

Robust Analysis of Multi-Task Learning Efficiency: New Benchmarks on Light-Weighed Backbones and Effective Measurement of Multi-Task Learning Challenges by Feature Disentanglement

Dayou Mao¹

daniel.mao@uwaterloo.ca

Yuhao Chen¹

yuhao.chen1@uwaterloo.ca

Yifan Wu¹

yifan.wu1@uwaterloo.ca

Maximilian Gilles²

maximilian.gilles@kit.edu

Alexander Wong¹

a28wong@uwaterloo.ca

¹Vision and Image Processing Research Group, Waterloo, Canada²Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany

Abstract

One of the main motivations of MTL is to develop neural networks capable of inferring multiple tasks simultaneously. While countless methods have been proposed in the past decade investigating robust model architectures and efficient training algorithms, there is still lack of understanding of these methods when applied on smaller feature extraction backbones, the generalizability of the commonly used fast approximation technique of replacing parameter-level gradients with feature level gradients, and lack of comprehensive understanding of MTL challenges and how one can efficiently and effectively identify the challenges. In this paper, we focus on the aforementioned efficiency aspects of existing MTL methods. We first carry out large-scale experiments of the methods with smaller backbones and on a the MetaGraspNet dataset as a new test ground. We also compare the existing methods with and without using the fast gradient surrogate and empirically study the generalizability of this technique. Lastly, we propose Feature Disentanglement measure as a novel and efficient identifier of the challenges in MTL, and propose Ranking Similarity score as an evaluation metric for different identifiers to prove the faithfulness of our method.

1. Introduction

Multi-task learning (MTL) is a learning paradigm that aims to develop systems capable of performing multiple tasks simultaneously. At a high-level, MTL designs systems with two key components. They learn a single backbone that encodes raw inputs into shared representations and append prediction heads that consume the shared representa-

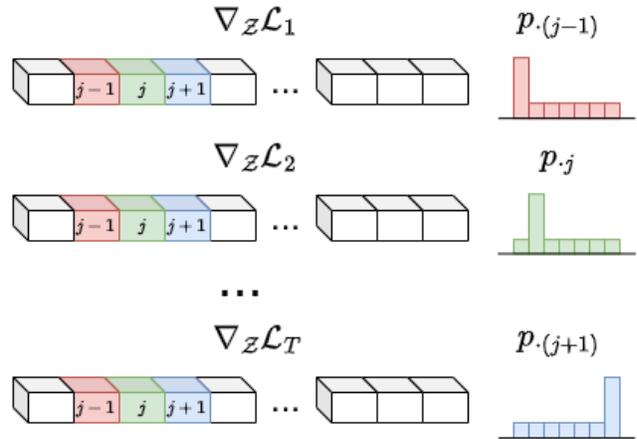


Figure 1. Illustration of feature disentanglement calculation. In the above, $p_{\cdot,j}$ denotes the mapping $i \mapsto p_{ij}$, which is the (smoothened) distribution of feature saliency at location j across all tasks. Same for $p_{\cdot,(j-1)}$ and $p_{\cdot,(j+1)}$. If an extracted feature is disentangled for the T down stream tasks, then each distribution $p_{\cdot,j}$ should be concentrated on fewer tasks and have lower entropy.

tion and each learning a different down stream task atop the backbone. Multi-task models are efficient because the majority of parameters lie in the feature extraction backbone and are *shared parameters*, while the *task-specific parameters* are usually very light-weighted. The merit of designing MTL systems is believed to be two-fold. Firstly, compared to developing a single-task model for each task, multi-task models significantly reduces overall model size because of shared backbone, and benefits of faster inference speed, reduced memory footprint, and lower power consumption all follow. Secondly, it is a common belief

that the more complex the supervision signals, the richer the learned representation [2, 29, 31]. Introducing supervision signals from a diverse range of downstream tasks has been proven to be an effective approach to improve performance compared to training single-task learning systems [4, 9, 19, 33, 35, 38, 42, 53, 57]. The idea of MTL has already been exploited in classical discriminative computer vision algorithms well before it becomes an active research topic on its own [12, 15, 44].

Despite all the potential benefits MTL can bring, training a multi-task models is also more challenging, as observed in the fields of computer vision [15, 19, 20, 46, 47, 58], natural language processing [49, 50], meta learning [1], and reinforcement learning [16, 34, 39, 40, 48]. Challenges of MTL arise because it requires consideration of multiple objectives when training the single shared backbone. Given vision tasks with a wide range of difficulties, output dimensions, and types of training loss functions, it is rarely the case that all tasks “align well” during training. Namely, a parameter update on the backbone that improves performance of one task may lead to worse performance of another task at the same time. This phenomenon is known as *negative transfer* or *destructive interference* [58]. It is widely accepted that this phenomenon can be explained by two factors: *task conflicts* and *task dominance* (Section 5.1).

Extensive research has delved into MTL techniques aimed at mitigating the issues of task conflicts and task dominance, as we will elaborate upon in more detail in Section 2. However, the literature still lacks understanding in the following 4 aspects. Firstly, the literature has been focusing on ResNet50 [28, 42, 43] and SegNet [23, 26, 32, 43, 55] backbones, but lack attention to smaller backbones like ResNet18. Secondly, the existing literature is mostly focused on toy examples [5, 6, 22, 25–27, 32, 43, 55], Celeb-A [6, 23, 27, 28, 42, 55], CityScapes [19, 23, 26–28, 32, 42, 43, 55], NYU/NYU-v2 [5, 23, 26–28, 32, 43, 47, 55, 56], Multi-MNIST [21, 42, 47, 55], Multi-Fashion-MNIST [25, 26], Multi-Task CIFAR-100 [6, 55], and QM-9 [27, 32], but lack common test ground on complex vision tasks across diverse range of tasks with a large-scale dataset. Thirdly, prior work [6, 18, 42, 43] have proposed the technique to use feature-level gradients to replace parameter-level gradients for more efficient computation, which we call the “Fast Gradient Surrogate” technique. However, some [18] lack theoretical guarantee or solid empirical analysis to prove effectiveness of this surrogate and others [32] have reported poor results when doing so. A solid understanding of its generalizability is still missing. Fourthly, prior work (Sections 2.1, 2.2, 2.3) hypothesize that task conflicts and task dominance are the root causes for poor performance of MTL models. But there is lack of analysis on the causal relationship between these claimed root cause and model performance. Moreover, both

of these measures require computing the gradients of all losses w.r.t. shared parameters and hence requires back propagation through the entire backbone T times if T is the number of tasks, which is super computationally expensive [18, 42, 43]. A more efficient way to understand MTL challenges is needed.

In this paper, we conduct large-scale experiments to address the efficiency issues of optimization algorithms for MTL from the following angles: We provide comprehensive benchmark results with ResNet18 [14] backbones on MetaGraspNet [11], CityScapes [7], and NYU-v2 [45] datasets, compare existing methods with their fast gradient surrogates, and propose Feature Disentanglement measure, shown in Figure 1, as a novel perspective to identify MTL challenges and propose Ranking Similarity as an evaluation protocol for comparing different identifiers. Our contribution in this paper is three-fold:

- We complement the existing performance benchmarks in the literature with smaller feature extraction backbones, ResNet18 [14], and a more robust benchmark dataset, MetaGraspNet [11]. (Section 3).
- We provide solid empirical evidence and show that the fast gradient surrogate technique cannot not be generalized to all methods and has to be analyzed and tested depending on the specific algorithm (Section 4);
- We propose Feature Disentanglement measurement that can more efficiently and faithfully reflect MTL challenge compared to traditional task conflicts and task dominance measures, evaluated by Ranking Similarity against test-time performance. (Section 5).

2. Problem Definition and Related Work

The problem of Multi-Task Learning aims to train a model to learn several tasks simultaneously. Formally, we assume for the following notations. T denotes the number of tasks. \mathcal{X} denotes the set of training inputs. \mathcal{Y}_i denotes the label space for task i and we define $\mathcal{Y} := \bigoplus_{i=1}^T \mathcal{Y}_i$ as the collection of all task labels. $\mathcal{D} \subset \mathcal{X} \times \mathcal{Y}$ denotes the training dataset. $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$ denotes a neural network parametrized by $\theta \in \Omega$ where Ω is the parameter space. $\mathcal{L}_i : \mathcal{Y}_i \times \mathcal{Y}_i \rightarrow \mathbb{R}_{\geq 0}$ denotes the loss functions for the i -th task. In general, multi-task learning can be formulated as the following optimization problem:

$$\min_{\theta \in \Omega} \sum_{(x,y) \in \mathcal{D}} \mathcal{L}(f_\theta(x), y), \quad (1)$$

where \mathcal{L} is some loss function, either scalar-valued or vector-valued, defined on $\mathcal{Y} \times \mathcal{Y}$. The average of all task losses $\mathcal{L} := \frac{1}{T} \sum_{i=1}^T \mathcal{L}_i \in \mathbb{R}_{\geq 0}$, or Equal Weighting (EW), is the most commonly used baseline. Pareto optimization minimizes the loss vector $\mathcal{L} := (\mathcal{L}_1, \dots, \mathcal{L}_T)^\top \in \mathbb{R}_{\geq 0}^T$ w.r.t. the partial-order \leq_K on \mathbb{R}^T induced by the pointed, closed,

and convex cone $K := \mathbb{R}_{\geq 0}^T$ [25, 32, 42, 54]. As we discuss in the remaining of this section, wealth of research has been done on efficient and effective optimization algorithms that solve Problem 1 and on designing the specific forms of the loss function \mathcal{L} .

We make two remarks here on the differences and relationships between gradient manipulation methods (Section 2.1) and gradient balancing methods (Section 2.2), before we go into depth. (1) Due to the linearity of the gradient operator ∇ , scaling the loss function and scaling the gradients are essentially the same. However, gradient manipulation methods aim to manipulate the *directions* of the gradients to resolve *task conflicts*, whereas gradient balancing methods aim to manipulate the *magnitudes* of the gradients to resolve *task dominance*. (2) Gradient manipulation methods focus on manipulating gradients for the shared parameters and tune the task-specific parameters as usual single-task learning, whereas gradient balancing methods scales gradients for all model parameters.

Additional Notations and Terminologies. Throughout, we adopt the following notations: (1) for any natural number n , $[n] := \{1, \dots, n\}$, (2) $\theta^{\text{sh}} \in \mathbb{R}^d$ denotes shared parameters where d is the number of shared parameters and \mathcal{Z} denotes shared representation, (3) $g_i := \nabla_{\theta^{\text{sh}}} \mathcal{L}_i$ denotes each task gradient, or “parameter-level gradients”, and $G := [g_1, \dots, g_T]^\top$ denotes the gradient matrix whose columns are the task gradients, (4) $\nabla_{\mathcal{Z}} \mathcal{L}_i$ are the “feature-level gradients”.

2.1. Gradient Manipulation

To update the shared parameters taking all tasks into consideration, gradient manipulation methods [6, 10, 23, 26, 32, 42, 43, 51, 55] compute task gradients $g_i := \nabla_{\theta^{\text{sh}}} \mathcal{L}_i$ and each propose different method to compute $\alpha = (\alpha_1, \dots, \alpha_T)^\top \in \mathbb{R}^T$ so that the final update on the shared parameters is done by the aggregation $\hat{g} := \sum_{i=1}^T \alpha_i g_i$. Prediction heads are trained as in usual single-task learning, each supervised by their own loss function. Computation of α can be done either explicitly or implicitly.

Explicit Methods [6, 23, 51, 55] derive closed-form formulae for α based on some heuristics and may rely on some stochasticity. PCGrad [55] proposed to reduce task conflicts by projecting gradients onto the normal planes of each other, whenever the angle between two gradient vectors is obtuse. GradVac [51] expanded along this direction and proposed to encourage acute angles between gradients by maintaining an Exponential Moving Average (EMA) of cosine similarity between task gradients as upper bound on angles between. GradDrop [6] propose to mask the gradients with Gradient Sign Purity so that the gradient signs are more aligned. Random Gradient Weighting (RGW) [23] draws random samples from a Gaussian distribution, normalize them into a probability simplex, and re-weigh the

task gradients with normalized samples.

Implicit Methods [10, 26, 32, 42, 43] hypothesize different objectives and compute the gradient weights α by solving either an optimization problem or a system of equations. MGDA [42] re-weighs task gradients so that the result is the norm minimizer in the convex hull enclosed by the task gradients. CAGrad [26] proposed to maximize the minimum amount of decrease (in absolute value) in the individual losses and could be viewed as a generalized version of MGDA by adding a search region. Nash-MTL [32] follows a similar idea as CAGrad, but rather than maximizing the minimum amount of decrease, it maximizes the sum of the log decreases in each individual loss. This eventually resolves to solving a (non-linear) system of equations. Aligned-MTL [43] proposed to first approximate the gradient matrix $G := [g_1, \dots, g_T] \in \mathbb{R}^{d \times T}$ with its best unitary matrix approximation, which is sure to have stability number 1, and then use the summation of the approximated task gradients to update θ^{sh} .

2.2. Gradient Balancing

Similar to gradient manipulation methods reviewed in the previous section, gradient balancing methods [5, 13, 17, 19, 23, 25, 27, 28, 30, 37, 56], or loss balancing methods, aim to assign weights $w = (w_1, \dots, w_T)^\top \in \mathbb{R}_{\geq 0}^T$ to the loss functions and solve the following optimization problem, so that all tasks are learned at compatible pace:

$$\min_{\theta \in \Omega} \sum_{(x,y) \in \mathcal{D}} \sum_{i=1}^T w_i \mathcal{L}_i(f_\theta(x), y_i) \quad (2)$$

Computation of w can be done either explicitly or implicitly, or even by an extra optimizer.

Explicit Methods [5, 13, 23, 28, 30, 56] compute w explicitly based on different heuristics. GradNorm [5] proposed to control the learning pace of different tasks based on the relative norm of the gradients and does so by assigning task weights and updating them at each training iteration. Exploiting the same idea as RGW [23], the same paper also proposed Random Loss Weighting (RLW). In [30], the authors propose yet another simple weighting mechanism named Dynamic Weight Average (DWA) to re-weight the losses based on the relative descending rate of each loss.

Implicit Methods [25, 27] compute the loss weights w by solving an optimization subproblem at each training iteration. In particular, FAMO [27] proposed that the parameter update at each training step should maximize the lowest relative improvement of the task losses. The subtle difference from CAGrad [26] is that FAMO uses *relative* improvements, so that solving the optimization subproblem results in reducing gradient dominance, whereas CAGrad uses *absolute* improvements and would result in reduced gradient conflicts.

Optimization-Based Methods [17, 19] dynamically update the loss weights with an extra set of objective and optimizer.

2.3. Gradient Regularization

Gradient regularization methods [18, 47] design regularization terms in the loss function and solves the following optimization problem:

$$\min_{\theta \in \Omega} \sum_{(x,y) \in \mathcal{D}} \mathcal{L}(f_{\theta}(x), y) + \mathcal{L}_{\text{reg}}(G) \quad (3)$$

where $G \in \mathbb{R}^{d \times T}$ is the gradient matrix. In particular, Suteu [47] argues that orthogonal task gradients are beneficial to learning and adds squared cosine similarity as a regularization term. Javaloy [18] proposed to regularize the gradients by minimizing the angles between each task gradient and their average.

2.4. Fast Gradient Surrogate

Most of the existing MTL algorithms rely on computing parameter-level gradients $\nabla_{\theta^{\text{sh}}} \mathcal{L}_i$, which requires back propagation through the entire backbone where the shared parameters lie. Feature-level gradients $\nabla_{\mathcal{Z}} \mathcal{L}_i$ are a lot cheaper to compute as back propagation is only required through the prediction heads, which are usually very lightweight. However, some works [42, 43] provide theoretical reasoning that the optimization problem could be solved with feature-level gradients $\nabla_{\mathcal{Z}} \mathcal{L}_i$, as they define an upper bound for the objective function. Other works [18, 32] have also empirically investigated application of this feature-level surrogate but [18] obtained reasonable results while [32] observed poor results.

2.5. Saliency Maps in Explainable AI

Explainable AI seeks to explain behaviours of AI systems. The representative work, GradCam [41], of top-down approaches proposed to compute a saliency map for the intermediate activations via the (absolute value of) gradients of class scores w.r.t. the activations. This has inspired us to quantify feature disentanglement by measuring the saliency for each task. More details are explained in Section 5.

3. Benchmarks

In this section, we carry out large-scale experiments to complement the understanding of MTL optimization algorithms in the following two dimensions: (1) application on smaller models, namely ResNet18 backbones, and (2) application in efficiency demanding robotics vision application, on a more complex and super large-scale real-world dataset, the MetaGraspNet dataset [11]. We also provide results on CityScapes [7] and NYU-v2 [45] for completeness.

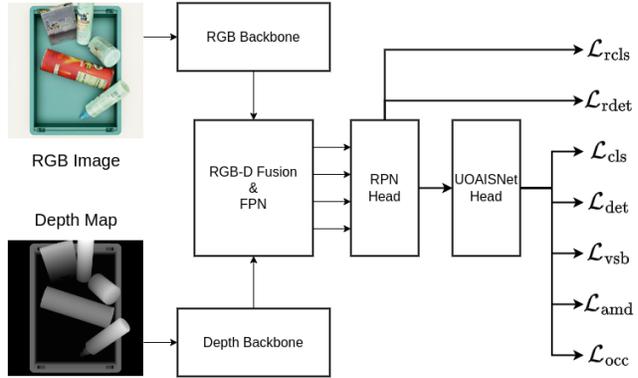


Figure 2. Illustration of model architecture used on the MetaGraspNet [11] benchmark.

Selected Methods. We studied 15 MTL optimization algorithms, from three categories: (1) we selected PCGrad [55], GradVac [51], GradDrop [6], RGW [23], MGDA [42], CA-Grad [26], Nash-MTL [32], and Aligned-MTL [43] from the gradient manipulation category (8 in total), (2) we selected Uncertainty [19], GradNorm [5], IMTL [28], FAMO [27], RLW [23], and DWA [30] from the gradient balancing category (6 in total), and (3) CosReg [47] from the gradient regularization category. Among these methods, MGDA and Aligned-MTL have provided theoretical analysis on replacing parameter-level gradients with feature-level gradients, known as MGDA-UB and Aligned-MTL-UB, respectively. These variants are also benchmarked. A comprehensive study on this fast approximation technique is reported in Section 4.

General Setup. For stochasticity consideration, all experiments were repeated 3 times and all results in this paper are average results across the 3 repetitions. As the focus of this paper is to study relative performance as compared to the baseline, rather than proposing novel MTL methods, we focus our experiments on the early stage of training. We refer the readers to Appendix 7 for more details.

3.1. Experiments on MetaGraspNet Dataset

The Dataset. We provide a new test ground for multi-task learning on the MetaGraspNet dataset [11], due to its significantly larger dataset size, increased task complexity, and higher real-world value.

Network Architecture. We follow [3, 52] and utilize two ResNet [14] backbones, one for RGB image input and one for depth map input. Results at each stage of the ResNet from the RGB image input and the depth map input are fused by convolution layers. These fused features collectively yield the output from the backbone network. Then we feed the backbone outputs to a Feature Pyramid Network (FPN) [24] to fuse the features from different levels. On top of this extracted feature from the FPN neck we at-

tach Region Proposal Network (RPN) [36] and UOAINet [3] as prediction heads for amodal object bounding boxes, visible object masks, amodal object masks, and occlusion predictions. An architecture overview is shown in Figure 2. **Training Objective and Baseline Definition.** We follow [52] and use the following loss functions: $\mathcal{L}_{\text{rcls}}$ and $\mathcal{L}_{\text{rdet}}$ for foreground/background classification and bounding boxes regression by the RPN head and \mathcal{L}_{cls} , and \mathcal{L}_{det} by the amodal detection head; \mathcal{L}_{vsb} for visible object masks prediction, \mathcal{L}_{amd} for amodal object masks prediction, and \mathcal{L}_{occ} for object occlusion prediction. We set the baseline loss function to be

$$\mathcal{L}_{\text{tot}} := \mathcal{L}_{\text{rcls}} + \mathcal{L}_{\text{rdet}} + \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{det}} + \mathcal{L}_{\text{vsb}} + \mathcal{L}_{\text{amd}} + \mathcal{L}_{\text{occ}}, \quad (4)$$

i.e., the sum of all 7 loss functions.

Experiment Results. Note that experiments are not meant to replicate existing results but rather comparing performance of different MTL optimization algorithms against the baseline. However, we observed extremely poor performance with GradNorm [5] and Nash-MTL [32], and the feature-level gradient counterpart of GradDrop [6], which is what originally proposed, so these methods are not reported on MetaGraspNet dataset. Results have shown that GradVac [51], GradDrop [6], IMTL [28], DWA [30], MGDA-UB [42], (rep) CAGrad [26], and (rep) CosReg [47] achieved consistent performance gain compared to the baseline on all 6 evaluation metrics. GradVac [51], IMTL [28], and (rep) MGDA achieved top-three performance under a majority (≥ 3) of the metrics. Full results on MetaGraspNet are summarized in Table 1.

3.2. Experiments on CityScapes and NYU-v2

We largely base our experiment setup for these two datasets on existing work in the literature and refer the readers to Appendix 7 for details. Full results on CityScapes and NYU-v2 are summarized in Tables 2 and 3, respectively. Results have shown that CosReg [47] consistently reaches top 3 on both datasets and all evaluation metrics. Moreover, GradDrop [6] also achieved top 3 scores except normal estimation (NE) on NYU-v2 [45], but still improved the baseline.

4. Generalizability of Fast Gradient Surrogate

It has been a common technique to replace $\nabla_{\theta^{\text{sh}}}\mathcal{L}_i$ with $\nabla_{\mathcal{Z}}\mathcal{L}_i$ for reduced computation cost [18, 42, 43]. While it might be reasonable to do so due to chain rule and sub-additivity of norms [42, 43], other methods [32] has reported significant performance degrade with feature-level gradients in their method. To the best of our knowledge, there is no prior work addressing the generalizability of this approximation via comprehensive empirical study across a large basket of existing algorithms.

We compared the performance using $\nabla_{\theta^{\text{sh}}}\mathcal{L}_i$ versus $\nabla_{\mathcal{Z}}\mathcal{L}_i$ on 8 optimization algorithms on the MetaGraspNet

[11] dataset. Full results are summarized in Table 4. Results have displayed the following: Firstly, only MGDA [42] and Aligned-MTL [43] achieved consistent performance gain under all evaluation metrics, and these are exactly the two selected methods that argued that feature-level gradients could be used as an upper bound (up to scaling) during the optimization process. On the other hand, GradVac [51], RGW [23], and IMTL [28] got significant performance degradation and hence this surrogate is clearly not applicable to these algorithms. We conclude that this fast gradient surrogate is not generalizable and we encourage more theoretical analysis to be done for each method.

5. Feature Disentanglement Measure

In this section, we propose a novel measurement using feature disentanglement for identifying the challenges in MTL problems. To the best of our knowledge, we are the first to explicitly quantify the severity of feature disentanglement and to monitor its training dynamics.

The intuition behind lies in the gap between Single-task Learning (STL) and MTL. STL is easier in the sense that each task could be solved independently. The gap between MTL and STL could be bridged by allocating disjoint subsets of the extracted features for each task, while still using a single backbone. We propose to understand the challenges in MTL from the perspective of learnt shared representation and study MTL by asking fundamental question: *what kind of features are beneficial for a given set of tasks?*

5.1. Preliminaries

The literature is familiar with how task conflicts and task dominance are defined for two tasks. In this section, we make it clear how these measures are defined for a set of T tasks.

Task Conflicts. We follow [47, 51, 55] and define the Gradient Direction Similarity (GDS) measure for T tasks as

$$\alpha_{ij} := \frac{\langle \nabla_{\theta^{\text{sh}}}\mathcal{L}_i, \nabla_{\theta^{\text{sh}}}\mathcal{L}_j \rangle}{\|\nabla_{\theta^{\text{sh}}}\mathcal{L}_i\|_2 \|\nabla_{\theta^{\text{sh}}}\mathcal{L}_j\|_2} \text{ for } i, j \in [T]; \quad (5a)$$

$$\text{GDS} := \frac{1}{T(T-1)} \sum \left\{ \alpha_{ij} : i, j \in [T], i \neq j \right\}, \quad (5b)$$

where $\alpha_{ij} \in [-1, +1]$ is the cosine value of the angle between $\nabla_{\theta^{\text{sh}}}\mathcal{L}_i$ and $\nabla_{\theta^{\text{sh}}}\mathcal{L}_j$ and quantifies the relationship between the *directions* of the task gradients. A lower GDS score indicates less agreement between the supervision from different losses.

Task Dominance. We follow [55] and define the Gradient Magnitude Similarity (GMS) measure for T tasks as

$$\beta_{ij} := \frac{2\|\nabla_{\theta^{\text{sh}}}\mathcal{L}_i\|_2 \|\nabla_{\theta^{\text{sh}}}\mathcal{L}_j\|_2}{\|\nabla_{\theta^{\text{sh}}}\mathcal{L}_i\|_2^2 + \|\nabla_{\theta^{\text{sh}}}\mathcal{L}_j\|_2^2}, \text{ for } i, j \in [T]; \quad (6a)$$

$$\text{GMS} := \frac{1}{T(T-1)} \sum \left\{ \beta_{ij} : i, j \in [T], i \neq j \right\} \quad (6b)$$

	BBox mAP	BBox mAR	VMask mAP	VMask mAR	AMask mAP	AMask mAR
Baseline	0.383	0.519	0.518	0.647	0.490	0.617
RGW [23]	0.358 (↓)	0.513 (↓)	0.490 (↓)	0.644 (−)	0.462 (↓)	0.614 (−)
PCGrad [55]	0.370 (↓)	0.526 (↑)	0.500 (↓)	0.657 (↑)	0.454 (↓)	0.595 (↓)
GradVac [51]	0.393 (↑)	0.560 (↑)	0.521 (−)	0.680 (↑)	0.495 (−)	0.654 (↑)
MGDA [42]	0.371 (↓)	0.531 (↑)	0.452 (↓)	0.594 (↓)	0.430 (↓)	0.564 (↓)
CAGrad [26]	0.411 (↑)	0.557 (↑)	0.522 (−)	0.648 (−)	0.488 (−)	0.604 (↓)
GradDrop [6]	0.399 (↑)	0.541 (↑)	0.533 (↑)	0.666 (↑)	0.505 (↑)	0.634 (↑)
Aligned-MTL [43]	0.400 (↑)	0.547 (↑)	0.477 (↓)	0.610 (↓)	0.460 (↓)	0.580 (↓)
IMTL [28]	0.410 (↑)	0.534 (↑)	0.550 (↑)	0.667 (↑)	0.541 (↑)	0.658 (↑)
RLW [23]	0.360 (↓)	0.504 (↓)	0.499 (↓)	0.649 (−)	0.466 (↓)	0.614 (−)
DWA [30]	0.390 (↑)	0.533 (↑)	0.533 (↑)	0.664 (↑)	0.499 (↑)	0.628 (↑)
Uncertainty [19]	0.206 (↓)	0.349 (↓)	0.345 (↓)	0.507 (↓)	0.319 (↓)	0.482 (↓)
FAMO [27]	0.431 (↑)	0.564 (↑)	0.517 (−)	0.623 (↓)	0.510 (↑)	0.613 (−)
CosReg [47]	0.387 (−)	0.545 (↑)	0.522 (−)	0.672 (↑)	0.488 (−)	0.631 (↑)

Table 1. Benchmark results of all selected methods with ResNet-18 backbone on MetaGraspNet [11] dataset. Performance increase (with ↑) or decrease (with ↓) that’s more than 0.01 are shown in brackets after each table entry. Scores within 0.01 offset from the baseline are treated as comparable performance and labeled by “−”. Best viewed in color.

	DE (↓)	SS (↑)	IS (↓)
Baseline	1.453	15.0%	0.131
RGW [23]	1.465	14.9%	0.131
PCGrad [55]	1.462	15.0%	0.133
GradVac [51]	1.466	15.0%	0.132
MGDA [42]	1.490	15.1%	0.134
CAGrad [26]	1.446	15.3%	0.128
GradDrop [6]	1.367	15.5%	0.117
Nash-MTL [32]	1.472	15.1%	0.132
Aligned-MTL [43]	1.499	14.8%	0.134
IMTL [28]	1.467	15.0%	0.128
RLW [23]	1.507	14.6%	0.133
DWA [30]	1.457	15.0%	0.131
Uncertainty [19]	15.597	1.4%	0.158
GradNorm [5]	1.454	15.0%	0.133
FAMO [27]	15.059	1.3%	0.055
CosReg [47]	1.344	16.6%	0.081

Table 2. Benchmark results with ResNet18 backbone on CityScapes [7]. DE stands for depth estimation, evaluated by L_1 distance; SS stands for semantic segmentation, evaluated by mIoU; and IS stands for instance segmentation, evaluated by L_1 distance.

where $\beta_{ij} \in [0, 1]$ quantifies the relationship between the *magnitudes* of the task gradients. A lower GMS score indicates less aligned learning pace between the losses.

5.2. Method: Feature Disentanglement Measure

Our method, Feature Disentanglement measure, is defined as follows. Given an extracted feature \mathcal{Z} and a task loss \mathcal{L}_i , how much a change in \mathcal{Z}_j can affect \mathcal{L}_i can be measured by $|\nabla_{\mathcal{Z}_j} \mathcal{L}_i| \in \mathbb{R}_{\geq 0}$, and the tensor $|\nabla_{\mathcal{Z}} \mathcal{L}_i|$ of gradient magnitudes, which has the same shape as \mathcal{Z} , can be interpreted as a saliency map on \mathcal{Z} , as inspired by GradCam [41] in

	DE (↓)	NE (↓)	SS (↑)
Baseline	1.116	44.227	2.7%
RGW [23]	1.111	43.759	2.7%
PCGrad [55]	1.125	43.834	2.7%
GradVac [51]	1.118	43.954	2.8%
MGDA [42]	1.155	44.632	2.6%
CAGrad [26]	1.143	44.042	2.7%
GradDrop [6]	1.093	43.912	2.9%
Nash-MTL [32]	1.119	44.013	2.7%
Aligned-MTL [43]	1.149	46.168	2.7%
IMTL [28]	1.120	44.040	2.6%
RLW [23]	1.541	43.524	2.4%
DWA [30]	1.120	44.196	2.6%
Uncertainty [19]	9.099	47.985	0.3%
GradNorm [5]	1.115	45.786	2.6%
FAMO [27]	1.097	39.148	0.5%
CosReg [47]	0.896	40.586	3.5%

Table 3. Benchmark results with ResNet18 backbone on NYU-v2 [45]. DE stands for depth estimation, evaluated by L_1 distance; NE stands for normal estimation, evaluated by angle in degrees, and SS stands for semantic segmentation, evaluated by mIoU.

the Explainable AI (XAI) literature. At location j , we can quantify the entropy of the saliencies across T tasks by

$$\mathcal{E}_j(\mathcal{Z}) := - \sum_{i=1}^T p_{ij} \log p_{ij}, \text{ where} \quad (7a)$$

$$p_{ij} := |\nabla_{\mathcal{Z}_j} \mathcal{L}_i| / \sum_{k=1}^T |\nabla_{\mathcal{Z}_j} \mathcal{L}_k| \quad (7b)$$

The feature disentanglement measure for the entire shared representation \mathcal{Z} is defined to be the average entropy across

	BBox mAP	BBox mAR	VMask mAP	VMask mAR	AMask mAP	AMask mAR
RGW [23]	-6.4%	-3.6%	-6.9%	-4.2%	-5.9%	-3.4%
PCGrad [55]	-1.4%	-0.7%	0.7%	-0.2%	3.2%	3.0%
GradVac [51]	-13.9%	-7.8%	-11.2%	-7.6%	-14.2%	-10.6%
MGDA [42]	14.3%	6.8%	17.8%	12.0%	19.8%	14.1%
CAGrad [26]	-0.9%	-2.9%	2.9%	2.6%	5.1%	4.8%
Aligned-MTL [43]	2.5%	1.3%	8.1%	5.9%	8.3%	7.7%
IMTL [28]	-7.4%	-1.5%	-8.2%	-1.6%	-8.1%	-1.1%
CosReg [47]	0.2%	-2.3%	1.5%	0.3%	0.8%	-0.4%

Table 4. Benchmark results with ResNet18 backbone on MetaGraspNet dataset but all gradients in the algorithms replaced with feature-level gradients. Table entries are relative performance change compared to their parameter-level gradient counterparts.

	BBox mAP	BBox mAR	VMask mAP	VMask mAR	AMask mAP	AMask mAR
GDS	64.3%	63.3%	50.5%	60.0%	56.7%	51.0%
GMS	67.6%	70.5%	53.8%	59.5%	50.5%	59.0%
FD	56.7%	58.6%	65.7%	65.7%	65.2%	69.0%

Table 5. Ranking similarity results on MetaGraspNet [11] dataset.

all positions:

$$\text{FD} := \text{mean} \left\{ \mathcal{E}_j(\mathcal{Z}) : j \in [\dim(\mathcal{Z})] \right\}, \quad (8)$$

where $\dim(\mathcal{Z})$ denotes the dimension of the Euclidean space \mathcal{Z} lies in. A lower feature disentanglement measurement indicates that activations are salient to fewer tasks, and hence larger disentangled-ness. Note that monitoring task conflicts and task dominance is extremely expensive as they require back propagation through until the first layer T times to compute $\nabla_{\theta^{\text{sh}}} \mathcal{L}_i$. However, the feature disentanglement measure (ours) is a lot cheaper as it only relies on $\nabla_{\mathcal{Z}} \mathcal{L}_i$ and only need to back propagates through the T prediction heads. An illustration of this definition is shown in Figure 1.

5.3. Evaluation Protocol: Ranking Similarity

In order to quantitatively evaluate the faithfulness of different measures (GDS, GMS, and FD) for revealing the challenges in MTL problems, we propose Ranking Similarity to quantify the alignment against test-time performance.

Definition. Given a set of n scalars $A := \{a_1, \dots, a_n\} \subset \mathbb{R}$ and two rankings $R_1, R_2 : A \rightarrow [n]$ on A , we define the *ranking similarity* $\mathcal{S}(R_1, R_2)$ between R_1 and R_2 to be the following average:

$$\mathcal{S}(R_1, R_2) := \frac{1}{n(n-1)} \sum_{\substack{i,j=1 \\ i \neq j}}^n \mathbb{I} \left[R_1 \text{ and } R_2 \text{ agree on } a_i \text{ and } a_j \right], \quad (9)$$

where for any $i, j \in [n]$ with $i \neq j$, R_1 and R_2 agree on a_i and a_j if and only if “ $R_1(a_i) > R_1(a_j)$ ” and “ $R_2(a_i) > R_2(a_j)$ ” have the same truth value. i.e., $\mathcal{S}(R_1, R_2)$ is the

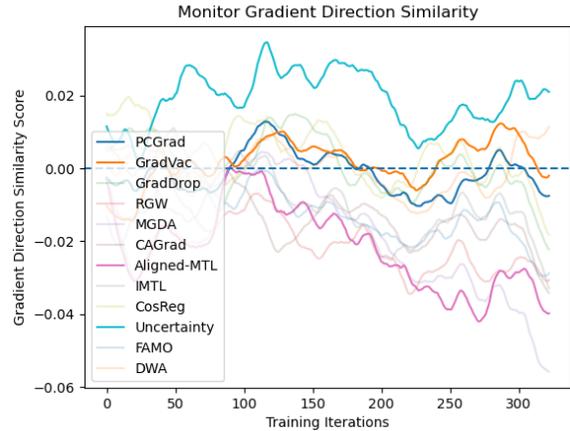


Figure 3. Training dynamics of GDS using gradients w.r.t. shared parameters.

percentage of pairs (a_i, a_j) with the same ordering under R_1 and R_2 . Larger ranking similarity means more agreement under different ranking methods.

Symmetry. Let R_1 and R_2 be two arbitrary rankings and let R'_2 be the reverse of R_2 . Then R_1 and R_2 agree on (a_i, a_j) if and only if R_1 and R'_2 disagree. So $\mathcal{S}(R_1, R'_2) = 1 - \mathcal{S}(R_1, R_2)$. As we can always reverse a ranking when making comparisons, we only consider $|\mathcal{S}(R_1, R_2) - 0.5|$. The larger the absolute value, the clearer R_1 and R_2 aligns. We only report this symmetric version between test-time performance scores and MTL challenge measures.

5.4. Qualitative Results

We show interesting examples and qualitatively demonstrate the effectiveness of our method from the training

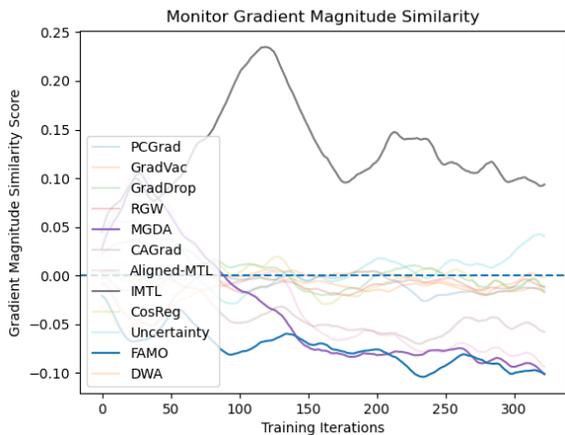


Figure 4. Training dynamics of GMS using gradients w.r.t. shared parameters.

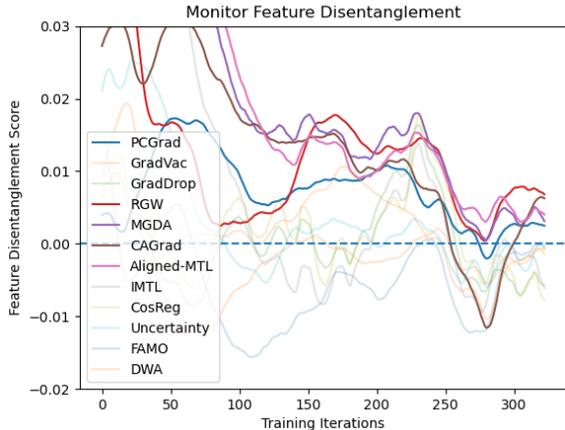


Figure 5. Training dynamics of Feature Disentanglement (FD) using gradients w.r.t. shared parameters.

dynamics on MetaGraspNet [11] dataset. Trajectories are plotted in Figures 3, 4, and 5. For clarity, we only plot the relative values to the baseline method. All curves are smoothed by taking the moving average with window size equal to 1/10 of the total trajectory length. We also added transparency to those not of interest and emphasized ones from which we make important qualitative observations. See Appendix 8 for more details on how trajectories were plotted.

Task Conflicts and Task Dominance. Figure 3 and Table 1 have shown that Uncertainty Weighting, which had poor performance on the test set (Table 1), and GradVac, which had stronger performance, both achieved high GDS scores. In contrast, PCGrad and Aligned-MTL, which are hard to conclude one is superior to the other on the test set, lie far

	DE	SS	IS
GMS	56.8%	58.2%	61.3%
GDS	58.2%	55.3%	55.7%
FD	57.3%	59.0%	61.3%

Table 6. Ranking similarity results on CityScapes [7].

	DE	NE	SS
GMS	61.3%	57.5%	67.8%
GDS	70.0%	72.7%	60.5%
FD	67.9%	70.6%	57.9%

Table 7. Ranking similarity results on NYU-v2 [45].

apart in the GDS plot. Figure 4 and Table 1 have shown that MGDA and FAMO achieved almost the same GDS curves, but FAMO achieved significantly better performance results than MGDA.

Feature Disentanglement. Figure 5 and Table 1 have shown that most methods applied parameter-level gradients achieved feature entangled-ness lower than baseline close to the end of training, with the exception for RGW, PCGrad, MGDA, and Aligned-MTL. These four methods form exactly the complement of the three methods that achieved performance gain among the gradient manipulation methods, as reported in Table 1. Nevertheless, clear decrease trends are displayed in PCGrad, MGDA, and Aligned-MTL. This provides strong evidence that the previous success in these methods can be attributed to learning disentangled features for down stream tasks.

5.5. Quantitative Results

When defining the ordering of training trajectories, we took the mean of the last 50 elements, which is the closest to the end of training. With RS, we are able to quantitatively report the faithfulness of the MTL measures. Results on three datasets are summarized in Tables 5, 6, and 7. Results have shown that on MetaGraspNet [11] dataset, FD has lower ranking similarities for bounding box predictions compared to GDS or GMS, but consistently out-performs traditional GDS and GMS for visible and amodal mask predictions). On CityScapes [7], FD out-performed GMS and GDS on semantic segmentation and instance segmentation, and out-performed GMS on depth estimation. On NYU-v2, FD still out-performed GMS on depth estimation and normal estimation.

6. Conclusions

In this paper, we first conducted large-scale experiments with ResNet18 [14] backbone on CityScapes [7], NYU-v2 [45], a new large-sized and complex test ground, MetaGraspNet [11] dataset. We showed that GradDrop [6] and CosReg [47] were the best-performing methods on all

three datasets. Secondly, we compared each method with parameter-level gradients versus feature-level gradients and showed that only MGDA [42] and Aligned-MTL [43] which have theoretical guarantees achieved a performance gain, while others were either similar or had significant performance degrade. Lastly, we proposed a novel efficient method to identify the challenges in MTL, and showed its faithfulness for visible and amodal mask predictions on MetaGraspNet and for semantic and instance segmentation on CityScapes. We leave proposing improved measures based on shared representations as a future direction.

References

- [1] Milad Abdollahzadeh, Toubia Malekzadeh, and Ngai-Man Man Cheung. Revisit multimodal meta-learning through the lens of multi-task learning. *Advances in Neural Information Processing Systems*, 34:14632–14644, 2021. [2](#)
- [2] Idan Achituve, Haggai Maron, and Gal Chechik. Self-supervised learning for domain adaptation on point clouds. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 123–133, 2021. [2](#)
- [3] Seunghyeok Back, Joosoon Lee, Taewon Kim, Sangjun Noh, Raeyoung Kang, Seongho Bak, and Kyoobin Lee. Unseen object amodal instance segmentation via hierarchical occlusion modeling. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 5085–5092. IEEE, 2022. [4](#), [5](#)
- [4] Rich Caruana. Multitask learning. *Machine learning*, 28: 41–75, 1997. [2](#)
- [5] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International conference on machine learning*, pages 794–803. PMLR, 2018. [2](#), [3](#), [4](#), [5](#), [6](#)
- [6] Zhao Chen, Jiquan Ngiam, Yanping Huang, Thang Luong, Henrik Kretschmar, Yuning Chai, and Dragomir Anguelov. Just pick a sign: Optimizing deep multitask models with gradient sign dropout. *Advances in Neural Information Processing Systems*, 33:2039–2050, 2020. [2](#), [3](#), [4](#), [5](#), [6](#), [8](#)
- [7] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. [2](#), [4](#), [6](#), [8](#), [1](#)
- [8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. [1](#)
- [9] Nikita Dvornik, Konstantin Shmelkov, Julien Mairal, and Cordelia Schmid. Blitznet: A real-time deep network for scene understanding. In *Proceedings of the IEEE international conference on computer vision*, pages 4154–4162, 2017. [2](#)
- [10] Christopher Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. Measuring and harnessing transference in multi-task learning. *arXiv preprint arXiv:2010.15413*, 2020. [3](#)
- [11] Maximilian Gilles, Yuhao Chen, Emily Zhixuan Zeng, Yifan Wu, Kai Furmans, Alexander Wong, and Rania Rayyes. Metagraspnetv2: All-in-one dataset enabling fast and reliable robotic bin picking via object relationship reasoning and dexterous grasping. *IEEE Transactions on Automation Science and Engineering*, pages 1–19, 2023. [2](#), [4](#), [5](#), [6](#), [7](#), [8](#), [1](#)
- [12] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. [2](#)
- [13] 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)*, pages 270–287, 2018. [3](#)
- [14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. [2](#), [4](#), [8](#), [1](#)
- [15] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. [2](#)
- [16] 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 AAAI Conference on Artificial Intelligence*, pages 3796–3803, 2019. [2](#)
- [17] Geethu Miriam Jacob, Vishal Agarwal, and Björn Stenger. Online knowledge distillation for multi-task learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2359–2368, 2023. [3](#), [4](#)
- [18] Adrián Javaloy and Isabel Valera. Rotograd: Gradient homogenization in multitask learning. *arXiv preprint arXiv:2103.02631*, 2021. [2](#), [4](#), [5](#)
- [19] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7482–7491, 2018. [2](#), [3](#), [4](#), [6](#)
- [20] Iasonas Kokkinos. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6129–6138, 2017. [2](#)
- [21] M Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. *Advances in neural information processing systems*, 23, 2010. [2](#)
- [22] Changsheng Li, Junchi Yan, Fan Wei, Weishan Dong, Qingshan Liu, and Hongyuan Zha. Self-paced multi-task learning. In *Proceedings of the AAAI conference on artificial intelligence*, 2017. [2](#)
- [23] Baijiong Lin, Feiyang Ye, Yu Zhang, and Ivor W Tsang. Reasonable effectiveness of random weighting: A litmus test for multi-task learning. *arXiv preprint arXiv:2111.10603*, 2021. [2](#), [3](#), [4](#), [5](#), [6](#), [7](#)

- [24] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 4
- [25] Xi Lin, Hui-Ling Zhen, Zhenhua Li, Qing-Fu Zhang, and Sam Kwong. Pareto multi-task learning. *Advances in neural information processing systems*, 32, 2019. 2, 3
- [26] Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multi-task learning. *Advances in Neural Information Processing Systems*, 34:18878–18890, 2021. 2, 3, 4, 5, 6, 7
- [27] Bo Liu, Yihao Feng, Peter Stone, and Qiang Liu. Famo: Fast adaptive multitask optimization. *arXiv preprint arXiv:2306.03792*, 2023. 2, 3, 4, 6
- [28] Liyang Liu, Yi Li, Zhanghui Kuang, J Xue, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. Towards impartial multi-task learning. *iclr*, 2021. 2, 3, 4, 5, 6, 7
- [29] Shikun Liu, Andrew Davison, and Edward Johns. Self-supervised generalisation with meta auxiliary learning. *Advances in Neural Information Processing Systems*, 32, 2019. 2
- [30] Shikun Liu, Edward Johns, and Andrew J Davison. End-to-end multi-task learning with attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1871–1880, 2019. 3, 4, 5, 6
- [31] Aviv Navon, Idan Achituve, Haggai Maron, Gal Chechik, and Ethan Fetaya. Auxiliary learning by implicit differentiation. *arXiv preprint arXiv:2007.02693*, 2020. 2
- [32] Aviv Navon, Aviv Shamsian, Idan Achituve, Haggai Maron, Kenji Kawaguchi, Gal Chechik, and Ethan Fetaya. Multi-task learning as a bargaining game. *arXiv preprint arXiv:2202.01017*, 2022. 2, 3, 4, 5, 6
- [33] Davy Neven, Bert De Brabandere, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Fast scene understanding for autonomous driving. *arXiv preprint arXiv:1708.02550*, 2017. 2
- [34] Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. Actor-mimic: Deep multitask and transfer reinforcement learning. *arXiv preprint arXiv:1511.06342*, 2015. 2
- [35] Lerrel Pinto and Abhinav Gupta. Learning to push by grasping: Using multiple tasks for effective learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 2161–2168. IEEE, 2017. 2
- [36] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. 5
- [37] Ethan M Rudd, Manuel Günther, and Terrance E Boulton. Moon: A mixed objective optimization network for the recognition of facial attributes. In *Computer Vision—ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part V 14*, pages 19–35. Springer, 2016. 3
- [38] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017. 2
- [39] Andrei A Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. Policy distillation. *arXiv preprint arXiv:1511.06295*, 2015. 2
- [40] Tom Schaul, Diana Borsa, Joseph Modayil, and Razvan Pascanu. Ray interference: a source of plateaus in deep reinforcement learning. *arXiv preprint arXiv:1904.11455*, 2019. 2
- [41] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. 4, 6
- [42] Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. *Advances in neural information processing systems*, 31, 2018. 2, 3, 4, 5, 6, 7, 9
- [43] Dmitry Senushkin, Nikolay Patakin, Arseny Kuznetsov, and Anton Konushin. Independent component alignment for multi-task learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20083–20093, 2023. 2, 3, 4, 5, 6, 7, 9
- [44] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013. 2
- [45] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgb-d images. In *Computer Vision—ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7–13, 2012, Proceedings, Part V 12*, pages 746–760. Springer, 2012. 2, 4, 5, 6, 8, 1
- [46] Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Which tasks should be learned together in multi-task learning? In *International Conference on Machine Learning*, pages 9120–9132. PMLR, 2020. 2
- [47] Mihai Suteu and Yike Guo. Regularizing deep multi-task networks using orthogonal gradients. *arXiv preprint arXiv:1912.06844*, 2019. 2, 4, 5, 6, 7, 8
- [48] Yee Teh, Victor Bapst, Wojciech M Czarnecki, John Quan, James Kirkpatrick, Raia Hadsell, Nicolas Heess, and Razvan Pascanu. Distral: Robust multitask reinforcement learning. *Advances in neural information processing systems*, 30, 2017. 2
- [49] Zirui Wang, Zihang Dai, Barnabás Póczos, and Jaime Carbonell. Characterizing and avoiding negative transfer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11293–11302, 2019. 2
- [50] Zirui Wang, Zachary C Lipton, and Yulia Tsvetkov. On negative interference in multilingual models: Findings and a meta-learning treatment. *arXiv preprint arXiv:2010.03017*, 2020. 2
- [51] Zirui Wang, Yulia Tsvetkov, Orhan Firat, and Yuan Cao. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. *arXiv preprint arXiv:2010.05874*, 2020. 3, 4, 5, 6, 7

- [52] Alexander Wong, Yifan Wu, Saad Abbasi, Saejith Nair, Yuhao Chen, and Mohammad Javad Shafiee. Fast graspnext: A fast self-attention neural network architecture for multi-task learning in computer vision tasks for robotic grasping on the edge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2292–2296, 2023. [4](#), [5](#)
- [53] Dan Xu, Wanli Ouyang, Xiaogang Wang, and Nicu Sebe. Pad-net: Multi-tasks guided prediction-and-distillation network for simultaneous depth estimation and scene parsing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 675–684, 2018. [2](#)
- [54] Feiyang Ye, Baijiong Lin, Zhixiong Yue, Pengxin Guo, Qiao Xiao, and Yu Zhang. Multi-objective meta learning. *Advances in Neural Information Processing Systems*, 34: 21338–21351, 2021. [3](#)
- [55] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33:5824–5836, 2020. [2](#), [3](#), [4](#), [5](#), [6](#), [7](#)
- [56] Hayoung Yun and Hanjoo Cho. Achievement-based training progress balancing for multi-task learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16935–16944, 2023. [2](#), [3](#)
- [57] Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3712–3722, 2018. [2](#)
- [58] Xiangyun Zhao, Haoxiang Li, Xiaohui Shen, Xiaodan Liang, and Ying Wu. A modulation module for multi-task learning with applications in image retrieval. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 401–416, 2018. [2](#)

Robust Analysis of Multi-Task Learning Efficiency: New Benchmarks on Light-Weighed Backbones and Effective Measurement of Multi-Task Learning Challenges by Feature Disentanglement

Supplementary Material

7. Experiment Setup for Benchmark Results

All experiments were done on a pool of GPUs including NVIDIA RTX 6000 Ada Generation, NVIDIA RTX A6000, and NVIDIA GeForce RTX 4090. Throughout, we fixed the backbone to be ResNet18 [14] with weights pretrained on ImageNet [8], and fixed the optimizer to be the PyTorch SGD optimizer with learning rate 1.0×10^{-4} and momentum 0.9. Linear warmup learning rate scheduler was also applied in all experiments. All images, including training, validation, and testing, are resized to 512×512 . We used batch size of 4 for experiments on MetaGraspNet [11] and batch size of 32 for experiments on CityScapes [7] and NYU-v2 [45].

8. Experiment Setup for Training Dynamics of MTL Measures

To save computation, we only compute these measures every 10 training iterations. With around $3k$ training iterations on the MetaGraspNet [11] dataset, we get around 300 data points on the trajectories.