IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2024.0429000

Rethinking Resource Competition in Multi-Task Learning: from Shared Parameters to Shared Representation

Dayou Mao[1], Yuhao Chen[2] (Member, IEEE), Yifan Wu[3], Maximilian Gilles[4], and Alexander Wong[5] (Member, IEEE)
[1]Vision and Image Processing Research Group, Waterloo, Canada (e-mail: daniel.mao@uwaterloo.ca)
[2]Vision and Image Processing Research Group, Waterloo, Canada (e-mail: yuhao.chen1@uwaterloo.ca)
[3]Vision and Image Processing Research Group, Waterloo, Canada (e-mail: yifan.wu1@uwaterloo.ca)
[4]Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany (e-mail: maximilian.gilles@kit.edu)
[5]Vision and Image Processing Research Group, Waterloo, Canada (e-mail: a28wong@uwaterloo.ca)

Corresponding author: Yuhao Chen (e-mail: yuhao.chen1@uwaterloo.ca).

ABSTRACT The core idea of Multi-Task Learning (MTL) is to develop neural networks with a shared feature extraction backbone and multiple prediction heads, each capable of inferring a different task simultaneously. Parameters in the backbone contribute to all tasks while those in the prediction heads contribute to only one or fewer tasks. Challenges arise when multiple tasks compete for resource. Existing methods focus on resource competition in shared parameters and proposed explanatory factors of task conflicts, task dominance, and gradient stability. However the fundamental nature of MTL is still understudied. In this paper, instead of following the existing methodology research directions, we carry out large-scale empirical study and provide deeper insight on understanding MTL. In particular, instead of focusing on resource competition in the shared parameters in the backbone, we shift our attention to resource competition in the backbone output, which is the embedded representation that is shared by all prediction heads. We show that the existing explanatory factors display weak causal relationship with model performance. We propose a novel measurement, which we term Feature Disentanglement, and show that understanding MTL problems from the perspective of how the shared representation is leveraged by different prediction heads, is a more faithful and reliable way than that from the perspective of how supervision signals from different tasks are interfering in the shared parameters. Additionally, it has been a commonly employed technique to replace gradients w.r.t. shared parameters with gradients w.r.t. shared representation for reduced computation. We conduct a comprehensive study and show that unless a theoretical analysis could be developed, there is not general guarantee that this fast approximation technique would work in practice.

INDEX TERMS Attention mechanism, computer vision, deep learning, explainable AI, machine learning, multi-objective optimization, multi-task learning, neural networks, representation, robotics.

I. INTRODUCTION

Multi-task learning (MTL) is a learning paradigm that arises from the need to develop machine learning systems capable of performing multiple tasks simultaneously. At a high-level, MTL systems comprises two key components: a single backbone and multiple prediction heads. The backbone encodes input data into a shared representation that is consumed by the prediction heads, and each prediction head learns to infer for one task from the extracted shared representation. 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-weighed. The merit of designing MTL systems is two-fold. Firstly, compared to developing a single-task model for each task, multi-task models significantly reduces overall model size because the backbone is shared, and hence during inference it brings faster speed, reduced memory footprint, and lower power consumption. Secondly, it is a common belief that the more complex the supervision signals, the richer the learned representation [1]–[3]. Introducing supervision signals from a diverse range of down stream tasks has been proven to be an effective approach to improve task performance of each other compared to training single-task learning systems [4]–[12]. The idea of MTL has already

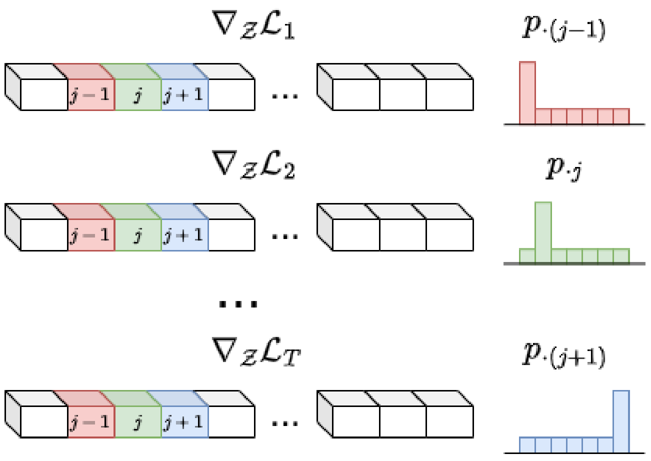
$\nabla_{\mathcal{Z}}\mathcal{L}_1$
$p_{\cdot(j-1)}$
$j-1$
j
$j+1$
$\nabla_{\mathcal{Z}}\mathcal{L}_2$
$p_{\cdot j}$
$\nabla_{\mathcal{Z}}\mathcal{L}_T$
$p_{\cdot(j+1)}$

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.

been exploited in classical discriminative computer vision algorithms well before it becomes an active research topic on its own [13]–[15].

Despite all the potential benefits MTL can bring, training a multi-task models is often more challenging, as observed in the fields of computer vision [11], [15]–[19], natural language processing [20], [21], meta learning [22], and reinforcement learning [23]–[27]. Challenges of MTL arise because it requires consideration of multiple objectives when deciding how parameters in the shared backbone should be updated. 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* [18]. It is widely accepted that this phenomenon can be explained by two factors: *task conflicts* and *task dominance* (Section III-A). However, empirically found that task conflicts and task dominance have weak relationship with actual performance.

From the computation perspective, many MTL methods [10], [28]–[31] involve computing gradients w.r.t. the shared parameters of all the loss functions, and hence back-propagation through the backbone is done $\mathcal{O}(T)$ times if T denotes the number of tasks. This procedure in training is extremely expensive and prior work [10], [32]–[34] have proposed to use feature-level gradients (gradients w.r.t. shared representation) to replace parameter-level gradients (gradients w.r.t. shared parameters) for more efficient computation, which we will address as the "Fast Gradient Surrogate" technique in this paper. However, some work [34] lack theoretical guarantee or solid empirical analysis to prove effectiveness of this surrogate and others [35] have reported poor results when doing so. While the huge saving in computation cost is appealing, this technique is unfortunately not generalizable to all gradient-based methods, as we will show in Section V.

In this paper, we take the innovative step to shift research focus from shared parameters to shared representation. We borrow ideas from the Explainable AI (XAI) literature and investigate how the shared representation is used by the prediction heads from both the bottom-up approach, which requires back-propagation through the prediction heads, and the top-down approach, which requires an additional attention module for each task. We convert the attributions into saliency maps and draw connection between the saliency maps and the model performance. Moreover, we implement the feature-level gradient counterparts of several MTL methods and empirically compare with the originally proposed methods. Our contribution is two-fold:

- We challenge the traditional assumptions that task conflicts and task dominance are reasons why MTL is harder and propose Feature Disentanglement measurement which can more faithfully and efficiently reveal problem nature.
- We provide solid empirical evidence and show that there is no guarantee in practice that the fast gradient surrogate technique will preserve model performance or even improve it, unless a theoretical analysis could be developed.

II. BACKGROUND AND PRIOR WORK

Notations and Terminologies. Throughout the paper, we follow 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. $f_\theta : \mathcal{X} \to \mathcal{Y}$ denotes a neural network parametrized by $\theta \in \Omega$ where Ω is the parameter space. $\theta^{\text{sh}} \in \mathbb{R}^d$ denotes shared parameters where d is the total number of shared parameters. $\mathcal{Z} \in \mathbb{R}^h$ denotes shared representation where h is the dimension of the embedded feature space. $\mathcal{L}_i : \mathcal{Y}_i \times \mathcal{Y}_i \to \mathbb{R}_{\geq 0}$ denotes the loss functions for the i-th task. $g_i := \nabla_{\theta^{\text{sh}}} \mathcal{L}_i \in \mathbb{R}^d$ denotes each task gradient, or "parameter-level gradients". $G := [g_1, ..., g_T] \in \mathbb{R}^{d \times T}$ denotes the gradient matrix whose columns are the task gradients. $\nabla_{\mathcal{Z}} \mathcal{L}_i$ are the "feature-level gradients". The data point at which these gradients are computed is omitted from the notations for simplicity and should be inferred from the context. For any natural number n, $[n] := \{1, ..., n\}$.

Problem Framework. In general, multi-task learning can be formulated as the following optimization problem:

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

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 $\mathcal{L} := (\mathcal{L}_1, ..., \mathcal{L}_T)^\top \in \mathbb{R}^T_{\geq 0}$ w.r.t. the partial-order

$\leq_K$ on $\mathbb{R}^T$ induced by the pointed, closed, and convex cone $K := \mathbb{R}^T_{\geq 0}$ [10], [35]–[37]. 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 $\mathcal{L}$.

Methodology Overview. MTL methods can be broadly divided into optimization methods and architecture design, and optimization methods can be further divided into gradient manipulation methods, gradient balancing methods, and gradient regularization methods. In this paper, we use well-studied MTL models and focus on their optimization methods. Before diving into the details of prior work, we make two remarks here on the differences and relationships between gradient manipulation methods (Section II-A) and gradient balancing methods (Section II-B). (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.

A. GRADIENT MANIPULATION

To update the shared parameters θ^{sh} taking all T tasks into consideration, gradient manipulation methods [10], [28]–[30], [32], [33], [35], [38], [39] first compute task gradients $g_i = \nabla_{\theta^{\text{sh}}}\mathcal{L}_i$ and propose different methods to compute coefficients $\alpha = (\alpha_1, ..., \alpha_T)^\top \in \mathbb{R}^T$ and set

$$\hat{g} := G\alpha = \sum_{i=1}^{T} \alpha_i g_i \in \mathbb{R}^d \tag{2}$$

as gradient for the shared parameters. 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 [28], [29], [33], [38] derive closed-form formulae for α based on some heuristics and may also rely on some stochasticity. PCGrad [28] proposed to reduce task conflicts by projecting the conflicting gradients onto the normal planes of each other, removing the conflicting component. GradVac [29] expanded this idea and proposed to encourage acute angles between gradients by maintaining an Exponential Moving Average (EMA) of cosine similarity between task gradients. GradDrop [33] propose to mask the gradients with Gradient Sign Purity so that the gradient signs are more aligned. Random Gradient Weighting (RGW) [38] proposed to randomize α based on Gaussian distribution.

Implicit Methods [10], [30], [32], [35], [39] hypothesize different objectives and compute the gradient weights α by solving either an optimization problem or a system of equations. MGDA [10] re-weighs task gradients so that the result is the norm minimizer in the convex hull enclosed by the task gradients. CAGrad [30] 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 [35] 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 [32] proposed to first approximate the gradient matrix G by the closest unitary matrix $\hat{G}$, which has stability number 1, so that the linear system $\hat{g} = G\alpha$ (Equation 2) is well-defined.

B. GRADIENT BALANCING

Gradient balancing methods [11], [31], [36], [38], [40]–[46], or loss balancing methods, on the other hand, aim to re-weigh task losses dynamically so that all tasks are learned at compatible pace. Let $w = (w_1, ..., w_T)^\top \in \mathbb{R}^T_{\geq 0}$ denote the task weights, then gradient balancing methods solve the following dynamically re-weighed problem:

$$\min_{\theta \in \Omega} \sum_{(x,y)\in\mathcal{D}} \sum_{i=1}^{T} w_i \mathcal{L}_i(f_\theta(x), y_i) \tag{3}$$

Note that this family of algorithms does not emphesize on the distinction between shared parameters and task-specific parameters. Computation of w can be done either explicitly or implicitly, or even by an extra optimizer.

Explicit Methods [31], [38], [40]–[43] compute w explicitly based on different heuristics. GradNorm [31] proposed to control the learning pace of different tasks based on the relative norm of the gradients g_i and does so by assigning task weights and updating them at each training iteration. Exploiting the same idea as RGW [38], the same paper also proposed Random Loss Weighting (RLW). In [43], 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 [36], [44] compute the loss weights w by solving an optimization subproblem at each training iteration. In particular, FAMO [44] proposed that the parameter update at each training step should maximize the lowest relative improvement of the task losses. The subtle difference from CAGrad [30] 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 [11], [45] dynamically update the loss weights with an extra set of objective and optimizer. This family of methods optimize the loss weights together with model parameters during training and hence no explicit computation or subproblems are needed.

C. GRADIENT REGULARIZATION

Gradient regularization methods [16], [34] design regularization terms in the loss function and solves the following optimization problem:

$$\min_{\theta \in \Omega} \sum_{(x,y)\in\mathcal{D}} \mathcal{L}(f_\theta(x), y) + \mathcal{L}_{\text{reg}}(G). \tag{4}$$

In particular, Suteu [16] hypothesized that orthogonal task gradients are beneficial for learning and added pairwise squared cosine similarity as the regularization term to encourage orthogonal columns in G. Javaloy [34] proposed a similar idea to regularize the gradients by minimizing the angles between each task gradient and the average gradient.

D. SALIENCY MAPS IN EXPLAINABLE AI

Explainable AI (XAI) seeks to explain behaviours of AI systems. Visual explanation method can be broadly categorized into bottom-up approaches and top-down approaches [47]. Bottom-up approaches rely on gradients of the network prediction w.r.t. intermediate activations [48]. The representative work, GradCAM [48], of bottom-up approaches proposed to compute saliency maps for the intermediate activations via the magnitude of gradients of class scores w.r.t. the activations. On the other hand, top-down approaches augment the network with a small attention module and train together with the original network [47]. We leverage both types of attribution technique, together with our proposed Feature Disentanglement measurement, to answer the question of "how the shared representation is used by the prediction heads". More details are explained in Section III.

E. 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. In contrast, 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 light-weighed compared to the backbone. Some works [10], [32] have 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. However, other works [34], [35] empirically investigated application of this feature-level surrogate but [34] obtained reasonable results while [35] observed poor results. We now empirically prove in this paper that the assumption on generalizability does not hold.

III. METHOD

Shared resource propose a challenge for MTL. Instead of focusing on the fact that *parameters* are shared by multiple tasks, we shift our attention to the fact that extracted image *features* are shared by the prediction heads. 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.

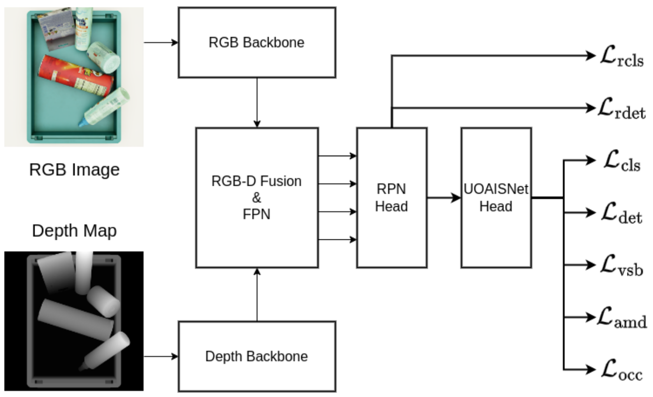
RGB Image
Depth Map
RGB Backbone
RGB-D Fusion & FPN
Depth Backbone
RPN Head
UOAISNet Head
$\mathcal{L}_{\text{rcls}}$
$\mathcal{L}_{\text{rdet}}$
$\mathcal{L}_{\text{cls}}$
$\mathcal{L}_{\text{det}}$
$\mathcal{L}_{\text{vsb}}$
$\mathcal{L}_{\text{amd}}$
$\mathcal{L}_{\text{occ}}$

FIGURE 2. Illustration of RGB-D inputs and model architecture used on the MetaGraspNet [49] benchmark.

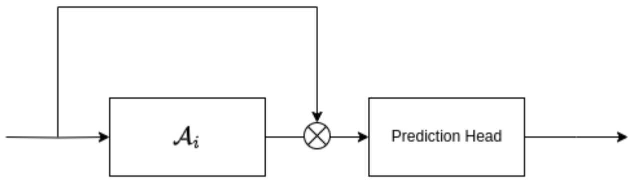

FIGURE 3. Illustration of attention-augmented prediction heads for top-down attribution. The ⊗ module denotes element-wise multiplication. A separate attention module is created for each prediction head. All *T* attention-augmented prediction heads share the extracted feature from the backbone, as in the original architecture design.

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 MTL problem nature by answering the following research question: *what is the best way for the prediction heads to share the embedded representation?*

A. 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 in our experiments.

Task Conflicts. We follow [16], [28], [29] and define the Gradient Direction Similarity (GDS) measure for T tasks as

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

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

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.

	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 [38]	0.358 (↓)	0.513 (↓)	0.490 (↓)	0.644 (–)	0.462 (↓)	0.614 (–)
PCGrad [28]	0.370 (↓)	0.526 (↑)	0.500 (↓)	0.657 (↑)	0.454 (↓)	0.595 (↓)
GradVac [29]	0.393 (↑)	**0.560** (↑)	0.521 (–)	**0.680** (↑)	0.495 (–)	**0.654** (↑)
MGDA [10]	0.371 (↓)	0.531 (↑)	0.452 (↓)	0.594 (↓)	0.430 (↓)	0.564 (↓)
CAGrad [30]	**0.411** (↑)	0.557 (↑)	0.522 (–)	0.648 (–)	0.488 (–)	0.604 (↓)
GradDrop [33]	0.399 (↑)	0.541 (↑)	0.533 (↑)	0.666 (↑)	0.505 (↑)	0.634 (↑)
Aligned-MTL [32]	0.400 (↑)	0.547 (↑)	0.477 (↓)	0.610 (↓)	0.460 (↓)	0.580 (↓)
IMTL [42]	0.410 (↑)	0.534 (↑)	**0.550** (↑)	0.667 (↑)	**0.541** (↑)	**0.658** (↑)
RLW [38]	0.360 (↓)	0.504 (↓)	0.499 (↓)	0.649 (–)	0.466 (↓)	0.614 (–)
DWA [43]	0.390 (↑)	0.533 (↑)	**0.533** (↑)	0.664 (↑)	0.499 (↑)	0.628 (↑)
Uncertainty [11]	0.206 (↓)	0.349 (↓)	0.345 (↓)	0.507 (↓)	0.319 (↓)	0.482 (↓)
FAMO [44]	**0.431** (↑)	**0.564** (↑)	0.517 (–)	0.623 (↓)	0.510 (↑)	0.613 (–)
CosReg [16]	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 [49] 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.

	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 2. Ranking similarity results on MetaGraspNet [49] dataset.

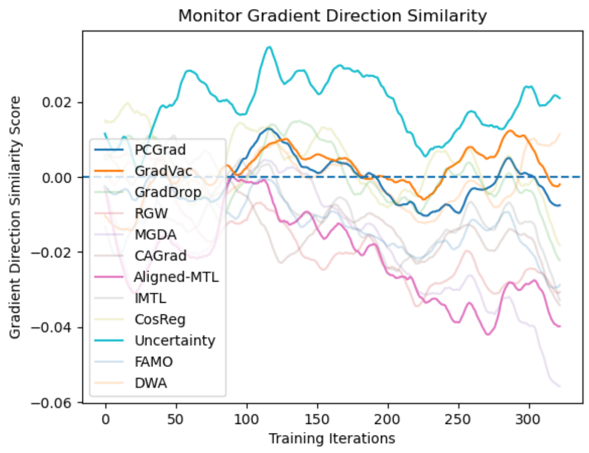

FIGURE 4. Training dynamics of GDS using gradients w.r.t. shared parameters.

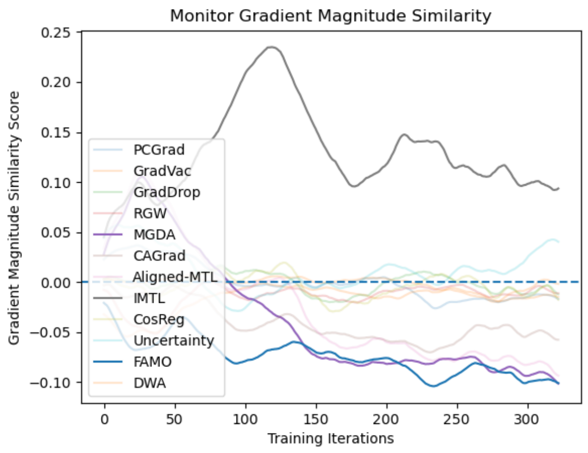

FIGURE 5. Training dynamics of GMS using gradients w.r.t. shared parameters.

Task Dominance. We follow [28] and define the Gradient Magnitude Similarity (GMS) measure for T tasks as

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

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

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.

B. BOTTOM-UP AND TOP-DOWN ATTRIBUTION

We leverage both bottom-up and top-down approaches to investigate how the shared representation is used by the prediction heads. We introduce the details of how these attributions are defined in this section.

Bottom-up. Inspired by GradCAM [48], we use the feature-level gradient $\nabla_{\mathcal{Z}}\mathcal{L}_i$ to measure how the shared representation $\mathcal{Z}$ contributes to the prediction of each task i. This computation could be treated as a light-weighed add-on to the evaluation procedure and does not change the model itself or its performance.

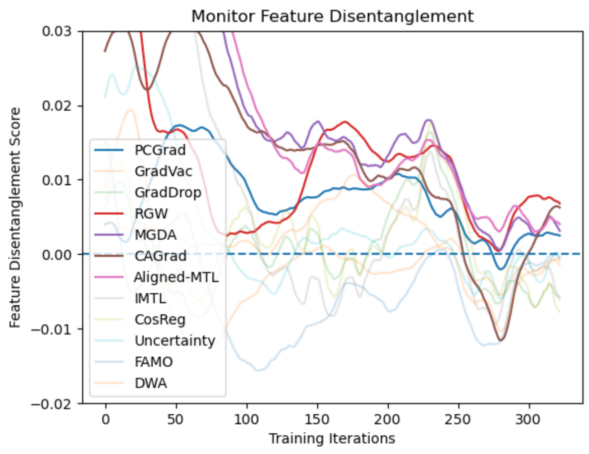

FIGURE 6. **Training dynamics of Feature Disentanglement (FD) using gradients w.r.t. shared parameters.**

Top-down. We follow the XAI literature on attention mechanisms for visual explanation [47] and design add-ons for the model architecture. Mathematically, if $\mathcal{E}$ is the backbone and $\mathcal{H}_i$ is the prediction head for task i, then the original network f can be written as:

$$f(x) = [\mathcal{H}_i(\mathcal{E}(x)), ..., \mathcal{H}_T(\mathcal{E}(x))]_{i=1}^T. \tag{7}$$

With the additional attention modules $\mathcal{A}_i$, the new network $f_\mathcal{A}$ can be written as:

$$f_\mathcal{A}(x) := [\mathcal{H}_i(\mathcal{A}_i(\mathcal{E}(x)) \circ \mathcal{E}(x))]_{i=1}^T \tag{8}$$

where $\circ$ denotes element-wise multiplication of two equal-shaped tensors. During inference, we use the tensor $\mathcal{A}_i(\mathcal{E}(x))$ as the explanatory tensor of how the shared representation $\mathcal{Z} = \mathcal{E}(x)$ is used by the prediction heads. An illustrative figure is shown in Figure 3. In our experiments, we use a 3x3 conv layer followed by a depthwise separable conv layer as the attention module $\mathcal{A}_i$.

C. FEATURE DISENTANGLEMENT MEASURE

With the task attribution, or saliency maps $\mathcal{M}_i$, either obtained by bottom-up approach $\mathcal{M}_i := \nabla_\mathcal{Z}\mathcal{L}_i$ or top-down approach $\mathcal{M}_i := \mathcal{A}_i(\mathcal{E}(x))$, we now develop a novel quantitative measurement of how challenging the MTL problem of interest is.

Definition. At location $j \in [h]$, we can quantify the entropy $\mathcal{S}$ of the saliencies across T tasks by

$$\mathcal{S}_j(\mathcal{Z}) := -\sum_{i=1}^T p_{ij} \log p_{ij}, \text{ where} \tag{9a}$$

$$p_{ij} := |\mathcal{M}_{ij}| / \sum_{k=1}^T |\mathcal{M}_{ik}|, \text{ for } i \in [T]. \tag{9b}$$

The *feature disentanglement* (FD) measure for the entire shared representation $\mathcal{Z}$ is defined to be the average entropy across all positions:

$$\text{FD} := \mathcal{S}(\mathcal{Z}) := \frac{1}{h}\sum_{j=1}^h \mathcal{S}_j(\mathcal{Z}), \tag{10}$$

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 either only back propagates through the T prediction heads (bottom-up) or does not compute back propagation at all (top-down). An illustration of this definition is shown in Figure 1.

D. EVALUATION PROTOCOL: RANKING SIMILARITY

To quantitatively evaluate the faithfulness of different measures (GDS, GMS, gradient stability [32], 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 $A := \{a_1, ..., a_n\} \subset \mathbb{R}$ and two rankings $R_1, R_2 : A \to [n]$ of elements of A, we define the *ranking similarity* $\mathcal{S}(R_1, R_2) \in \mathbb{R}$ between R_1 and R_2 to be the following average:

$$\mathcal{S}(R_1, R_2) := \frac{1}{n(n-1)} \sum_{\substack{i,j=1 \\ i \neq j}}^n \mathbb{I}\Big[R_1 \text{ and } R_2 \text{ agree on } a_i \text{ and } a_j\Big], \tag{11}$$

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 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. i.e., $R_2'(a) = n + 1 - R_2(a)$. Then R_1 and R_2 agree on a pair $(a_i, a_j) \in A \times A$ 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 rankings are equivalent to their reverse, we can always reverse a ranking when making comparisons. So we only consider $|\mathcal{S}(R_1, R_2) - 0.5|$ in our experiments. The larger the absolute value, the closer R_1 and R_2 align. We only report this symmetric version between test-time performance scores and MTL challenge measures in the following section.

IV. EXPERIMENTS

A. SETUP

Datasets. We carry out our stud on three benchmark datasets: MetaGraspNet [49], CityScapes [50], and NYU-v2 [51]. We provide a new test ground for multi-task learning on the MetaGraspNet dataset [49], due to its significantly larger

	DE score	DE rank	SS score	SS rank	IS score	IS rank
Baseline	**1.134** (-)	3	17.7% (-)	5	0.034 (-)	4
RGW [38]	1.160 (↓)	6	16.8% (↓)	14	0.046 (↓)	15
PCGrad [28]	1.170 (↓)	9	17.0% (↓)	8	0.046 (↓)	14
GradVac [29]	1.169 (↓)	7	16.8% (↓)	12	0.044 (↓)	9
MGDA [10]	1.215 (↓)	14	17.0% (↓)	9	0.045 (↓)	12
CAGrad [30]	1.169 (↓)	8	17.1% (↓)	7	0.042 (↓)	8
GradDrop [33]	**1.101** (↑)	2	17.6% (-)	6	0.045 (↓)	10
Nash-MTL [35]	1.177 (↓)	11	16.8% (↓)	11	0.045 (↓)	11
Aligned-MTL [32]	1.207 (↓)	13	16.8% (↓)	13	0.045 (↓)	13
IMTL [42]	1.174 (↓)	10	16.8% (↓)	10	0.037 (↓)	6
RLW [38]	1.178 (↓)	12	**18.5%** (↑)	2	0.037 (↓)	7
DWA [43]	1.135 (-)	4	**17.8%** (-)	3	**0.034** (↑)	3
Uncertainty [11]	8.987 (↓)	15	9.5% (↓)	15	0.081 (↓)	16
GradNorm [31]	1.142 (-)	5	17.8% (-)	4	0.036 (↓)	5
FAMO [44]	15.048 (↓)	16	1.3% (↓)	16	**0.026** (↑)	2
CosReg [16]	**1.067** (↑)	1	**20.9%** (↑)	1	**0.023** (↑)	1

TABLE 3. Benchmark results with ResNet18 backbone on CityScapes [50]. 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.

	DE score	DE rank	SS score	SS rank	IS score	IS rank
Baseline	0.788 (-)	7	**18.8%** (-)	2	1.340 (-)	7
RGW [38]	1.064 (↓)	8	18.2% (↓)	4	0.029 (↑)	5
PCGrad [28]	0.000 (↑)	4	17.6% (↓)	8	3.973 (↓)	10
GradVac [29]	**0.000** (↑)	2	17.5% (↓)	10	3.971 (↓)	9
MGDA [10]	1.317 (↓)	13	18.0% (↓)	5	**0.028** (↑)	3
CAGrad [30]	**0.000** (↑)	3	17.6% (↓)	9	3.969 (↓)	8
Nash-MTL [35]	1.238 (↓)	11	17.7% (↓)	7	0.028 (↑)	4
Aligned-MTL [32]	1.302 (↓)	12	18.0% (↓)	6	0.031 (↑)	6
IMTL [42]	**0.000** (↑)	1	17.5% (↓)	11	4.002 (↓)	11
RLW [38]	1.125 (↓)	9	**20.1%** (↑)	1	**0.026** (↑)	2
DWA [43]	1.153 (↓)	10	**18.6%** (↓)	3	**0.025** (↑)	1
Uncertainty [11]	0.058 (↑)	6	8.3% (↓)	12	4.068 (↓)	13
FAMO [44]	0.000 (↑)	5	1.1% (↓)	13	4.062 (↓)	12

TABLE 4. Benchmark results with ResNet50 backbone on CityScapes [50]. 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.

	DE score	DE rank	SS score	SS rank	IS score	IS rank
Baseline	**0.000** (-)	3	**17.6%** (-)	2	3.971 (-)	4
GradVac [29]	0.003 (↓)	8	15.8% (↓)	5	3.982 (-)	7
MGDA [10]	0.001 (↓)	4	16.0% (↓)	4	3.973 (-)	5
Nash-MTL [35]	0.001 (↓)	5	15.4% (↓)	7	**3.969** (-)	2
Aligned-MTL [32]	0.001 (↓)	6	15.5% (↓)	6	3.978 (-)	6
RLW [38]	0.001 (↓)	7	**18.0%** (↑)	1	**3.863** (↑)	1
DWA [43]	**0.000** (↑)	2	**17.5%** (-)	3	**3.969** (-)	3
Uncertainty [11]	0.064 (↓)	9	8.0% (↓)	8	4.127 (↓)	9
FAMO [44]	**0.000** (↑)	1	1.0% (↓)	9	4.111 (↓)	8

TABLE 5. Benchmark results with ResNet18 backbone and attention modules on CityScapes [50]. 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.

	DE	SS	IS
GMS	58.8%	63.6%	61.0%
GDS	62.6%	65.1%	75.4%
Grad Stability	**64.0%**	**65.4%**	76.1%
FD	57.4%	64.0%	**80.1%**

TABLE 6. Ranking similarity results with ResNet18 backbone on CityScapes [50] using bottom-up approach.

	DE	SS	IS
GMS	68.6%	67.8%	63.1%
GDS	83.9%	55.2%	52.1%
Grad Stability	**84.0%**	54.9%	51.8%
FD	58.0%	**74.4%**	**63.6%**

TABLE 7. Ranking similarity results with ResNet18 backbone on CityScapes [50] using top-down approach.

dataset size (∼12k training examples), increased task complexity, and higher real-world value. We refer the readers to Section IV-B for details of network architecture. For semantic segmentation, we consider all 19 classes in CityScapes and all 41 classes in NYU-v2. For solidness of our results, we do not follow the simplified coarse class semantic segmentation

	DE score	DE rank	NE score	NE rank	SS score	SS rank
Baseline	0.747 (-)	4	42.323 (-)	11	**4.8%** (-)	3
PCGrad [28]	0.782 (↓)	9	42.105 (-)	10	4.6% (↓)	8
GradVac [29]	0.782 (↓)	8	41.932 (-)	9	4.7% (-)	4
MGDA [10]	0.849 (↓)	14	41.604 (↑)	5	4.1% (↓)	13
CAGrad [30]	0.779 (↓)	7	**40.479** (↑)	3	4.6% (↓)	7
GradDrop [33]	**0.731** (↑)	2	41.623 (↑)	6	**5.5%** (↑)	2
Nash-MTL [35]	0.808 (↓)	12	41.692 (↑)	7	4.4% (↓)	10
Aligned-MTL [32]	0.836 (↓)	13	43.043 (↓)	13	4.2% (↓)	12
IMTL [42]	0.806 (↓)	11	41.879 (↑)	8	4.4% (↓)	9
RLW [38]	0.794 (↓)	10	41.244 (↑)	4	4.3% (↓)	11
DWA [43]	0.760 (↓)	6	42.710 (-)	12	4.7% (↓)	6
Uncertainty [11]	9.083 (↓)	15	45.334 (↓)	15	0.3% (↓)	15
GradNorm [31]	0.760 (↓)	5	43.581 (↓)	14	4.7% (↓)	5
FAMO [44]	**0.742** (-)	3	**35.479** (↑)	1	0.5% (↓)	14
CosReg [16]	**0.703** (↑)	1	**39.241** (↑)	2	**7.8%** (↑)	1

TABLE 8. Benchmark results with ResNet18 backbone on NYU-v2 [51]. 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.

	DE score	DE rank	NE score	NE rank	SS score	SS rank
Baseline	5.534 (-)	5	nan (-)	3	2.4% (-)	6
RGW [38]	8.233 (↓)	9	nan (-)	4	0.9% (↓)	12
PCGrad [28]	6.637 (↓)	7	nan (-)	5	2.2% (↓)	7
GradVac [29]	8.908 (↓)	11	nan (-)	6	1.2% (↓)	8
MGDA [10]	**0.899** (↑)	1	43.050 (-)	1	**4.6%** (↑)	1
CAGrad [30]	4.173 (↑)	4	nan (-)	7	3.1% (↑)	4
Aligned-MTL [32]	8.013 (↓)	8	nan (-)	8	1.1% (↓)	9
IMTL [42]	**2.953** (↑)	2	nan (-)	9	**3.6%** (↑)	3
RLW [38]	**3.712** (↑)	3	nan (-)	10	**3.7%** (↑)	2
DWA [43]	6.165 (↓)	6	nan (-)	11	2.5% (↑)	5
Uncertainty [11]	8.718 (↓)	10	47.429 (-)	2	1.1% (↓)	10
GradNorm [31]	9.229 (↓)	12	nan (-)	12	1.0% (↓)	11

TABLE 9. Benchmark results with ResNet50 backbone on NYU-v2 [51].

	DE score	DE rank	NE score	NE rank	SS score	SS rank
Baseline	**3.212** (-)	2	nan (-)	2	**3.7%** (-)	2
RGW [38]	**5.751** (↓)	3	nan (-)	3	**2.3%** (↓)	3
PCGrad [28]	7.689 (↓)	6	nan (-)	4	1.2% (↓)	6
GradVac [29]	**0.885** (↑)	1	42.581 (-)	1	**4.8%** (↑)	1
RLW [38]	6.842 (↓)	5	nan (-)	5	2.1% (↓)	5
DWA [43]	6.322 (↓)	4	nan (-)	6	2.3% (↓)	4

TABLE 10. Benchmark results with ResNet18 backbone and attention modules on NYU-v2 [51]. 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.

	DE	NE	SS
GMS	56.8%	61.8%	62.9%
GDS	63.5%	64.9%	**68.8%**
Grad Stability	**66.7%**	68.0%	68.2%
FD	60.5%	**69.5%**	60.7%

TABLE 11. Ranking similarity results with ResNet18 backbone on NYU-v2 [51] using bottom-up approach.

	DE	NE	SS
GMS	**65.3%**	60.0%	**65.3%**
GDS	60.5%	64.6%	60.5%
Grad Stability	62.7%	**66.7%**	62.7%
FD	62.7%	54.4%	62.7%

TABLE 12. Ranking similarity results with ResNet18 backbone on NYU-v2 [51] using top-down approach.

setup as in [28], [38].

Methods. We studied 15 MTL optimization algorithms, from three categories: (1) we selected PCGrad [28], GradVac [29], GradDrop [33], RGW [38], MGDA [10], CAGrad [30], Nash-MTL [35], and Aligned-MTL [32] from the gradient manipulation category (8 in total), (2) we selected Uncertainty [11], GradNorm [31], IMTL [42], FAMO [44], RLW [38], and DWA [43] from the gradient balancing category (6 in total), and (3) CosReg [16] 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 V.

Training. 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 A for more details.

B. PRELIMINARY STUDY: METAGRASPNET DATASET

Network Architecture. We follow [52], [53] and utilize two ResNet [54] 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) [55] to fuse the features from different levels. On top of this extracted feature from the FPN neck we attach Region Proposal Network (RPN) [56] and UOAISNet [52] 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 [53] 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 $\mathcal{L}_{\text{cls}}$, and $\mathcal{L}_{\text{det}}$ by the amodal detection head; $\mathcal{L}_{\text{vsb}}$ for visible object masks prediction, $\mathcal{L}_{\text{amd}}$ for amodal object masks prediction, and $\mathcal{L}_{\text{occ}}$ for object occlusion prediction. We set the baseline loss function to be

$$\mathcal{L}_{\text{tot}} := \mathcal{L}_{\text{rcls}} + \mathcal{L}_{\text{rdet}} + \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{det}} + \mathcal{L}_{\text{vsb}} + \mathcal{L}_{\text{amd}} + \mathcal{L}_{\text{occ}}, \quad (12)$$

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 [31] and Nash-MTL [35], and the feature-level gradient counterpart of GradDrop [33], which is what originally proposed, so these methods are not reported on MetaGraspNet dataset. Results have shown that GradVac [29], GradDrop [33], IMTL [42], DWA [43], MGDA-UB [10], (rep) CAGrad [30], and (rep) CosReg [16] achieved consistent performance gain compared to the baseline on all 6 evaluation metrics. GradVac [29], IMTL [42], and (rep) MGDA achieved top-three performance under a majority (≥ 3) of the metrics. Full results on MetaGraspNet are summarized in Table 1.

We show interesting examples and qualitatively demonstrate the effectiveness of our method from the training dynamics on MetaGraspNet [49] dataset. Training dynamics of GDS, GMS, and feature disentanglement scores are plotted in Figures 4, 5, and 6, respectively. For clarity, we only plot the relative values to the baseline method. All curves are smoothened 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. 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. See Appendix B for more details on how trajectories were plotted.

Task Conflicts and Task Dominance. Figure 4 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 apart in the GDS plot. Figure 5 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 6 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.

Quantitative results are shown in Table 2. Results have shown that on MetaGraspNet [49] 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).

C. EXPERIMENTS ON CITYSCAPES

Full benchmark results on CityScapes are summarized in Tables 3, 4, and 5. Ranking similarity scores are summarized in Tables 6 and 7. Results have shown that feature disentanglement measure our-performed GMS, GDS, and gradient stability for instance segmentation task when using the bottom-up approach and for both semantic and instance segmentation when using the top-down approach.

D. EXPERIMENTS ON NYU-V2

Full benchmark results on NYU-v2 [51] are summarized in Tables 8, 9, and 10. Ranking similarity scores are shown in Tables 11 and 12. Results have shown that feature disentanglement measure out-performed GMS for the depth estimation task and performed the best for normal estimation, when using bottom-up approach. On the other hand, when using top-down approach, FD only out-performed GDS for depth estimation.

V. GENERALIZABILITY OF FAST GRADIENT SURROGATE

We now detail our second main contribution of this paper: we show that the fast gradient technique is not applicable in general. 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 [10], [32], [34]. While it might be reasonable to do so due to chain rule and

	BBox mAP	BBox mAR	VMask mAP	VMask mAR	AMask mAP	AMask mAR
RGW [38]	-6.4%	-3.6%	-6.9%	-4.2%	-5.9%	-3.4%
PCGrad [28]	-1.4%	-0.7%	0.7%	-0.2%	3.2%	3.0%
GradVac [29]	-13.9%	-7.8%	-11.2%	-7.6%	-14.2%	-10.6%
MGDA [10]	14.3%	6.8%	17.8%	12.0%	19.8%	14.1%
CAGrad [30]	-0.9%	-2.9%	2.9%	2.6%	5.1%	4.8%
Aligned-MTL [32]	2.5%	1.3%	8.1%	5.9%	8.3%	7.7%
IMTL [42]	-7.4%	-1.5%	-8.2%	-1.6%	-8.1%	-1.1%
CosReg [16]	0.2%	-2.3%	1.5%	0.3%	0.8%	-0.4%

TABLE 13. 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.

sub-additivity of norms [10], [32], other methods [35] 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 $\nabla_{\theta^{\mathrm{sh}}}\mathcal{L}_i$ versus $\nabla_{\mathcal{Z}}\mathcal{L}_i$ on 8 optimization algorithms on the MetaGraspNet [49] dataset. Full results are summarized in Table 13. Results have displayed the following: Firstly, only MGDA [10] and Aligned-MTL [32] 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 [29], RGW [38], and IMTL [42] 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.

VI. CONCLUSIONS

In this paper, we shift our attention from the traditional studies on task interference to investigating resource competition in the shared representation. Through bottom-up and top-down approaches inspired by the XAI literature, we construct per-task saliency maps from either the loss gradients w.r.t. shared representation or the attention module output, which serves as a gating mechanism on the shared representation. These saliency maps summarized by our novel feature disentanglement measurement as an explanatory factor for the causal relationship between MTL problem nature and model performance. Through large-scale comprehensive empirical study, we have shown the effectiveness of feature disentanglement measure compared to traditional task conflicts, task dominance, and gradient stability on three benchmark datasets: MetaGraspNet, CityScapes, and NYU-v2. Additionally, we also provide empirical evidence to disprove the generalizability of the fast gradient surrogate technique on MetaGraspNet. This shows that the prior work on gradient-based methods do rely on the assumption that resource are competing by the multiple tasks in the shared parameters, and is not directly translatable to resource competition in the latent space where the shared representation lies. The proposed novel perspectives and reported experimental results provide the community with deeper insight on the fundamental nature of MTL problems.

APPENDIX A 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 use ResNet architectures with pretrained weights on ImageNet [57], and fixed the optimizer to be the PyTorch SGD optimizer with learning rate 1.0×10^{-4} and momentum 0.9. Linear warm-up 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 [49] and batch size of 32 for experiments on CityScapes [50] and NYU-v2 [51].

APPENDIX B EXPERIMENT SETUP FOR TRAINING DYNAMICS OF MTL MEASURES

For preliminary studies on the MetaGraspNet [49] dataset, we computed the task conflicts and other measures during training. To save computation, we only compute these measures every 10 training iterations. With around $3k$ training iterations on the MetaGraspNet [49] dataset, we get around 300 data points on the trajectories. For experiments on CityScapes and NYU-v2 datasets, these measures are computed during validation epochs, together with the model performance metrics.

REFERENCES

[1] S. Liu, A. Davison, and E. Johns, "Self-supervised generalisation with meta auxiliary learning," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[2] I. Achituve, H. Maron, and G. Chechik, "Self-supervised learning for domain adaptation on point clouds," in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2021, pp. 123–133.

[3] A. Navon, I. Achituve, H. Maron, G. Chechik, and E. Fetaya, "Auxiliary learning by implicit differentiation," *arXiv preprint arXiv:2007.02693*, 2020.

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

[5] A. R. Zamir, A. Sax, W. Shen, L. J. Guibas, J. Malik, and S. Savarese, "Taskonomy: Disentangling task transfer learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3712–3722.

[6] R. Caruana, "Multitask learning," *Machine learning*, vol. 28, pp. 41–75, 1997.

[7] D. Neven, B. De Brabandere, S. Georgoulis, M. Proesmans, and L. Van Gool, "Fast scene understanding for autonomous driving," *arXiv preprint arXiv:1708.02550*, 2017.

[8] N. Dvornik, K. Shmelkov, J. Mairal, and C. Schmid, "Blitznet: A real-time deep network for scene understanding," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 4154–4162.

[9] D. Xu, W. Ouyang, X. Wang, and N. 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*, 2018, pp. 675–684.

[10] O. Sener and V. Koltun, "Multi-task learning as multi-objective optimization," *Advances in neural information processing systems*, vol. 31, 2018.

[11] A. Kendall, Y. Gal, and R. 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*, 2018, pp. 7482–7491.

[12] L. Pinto and A. Gupta, "Learning to push by grasping: Using multiple tasks for effective learning," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 2161–2168.

[13] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *arXiv preprint arXiv:1312.6229*, 2013.

[14] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[15] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[16] M. Suteu and Y. Guo, "Regularizing deep multi-task networks using orthogonal gradients," *arXiv preprint arXiv:1912.06844*, 2019.

[17] I. 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*, 2017, pp. 6129–6138.

[18] X. Zhao, H. Li, X. Shen, X. Liang, and Y. Wu, "A modulation module for multi-task learning with applications in image retrieval," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 401–416.

[19] T. Standley, A. Zamir, D. Chen, L. Guibas, J. Malik, and S. Savarese, "Which tasks should be learned together in multi-task learning?" in *International Conference on Machine Learning*. PMLR, 2020, pp. 9120–9132.

[20] Z. Wang, Z. C. Lipton, and Y. Tsvetkov, "On negative interference in multilingual models: Findings and a meta-learning treatment," *arXiv preprint arXiv:2010.03017*, 2020.

[21] Z. Wang, Z. Dai, B. Póczos, and J. Carbonell, "Characterizing and avoiding negative transfer," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 11 293–11 302.

[22] M. Abdollahzadeh, T. Malekzadeh, and N.-M. M. Cheung, "Revisit multimodal meta-learning through the lens of multi-task learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 14 632–14 644, 2021.

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

[24] A. A. Rusu, S. G. Colmenarejo, C. Gulcehre, G. Desjardins, J. Kirkpatrick, R. Pascanu, V. Mnih, K. Kavukcuoglu, and R. Hadsell, "Policy distillation," *arXiv preprint arXiv:1511.06295*, 2015.

[25] Y. Teh, V. Bapst, W. M. Czarnecki, J. Quan, J. Kirkpatrick, R. Hadsell, N. Heess, and R. Pascanu, "Distral: Robust multitask reinforcement learning," *Advances in neural information processing systems*, vol. 30, 2017.

[26] M. Hessel, H. Soyer, L. Espeholt, W. Czarnecki, S. Schmitt, and H. Van Hasselt, "Multi-task deep reinforcement learning with popart," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 3796–3803.

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

[28] T. Yu, S. Kumar, A. Gupta, S. Levine, K. Hausman, and C. Finn, "Gradient surgery for multi-task learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 5824–5836, 2020.

[29] Z. Wang, Y. Tsvetkov, O. Firat, and Y. Cao, "Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models," *arXiv preprint arXiv:2010.05874*, 2020.

[30] B. Liu, X. Liu, X. Jin, P. Stone, and Q. Liu, "Conflict-averse gradient descent for multi-task learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 18 878–18 890, 2021.

[31] Z. Chen, V. Badrinarayanan, C.-Y. Lee, and A. Rabinovich, "Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks," in *International conference on machine learning*. PMLR, 2018, pp. 794–803.

[32] D. Senushkin, N. Patakin, A. Kuznetsov, and A. Konushin, "Independent component alignment for multi-task learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 20 083–20 093.

[33] Z. Chen, J. Ngiam, Y. Huang, T. Luong, H. Kretzschmar, Y. Chai, and D. Anguelov, "Just pick a sign: Optimizing deep multitask models with gradient sign dropout," *Advances in Neural Information Processing Systems*, vol. 33, pp. 2039–2050, 2020.

[34] A. Javaloy and I. Valera, "Rotograd: Gradient homogenization in multitask learning," *arXiv preprint arXiv:2103.02631*, 2021.

[35] A. Navon, A. Shamsian, I. Achituve, H. Maron, K. Kawaguchi, G. Chechik, and E. Fetaya, "Multi-task learning as a bargaining game," *arXiv preprint arXiv:2202.01017*, 2022.

[36] X. Lin, H.-L. Zhen, Z. Li, Q.-F. Zhang, and S. Kwong, "Pareto multi-task learning," *Advances in neural information processing systems*, vol. 32, 2019.

[37] F. Ye, B. Lin, Z. Yue, P. Guo, Q. Xiao, and Y. Zhang, "Multi-objective meta learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 21 338–21 351, 2021.

[38] B. Lin, F. Ye, Y. Zhang, and I. W. Tsang, "Reasonable effectiveness of random weighting: A litmus test for multi-task learning," *arXiv preprint arXiv:2111.10603*, 2021.

[39] C. Fifty, E. Amid, Z. Zhao, T. Yu, R. Anil, and C. Finn, "Measuring and harnessing transference in multi-task learning," *arXiv preprint arXiv:2010.15413*, 2020.

[40] M. Guo, A. Haque, D.-A. Huang, S. Yeung, and L. Fei-Fei, "Dynamic task prioritization for multitask learning," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 270–287.

[41] H. Yun and H. Cho, "Achievement-based training progress balancing for multi-task learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 16 935–16 944.

[42] L. Liu, Y. Li, Z. Kuang, J. Xue, Y. Chen, W. Yang, Q. Liao, and W. Zhang, "Towards impartial multi-task learning." iclr, 2021.

[43] S. Liu, E. Johns, and A. J. Davison, "End-to-end multi-task learning with attention," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 1871–1880.

[44] B. Liu, Y. Feng, P. Stone, and Q. Liu, "Famo: Fast adaptive multitask optimization," *arXiv preprint arXiv:2306.03792*, 2023.

[45] G. M. Jacob, V. Agarwal, and B. Stenger, "Online knowledge distillation for multi-task learning," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 2359–2368.

[46] E. M. Rudd, M. Günther, and T. E. Boult, "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*. Springer, 2016, pp. 19–35.

[47] K. Mori, H. Fukui, T. Murase, T. Hirakawa, T. Yamashita, and H. Fujiyoshi, "Visual explanation by attention branch network for end-to-end learning-based self-driving," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, 2019, pp. 1577–1582.

[48] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 618–626.

[49] M. Gilles, Y. Chen, E. Z. Zeng, Y. Wu, K. Furmans, A. Wong, and R. 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*, pp. 1–19, 2023.

[50] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3213–3223.

[51] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part V 12*. Springer, 2012, pp. 746–760.

[52] S. Back, J. Lee, T. Kim, S. Noh, R. Kang, S. Bak, and K. Lee, "Unseen object amodal instance segmentation via hierarchical occlusion modeling,"

in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 5085–5092.

[53] A. Wong, Y. Wu, S. Abbasi, S. Nair, Y. Chen, and M. J. 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*, 2023, pp. 2292–2296.

[54] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[55] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[56] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.

[57] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

DAYOU MAO was an Undergraduate Research Assistant at the Vision and Image Processing (VIP) Lab in the department of Systems Design Engineering at the University of Waterloo. He is graduating with a Bachelor of Mathematics degree from the University of Waterloo in June 2024. His research interest mainly lies within topics on core deep learning and is dedicated to develop scientific theory in deep learning for vision modelling.

YUHAO CHEN (Member, IEEE) received the B.A.Sc. and Ph.D. degrees in electrical and computer engineering from Purdue University in 2015 and 2019, respectively. He is currently a Research Assistant Professor in systems design engineering with the Vision and Image Processing Laboratory (VIP), University of Waterloo. His research has been focused on developing computer vision and artificial intelligence solutions for industrial applications. He was a member of the Video and Image Processing (VIPER) Laboratory.

YIFAN WU received the B.Eng. degree in computing from the Imperial College London, London, U.K., in 2017, and the M.A.Sc. degree in systems design engineering from the University of Waterloo, Waterloo, ON, Canada, in 2022. His research interests include optical remote sensing image processing, building extraction, and machine learning.

MAXIMILIAN GILLES received the B.Sc. and M.Sc. degrees in mechanical engineering from the Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany, in 2017 and 2020, respectively, where he is currently pursuing the Ph.D. degree. His research interests include robot perception and manipulation, with a special emphasis on material handling applications. He is also a member of the Robotics and Interactive System Group, Institute for Material Handling and Logistics (IFL).

ALEXANDER WONG (Member, IEEE) is currently the Canada Research Chair of Artificial Intelligence and Medical Imaging, a member of the College of the Royal Society of Canada, a fellow of the Institute of Engineering and Technology and International Society for Design and Development in Education, the Co-Director of the Vision and Image Processing Research Group, and a Professor with the Department of Systems Design Engineering, University of Waterloo. He is also a P.Eng. He has published over 650 refereed journals and conference papers in various fields, such as computational imaging, artificial intelligence, computer vision, and multimedia systems.