We humbly thank all of our reviewers for their time and effort in considering our paper. We are encouraged by the general appreciation of the simplicity and wide applicability of GradDrop in our diverse test scenarios. If given the privilege of proceeding, we will address all comments (including any we couldn't address here) within the final version.

[**R1**] We want to reassure **R1** that our toy example methods are well-trained (and repeated 200 times), but SGD/PCGrad perform poorly due to the 1D setting. E.g. it is well known that SGD performs poorly with local minima, and our 1D setting manifests this drawback in a dramatic way. In regards to Table 2, our improvements lie well outside error bars and milder improvements on CelebA are typical (e.g. see [34] and [41]). We included fewer setting descriptions in Section 4.3 because most details were standard, but please reference Section A.4 for a fuller description. We can also certainly apply GradDrop to NYUv2, but chose the Waymo Open Dataset as it is more modern, larger, and difficult.

[**R2**] Regarding experiment stats in Tables 3/4, we did perform multiple runs, and we will include the error bars in a final version. But we should mention that our method's improvements lie well outside any error bars. We did not include per-method variances as they were similar across methods, but can clarify that within the paper. We also appreciate your suggestion on the synthetic experiments; we can already show that more closely overlapping sines within our toy setting will reduce the performance gap between methods, and will include such a study in a revision.

[**R3**] First, there may have been a small misunderstanding regarding page limits, as the Broader Impacts section is not included in the 8-page limit. But more importantly, new Grad Clipping (GC) and Grad Penalty (GP) baselines are shown in Table I. GradDrop handily beats both methods. During training, GP tends to converge faster, but often converges to a worse value. GP is also ≈30% slower per step than GradDrop on CelebA, and ≈45% slower on CIFAR-100. For 3D detection, we cannot use GP as it takes up too much memory, but in fact GC *was already in use* in our main paper results. So, GradDrop not only beats GC for 3D detection, but can be applied *together* with GC in synergistic fashion. Indeed, we also tested GC+GradDrop on CIFAR-100 and it beats GC-Only by 0.3% accuracy (not shown in Table I).

[**R4**] Regarding pre-multiplication, the 1.0 initialization is necessary as otherwise the layer nontrivially transforms its inputs and is no longer "virtual." Please also see additional discussion in A.1. Thanks for citing Tseng et al; although this counts as contemporaneous work, we will add discussion in a revision but also note that despite the cosmetic similarity of randomly dropping gradients, their work operates on a specific metalearning setting, assumes a random distribution of added grads, and does not consider the *sign* of the gradient, which is at the heart of multitask gradient conflict. We thus believe GradDrop is significantly different and provides important insight not present elsewhere.

[**R1**, **R4**] Regarding MGDA/GradNorm performance, please see Section A.5. We also agree that combining methods is optimal only sometimes (e.g. CIFAR-100), but just having this option is compelling. As to the crux of GradDrop's efficacy: **Proposition 2:** Given continuous, lower-bounded component loss functions $L_i(\mathbf{w})$ with local minima $\mathbf{w^{(i)}}$ and a GradDrop update $\nabla^{(GD)}$, then to second order around each $\mathbf{w^{(i)}}$, $E[|\nabla^{(GD)} L|_2]$ is monotonically increasing w.r.t. $|\mathbf{w} - \mathbf{w^{(i)}}|, \forall i$. **Sketch of proof:** Expand L_i around $\mathbf{w^{(i)}}$ to second order to show that $|\nabla L_i|$ increases w.r.t. $|\mathbf{w} - \mathbf{w^{(i)}}|$.□ As **R4** mentions, we do not claim convergence properties, but GradDrop produces larger gradients when *any* loss term is far from a minimum. Thus, if a model converges, the convergence point *likely* has better overlap between component loss minima. Similarly, any "minimum hopping" as posited by **R4** will favor overlapping minima.

[**R1**, **R4**] More on why we outperform SGD and Random GradDrop (RGD): **Proposition 3:** Suppose for 1D loss function $L = \sum_i L_i(w)$ an SGD grad reduces total loss by a linear estimate $|\Delta L^{(SGD)}|$. For GradDrop (GD) with prob keep fn $f(x) = x$ and RGD, we have $|\Delta L^{(SGD)}| = E[|\Delta L^{(GD)}|] \geq E[|\Delta L^{(RGD)}|]$. **Proof:** Set $p = \sum_{\nabla_i \geq 0} |\nabla_i|$ and $n = \sum_{\nabla_i < 0} |\nabla_i|$. Then $E[|\Delta L^{(GD)}|] = (p-n)(\mathcal{P}p - (1-\mathcal{P})n) = (p-n)^2 = \Delta L^{(SGD)} \geq 0.5(p-n)^2 = E[|\Delta L^{(RGD)}|]$.□ Thus, GradDrop preserves SGD statistics on average, but unlike SGD can also detect and penalize inconsistent task grads. We also beat RGD both theoretically (Prop 3) and empirically (Table I and main paper Table 3).

[**R1**, **R2**, **R3**, **R4**] Thank you all again for your feedback. We believe GradDrop is not only general and practical, but also gives new insight into the challenges of multitask optimization that will be of interest to the NeurIPS community.

Table I: New Baselines. Std dev per metric, CelebA: ($\pm 0.02\%$, ± 0.04 and ± 0.05) and CIFAR-100: ($\pm 0.2\%, \pm 0.02$)

	CelebA			**CIFAR-100**	
Method	Err Rate (%) ↓	F1_{max} ↑	Test Loss ↓	Err Rate (%) ↓	Test Loss ↓
Baseline	8.71	29.35	8.00	29.8	1.22
Gradient Clipping (GC) **R3**	8.70	29.34	7.93	29.4	1.22
Gradient Penalty (GP) **R3**	8.63	29.43	7.96	30.6	1.28
Random GradDrop (RGD) **R1**, **R4**	8.60	29.42	7.86	29.6	1.16
Ours	**8.52**	**29.57**	**7.80**	**28.9**	**1.06**