{D}_i$ consisting of labeled training examples, i.e. $\mathcal{D}_i = \{(x, y)_n\}$. The objective for each task in this supervised setting is then defined as $\mathcal{L}_i(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}_i}\left[-\log f_\theta(y|x, z_i)\right]$, where z_i is a one-hot encoding of task $\mathcal{T}_i$.

At each training step, we randomly sample a batch of data points $\mathcal{B}$ from the whole dataset $\bigcup_i \mathcal{D}_i$ and then group the sampled data with the same task encoding into small batches denoted as $\mathcal{B}_i$ for each $\mathcal{T}_i$ represented in $\mathcal{B}$. We denote the set of tasks appearing in $\mathcal{B}$ as $\mathcal{B}_\mathcal{T}$. After sampling, we precompute the gradient of each task in $\mathcal{B}_\mathcal{T}$ as

$$\nabla_\theta \mathcal{L}_i(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{B}_i}\left[-\nabla_\theta \log f_\theta(y|x, z_i)\right].$$

Given the set of precomputed gradients $\nabla_\theta \mathcal{L}_i(\theta)$, we also precompute the cosine similarity between all pairs of the gradients in the set. Using the pre-computed gradients and their similarities, we can obtain the PCGrad update by following Algorithm 1, without re-computing task gradients nor backpropagating into the network.

Since the PCGrad procedure is only modifying the gradients of shared parameters in the optimization step, it is model-agnostic and can be readily applied to any architecture designed for supervised multi-task learning. In Section 5, we combine PCGrad with two state-of-the-art architectures for multi-task learning, which leads to noticeable improvement over their original performance.

For multi-task reinforcement learning, PCGrad can be readily applied to policy gradient methods by directly updating the computed policy gradient of each task, following Algorithm 1, analogous to the supervised learning setting. For actor-critic algorithms, it is also straightforward to apply PCGrad: we simply replace the task gradients for both the actor and the critic by their gradients computed via PCGrad.

In our experiments, we apply PCGrad to the soft actor-critic (SAC) algorithm (Haarnoja et al., 2018), an off-policy RL method. In the context of SAC specifically, we also propose to learn the temperature α for adjusting entropy of the policy on a per-task basis. This allows the method to control the entropy of the multi-task policy per-task. For more details on the practical instantiations of multi-task and goal-conditioned reinforcement learning, see Appendix D.

4 Related Work

Algorithms for multi-task learning typically consider how to train a single model that can solve a variety of different tasks (Caruana, 1997; Bakker & Heskes, 2003; Ruder, 2017). The multi-task formulation has been applied to many different settings, including supervised learning (Zhang et al., 2014; Long & Wang, 2015; Yang & Hospedales, 2016; Sener & Koltun, 2018; Zamir et al., 2018) and reinforcement-learning (Espeholt et al., 2018; Wilson et al., 2007), as well as many different domains, such as vision (Bilen & Vedaldi, 2016; Misra et al., 2016a; Kokkinos, 2017; Liu et al., 2018; Zamir et al., 2018), language (Collobert & Weston, 2008; Dong et al., 2015; McCann et al., 2018; Radford et al., 2019) and robotics (Riedmiller et al., 2018; Wulfmeier et al., 2019; Hausman et al., 2018). While multi-task learning has the promise of accelerating acquisition of large task repertoires, in practice it presents a challenging optimization problem, which has been tackled in several ways in prior work.

A number of architectural solutions have been proposed to the multi-task learning problem based on multiple modules or paths (Fernando et al., 2017; Devin et al., 2016; Misra et al., 2016b; Rusu et al., 2016b; Rosenbaum et al., 2018; Vandenhende et al., 2019; Rosenbaum et al., 2018), or using attention-based architectures (Liu et al., 2018; Maninis et al., 2019). Our work is agnostic to the model architecture and can be combined with prior architectural approaches in a complementary fashion. A different set of multi-task learning approaches aim to decompose the problem into multiple local problems, often corresponding to each task, that are significantly easier to learn, akin to divide and conquer algorithms (Levine et al., 2016; Rusu et al., 2016a; Parisotto et al., 2015; Teh et al., 2017; Ghosh et al., 2017; Czarnecki et al., 2019). Eventually, the local models are combined into a single, multi-task policy using different distillation techniques (outlined in (Hinton et al., 2015; Czarnecki et al., 2019)). In contrast to these methods, we propose a simple and cogent scheme for multi-task learning that allows us to learn the tasks simultaneously using a single, shared model without the need for network distillation.

Similarly to our work, a number of prior approaches have observed the difficulty of optimization in the multi-task learning setting (Hessel et al., 2019; Kendall et al., 2018b; Schaul et al., 2019). Our work, in contrast, suggests that the challenge in multi-task learning may be attributed to what we describe as the tragic triad of multi-task learning (i.e., conflicting gradients, high curvature, and large gradient differences), which we address directly by introducing a simple and practical algorithm that deconflicts gradients from different tasks. Prior works combat optimization challenges by rescaling task gradients (Sener & Koltun, 2018; Chen et al., 2018). We alter both the magnitude and direction of the gradient, which we find to be critical for good performance (see Fig. 3). Prior work has also used the cosine similarity between gradients to define when an auxiliary task might be useful for single-task learning (Du et al., 2018). We similarly use cosine similarity between gradients to determine if the gradients between a pair of tasks are in conflict. Unlike Du et al. (2018), we use this measure for effective multi-task learning, instead of ignoring auxiliary objectives.

Multiple approaches to continual learning have studied how to prevent gradient updates from adversely affecting previously-learned tasks through various forms of gradient projection (Lopez-Paz & Ranzato, 2017; Chaudhry et al., 2018). These methods focus on sequential learning settings, and either solve for the gradient projections using quadratic programming (Lopez-Paz & Ranzato, 2017), or only project onto the normal plane of the average gradient of past tasks (Chaudhry et al., 2018). In contrast, our work

#P.	Architecture	Weighting	Segmentation		Depth		Surface Normal				
			(Higher Better)		(Lower Better)		Angle Distance (Lower Better)		Within t° (Higher Better)		
			mIoU	Pix Acc	Abs Err	Rel Err	Mean	Median	11.25	22.5	30
$\approx$3	Cross-Stitch[‡]	Equal Weights	14.71	50.23	0.6481	0.2871	33.56	28.58	20.08	40.54	51.97
		Uncert. Weights[*]	15.69	52.60	0.6277	0.2702	32.69	27.26	21.63	42.84	54.45
		DWA[†], $T = 2$	**16.11**	**53.19**	**0.5922**	**0.2611**	**32.34**	**26.91**	**21.81**	**43.14**	**54.92**
1.77	MTAN[†]	Equal Weights	**17.72**	55.32	**0.5906**	0.2577	31.44	**25.37**	$\boxed{23.17}$	45.65	57.48
		Uncert. Weights[*]	17.67	**55.61**	0.5927	0.2592	**31.25**	25.57	22.99	**45.83**	**57.67**
		DWA[†], $T = 2$	17.15	54.97	0.5956	**0.2569**	31.60	25.46	22.48	44.86	57.24
1.77	MTAN[†]+ PCGrad (ours)	Uncert. Weights[*]	$\boxed{\mathbf{20.17}}$	$\boxed{\mathbf{56.65}}$	$\boxed{\mathbf{0.5904}}$	$\boxed{\mathbf{0.2467}}$	$\boxed{\mathbf{30.01}}$	$\boxed{\mathbf{24.83}}$	22.28	$\boxed{\mathbf{46.12}}$	$\boxed{\mathbf{58.77}}$

Table 1: Three-task learning on the NYUv2 dataset: 13-class semantic segmentation, depth estimation, and surface normal prediction results. #P shows the total number of network parameters. We highlight the best performing combination of multi-task architecture and weighting in bold. The top validation scores for each task are annotated with boxes. The symbols indicate prior methods: [*]: (Kendall et al., 2018a), [†]: (Liu et al., 2018), [‡]: (Misra et al., 2016b). Performance of other methods as reported in Liu et al. (2018).

	% accuracy
task specific, 1-fc (Rosenbaum et al., 2018)	42
task specific, all-fc (Rosenbaum et al., 2018)	49
cross stitch, all-fc (Misra et al., 2016b)	53
routing, all-fc + WPL (Rosenbaum et al., 2019)	74.7
independent	67.7
PCGrad (ours)	71
routing-all-fc + WPL + PCGrad (ours)	**77.5**

Table 2: CIFAR-100 multi-task results. When combined with routing networks, PCGrad leads to a large improvement.

focuses on positive transfer when simultaneously learning multiple tasks, does not require solving a QP, and *iteratively* projects the gradients of each task instead of *averaging*. Finally, our method is distinct from and solves a different problem than the projected gradient method (Calamai & Moré, 1987), which is an approach for constrained optimization that projects gradients onto the constraint manifold.

5 Experiments

The goal of our experiments is to study the following questions: (1) Is the tragic triad of multi-task learning a major factor in making optimization for multi-task learning challenging? (2) Does PCGrad make the optimization problems easier for various multi-task learning problems including supervised, reinforcement, and goal-conditioned reinforcement learning settings across different task families? (3) Can PCGrad be combined with other multi-task learning approaches to further improve performance? To broadly evaluate PCGrad, we consider multi-task supervised learning, multi-task RL, and goal-conditioned RL problem setups. For details on the experimental set-up and model architectures see Appendix I.

5.1 Multi-Task Supervised Learning

To answer question (3), we perform experiments on three standard multi-task supervised learning datasets: MultiMNIST, multi-task CIFAR-100 and NYUv2. We include the results on MultiMNIST in Appendix F.

For CIFAR-100, we follow (Rosenbaum et al., 2018) to treat 20 coarse labels in the dataset as distinct tasks and create a dataset with 20 tasks and 2500 training instances as well as 500 test instances per task. We combine PCGrad with a powerful multi-task learning architecture, routing networks (Rosenbaum et al., 2018; 2019), by simply projecting gradients of the shared parameters in routing networks. As shown in Table 2, applying PCGrad to a single network achieves 71% classification accuracy, which outperforms most of the prior methods such as independent training and cross-stitch (Misra et al., 2016b). Though routing networks achieve better performance than PCGrad on its own, PCGrad is complementary to routing networks and combining PCGrad with routing networks leads to a 2.8% absolute improvement in test accuracy averaged over 3 runs.

We also combine PCGrad with another state-of-art multi-task learning algorithm, MTAN (Liu et al., 2018), and evaluate the performance on a more challenging indoor scene dataset, NYUv2, which contains 3 tasks: 13-class semantic segmentation, depth estimation, and surface normal prediction. We compare MTAN with PCGrad to a list of methods mentioned in Appendix I.1, where each method is trained with three different weighting schemes as in (Liu et al., 2018), equal weighting, weight uncertainty (Kendall et al., 2018a), and DWA (Liu et al., 2018). We only run MTAN with PCGrad with weight uncertainty as we find weight uncertainty as the most effective scheme for training MTAN. The results comparing Cross-Stitch, MTAN and MTAN + PCGrad are presented in Table 1 while the full comparison can be found in Table 4 in the Appendix I.3. MTAN with PCGrad is able to achieve the best scores in 8 out of the 9 categories where there are 3 categories per task.

Our multi-task supervised learning results indicate that PCGrad can be seamlessly combined with state-of-art multi-task learning architectures and further improve their results on established supervised multi-task learning benchmarks.

5.2 Multi-Task Reinforcement Learning

To answer question (2), we first consider the multi-task reinforcement learning problem and evaluate our algorithm

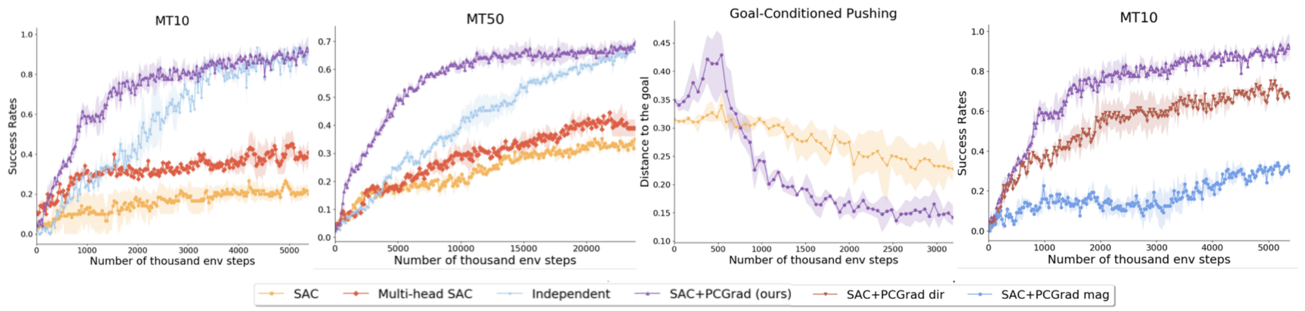

Figure 3: For the three plots on the left, we show learning curves on MT10, MT50 and goal-conditioned pushing respectively. PCGrad outperforms the other methods in the three settings in terms of both success rates / average distance to the goal and data efficiency. In the rightmost plot, we present the ablation study on only using the magnitude and the direction of the gradients modified by PCGrad. PCGrad outperforms both ablations, indicating the importance of modifying both the gradient directions and magnitudes in multi-task learning.

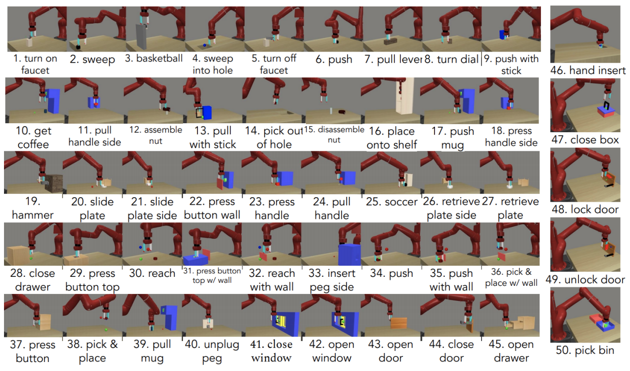

Figure 4: Visualization of 50 tasks used in MT50 from Meta-World (Yu et al., 2019), used in our multi-task RL experiments. MT10 is a subset of these tasks, which includes reach, push, pick & place, open drawer, close drawer, open door, press button top, open window, close window, and insert peg inside.

on the recently proposed Meta-World benchmark (Yu et al., 2019). In particular, we test all methods on the MT10 and MT50 benchmarks in Meta-World, which contains 10 and 50 manipulation tasks respectively shown in Figure 4. At each data collection step, we collect 600 samples for each task, and at each training step, we sample 128 datapoints per task from corresponding replay buffers. The results are shown in left two plots in Figure 3. We measure success according to the metrics used in the Meta-World benchmark where the reported the success rates are averaged across tasks. For all methods, we apply the temperature adjustment strategy as discussed in Section 3 to learn a separate alpha term per task as the task encoding in MT10 and MT50 is just a one-hot encoding. PCGrad combined with SAC learns all tasks with the best data efficiency and successfully solves all of the 10 tasks in MT10 and about 70% of the 50 tasks in MT50. Training a single SAC policy and a multi-head policy turns out to be unable to acquire half of the skills in both MT10 and MT50, suggesting that eliminating gradient interference across tasks can significantly boost performance of multi-task RL. Training independent SAC agents is able to eventually solve all tasks in MT10 and 70% of the tasks in MT50, but requires about 2 millions and 15 millions more samples than PCGrad with SAC in MT10 and MT50 respectively, implying that applying PCGrad can result in leveraging shared structure among tasks that expedites multi-task learning.

As noted by Yu et al. (2019), these tasks involve fairly distinct behavior motions, which makes learning all of them with a single policy challenging as demonstrated by poor baseline performance. The ability to learn these tasks together opens the door for a number of interesting extensions to meta-learning, goal conditioned RL and generalization to novel task families. We present the results of PCGrad on goal-conditioned RL in the following subsection.

We also provide an ablation study on the importance of correcting the gradient direction and scaling the gradient magnitudes in PCGrad. We construct two variants of PCGrad: (1) only applying the gradient direction corrected with PCGrad while keeping the gradient magnitude unchanged and (2) only applying the gradient magnitude computed by PCGrad while keeping the gradient direction unchanged. As shown in the rightmost plot in Figure 3, both variants perform worse than PCGrad and the variant where we only vary the gradient magnitude is much worse than PCGrad. This emphasizes the importance of the orientation change, which is particularly notable as multiple prior works only alter gradient magnitudes (Chen et al., 2017; Sener & Koltun, 2018). For a direct comparison to Chen et al. (2017), see Figure 6 in Appendix G.

5.3 Goal-Conditioned Reinforcement Learning

For our goal-conditioned RL evaluation, we adopt the goal-conditioned robotic pushing task with a Sawyer robot where the goals are represented as the concatenations of the initial positions of the puck to be pushed and the its goal location, both of which are uniformly sampled (details in Appendix I.2). We also apply the temperature adjustment strategy as discussed in Section 3 to predict the temperature for entropy term given the goal. We summarize the results in the plot second from right in Figure 3. PCGrad with SAC achieves better performance in terms of average distance to the goal position, while the vanilla SAC agent is struggling

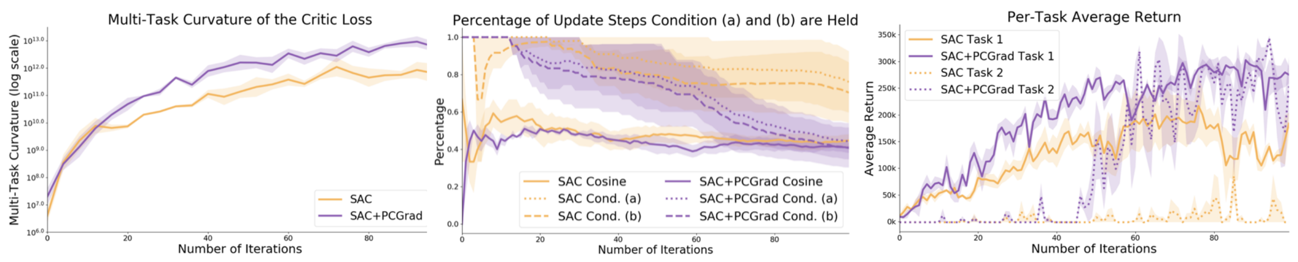

Figure 5: An empirical analysis of the theoretical conditions discussed in Theorem 2, showing the first 100 iterations of training on two RL tasks, reach and press button top. **Left**: The estimated value of the multi-task curvature. We observe high multi-task curvatures exist throughout training, providing evidence for condition (b) in Theorem 2. **Middle**: The solid lines show the percentage gradients with positive cosine similarity between two task gradients, while the dotted lines and dashed lines show the percentage of iterations in which condition (a) and the implication of condition (b) ($\xi(\mathbf{g}_1, \mathbf{g}_2) \leq 1$) in Theorem 2 are held respectively, among iterations when the cosine similarity is negative. **Right**: The average return of each task achieved by SAC and SAC combined with PCGrad. From the plots in the **Middle** and on the **Right**, we can tell that condition (a) holds most of the time for both Adam and Adam combined with PCGrad when they haven't solved Task 2 and as soon as Adam combined PCGrad starts to learn Task 2, the percentage of condition (a) held starts to decline. This observation suggests that condition (a) is a key factor for PCGrad excelling in multi-task learning.

to successfully accomplish the task. This suggests that PCGrad is able to ease the RL optimization problem also when the task distribution is continuous.

5.4 Empirical Analysis of the Tragic Triad

Finally, to answer question (1), we compare the performance of the network trained with standard multi-task learning conditioned on ground truth task labels and Adam optimizer and the network trained with multi-task learning and Adam optimizer with PCGrad-modified gradients on two tasks, reach and press button top, in the Meta-World (Yu et al., 2019) benchmark. As shown in the leftmost plot in Figure 5, we plot the multi-task curvature, which is computed as

$$\mathbf{H}(\mathcal{L};\theta^t,\theta^{t+1}) = 2\cdot\left[\mathcal{L}(\theta^{t+1}) - \mathcal{L}(\theta^t) - \nabla_{\theta^t}\mathcal{L}(\theta^t)^T(\theta^{t+1} - \theta^t)\right]$$

by Taylor's Theorem where $\mathcal{L}$ is the multi-task loss, and θ^t and θ^{t+1} are the parameters at iteration t and $t+1$. During the training process, the multi-task curvature stays positive and is increasing for both Adam and Adam combined PCGrad, suggesting that condition (b) in Theorem 2 that the multi-task curvature is lower bounded by some positive value is widely held empirically. To further analyze conditions in Theorem 2 empirically, we plot the percentage of condition (a) (i.e. conflicting gradients) and the implication of condition (b) ($\xi(\mathbf{g}_1, \mathbf{g}_2) \leq 1$) in Theorem 2 being held among the total number of iterations where the cosine similarity is negative in the plot in the middle of Figure 5. Along with the plot on the right in Figure 5, which presents the average return of the two tasks during training, we can see that while Adam and Adam with PCGrad haven't received reward signal from Task 2, condition (a) and the implication of condition (b) stay held and as soon as Adam with PCGrad begins to solve Task 2, the percentage of condition (a) and the implication of condition (b) being held start to decrease. Such a pattern suggests that conflicting gradients, high curvatures and dominating gradients indeed produce considerable challenges in optimization before multi-task learner gains any useful learning signal, which also implies that the tragic triad may indeed be the determining factor where PCGrad can lead to better performance gain over standard multi-task learning in practice.

6 Conclusion

In this work, we identified a set of conditions that underlies major challenges in multi-task optimization: conflicting gradients, high positive curvature, and large gradient differences. We proposed a simple algorithm (PCGrad) to mitigate these challenges via "gradient surgery." PCGrad provides a simple solution to mitigating gradient interference, which substantially improves optimization performance. We provide simple didactic examples and subsequently show significant improvement in optimization for a variety of multi-task supervised learning and reinforcement learning problems. We show that, when some optimization challenges of multi-task learning are alleviated by PCGrad, we can obtain the hypothesized benefits in efficiency and asymptotic performance of multi-task settings.

While we studied multi-task supervised learning and multi-task reinforcement learning in this work, we suspect the problem of conflicting gradients to be prevalent in a range of other settings and applications, such as meta-learning, continual learning, multi-goal imitation learning (Codevilla et al., 2018), and multi-task problems in natural language processing applications (McCann et al., 2018). Due to its simplicity and model-agnostic nature, we expect that applying PCGrad in these domains to be a promising avenue for future investigation. Further, the general idea of gradient surgery may be an important ingredient for alleviating a broader class of optimization challenges in deep learning, such as the challenges in the stability challenges in two-player games (Roth et al., 2017) and multi-agent optimizations (Nedic & Ozdaglar, 2009). We believe this work to be a step towards simple yet general techniques for addressing some of these challenges.

Acknowledgements

The authors would like to thank Eric Mitchell for technical guidance, and Aravind Rajeswaran and Deirdre Quillen for helpful discussions.

References

Badrinarayanan, V., Kendall, A., and Cipolla, R. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12), 2017.

Bakker, B. and Heskes, T. Task clustering and gating for bayesian multitask learning. *J. Mach. Learn. Res.*, 2003.

Bilen, H. and Vedaldi, A. Integrated perception with recurrent multi-task neural networks. In *Advances in Neural Information Processing Systems*, 2016.

Boyd, S. and Vandenberghe, L. *Convex optimization*. Cambridge university press, 2004.

Calamai, P. H. and Moré, J. J. Projected gradient methods for linearly constrained problems. *Mathematical programming*, 39(1), 1987.

Caruana, R. Multitask learning. *Machine Learning*, 1997.

Chaudhry, A., Ranzato, M., Rohrbach, M., and Elhoseiny, M. Efficient lifelong learning with a-gem. *arXiv:1812.00420*, 2018.

Chen, Z., Badrinarayanan, V., Lee, C.-Y., and Rabinovich, A. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. *arXiv:1711.02257*, 2017.

Chen, Z., Badrinarayanan, V., Lee, C., and Rabinovich, A. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International Conference on Machine Learning*, 2018.

Codevilla, F., Miiller, M., López, A., Koltun, V., and Dosovitskiy, A. End-to-end driving via conditional imitation learning. In *International Conference on Robotics and Automation (ICRA)*, 2018.

Collobert, R. and Weston, J. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *International Conference on Machine Learning*, 2008.

Czarnecki, W. M., Pascanu, R., Osindero, S., Jayakumar, S. M., Swirszcz, G., and Jaderberg, M. Distilling policy distillation. In *International Conference on Artificial Intelligence and Statistics, AISTATS*, 2019.

Devin, C., Gupta, A., Darrell, T., Abbeel, P., and Levine, S. Learning modular neural network policies for multi-task and multi-robot transfer. *CoRR*, abs/1609.07088, 2016.

Dong, D., Wu, H., He, W., Yu, D., and Wang, H. Multi-task learning for multiple language translation. In *Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2015.

Du, Y., Czarnecki, W. M., Jayakumar, S. M., Pascanu, R., and Lakshminarayanan, B. Adapting auxiliary losses using gradient similarity. *CoRR*, abs/1812.02224, 2018.

Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., Legg, S., and Kavukcuoglu, K. IMPALA: scalable distributed deep-rl with importance weighted actor-learner architectures. In *International Conference on Machine Learning*, 2018.

Fernando, C., Banarse, D., Blundell, C., Zwols, Y., Ha, D., Rusu, A. A., Pritzel, A., and Wierstra, D. Pathnet: Evolution channels gradient descent in super neural networks. *CoRR*, abs/1701.08734, 2017. URL http://arxiv.org/abs/1701.08734.

Ghosh, D., Singh, A., Rajeswaran, A., Kumar, V., and Levine, S. Divide-and-conquer reinforcement learning. *CoRR*, abs/1711.09874, 2017.

Goodfellow, I. J., Vinyals, O., and Saxe, A. M. Qualitatively characterizing neural network optimization problems. *arXiv:1412.6544*, 2014.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *International Conference on Machine Learning*, 2018.

Hausman, K., Springenberg, J. T., Wang, Z., Heess, N., and Riedmiller, M. Learning an embedding space for transferable robot skills. 2018.

Hessel, M., Soyer, H., Espeholt, L., Czarnecki, W., Schmitt, S., and van Hasselt, H. Multi-task deep reinforcement learning with popart. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 2019.

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv:1503.02531*, 2015.

Kendall, A., Gal, Y., and Cipolla, R. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Computer Vision and Pattern Recognition*, 2018a.

Kendall, A., Gal, Y., and Cipolla, R. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*, 2018b.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014.

Kokkinos, I. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *Computer Vision and Pattern Recognition*, 2017.

Levine, S., Finn, C., Darrell, T., and Abbeel, P. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1), 2016.

Liu, S., Johns, E., and Davison, A. J. End-to-end multi-task learning with attention. *CoRR*, abs/1803.10704, 2018.

Long, M. and Wang, J. Learning multiple tasks with deep relationship networks. *arXiv:1506.02117*, 2, 2015.

Lopez-Paz, D. and Ranzato, M. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, 2017.

Maninis, K., Radosavovic, I., and Kokkinos, I. Attentive single-tasking of multiple tasks. *CoRR*, abs/1904.08918, 2019.

McCann, B., Keskar, N. S., Xiong, C., and Socher, R. The natural language decathlon: Multitask learning as question answering. *arXiv:1806.08730*, 2018.

Misra, I., Shrivastava, A., Gupta, A., and Hebert, M. Cross-stitch networks for multi-task learning. In *Computer Vision and Pattern Recognition*, 2016a.

Misra, I., Shrivastava, A., Gupta, A., and Hebert, M. Cross-stitch networks for multi-task learning. In *Conference on Computer Vision and Pattern Recognition, CVPR*, 2016b.

Nedic, A. and Ozdaglar, A. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1), 2009.

Parisotto, E., Ba, J. L., and Salakhutdinov, R. Actor-mimic: Deep multitask and transfer reinforcement learning. *arXiv:1511.06342*, 2015.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8), 2019.

Riedmiller, M., Hafner, R., Lampe, T., Neunert, M., Degrave, J., Van de Wiele, T., Mnih, V., Heess, N., and Springenberg, J. T. Learning by playing-solving sparse reward tasks from scratch. *arXiv:1802.10567*, 2018.

Rosenbaum, C., Klinger, T., and Riemer, M. Routing networks: Adaptive selection of non-linear functions for multi-task learning. *International Conference on Learning Representations (ICLR)*, 2018.

Rosenbaum, C., Cases, I., Riemer, M., and Klinger, T. Routing networks and the challenges of modular and compositional computation. *arXiv:1904.12774*, 2019.

Roth, K., Lucchi, A., Nowozin, S., and Hofmann, T. Stabilizing training of generative adversarial networks through regularization. In *Advances in Neural Information Processing Systems*, 2017.

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

Rusu, A. A., Colmenarejo, S. G., Gülçehre, Ç., Desjardins, G., Kirkpatrick, J., Pascanu, R., Mnih, V., Kavukcuoglu, K., and Hadsell, R. Policy distillation. In *International Conference on Learning Representations, ICLR*, 2016a.

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., and Hadsell, R. Progressive neural networks. *arXiv:1606.04671*, 2016b.

Schaul, T., Borsa, D., Modayil, J., and Pascanu, R. Ray interference: a source of plateaus in deep reinforcement learning. *arXiv:1904.11455*, 2019.

Sener, O. and Koltun, V. Multi-task learning as multi-objective optimization. In *Advances in Neural Information Processing Systems*, 2018.

Teh, Y., Bapst, V., Czarnecki, W. M., Quan, J., Kirkpatrick, J., Hadsell, R., Heess, N., and Pascanu, R. Distral: Robust multitask reinforcement learning. In *Advances in Neural Information Processing Systems*, 2017.

Vandenhende, S., Brabandere, B. D., and Gool, L. V. Branched multi-task networks: Deciding what layers to share. *CoRR*, abs/1904.02920, 2019.

Wilson, A., Fern, A., Ray, S., and Tadepalli, P. Multi-task reinforcement learning: a hierarchical bayesian approach. In *International Conference on Machine Learning*, 2007.

Wulfmeier, M., Abdolmaleki, A., Hafner, R., Springenberg, J. T., Neunert, M., Hertweck, T., Lampe, T., Siegel, N., Heess, N., and Riedmiller, M. Regularized hierarchical policies for compositional transfer in robotics. *arXiv:1906.11228*, 2019.

Yang, Y. and Hospedales, T. M. Trace norm regularised deep multi-task learning. *arXiv:1606.04038*, 2016.

Yu, T., Quillen, D., He, Z., Julian, R., Hausman, K., Finn, C., and Levine, S. Meta-world: A benchmark and evaluation for multi-task and meta-reinforcement learning. *arXiv:1910.10897*, 2019.

Zamir, A. R., Sax, A., Shen, W., Guibas, L. J., Malik, J., and Savarese, S. Taskonomy: Disentangling task transfer learning. In *Computer Vision and Pattern Recognition*, 2018.

Zhang, Z., Luo, P., Loy, C. C., and Tang, X. Facial landmark detection by deep multi-task learning. In *European conference on computer vision*. Springer, 2014.

Appendix

A Proof of Theorem 1

Theorem 1. *Assume $\mathcal{L}_1$ and $\mathcal{L}_2$ are convex and differentiable. Suppose the gradient of $\mathcal{L}$ is Lipschitz continuous with constant $L > 0$. Then, the PCGrad update rule with step size $t \leq \frac{1}{L}$ will converge to either (1) a location in the optimization landscape where $\cos(\phi_{12}) = -1$ or (2) the optimal value $\mathcal{L}(\theta^*)$.*

Proof. We will use the shorthand $||\cdot||$ to denote the L_2-norm and $\nabla\mathcal{L} = \nabla_\theta\mathcal{L}$, where θ is the parameter vector. Following Definition 1 and 4, let $\mathbf{g_1} = \nabla\mathcal{L}_1$, $\mathbf{g_2} = \nabla\mathcal{L}_2$, $\mathbf{g} = \nabla\mathcal{L} = \mathbf{g_1} + \mathbf{g_2}$, and ϕ_{12} be the angle between $\mathbf{g_1}$ and $\mathbf{g_2}$.

At each PCGrad update, we have two cases: $cos(\phi_{12}) \geq 0$ or $\cos(\phi_{12}) < 0$.

If $\cos(\phi_{12}) \geq 0$, then we apply the standard gradient descent update using $t \leq \frac{1}{L}$, which leads to a strict decrease in the objective function value $\mathcal{L}(\theta)$ (since it is also convex) unless $\nabla\mathcal{L}(\theta) = 0$, which occurs only when $\theta = \theta^*$ (Boyd & Vandenberghe, 2004).

In the case that $\cos(\phi_{12}) < 0$, we proceed as follows:

Our assumption that $\nabla\mathcal{L}$ is Lipschitz continuous with constant L implies that $\nabla^2\mathcal{L}(\theta) - LI$ is a negative semi-definite matrix. Using this fact, we can perform a quadratic expansion of $\mathcal{L}$ around $\mathcal{L}(\theta)$ and obtain the following inequality:

$$\begin{aligned}\mathcal{L}(\theta^+) &\leq \mathcal{L}(\theta) + \nabla\mathcal{L}(\theta)^T(\theta^+ - \theta) + \frac{1}{2}\nabla^2\mathcal{L}(\theta)||\theta^+ - \theta||^2 \\ &\leq \mathcal{L}(\theta) + \nabla\mathcal{L}(\theta)^T(\theta^+ - \theta) + \frac{1}{2}L||\theta^+ - \theta||^2\end{aligned}$$

Now, we can plug in the PCGrad update by letting $\theta^+ = \theta - t\cdot(\mathbf{g} - \frac{\mathbf{g_1}\cdot\mathbf{g_2}}{||\mathbf{g_1}||^2}\mathbf{g_1} - \frac{\mathbf{g_1}\cdot\mathbf{g_2}}{||\mathbf{g_2}||^2}\mathbf{g_2})$. We then get:

$$\mathcal{L}(\theta^+) \leq \mathcal{L}(\theta) + t\cdot\mathbf{g}^T(-\mathbf{g} + \frac{\mathbf{g_1}\cdot\mathbf{g_2}}{||\mathbf{g_1}||^2}\mathbf{g_1} + \frac{\mathbf{g_1}\cdot\mathbf{g_2}}{||\mathbf{g_2}||^2}\mathbf{g_2}) + \frac{1}{2}Lt^2||\mathbf{g} - \frac{\mathbf{g_1}\cdot\mathbf{g_2}}{||\mathbf{g_1}||^2}\mathbf{g_1} - \frac{\mathbf{g_1}\cdot\mathbf{g_2}}{||\mathbf{g_2}||^2}\mathbf{g_2}||^2$$

(Expanding, using the identity $\mathbf{g} = \mathbf{g_1} + \mathbf{g_2}$)

$$\begin{aligned}&= \mathcal{L}(\theta) + t\left(-||\mathbf{g_1}||^2 - ||\mathbf{g_2}||^2 + \frac{(\mathbf{g_1}\cdot\mathbf{g_2})^2}{||\mathbf{g_1}||^2} + \frac{(\mathbf{g_1}\cdot\mathbf{g_2})^2}{||\mathbf{g_2}||^2}\right) + \frac{1}{2}Lt^2||\mathbf{g_1} + \mathbf{g_2} \\ &- \frac{\mathbf{g_1}\cdot\mathbf{g_2}}{||\mathbf{g_1}||^2}\mathbf{g_1} - \frac{\mathbf{g_1}\cdot\mathbf{g_2}}{||\mathbf{g_2}||^2}\mathbf{g_2}||^2\end{aligned}$$

(Expanding further and re-arranging terms)

$$= \mathcal{L}(\theta) - (t - \frac{1}{2}Lt^2)(||\mathbf{g_1}||^2 + ||\mathbf{g_2}||^2 - \frac{(\mathbf{g_1}\cdot\mathbf{g_2})^2}{||\mathbf{g_1}||^2} - \frac{(\mathbf{g_1}\cdot\mathbf{g_2})^2}{||\mathbf{g_2}||^2}) - Lt^2(\mathbf{g_1}\cdot\mathbf{g_2} - \frac{(\mathbf{g_1}\cdot\mathbf{g_2})^2}{||\mathbf{g_1}||^2||\mathbf{g_2}||^2}\mathbf{g_1}\cdot\mathbf{g_2})$$

(Using the identity $\cos(\phi_{12}) = \frac{\mathbf{g_1}\cdot\mathbf{g_2}}{||\mathbf{g_1}||||\mathbf{g_2}||}$)

$$= \mathcal{L}(\theta) - (t - \frac{1}{2}Lt^2)[(1-\cos^2(\phi_{12}))||\mathbf{g_1}||^2 + (1-\cos^2(\phi_{12}))||\mathbf{g_2}||^2] - Lt^2(1-\cos^2(\phi_{12}))||\mathbf{g_1}||||\mathbf{g_2}||\cos(\phi_{12}) \tag{1}$$

(Note that $\cos(\phi_{12}) < 0$ so the final term is non-negative)

Using $t \leq \frac{1}{L}$, we know that $-(1 - \frac{1}{2}Lt) = \frac{1}{2}Lt - 1 \leq \frac{1}{2}L(1/L) - 1 = \frac{-1}{2}$ and $Lt^2 \leq t$.

Plugging this into the last expression above, we can conclude the following:

$$\begin{aligned}
\mathcal{L}(\theta^+) &\leq \mathcal{L}(\theta) - \frac{1}{2}t[(1-\cos^2(\phi_{12}))||\mathbf{g_1}||^2 + (1-\cos^2(\phi_{12}))||\mathbf{g_2}||^2] - t(1-\cos^2(\phi_{12}))||\mathbf{g_1}||||\mathbf{g_2}||\cos(\phi_{12}) \\
&= \mathcal{L}(\theta) - \frac{1}{2}t(1-\cos^2(\phi_{12}))[||\mathbf{g_1}||^2 + 2||\mathbf{g_1}||||\mathbf{g_2}||\cos(\phi_{12}) + ||\mathbf{g_2}||^2] \\
&= \mathcal{L}(\theta) - \frac{1}{2}t(1-\cos^2(\phi_{12}))[||\mathbf{g_1}||^2 + 2\mathbf{g_1}\cdot\mathbf{g_2} + ||\mathbf{g_2}||^2] \\
&= \mathcal{L}(\theta) - \frac{1}{2}t(1-\cos^2(\phi_{12}))||\mathbf{g_1}+\mathbf{g_2}||^2 \\
&= \mathcal{L}(\theta) - \frac{1}{2}t(1-\cos^2(\phi_{12}))\|\mathbf{g}\|^2
\end{aligned}$$

If $\cos(\phi_{12}) > -1$, then $\frac{1}{2}t(1-\cos^2(\phi_{12}))\|\mathbf{g}\|^2$ will always be positive unless $\mathbf{g} = 0$. This inequality implies that the objective function value strictly decreases with each iteration where $\cos(\phi_{12}) > -1$.

Hence repeatedly applying PCGrad process can either reach the optimal value $\mathcal{L}(\theta) = \mathcal{L}(\theta^*)$ or $\cos(\phi_{12}) = -1$, in which case $\frac{1}{2}t(1-\cos^2(\phi_{12}))\|\mathbf{g}\|^2 = 0$. Note that this result only holds when we choose t to be small enough, i.e. $t \leq \frac{1}{L}$.

□

B Proof of Theorem 2

Theorem 2. *Suppose $\mathcal{L}$ is differentiable and the gradient of $\mathcal{L}$ is Lipschitz continuous with constant $L > 0$. Let θ^{MT} and θ^{PCGrad} be the parameters after applying one update to θ with $\mathbf{g}$ and PCGrad-modified gradient $\mathbf{g}^{PC}$ respectively, with step size $t > 0$. Moreover, assume $\mathbf{H}(\mathcal{L};\theta,\theta^{MT}) \geq \ell\|\mathbf{g}\|_2^2$ for some constant $\ell \leq L$, i.e. the multi-task curvature is lower-bounded. Then $\mathcal{L}(\theta^{PCGrad}) \leq \mathcal{L}(\theta^{MT})$ if*

(a) $\cos\phi_{12} \leq -\Phi(\mathbf{g_1},\mathbf{g_2})$,

(b) $\ell \geq \xi(\mathbf{g_1},\mathbf{g_2})L$, *and*

(c) $t \geq \frac{2}{\ell-\xi(\mathbf{g_1},\mathbf{g_2})L}$.

Proof. Note that $\theta^{\text{MT}} = \theta - t\cdot\mathbf{g}$ and $\theta^{\text{PCGrad}} = \theta - t(\mathbf{g} - \frac{\mathbf{g_1}\cdot\mathbf{g_2}}{||\mathbf{g_1}||^2}\mathbf{g_1} - \frac{\mathbf{g_1}\cdot\mathbf{g_2}}{||\mathbf{g_2}||^2}\mathbf{g_2})$. Based on the condition that $\mathbf{H}(\mathcal{L};\theta,\theta^{\text{MT}}) \geq \ell\|\mathbf{g}\|_2^2$, we first apply Taylor's Theorem to $\mathcal{L}(\theta^{\text{MT}})$ and obtain the following result:

$$\begin{aligned}
\mathcal{L}(\theta^{\text{MT}}) &= \mathcal{L}(\theta) + \mathbf{g}^T(-t\mathbf{g}) + \int_0^1 (-t\mathbf{g})^T \frac{\nabla^2\mathcal{L}(\theta + a\cdot(-t\mathbf{g}))}{2}(-t\mathbf{g})da \\
&\geq \mathcal{L}(\theta) + \mathbf{g}^T(-t\mathbf{g}) + t^2\cdot\frac{1}{2}\ell\cdot\|\mathbf{g}\|_2^2 \\
&= \mathcal{L}(\theta) - t\|\mathbf{g}\|_2^2 + \frac{1}{2}\ell t^2\|\mathbf{g}\|_2^2 \\
&= \mathcal{L}(\theta) + (\frac{1}{2}\ell t^2 - t)\|\mathbf{g}\|_2^2
\end{aligned} \tag{2}$$

where the first inequality follows from Definition 3 and the assumption $\mathbf{H}(\mathcal{L};\theta,\theta^{\text{MT}}) \geq \ell\|\mathbf{g}\|_2^2$. From equation 1, we have the simplified upper bound for $\mathcal{L}(\theta^{\text{PCGrad}})$:

$$\mathcal{L}(\theta^{\text{PCGrad}}) \leq \mathcal{L}(\theta) - (1-\cos^2\phi_{12})[(t - \frac{1}{2}Lt^2)\cdot(\|\mathbf{g_1}\|_2^2 + \|\mathbf{g_1}\|_2^2) + Lt^2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12}] \tag{3}$$

Apply Equation 2 and Equation 3 and we have the following inequality:

$$\begin{aligned}
&\mathcal{L}(\theta^{\text{MT}}) - \mathcal{L}(\theta^{\text{PCGrad}}) \geq \mathcal{L}(\theta) + (\frac{1}{2}\ell t^2 - t)\|\mathbf{g}\|_2^2 - \mathcal{L}(\theta) + (1-\cos^2\phi_{12})[(t - \frac{1}{2}Lt^2)(\|\mathbf{g_1}\|_2^2 + \|\mathbf{g_2}\|_2^2) \\
&+ Lt^2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12}] \\
&= (\frac{1}{2}\ell t^2 - t)\|\mathbf{g_1}+\mathbf{g_2}\|_2^2 + (1-\cos^2\phi_{12})\left[(t - \frac{1}{2}Lt^2)\cdot(\|\mathbf{g_1}\|_2^2 + \|\mathbf{g_2}\|_2^2) + Lt^2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12}\right] \\
&= \left(\frac{1}{2}\|\mathbf{g_1}+\mathbf{g_2}\|_2^2\ell - \frac{1-\cos^2\phi_{12}}{2}(\|\mathbf{g_1}\|_2^2 + \|\mathbf{g_2}\|_2^2 - 2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12})L\right)t^2 \\
&- \left((\|\mathbf{g_1}\|_2^2 + \|\mathbf{g_2}\|_2^2)\cos^2\phi_{12} + 2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12}\right)t \\
&= \left(\frac{1}{2}\|\mathbf{g_1}+\mathbf{g_2}\|_2^2\ell - \frac{1-\cos^2\phi_{12}}{2}\|\mathbf{g_1}-\mathbf{g_2}\|_2^2 L\right)t^2 - \left((\|\mathbf{g_1}\|_2^2 + \|\mathbf{g_2}\|_2^2)\cos^2\phi_{12} + 2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12}\right)t \\
&= t\cdot\left[\left(\frac{1}{2}\|\mathbf{g_1}+\mathbf{g_2}\|_2^2\ell - \frac{1-\cos^2\phi_{12}}{2}(\|\mathbf{g_1}-\mathbf{g_2}\|_2^2)L\right)t - \left((\|\mathbf{g_1}\|_2^2 + \|\mathbf{g_2}\|_2^2)\cos^2\phi_{12} + 2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12}\right)\right]
\end{aligned} \tag{4}$$

Since $\cos\phi_{12} \leq -\Phi(\mathbf{g_1},\mathbf{g_2}) = -\frac{2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2}{\|\mathbf{g_1}\|_2^2+\|\mathbf{g_2}\|_2^2}$ and $\ell \geq \xi(\mathbf{g_1},\mathbf{g_2}) = \frac{(1-\cos^2\phi_{12})(\|\mathbf{g_1}-\mathbf{g_2}\|_2^2)}{\|\mathbf{g_1}+\mathbf{g_2}\|_2^2}L$, we have

$$\frac{1}{2}\|\mathbf{g_1}+\mathbf{g_2}\|_2^2\ell - \frac{1-\cos^2\phi_{12}}{2}\|\mathbf{g_1}-\mathbf{g_2}\|_2^2 L \geq 0$$

and

$$(\|\mathbf{g_1}\|_2^2 + \|\mathbf{g_2}\|_2^2)\cos^2\phi_{12} + 2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12} \geq 0.$$

By the condition that $t \geq \frac{2}{\ell - \xi(\mathbf{g_1},\mathbf{g_2})L} = \frac{2}{\ell - \frac{(1-\cos^2\phi_{12})\|\mathbf{g_1}-\mathbf{g_2}\|_2^2}{\|\mathbf{g_1}+\mathbf{g_2}\|_2^2}L}$ and monotonicity of linear functions, we have the following:

$$\begin{aligned}
&\mathcal{L}(\theta^{\text{MT}}) - \mathcal{L}(\theta^{\text{PCGrad}}) \geq [\left(\frac{1}{2}\|\mathbf{g_1}+\mathbf{g_2}\|_2^2\ell - \frac{1-\cos^2\phi_{12}}{2}\cdot\|\mathbf{g_1}-\mathbf{g_2}\|_2^2 L\right)\cdot\frac{2}{\ell - \frac{(1-\cos^2\phi_{12})\|\mathbf{g_1}-\mathbf{g_2}\|_2^2}{\|\mathbf{g_1}+\mathbf{g_2}\|_2^2}L} \\
&- \left((\|\mathbf{g_1}\|_2^2 + \|\mathbf{g_2}\|_2^2)\cos^2\phi_{12} + 2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12}\right)]\cdot t \\
&= [\|\mathbf{g_1}+\mathbf{g_2}\|_2^2\cdot\left(\ell - \frac{(1-\cos^2\phi_{12})\cdot\|\mathbf{g_1}-\mathbf{g_2}\|_2^2}{\|\mathbf{g_1}+\mathbf{g_2}\|_2^2})L\right)\cdot\frac{1}{\ell - \frac{(1-\cos^2\phi_{12})\|\mathbf{g_1}-\mathbf{g_2}\|_2^2}{\|\mathbf{g_1}+\mathbf{g_2}\|_2^2}L} \\
&- \left((\|\mathbf{g_1}\|_2^2 + \|\mathbf{g_2}\|_2^2)\cos^2\phi_{12} + 2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12}\right)]\cdot t \\
&= \left[\|\mathbf{g_1}+\mathbf{g_2}\|_2^2 - \left((\|\mathbf{g_1}\|_2^2 + \|\mathbf{g_2}\|_2^2)\cos^2\phi_{12} + 2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12}\right)\right]\cdot t \\
&= \left[\|\mathbf{g_1}\|_2^2 + \|\mathbf{g_2}\|_2^2 + 2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12} - \left((\|\mathbf{g_1}\|_2^2 + \|\mathbf{g_2}\|_2^2)\cos^2\phi_{12} + 2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12}\right)\right]\cdot t \\
&= (1-\cos^2\phi_{12})(\|\mathbf{g_1}\|_2^2 + \|\mathbf{g_2}\|_2^2)\cdot t \\
&\geq 0
\end{aligned}$$

□

C PCGrad: Sufficient and Necessary Conditions for Loss Improvement

Beyond the sufficient conditions shown in Theorem 2, we also present the sufficient and necessary conditions under which PCGrad achieves lower loss after one gradient update in Theorem 3 in the two-task setting.

Theorem 3. *Suppose $\mathcal{L}$ is differentiable and the gradient of $\mathcal{L}$ is Lipschitz continuous with constant $L > 0$. Let θ^{MT} and θ^{PCGrad} be the parameters after applying one update to θ with $\mathbf{g}$ and PCGrad-modified gradient $\mathbf{g}^{PC}$ respectively, with step size $t > 0$. Moreover, assume $\mathbf{H}(\mathcal{L};\theta,\theta^{MT}) \geq \ell\|\mathbf{g}\|_2^2$ for some constant $\ell \leq L$, i.e. the multi-task curvature is lower-bounded. Then $\mathcal{L}(\theta^{PCGrad}) \leq \mathcal{L}(\theta^{MT})$ if and only if*

- $-\Phi(\mathbf{g_1},\mathbf{g_2}) \leq \cos\phi_{12} < 0$

- $\ell \leq \xi(\mathbf{g_1}, \mathbf{g_2})L$
- $0 < t \leq \frac{(\|\mathbf{g_1}\|_2^2+\|\mathbf{g_2}\|_2^2)\cos^2\phi_{12}+2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12}}{\frac{1}{2}\|\mathbf{g_1}+\mathbf{g_2}\|_2^2\ell-\frac{1-\cos^2\phi_{12}}{2}(\|\mathbf{g_1}-\mathbf{g_2}\|_2^2)L}$

or

- $\cos\phi_{12} \leq -\Phi(\mathbf{g_1}, \mathbf{g_2})$
- $\ell \geq \xi(\mathbf{g_1}, \mathbf{g_2})L$
- $t \geq \frac{(\|\mathbf{g_1}\|_2^2+\|\mathbf{g_2}\|_2^2)\cos^2\phi_{12}+2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12}}{\frac{1}{2}\|\mathbf{g_1}+\mathbf{g_2}\|_2^2\ell-\frac{1-\cos^2\phi_{12}}{2}(\|\mathbf{g_1}-\mathbf{g_2}\|_2^2)L}$.

Proof. To show the necessary conditions, from Equation 4, all we need is

$$t\cdot[(\frac{1}{2}\|\mathbf{g_1}+\mathbf{g_2}\|_2^2\ell - \frac{1-\cos^2\phi_{12}}{2}(\|\mathbf{g_1}-\mathbf{g_2}\|_2^2)L)t - \big((\|\mathbf{g_1}\|_2^2+\|\mathbf{g_2}\|_2^2)\cos^2\phi_{12}+2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12})] \geq 0 \quad (5)$$

Since $t \geq 0$, it reduces to show

$$(\frac{1}{2}\|\mathbf{g_1}+\mathbf{g_2}\|_2^2\ell - \frac{1-\cos^2\phi_{12}}{2}(\|\mathbf{g_1}-\mathbf{g_2}\|_2^2)L)t - \big((\|\mathbf{g_1}\|_2^2+\|\mathbf{g_2}\|_2^2)\cos^2\phi_{12}+2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12}\big) \geq 0 \quad (6)$$

For Equation 6 to hold while ensuring that $t \geq 0$, there are two cases:

- $\frac{1}{2}\|\mathbf{g_1}+\mathbf{g_2}\|_2^2\ell - (1-\cos^2\phi_{12})(\|\mathbf{g_1}\|_2^2+\|\mathbf{g_2}\|_2^2)L \geq 0$,
  $(\|\mathbf{g_1}\|_2^2+\|\mathbf{g_2}\|_2^2)\cos^2\phi_{12}+2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12} \geq 0$,
  $t \geq \frac{(\|\mathbf{g_1}\|_2^2+\|\mathbf{g_2}\|_2^2)\cos^2\phi_{12}+2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12}}{\frac{1}{2}\|\mathbf{g_1}+\mathbf{g_2}\|_2^2\ell-\frac{1-\cos^2\phi_{12}}{2}(\|\mathbf{g_1}-\mathbf{g_2}\|_2^2)L}$
- $\frac{1}{2}\|\mathbf{g_1}+\mathbf{g_2}\|_2^2\ell - (1-\cos^2\phi_{12})(\|\mathbf{g_1}\|_2^2+\|\mathbf{g_2}\|_2^2)L \leq 0$,
  $(\|\mathbf{g_1}\|_2^2+\|\mathbf{g_2}\|_2^2)\cos^2\phi_{12}+2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12} \leq 0$,
  $t \geq \frac{(\|\mathbf{g_1}\|_2^2+\|\mathbf{g_2}\|_2^2)\cos^2\phi_{12}+2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12}}{\frac{1}{2}\|\mathbf{g_1}+\mathbf{g_2}\|_2^2\ell-\frac{1-\cos^2\phi_{12}}{2}(\|\mathbf{g_1}-\mathbf{g_2}\|_2^2)L}$

, which can be simplified to

- $\cos\phi_{12} \leq -\frac{2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2}{\|\mathbf{g_1}\|_2^2+\|\mathbf{g_2}\|_2^2} = -\Phi(\mathbf{g_1}, \mathbf{g_2})$,
  $\ell \geq \frac{(1-\cos^2\phi_{12})(\|\mathbf{g_1}\|_2^2+\|\mathbf{g_2}\|_2^2)}{\|\mathbf{g_1}+\mathbf{g_2}\|_2^2}L = \xi(\mathbf{g_1}, \mathbf{g_2})$,
  $t \geq \frac{(\|\mathbf{g_1}\|_2^2+\|\mathbf{g_2}\|_2^2)\cos^2\phi_{12}+2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12}}{\frac{1}{2}\|\mathbf{g_1}+\mathbf{g_2}\|_2^2\ell-\frac{1-\cos^2\phi_{12}}{2}(\|\mathbf{g_1}-\mathbf{g_2}\|_2^2)L}$
- $-\frac{2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2}{\|\mathbf{g_1}\|_2^2+\|\mathbf{g_2}\|_2^2} = -\Phi(\mathbf{g_1}, \mathbf{g_2}) \leq \cos\phi_{12} < 0$,
  $\ell \leq \frac{(1-\cos^2\phi_{12})(\|\mathbf{g_1}\|_2^2+\|\mathbf{g_2}\|_2^2)}{\|\mathbf{g_1}+\mathbf{g_2}\|_2^2}L = \xi(\mathbf{g_1}, \mathbf{g_2})$,
  $0 < t \leq \frac{(\|\mathbf{g_1}\|_2^2+\|\mathbf{g_2}\|_2^2)\cos^2\phi_{12}+2\|\mathbf{g_1}\|_2\|\mathbf{g_2}\|_2\cos\phi_{12}}{\frac{1}{2}\|\mathbf{g_1}+\mathbf{g_2}\|_2^2\ell-\frac{1-\cos^2\phi_{12}}{2}(\|\mathbf{g_1}-\mathbf{g_2}\|_2^2)L}$.

The sufficient conditions hold as we can plug the conditions to RHS of Equation 6 and achieve non-negative result. □

D Practical Details of PCGrad on Multi-Task and Goal-Conditioned Reinforcement Learning

When applying PCGrad to goal-conditioned RL, we represent $p(\mathcal{T})$ as a distribution of goals and let z_i be the encoding of a goal. Similar to the multi-task supervised learning setting discussed in Section 3, PCGrad may be combined with various architectures designed for multi-task and goal-conditioned RL (Fernando et al., 2017; Devin et al., 2016), where PCGrad operates on the gradients of shared parameters, leaving task-specific parameters untouched.

	left digit	right digit
Sener & Koltun (2018)	96.45	94.95
PCGrad (ours)	**96.58**	**95.50**

Table 3: MultiMNIST results. PCGrad achieves improvements over the approach by Sener & Koltun (2018) in both left and right digit classfication accuracy.

As discussed in Section 3, we use SAC for our experiments. In SAC, we employ a Q-learning style gradient to compute the gradient of the Q-function network, $Q_\phi(s, a, z_i)$, often known as the critic, and a reparameterization-style gradient to compute the gradient of the policy network $\pi_\theta(a|s, z_i)$, often known as the actor. For sampling, we instantiate a set of replay buffers $\{\mathcal{D}_i\}_{\mathcal{T}_i\sim p(\mathcal{T})}$. Training and data collection are alternated throughout training. During a data collection step, we run the policy π_θ on all the tasks $\mathcal{T}_i \sim p(\mathcal{T})$ to collect an equal number of paths for each task and store the paths of each task $\mathcal{T}_i$ into the corresponding replay buffer $\mathcal{D}_i$. At each training step, we sample an equal amount of data from each replay buffer $\mathcal{D}_i$ to form a stratified batch. For each task $\mathcal{T}_i \sim p(\mathcal{T})$, the parameters of the critic θ are optimized to minimize the soft Bellman residual:

$$J_Q^{(i)}(\phi) = \mathbb{E}_{(s_t,a_t,z_i)\sim\mathcal{D}_i}\left[Q_\phi(s_t, a_t, z_i) - (r(s_t, a_t, z_i) + \gamma V_{\bar{\phi}}(s_{t+1}, z_i))\right], \tag{7}$$

$$V_{\bar{\phi}}(s_{t+1}, z_i) = \mathbb{E}_{a_{t+1}\sim\pi_\theta}\left[Q_{\bar{\phi}}(s_{t+1}, a_{t+1}, z_i) - \alpha \log \pi_\theta(a_{t+1}|s_{t+1}, z_i)\right], \tag{8}$$

where γ is the discount factor, $\bar{\phi}$ are the delayed parameters, and α is a learnable temperature that automatically adjusts the weight of the entropy term. For each task $\mathcal{T}_i \sim p(\mathcal{T})$, the parameters of the policy π_θ are trained to minimize the following objective

$$J_\pi^{(i)}(\theta) = \mathbb{E}_{s_t\sim\mathcal{D}_i}\left[\mathbb{E}_{a_t\sim\pi_\theta(a_t|s_t,z_i))}\left[\alpha \log \pi_\theta(a_t|s_t, z_i) - Q_\phi(s_t, a_t, z_i)\right]\right]. \tag{9}$$

We compute $\nabla_\phi J_Q^{(i)}(\phi)$ and $\nabla_\theta J_\pi^{(i)}(\theta)$ for all $\mathcal{T}_i \sim p(\mathcal{T})$ and apply PCGrad to both following Algorithm 1.

For details of the temperature adjustment strategy, the motivation is that if we use a single learnable temperature for adjusting entropy of the multi-task policy $\pi_\theta(a|s, z_i)$, SAC may stop exploring once all easier tasks are solved, leading to poor performance on tasks that are harder or require more exploration. To address this issue, we propose to learn the temperature on a per-task basis as mentioned in Section 3, i.e. using a parametrized model to represent $\alpha_\psi(z_i)$. This allows the method to control the entropy of $\pi_\theta(a|s, z_i)$ per-task. We optimize the parameters of $\alpha_\psi(z_i)$ using the same constrained optimization framework as in (Haarnoja et al., 2018).

E 2D Optimization Landscape Details

To produce the 2D optimization visualizations in Figure 1, we used a parameter vector $\theta = [\theta_1, \theta_2] \in \mathbb{R}^2$ and the following task loss functions:

$$\mathcal{L}_1(\theta) = 20\log(\max(|.5\theta_1 + \tanh(\theta_2)|, 0.000005))$$
$$\mathcal{L}_2(\theta) = 25\log(\max(|.5\theta_1 - \tanh(\theta_2) + 2|, 0.000005))$$

The multi-task objective is $\mathcal{L}(\theta) = \mathcal{L}_1(\theta) + \mathcal{L}_2(\theta)$. We initialized $\theta = [0.5, -3]$ and performed 500,000 gradient updates to minimize $\mathcal{L}$ using the Adam optimizer with learning rate 0.001. We compared using Adam for each update to using Adam in conjunction with the PCGrad method presented in Section 2.3.

F Experimental Results on MultiMNIST

Following the same set-up of Sener & Koltun (2018), for each image, we sample a different one uniformly at random. Then we put one of the image on the top left and the other one on the bottom right. The two tasks in the multi-task learning problem are to classify the digits on the top left (task-L) and bottom right (task-R) respectively. We construct such 60K examples. We combine PCGrad with the same backbone architecture used in (Sener & Koltun, 2018) and compare its performance to Sener & Koltun (2018) by running the open-sourced code provided in (Sener & Koltun, 2018). As shown in Table 3, our method results 0.13% and 0.55% improvement over (Sener & Koltun, 2018) in left and right digit accuracy respectively.

G Comparison to GradNorm

We compare PCGrad to a prior method GradNorm (Chen et al., 2018), which scales the magnitude of gradients of all the tasks. As shown in Figure 6, PCGrad significantly outperforms GradNorm. We also notice that the variant of PCGrad where only the gradient magnitudes change (see the rightmost plot in Figure 3) achieves comparable performance to GradNorm, which suggests that it is important to modify both the gradient directions and magnitudes to eliminate interference and achieve good multi-task learning results.

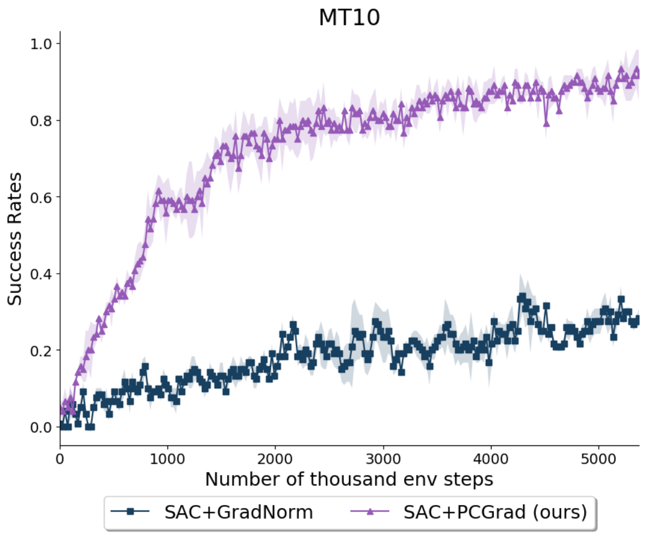

Figure 6: Comparison between PCGrad and GradNorm (Chen et al., 2018). PCGrad outperforms GradNorm with a large margin, indicating the importance of modifying both the gradient directions and magnitudes in multi-task learning.

H Ablation study on the task order

As stated on line 4 in Algorithm 1, we sample the tasks from the batch and randomly shuffle the order of the tasks before performing the update steps in PCGrad. With random shuffling, we make PCGrad symmetric w.r.t. the task order in expectation. In Figure 7, we observe that PCGrad with a random task order achieves better performance between PCGrad with a fixed task order in the setting of MT50 where the number of tasks is large and the conflicting gradient phenomenon is much more likely to happen.

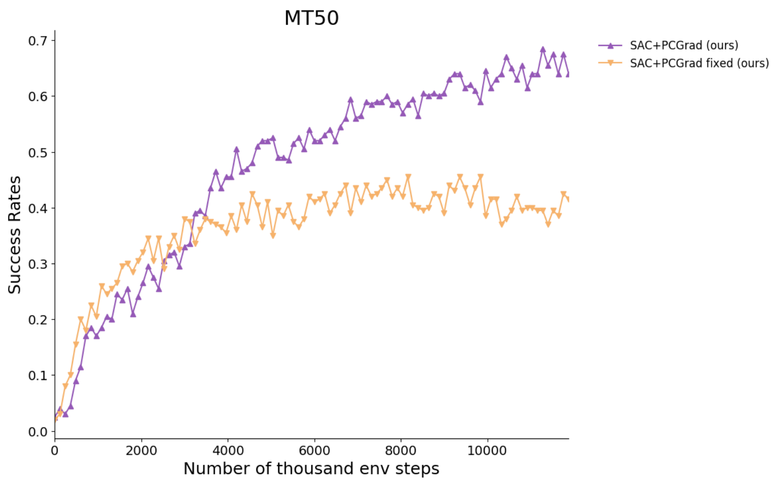

Figure 7: Ablation study on using a fixed task order during PCGrad. PCGrad with a random task order does significantly better PCGrad with a fixed task order in MT50 benchmark.

I Experiment Details

I.1 Detailed Experiment Set-up

For our CIFAR-100 multi-task experiment, we adopt the architecture used in (Rosenbaum et al., 2019), which is a convolutional neural network that consists of 3 convolutional layers with 160 3×3 filters each layer and 2 fully connected layers with 320 hidden units. As for experiments on the NYUv2 dataset, we follow (Liu et al., 2018) to use SegNet (Badrinarayanan et al., 2017) as the backbone architecture.

Our reinforcement learning experiments all use the SAC (Haarnoja et al., 2018) algorithm as the base algorithm, where the actor and the critic are represented as 6-layer fully-connected feedforward neural networks for all methods. The numbers of hidden units of each layer of the neural networks are 160, 300 and 200 for MT10, MT50 and goal-conditioned RL respectively.

We use five algorithms as baselines in the CIFAR-100 multi-task experiment: **task specific-1-fc** (Rosenbaum et al., 2018): a convolutional neural network shared across tasks except that each task has a separate last fully-connected layer, **task specific-1-fc** (Rosenbaum et al., 2018) : all the convolutional layers shared across tasks with separate fully-connected layers for each task, **cross stitch-all-fc** (Misra et al., 2016b): one convolutional neural network per task along with cross-stitch units to share features across tasks, **routing-all-fc + WPL** (Rosenbaum et al., 2019): a network that employs a trainable router trained with multi-agent RL algorithm (WPL) to select trainable functions for each task, **independent**: training separate neural networks for each task.

For comparisons on the NYUv2 dataset, we consider 5 baselines: **Single Task, One Task**: the vanilla SegNet used for single-task training, **Single Task, STAN** (Liu et al., 2018): the single-task version of MTAN as mentioned below, **Multi-Task, Split, Wide / Deep** (Liu et al., 2018): the standard SegNet shared for all three tasks except that each task has a separate last layer for final task-specific prediction with two variants **Wide** and **Deep** specified in (Liu et al., 2018), **Multi-Task Dense**: a shared network followed by separate task-specific networks, **Multi-Task Cross-Stitch** (Misra et al., 2016b): similar to the baseline used in CIFAR-100 experiment but with SegNet as the backbone, **MTAN** (Liu et al., 2018): a shared network with a soft-attention module for each task.

In the case of multi-task reinforcement learning, we evaluate our algorithm on the recently proposed Meta-World benchmark (Yu et al., 2019). This benchmark includes a variety of simulated robotic manipulation tasks contained in a shared, table-top environment with a simulated Sawyer arm (visualized as the ”Push” environment in Fig. 4). In particular, we use the multi-task benchmarks MT10 and MT50, which consists of the 10 tasks and 50 tasks respectively depicted in Fig. 4 that require diverse strategies to solve them, which makes them difficult to optimize jointly with a single policy. Note that MT10 is a subset of MT50.

On the multi-task and goal-conditioned RL domain, we apply PCGrad to the vanilla SAC algorithm with task encoding as part of the input to the actor and the critic as described in Section 3 and compare our method to the vanilla **SAC** without PCGrad and training actors and critics for each task individually (**Independent**).

I.2 Goal-conditioned Experiment Details

We use the pushing environment from the Meta-World benchmark (Yu et al., 2019) as shown in Figure 4. In this environment, the table spans from $[-0.4, 0.2]$ to $[0.4, 1.0]$ in the 2D space. To construct the goals, we sample the intial positions of the puck from the range $[-0.2, 0.6]$ to $[0.2, 0.7]$ on the table and the goal positions from the range $[-0.2, 0.85]$ to $[0.2, 0.95]$ on the table. The goal is represented as a concatenation of the initial puck position and the goal position. Since in the goal-conditioned setting, the task distribution is continuous, we sample a minibatch of 9 goals and 128 samples per goal at each training iteration and also sample 600 samples per goal in the minibatch at each data collection step.

I.3 Full NYUv2 Results

We provide the full comparison on the NYUv2 dataset in Table 4.

				Segmentation		Depth		Surface Normal				
Type	#P.	Architecture	Weighting	(Higher Better)		(Lower Better)		Angle Distance (Lower Better)		Within t° (Higher Better)		
				mIoU	Pix Acc	Abs Err	Rel Err	Mean	Median	11.25	22.5	30
Single Task	3	One Task	n.a.	15.10	51.54	0.7508	0.3266	31.76	25.51	22.12	45.33	57.13
	4.56	STAN†	n.a.	15.73	52.89	0.6935	0.2891	32.09	26.32	21.49	44.38	56.51
Multi Task	1.75	Split, Wide	Equal Weights	15.89	51.19	0.6494	0.2804	33.69	28.91	18.54	39.91	52.02
			Uncert. Weights*	15.86	51.12	**0.6040**	0.2570	**32.33**	**26.62**	**21.68**	**43.59**	**55.36**
			DWA†, $T = 2$	**16.92**	**53.72**	0.6125	**0.2546**	32.34	27.10	20.69	42.73	54.74
	2	Split, Deep	Equal Weights	13.03	41.47	0.7836	0.3326	38.28	36.55	9.50	27.11	39.63
			Uncert. Weights*	**14.53**	43.69	0.7705	0.3340	**35.14**	**32.13**	**14.69**	**34.52**	**46.94**
			DWA†, $T = 2$	13.63	**44.41**	**0.7581**	**0.3227**	36.41	34.12	12.82	31.12	43.48
	4.95	Dense	Equal Weights	16.06	52.73	0.6488	0.2871	33.58	28.01	20.07	41.50	53.35
			Uncert. Weights*	**16.48**	**54.40**	0.6282	0.2761	**31.68**	**25.68**	**21.73**	**44.58**	**56.65**
			DWA†, $T = 2$	16.15	54.35	**0.6059**	**0.2593**	32.44	27.40	20.53	42.76	54.27
	≈3	Cross-Stitch‡	Equal Weights	14.71	50.23	0.6481	0.2871	33.56	28.58	20.08	40.54	51.97
			Uncert. Weights*	15.69	52.60	0.6277	0.2702	32.69	27.26	21.63	42.84	54.45
			DWA†, $T = 2$	**16.11**	**53.19**	**0.5922**	**0.2611**	**32.34**	**26.91**	**21.81**	**43.14**	**54.92**
	1.77	MTAN†	Equal Weights	**17.72**	55.32	**0.5906**	0.2577	31.44	**25.37**	$\boxed{23.17}$	45.65	57.48
			Uncert. Weights*	17.67	**55.61**	0.5927	0.2592	**31.25**	25.57	22.99	**45.83**	**57.67**
			DWA†, $T = 2$	17.15	54.97	0.5956	**0.2569**	31.60	25.46	22.48	44.86	57.24
	1.77	MTAN† + PCGrad (ours)	Uncert. Weights*	$\boxed{\textbf{20.17}}$	$\boxed{\textbf{56.65}}$	$\boxed{\textbf{0.5904}}$	$\boxed{\textbf{0.2467}}$	$\boxed{\textbf{30.01}}$	$\boxed{\textbf{24.83}}$	22.28	$\boxed{\textbf{46.12}}$	$\boxed{\textbf{58.77}}$

Table 4: We present the full results on three tasks on the NYUv2 dataset: 13-class semantic segmentation, depth estimation, and surface normal prediction results. #P shows the total number of network parameters. We highlight the best performing combination of multi-task architecture and weighting in bold. The top validation scores for each task are annotated with boxes. The symbols indicate prior methods: *: (Kendall et al., 2018a), †: (Liu et al., 2018), ‡: (Misra et al., 2016b). Performance of other methods taken from (Liu et al., 2018).