Towards Task-Conflicts Momentum-Calibrated Approach for Multi-task Learning

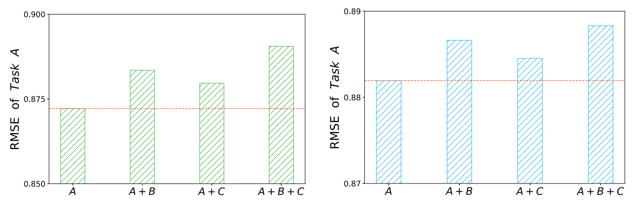
Heyan Chai[1], Zeyu Liu[1], Yongxin Tong[2], Ziyi Yao[1], Binxing Fang[1,3], Qing Liao[1,3,*]
[1]*School of Computer Science and Technology, Harbin Institute of Technology (Shenzhen), China*
[2] *School of Computer Science and Engineering, Beihang University, China*
[3]*Peng Cheng Laboratory, Shenzhen, China*
{chaiheyan,liuzeyu,yaoziyi}@stu.hit.edu.cn,yxtong@buaa.edu.cn
fangbx@cae.cn, liaoqing@hit.edu.cn

Abstract—**Multi-task learning (MTL) has succeeded in various industrial applications by utilizing common knowledge among joint training tasks to enhance the generalization of MTL models, resulting in improved performance across all training tasks simultaneously. Unfortunately, training all tasks simultaneously often causes performance degradation compared to single-task models since different tasks might conflict with each other. Despite existing MTL methods that aim to mitigate task conflicts by manipulating task gradients at each iteration, they ignore the potential influence of noisy data from different batches on task gradients. Consequently, the current iteration's task gradient may not accurately reflect the task itself, leading to inadequate alleviation of the dilemma of task conflicts. Moreover, existing works seldom explore the potential source of task conflicts and merely pose an assumption. In this paper, we conduct an in-depth empirical investigation into the potential sources of performance degradation of MTL and find that task gradient conflict is one of the primary reasons for the performance degradation of tasks. Then, to address the task conflicts problem, we propose a novel gradient manipulation approach, namely MoCoGrad, which manipulates task gradients by leveraging the momentum information of the task to calibrate the gradients of conflicting tasks. In addition, we derive theoretical guarantees for the convergence of our proposed MoCoGrad and theoretically analyze the convergence rate of MoCoGrad. Finally, to evaluate the effectiveness of MoCoGrad, extensive experiments are conducted on six real-world datasets from different domains. Our approach yields the best performance across all tasks in all six MTL benchmarks, demonstrating the effectiveness and superiority of our method.**

Index Terms—**multi-task learning, gradient manipulation, task conflicts**

I. Introduction

Over the last decade, deep learning has brought tremendous advantages in many fields. Many deep neural networks have been proposed to solve a large variety of tasks, such as image recognition [1], natural language processing [2], [3], and recommendation systems [4], [5]. However, most of them are designed to address one particular task. In many real-world scenarios, it is crucial to solve multiple tasks simultaneously while using limited computational or data resources. For example, autonomous driving [6] requires perception/object detection, path planning, and vehicle control task, which must all perform concurrently and in real-time. It is natural to solve this situation via multi-task learning (MTL), where a single model can tackle multiple tasks simultaneously [7]. Multi-task learning aims to jointly train multiple tasks to improve the generalization capability by exploiting general knowledge shared among tasks. It benefits individual tasks more efficiently and effectively [8]–[11]. It is vital to relieve researchers from the time-consuming and laborious effort of constructing a set of independent models [11]. MTL has achieved remarkable success in various research fields, including social media analysis [3], [12], [13], recommendation systems [5], [11], [14], and computer vision [15]–[17].



(a) Trained with HPS architecture. (b) Trained with MMoE architecture.

Fig. 1. The performance of task *A* by using HPS and MMoE [18] architectures trained on `MovieLens` dataset. The lower RMSE indicates better task performance and more experimental details are introduced in Section V. *A* denotes the single task learning, *A+B* denotes jointly training task *A* and *B*, and *A+B+C* denotes jointly training task *A*, *B* and *C*.

Nevertheless, MTL often causes performance deterioration compared to single-task models because optimizing multiple tasks simultaneously is a challenging optimization problem. Task conflict is a significant reason for such degradation [19]–[22]. Due to the discrepancy between joint training tasks, tasks in multi-task learning may correlate, conflict, or even compete with each other. As a result, some tasks are learned well while others are overlooked and far from being fully trained, leading to performance degradation [8], [13], [23], [24]. To intuitively illustrate our motivation, we give an example in Fig. 1, where we select three tasks from `MovieLens` dataset to train and present the performance of task *A* jointly. Joint training of different tasks can cause the performance of task *A* to fluctuate, and it degrades more as the number of training tasks

[*] Corresponding Author

increases, which experimentally demonstrates the existence of task conflicts.

Some previous works have been proposed to design specific MTL network architectures to reduce the interference from task conflicts in a soft-parameter sharing way [18], [25]–[27]. However, this type of method is scenario-dependent and only suitable for specific application scenarios, making it unable to be extended to other large-scale real-world application scenarios. Consequently, the generality of these methods is quite limited. Currently, there has been a significant amount of methods proposed to mitigate performance degradation problems caused by task conflicts in MTL from an optimization perspective [8], [9], [23], [28], using hard-parameters sharing (HPS) architectures. They attribute task conflicts to the optimization difficulty, especially the gradient conflicts between different tasks and propose to mitigate task conflicts by balancing the task losses or avoiding undesirable interference between task gradients.

However, there are two main limitations with existing MTL methods: 1) Existing methods seldom carry out a quantitative, in-depth exploration and analysis of the potential source of task conflicts, and most of them hold a qualitative assumption based on some empirical demonstrations, potentially leading to a less comprehensive understanding of the motivation. 2) They only consider the task gradient information at the current iteration, derived from the current batch data, which can be easily influenced by noisy data. Only considering current state task gradient information is insufficient to capture an adequate level of the informativeness that accurately reflects the current task gradient. Additionally, it does not guarantee the robustness of task gradients with regard to data noise. Therefore, it is desirable to explore the potential causes of task conflicts, making the existing assumption of task conflicts more explicit and solid.

To this end, in this paper, to intuitively reveal whether task gradient conflict is the major reason for task conflicts, we dug deeper into the correlation between model performance degradation and task gradient conflicts via detailed experimental analysis (**for Limitation 1**). Based on the above observations and analysis, only leveraging task gradient information in the current iteration may lead to interference from noisy data. Therefore, we propose a more rational and effective approach to alleviate the issue of task conflicts, which involves utilizing historical task gradient information to mitigate task gradient conflicts. In this paper, we propose a preemptive multi-task learning approach to mitigate task conflicts, referred to as Momentum-calibrated Conflicting Gradients (**MoCoGrad**), which leverages the historical task gradients information (also called momentum information[1]) to calibrate conflicting task gradient, so as to improve all task performance simultaneously (**for Limitation 2**). More concretely, we first propose an effective algorithm to manipulate the conflicting task gradients by leveraging the momentum information of tasks to pull conflicting task gradients closer to each other. Then, we analyze our approach theoretically and establish convergence guarantees. Finally, we empirically show that our MoCoGrad approach achieves state-of-the-art results on six MTL benchmarks on a variety of challenges ranging from recommendation systems and quantum chemistry to computer vision. The main contributions of our work are summarized as follows:

- We qualitatively and quantitatively describe the task conflicts problem characterized by *Task Conflict Intensity (TCI)* and provide additional investigation and exploration into the potential source of the task conflicts problem, which has rarely been discussed in previous works.
- We propose a novel gradient manipulating algorithm **MoCoGrad** to mitigate the task conflicts problem, which is via leveraging the momentum information of tasks to capture the practical relationship between different tasks at the current iteration, so as to achieve a relative trade-off between training tasks and effectively reduce the effects caused by task conflicts.
- We give in-depth theoretical proofs and analysis of our proposed MoCoGrad and establish converge guarantees in the convex setting.
- We extensively verify the effectiveness of our MoCoGrad on many real-world datasets on a variety of challenges ranging from recommendation systems and quantum chemistry to computer vision, such as click-through rate (CTR) prediction tasks, pixel labeling tasks, image classification tasks, MTL regression tasks.

II. Related Work

A. Multi-task Learning

Multi-task learning has been a vital research topic in the machine learning community over the last decade, through which one can simultaneously solve several tasks by leveraging the common knowledge contained in joint training tasks [7], [17], [32]. To achieve this, most MTL methods hold an assumption that all joint training tasks are related and the task parameter space is closer [33]–[35]. However, most real-world tasks tend to be unrelated and jointly training them is detrimental to the performance of all tasks due to task conflicts [11], [17]. To effectively mitigate task conflicts, significant efforts have been invested in designing different network architectures, which can be summarized into hard-parameter sharing [9], [19], [23], [26], [36]–[41] and soft-parameter sharing [18], [25]–[27], [42].

1) The hard-parameter sharing methods have the simplest network structure and are the most widely used MTL methods [7]. They consist of a shared network at the bottom of the overall network and branch out to several task-specific networks at the top. For example, UberNet [36] designs different task-specific networks for different tasks across different layers to improve the generalization of MTL. Long et al. [37] leverage the relationship between task parameters derived from task-specific networks to capture task correlations, so as to

[1]Momentum information can be viewed the historical gradient information of task [29]–[31].

enable the sharing of useful information between all tasks. MGDA [19] casts the multi-task learning problem as a multi-objective optimization problem, leveraging the task gradients derived from task-shared networks to find a Pareto optimal solution to mitigate the task conflicts. PCGrad [23] proposes to project the conflicting gradients into other gradients to mitigate task conflicts in a hard-parameter sharing way.

2) The soft-parameter sharing methods mainly focus on specific scenarios, where there is no specific task-shared network and each task has its own unique network structure, and cross-task information sharing can happen at any layer of the overall network. For example, Misra et al. [25] propose to adaptively fuse the features from different tasks by a learnable linear combination mechanism, aiming to perform effectively cross-task information sharing. Ma et al. [18] adopt a task-gate mechanism to fuse the different part of expert networks for each task adaptively. Liu et al. [26] adopt task-specific attention modules to help each task select useful features. Tang et al. [27] adopt a progressive routing mechanism for each task to fuse useful semantic knowledge gradually. Sun et al. [42] design an adaptive feature fusion mechanism to decide what to share in the task-sharing network.

B. Applications of Multi-task Learning

MTL has achieved remarkable success in various research and industrial applications, including social media analysis [3], [12], [13], recommendation systems [5], [11], [14], [43], and computer vision [6], [15]–[17]. For example, Liao et al. [3] adopt a task-gate mechanism to filter noisy information from cross-task information sharing in fake news detection tasks. A multi-task interaction network is proposed to perform stance detection and sentiment analysis tasks simultaneously [12]. Li et al. [12] propose a multi-task learning framework by leveraging heterogeneous information of complex objects and relations to improve recommendation system tasks. Teichmann et al. [6] propose a multi-task network to simultaneously perform joint classification, detection, and semantic segmentation tasks, which are essential tasks in autonomous driving field.

III. Task Conflicts Analysis in MTL

In this section, we will give detailed definitions of MTL and experimental analysis of the potential causes of task conflicts in MTL.

A. Problem Formulation

Definition 1 (Multi-task Learning, MTL). *Given K different learning tasks $\{\mathcal{T}^k|_{k=1}^{k=K}\}$ over one or multiple input space $\mathcal{X}$ and a collection of task spaces $\{\mathcal{Y}^k\}_{k\in[K]}$, where each task contains a set of i.i.d. training samples $\mathcal{D}_k = \{x_i^k, y_i^k\}_{i\in[n_k]}$, n_k is the number of training samples of task $\mathcal{T}^k$, the goal of multi-task learning is to improve the generalization performance across all training tasks over $\mathcal{X}$ by utilizing the knowledge contained in K tasks, formally,*

$$\min_{\theta_1,\ldots,\theta_K} \quad \sum_{k=1}^{K} \hat{\mathcal{L}}_k\left(x_i^k, y_i^k; \theta_k\right) \tag{1}$$

where $\hat{\mathcal{L}}_k(\theta_k)$ is the empirical loss of the task $\mathcal{T}^k$ which is defined as $\hat{\mathcal{L}}_k\left(x_i^k, y_i^k; \theta_k\right) \triangleq \frac{1}{|n_k|}\sum_i \mathcal{L}\left(\mathcal{F}\left(x_i^k, y_i^k; \theta_k\right), y_i^k\right)$, θ_k is the task-shared and task-specific parameters of task $\mathcal{T}^k$.

Note that, in this paper, we only focus on supervised learning tasks. Existing methods mainly consider a general setting for multi-task learning (i.e., multi-label learning, multi-domain classification, etc.), where all K tasks share the same training datasets, and we term this type of multi-task learning as Single-Input MTL. Differently, we consider a more practical multi-task learning setting, Multi-Input MTL, in which each task has individual and disjoint training datasets and only shares the same task-shared network not training examples.

B. Task Conflicts in Multi-task Learning

Unfortunately, directly optimizing Eq.(1) using common gradient descent optimization strategies may significantly deteriorate the model's performance. A major source of this phenomenon is that different tasks may compete with each other, named task conflicts [11], [20]–[22]. To determine whether task conflicts occur, we propose a quantitative measurement of task conflicts to evaluate their influence on task performance, defined as *Task Conflict Intensity (TCI)*.

Definition 2 (Task Conflict Intensity, TCI). *Denote the model obtained by K tasks as $\mathcal{F}(\mathcal{T}^1,...,\mathcal{T}^K)$ and the model obtained by single-task learning on task $\mathcal{T}^k$ as $\mathcal{F}(\mathcal{T}^k)$. Let $\mathcal{R}_{\mathcal{T}^k}$ be the standard expected risk on the task $\mathcal{T}^k$. Then the intensity of task conflicts about task $\mathcal{T}^k$ on model $\mathcal{F}$ can be evaluate by:*

$$TCI(\mathcal{T}^k, \mathcal{F}) = \mathcal{R}_{\mathcal{T}^k}(\mathcal{F}(\mathcal{T}^1,...,\mathcal{T}^K)) - \mathcal{R}_{\mathcal{T}^k}(\mathcal{F}(\mathcal{T}^k)), \tag{2}$$

where $\mathcal{R}_{\mathcal{T}^k}$ can be defined by:

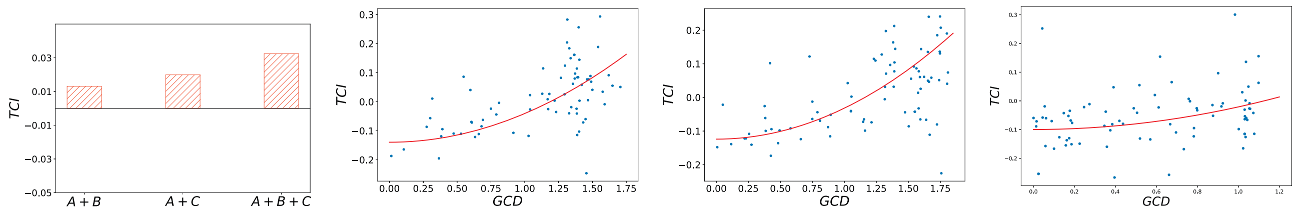
$$\mathcal{R}_{\mathcal{T}^k}(\mathcal{F}) = \mathbb{E}_{x,y\sim\mathcal{T}^k}(\mathcal{L}(\mathcal{F}(x), y)), \tag{3}$$

where $\mathcal{L}$ is the objective function.

Obviously, if the TCI is negative, the task conflict occurs, and vice versa. To present the task conflicts problem quantitatively, we conduct an experiment to show the TCI of selected tasks by evaluating some common MTL models on several task sets. As shown in Fig. 2(a), it is evident that the joint training of different tasks can cause significant task conflicts, leading to a decrease in model performance. Note that we use the $RMSE$ as a metric on the `MovieLens` dataset, therefore a lower value indicates better performance and a positive TCI means the task conflict has occurred.

C. Analysis of Potential Source of Task Conflicts

Previous works hold the assumption that degraded performance caused by task conflicts is owing to gradient conflicts between different tasks [9], [23], [24]. They therefore manipulate task gradients to mitigate this situation. However, existing works seldom carry out in-depth analysis of task gradient conflicts, which hinders the acquisition of a clear understanding of the correlation between gradient conflicts and the decline in model performance.



(a) TCI of HPS architecture. (b) Training with A and B (c) Training with A and C (d) Training with B and C

Fig. 2. The correlation between task conflict intensity (TCI) and gradient conflict degree (GCD) on selected tasks from MovieLens dataset (The details are in experimental section). (a) TCI of task A, training with different tasks, task B and C. (b) TCI of task A, training with task A and B. (c) TCI of task A, training with task A and C. (d) TCI of task B, training with task B and C.

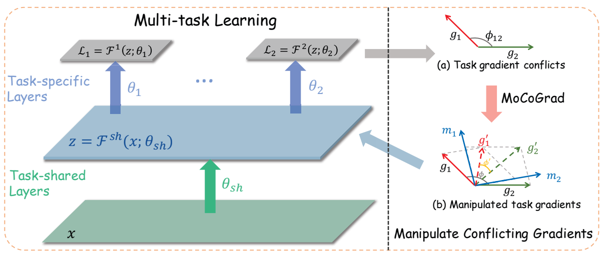

Multi-task Learning
$\mathcal{L}_1 = \mathcal{F}^1(z;\theta_1)$
$\mathcal{L}_2 = \mathcal{F}^2(z;\theta_2)$
Task-specific Layers
θ_1
θ_2
$z = \mathcal{F}^{sh}(x;\theta_{sh})$
Task-shared Layers
θ_{sh}
x
g_1
ϕ_{12}
g_2
(a) Task gradient conflicts
MoCoGrad
m_1
g_1'
g_2'
m_2
g_1
g_2
(b) Manipulated task gradients
Manipulate Conflicting Gradients

Fig. 3. Training process of using MoCoGrad algorithm. *Left.* The left part of the figure is a common MTL architecture. *Right.* We show how to work our proposed MoCoGrad algorithm in the case where gradients g_1 and g_2 in $\mathbb{R}^2$ are conflicting. The blue arrows denote the momentum information of task gradients, and the dashed arrows denote the manipulated task gradients, g_1' and g_2', by our MoCoGrad algorithm.

We now delve into a detailed exploration of the relationship between task gradient conflicts and task conflict intensity. To quantitatively analyze gradient conflicts, we define a metric to measure the degree of gradient conflicts between two tasks, referred to as *Gradient Conflict Degree (GCD)*.

Definition 3 (Gradient Conflict Degree, GCD). *Given two different tasks, $\mathcal{T}^i$ and $\mathcal{T}^j$, let ϕ_{ij} be the angle between two task gradients g_i and g_j. We define the degree of gradient conflict as,*

$$GCD(g_i, g_j) = 1 - \cos\phi_{ij}, \tag{4}$$

where $\cos\phi_{ij}$ is the cosine similarity of $\boldsymbol{g}_i$ and $\boldsymbol{g}_j$. The task gradient conflicts occurs when $GCD > 1$, and a larger GCD indicates more intense conflicts between gradients.

We plot the correlation curve between TCI and GCD in Fig. 2(b-d). Note that we use $RMSE$ as a metric and thus a lower TCI value means better performance. It has been observed that a strong positive correlation exists between TCI and GCD. Specifically, a larger GCD value corresponds to a larger TCI value. This implies that the more intense task gradient conflicts are, the poorer the task performance will be. Accordingly, we can solidify the hypothesis that the major reason for task conflicts is task gradient conflicts.

IV. Methodology

Analysis in Section III reveals that the relationship between task gradients plays a significant role, and task gradient conflicts indeed deteriorate the MTL model generalization. Based on it, in this section, we will give a detailed description of the proposed multi-task learning method on how to alleviate the dilemma of task conflicts in MTL, named **Momentum-calibrated Conflicting Gradients (MoCoGrad)**. We first introduce the preliminaries and notations in Section IV-A. Then, we discuss the overall MoCoGrad algorithm in detail in Section IV-B. Finally, we prove and analyze the convergence and effectiveness of our proposed MoCoGrad in Section IV-C.

A. Preliminaries and Notations

Gradient Manipulation-based Approaches. An existing line of work [9], [23], [24], [44] has used gradient-based methods to benefit MTL models. Notably, there are two works most relevant to our work, PCGrad [23] and GradVac [9]. PCGrad assumes that negative cosine similarity indicates task gradient conflicts, which is detrimental to the learning of MTL models. Based on it, as illustrated in Fig. 4(b), PCGrad proposes a gradient manipulating algorithm by replacing conflicting gradient g_i by its projection onto the normal plane of g_j, defined by

$$g_i' = g_i - \frac{g_i \cdot g_j}{\|g_j\|^2} g_j. \tag{5}$$

where g_i and g_j are the gradients of the i-th and j-th task respectively at an arbitrary iteration, and g_i' denotes the manipulated gradient.

Besides, GradVac [9] proposes to set adaptive gradient similarity objectives in a proper manner, aiming to capture the complex inter-task relationships. As illustrated in Fig. 4(c), where taking manipulating conflicting gradient g_i as an example, GradVac replaces the conflicting gradients g_i with a vector that is a linear combination of g_i and g_j, formally defined as

$$g_i' = g_i + \alpha \cdot g_j, \tag{6}$$

where α is the coefficient of linear combination calculated by Law of Sines. Formally,

$$\alpha = \frac{\|g_i\| \left(\cos\gamma\sqrt{1-\cos^2\phi_{ij}} - \cos\phi_{ij}\sqrt{1-\cos^2\gamma}\right)}{\|g_j\| \sqrt{1-\cos^2\gamma}} \tag{7}$$

where $\cos\phi_{ij} = \frac{g_i \cdot g_j}{\|g_i\|\|g_j\|}$ and $\cos\gamma = \frac{g_i' \cdot g_j}{\|g_i'\|\|g_j\|}$. ϕ_{ij} is the angle of g_i and g_j, and γ is the angle of manipulated gradients g_i' and g_j.

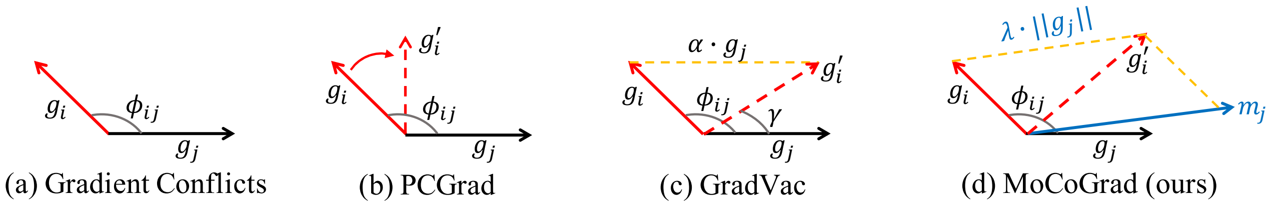


Fig. 4. The Comparison of PCGrad, GradVac, and MoCoGrad on a 2D multi-task learning system. **(a)** Tasks $\mathcal{T}^i$ and $\mathcal{T}^j$ have conflicting gradients, g_i and g_j respectively, leading to destructive interference. **(b)** and **(c)** are the process of PCGrad and GradVac algorithm manipulating conflicting gradients, respectively. **(d)** We illustrate how to work our proposed MoCoGrad algorithm in the case where gradients are conflicting, which is via leveraging the momentum information of another gradient to calibrate the conflicting gradients. Note that, we only use the conflicting gradient g_i as an example to illustrate our MoCoGrad. blue arrow denote corresponding momentum of task gradient m_j, and red dashed arrow denotes manipulated task gradient g_i'.

However, in the above methods, the gradients are derived from the data of the current batch only while the influence of other batches is ignored, making the task gradient of the current iteration not truly representative of the task itself.

Notations. Following existing MTL methods, we take the hard-parameter sharing MTL architecture as an example to present our MoCoGrad approach. As shown in Fig. 3, $\mathcal{F}^{sh}$ and $\mathcal{F}^k$ are parameterized by heavy-weight task-shared parameters θ_{sh} and light-weight task-specific parameters θ_k, respectively. Since all tasks take the identical intermediate feature $z = \mathcal{F}^{sh}(x; \theta_{sh})$ as input, it is inevitable that different tasks conflict with each other, leading to model performance deterioration. Note that our proposed MoCoGrad is not only applicable to hard-parameters sharing MTL architectures but also to other sparse-parameters sharing MTL architectures, such as MMoE [18], MTAN [26], and Cross-stitch [25]. In other words, our MoCoGrad is a general and model-agnostic approach that does not depend on specific architectures. For simplicity of analysis, in the following section, we omit the task-shared parameters θ_{sh} for the ease of notation, and only use θ_k to denote the task-shared and task-specific parameters of task $\mathcal{T}^k$. We use $\|\cdot\|$ to denote the spectral norm for the matrix and ℓ_2-norm for the vector.

B. MoCoGrad

For an arbitrary task $\mathcal{T}^k$, we define its gradient as $g_k = \nabla_{\theta_k}\mathcal{L}_k$ via back-propagation from the raw loss $\mathcal{L}_k$, and g_k represents the optimal update direction of task k. Due to the inconsistent optimal update directions of shared parameters for each task, conflicts can arise from the gradients of different tasks. These conflicts may impede the training process by causing networks to become excessively trained on certain tasks while inadequately trained on others, causing the model to deteriorate. As such, it is desirable to improve the generalization of the MTL model via alleviating task gradient conflicts, with the aim of benefiting all training tasks simultaneously. In this section, we propose a novel gradient manipulation algorithm to mitigate task conflicts by calibrating the conflicting gradients.

Inspired by PCGrad and GradVac, we first propose a measurement to quantitatively define task gradient conflicts, *GCD*, defined in Definition 3. We hypothesize that a larger GCD value is detrimental to multi-task optimization, especially when $GCD > 1$. In other words, when the GCD value between the task gradients g_i and g_j is greater than 1, $GCD(g_i, g_j) > 1$, task $\mathcal{T}^i$ and $\mathcal{T}^j$ compete with each other intensively, and the multi-task learning network composed of these two tasks exhibits task conflicts.

Existing gradient-based methods only use gradient information of the task at an arbitrary iteration derived by the current mini-batch data, and the gradients of tasks calculated through this approach are easily influenced by noisy data. Consequently, this leads to an inability to accurately capture the relationships between tasks, making it difficult to determine whether task conflicts have occurred. To address these limitations, a natural idea is to accurately capture relationship of the tasks by leveraging the historical gradient information. An example is shown in Fig. 4(d), where we use the momentum information of j-task to calibrate g_i.

In particular, for task gradients g_i and g_j derived at the t-th iteration, we first calculate the gradient conflict degree GCD to determine if they are conflicting. If the GCD is greater than 1, it indicates the occurrence of task gradient conflicts. In such cases, we replace the conflicting gradient g_i with a vector derived from the vector space spanned by g_i and $m_j^{(t-1)}$, which prevents conflicts between task gradients. To avoid the significant impact of momentum magnitude on the magnitude of gradients in the current iteration, we propose to leverage the relationship between task gradient g_j and task momentum $m_j^{(t-1)}$ at t iteration to scale the task momentum magnitude. We then derive the calibrated gradient for task $\mathcal{T}^i$ as:

$$\hat{g}_i = g_i + \lambda \frac{\|g_j\|}{\|m_j^{(t-1)}\|} \cdot m_j^{(t-1)} \tag{8}$$

where $\lambda \in (0, 1]$ is a pre-specified hyper-parameter and controls the degree of calibration of conflicting task gradient g_i. Note that we omit the superscript (t) of gradient $g_k^{(t)}$ of k-task for the ease of notation. $m_j^{(t-1)}$ denotes the first momentum of j-task at t-1 iteration, which is updated by,

$$m_j^{(t)} = \beta_1 m_j^{(t-1)} + (1 - \beta_1) \cdot \hat{g}_j \tag{9}$$

where the hyper-parameters $\beta_1 \in [0, 1)$ control the exponential decay rates of task gradients. In practice, $\beta_1 = 0.9$ is a typical

Algorithm 1 Training with MoCoGrad Algorithm

```
Input: Loss $\mathcal{L}_i$ of task $\mathcal{T}^i$, Network parameters $\theta$, Task set $\mathcal{T} = \{\mathcal{T}^i\}_{i=1}^K$, First moment estimates of $i$-task $m_i$, Hyper-parameter $\lambda, \beta_1$.
1:  Initialize model parameters
2:  Initialize moment of all tasks $\{m_i^{(0)}\}_{i=1}^K = 0$
3:  Initialize time step $t = 0$.
4:  while not converged do
5:      for $\mathcal{T}^i \in \mathcal{T}$ do
6:          $g_i = \nabla_\theta \mathcal{L}_i$
7:          for $\mathcal{T}^j \in \mathcal{T} \backslash \mathcal{T}^i$ in random order do
8:              $g_j = \nabla_\theta \mathcal{L}_i$
9:              if $GCD(g_i \cdot g_j) > 1$ then        ▷ using Eq. (4)
10:                 Set $\hat{g}_i = g_i + \lambda \frac{\|g_j\|}{\|m_j^{(t-1)}\|} \cdot m_j^{(t-1)}$.
11:             end if
12:             Update $m_j^{(t)} = \beta_1 m_j^{(t-1)} + (1-\beta_1) \cdot g_j$
13:         end for
14:     end for
15:     Update parameters with $g^{New} = \sum_{i=1}^{i=K} \hat{g}_i$
16: end while
```

setting [31].

Finally, the calibrated gradient $\hat{g}_i$ replaces the original g_i. Eq.(8) allows us to capture more accurate relationships between tasks at arbitrary iterations at the task gradient level, thereby effectively alleviating task conflicts. Our MoCoGrad repeats the conflicting gradient calibration process across all tasks via randomly sampling two tasks each time from the entire set of tasks. The detailed process of MoCoGrad is described in Algorithm 1.

C. Theoretical Analysis of MoCoGrad

We theoretically analyze the convergence of our method in the convex case. We first give some assumptions to help establish the following convergence analysis. We emphasize that these assumptions do not lose anything over typical SGD assumptions since our assumptions can be obtained from SGD [45].

Definition 4. *Consider the loss functions of two tasks $\mathcal{L}_1 : \mathbb{R}^n \to \mathbb{R}$ and $\mathcal{L}_2 : \mathbb{R}^n \to \mathbb{R}$. We define the two-task learning objective as $\mathcal{L}(\theta) = \mathcal{L}_1(\theta) + \mathcal{L}_2(\theta)$ for all $\theta \in \mathbb{R}^n$, where $g_1 = \nabla_\theta \mathcal{L}_1$, $g_2 = \nabla_\theta \mathcal{L}_2$, and $g = g_1 + g_2$.*

Assumption 1 (Upper bound)**.** *Assume that the model $\mathcal{F}$ has bounded diameter D and the gradients have bound G for all $t \in [T]$, then, for all $t \in [T]$, $\|\theta - \theta'\| \le D$, and $\|g^{(t)}\| \le G$.*

Theorem 1. *Suppose that **Assumption 1** holds. Let $\hat{g}$ be the calibrated gradients obtained from Algorithm 1, then $\hat{g}$ is also bounded.*

Proof. For any task $i, j \in [K]$ at t-th optimization step, we use the **Assumption 1**

$$
\begin{aligned}
\|\hat{g}\| &= \sum\nolimits_{i=1}^{K} (g_i + \lambda \frac{\|g_j\|}{\|m_j^{(t-1)}\|} \cdot m_j^{(t-1)}) \\
&= \sum\nolimits_i g_i + \lambda \sum\nolimits_{j=1}^{j=k} (\frac{\|g_j\|}{\|m_j^{(t-1)}\|} \cdot m_j^{(t-1)}) \\
&\le \sum\nolimits_i \|g_i\| + \lambda \sum\nolimits_{j=1}^{j=K} \| \frac{\|g_j\|}{\|m_j^{(t-1)}\|} \cdot m_j^{(t-1)}) \| \\
&\le \sum\nolimits_i \|g_i\| + \lambda \sum\nolimits_{j=1}^{j=K} \|g_j\| \\
&\le K(\lambda + 1)G \qquad (\text{using } \textbf{Assumption 1}) \\
&< 2KG \qquad (\text{using } \lambda \in (0,1] \text{ in Eq. } (8))
\end{aligned}
\tag{10}
$$

where K is the number of tasks. This inequality proves that the calibrated task gradients generated by MoCoGrad is bounded. □

Assumption 2 (L-smoothness)**.** *If $\mathcal{L}$ is differential and L-smooth, then $\mathcal{L}(\theta') \le \mathcal{L}(\theta) + \nabla \mathcal{L}(\theta)^\top (\theta' - \theta) + \frac{L}{2}\|\theta' - \theta\|^2$*

Theorem 2 (Convergence of Multi-task learning)**.** *Assume loss functions $\mathcal{L}_i$ are convex and differential, and $\nabla \mathcal{L}_i(\theta^{(t)})$ is L-lipschitz continuous with $L > 0$. The update rule is $\theta^{(t+1)} = \theta^{(t)} - \mu \hat{g}$, where $\hat{g}$ is derived by using MoCoGrad algorithm and $\mu > 0$ is the step size. For $\mu \le \frac{1}{L}$, the sequence $\{\theta^{(t}\}_{t=1}^{\infty}$ can converges to a optimal point θ^*. Moreover, all loss functions $(\mathcal{L}_1(\theta^{(t)}) \cdots \mathcal{L}_K(\theta^{(t)}))$ converges to $(\mathcal{L}_1(\theta^*) \cdots \mathcal{L}_K(\theta^*))$.*

Proof. We note that $\theta^{(t)}$ are the parameters of model at the t-th optimization step. Without loss of generality, we consider the two-task learning scenario based on **Definition 4** and we get calibrated gradients $\hat{g} = \hat{g}_1 + \hat{g}_2$. Then for each task $\mathcal{T}^i$ and the corresponding loss $\mathcal{L}_i$, we use **Assumption 2**

$$
\begin{aligned}
&\mathcal{L}_i(\theta^{(t+1)}) \le \mathcal{L}_i(\theta^{(t)}) - \nabla \mathcal{L}(\theta^{(t)})^\top \cdot \mu \hat{g} + \frac{L}{2}\|\mu \hat{g}\|^2 \\
&(\text{expanding } \hat{g} = \hat{g}_1 + \hat{g}_2 \text{ by using Eq. (8) }) \\
&\le \mathcal{L}_i(\theta^{(t)}) - \mu g^\top (g_1 + \lambda \frac{\|g_2\|}{\|m_2^{(t-1)}\|} \cdot m_2^{(t-1)} \\
&\qquad\qquad + g_2 + \lambda \frac{\|g_1\|}{\|m_1^{(t-1)}\|} \cdot m_1^{(t-1)}) \\
&(\text{expanding further and re-arranging terms }) \\
&\le \mathcal{L}_i(\theta^{(t)}) \underbrace{-(\mu - \frac{1}{2}L\mu^2 - \frac{1}{2}L\mu^2\lambda^2)}_{①} \\
&\underbrace{-(\mu\lambda g - L\mu^2\lambda g)\left(\frac{\|g_1\|}{\|m_1^{(t-1)}\|} m_1^{(t-1)} + \frac{\|g_2\|}{\|m_2^{(t-1)}\|} m_2^{(t-1)}\right)}_{②} \\
&\underbrace{-L\mu^2\lambda^2\|g_1\|\|g_2\| + L\mu^2\lambda^2 \frac{\|g_1\|\|g_2\|}{\|m_1^{(t-1)}\|\|m_2^{(t-1)}\|} m_1^{(t-1)} m_2^{(t-1)}}_{③}
\end{aligned}
\tag{11}
$$

Since $\mu \le \frac{1}{L}$ and $\lambda \in (0,1]$, we have

$$① \le -(\mu - \frac{1}{2}\mu - \frac{1}{2}\mu\lambda^2)g^2 = -\frac{1}{2}\mu(1-\lambda^2)g^2 \le 0, \quad (12)$$

and

$$② \le -(\mu\lambda g - \mu\lambda g)(\frac{\|g_1\|}{\|m_1^{(t-1)}\|}m_1^{(t-1)} + \frac{\|g_2\|}{\|m_2^{(t-1)}\|}m_2^{(t-1)}) \le 0. \quad (13)$$

With the triangle inequality and $\mu \le \frac{1}{L}$, we have

$$③ \le -(\mu\lambda^2\|g_1\|\|g_2\| - \frac{\|g_1\|\|g_2\|}{\|m_1^{(t-1)}\|\|m_2^{(t-1)}\|}\|m_1^{(t-1)}\|\|m_2^{(t-1)}\|) \le 0. \quad (14)$$

Then, we can submit Eq. (12), Eq. (13), Eq. (14) back into Eq. (11) to get the following inequality,

$$\mathcal{L}_i(\theta^{(t+1)}) - \mathcal{L}_i(\theta^{(t)}) \le 0 \quad (15)$$

Therefore, the $\mathcal{L}_i(\theta^{(t)})$ is bounded. As the sequence $\mathcal{L}_i(\theta^{(t)})$ is decreasing, the $\theta^{(t)}$ is in the subset $\{\theta : \mathcal{L}(\theta) \le \mathcal{L}(\theta^{(0)})\}$ that is closed and bounded, thus compact. As such, there exists a $\theta^{(\epsilon)}$ that converges to the point θ^*. Consequently, for $t \to \infty$, we can get that $\mathcal{L}_i(\theta^{(t)}) \xrightarrow{t\to\infty} \mathcal{L}_i(\theta^*)$ to complete proof. □

Moreover, we will explore the convergence rate of MoCograd in **Theorem 3**. Let $\mathcal{L}^{(t)}$ be the loss function at t iteration. Then the regret of MoCoGrad at the end T iterations of optimization process is given by

$$R(T) = \sum_{t=1}^{T} \mathcal{L}^{(t)}(\theta^{(t)}) - \min_\theta \sum_{t=1}^{T} \mathcal{L}^{(t)}(\theta) \quad (16)$$

Let $\theta^* = \arg\min_\theta \sum_{t=1}^{T} \mathcal{L}^{(t)}(\theta)$, we can get simplified form of regret by $R(T) = \sum_{t=1}^{T}(\mathcal{L}^{(t)}(\theta^{(t)}) - \mathcal{L}^{(t)}(\theta^*))$.

Theorem 3 (Convergence rate of MoCoGrad)**.** *Let $\{\theta^{(t)}\}$ be the sequence obtained from Algorithm 1, $\mu_t > 0$, $\lambda_t > 0$, $\beta_1 = \beta_{11}$, $\beta_{1t} \le \beta_1$ for all $t \in [T]$. Based on **Assumption 1**, we have the following bound on the regret*

$$R(T) \le \sum_{i=1}^{d} \frac{D_i^2}{2\mu_T} + K\sum_{t=1}^{T}\sum_{i=1}^{d} \lambda_t G_i D_i + \sum_{t=1}^{T}\sum_{i=1}^{d} \frac{\mu_t}{2}(1+k\lambda_t)^2 G_i^2 \quad (17)$$

Proof. We specifically split the upper bound of $R(T)$ onto each dimensional variable, for all $t \in [T]$ and $i \in [d]$, according to **Assumption 2**, Eq. (16) can be transformed by

$$R(T) \le \sum_{i=1}^{d} \underbrace{\sum_{t=1}^{T} g_{i,t}(\theta_i^{(t)} - \theta_i^*)}_{E_0} \quad (18)$$

where, for the sake of simplicity, $g_{i,t}$, $\theta_i^{(t)}$ denotes the i-th dimension of g and θ at t iteration. Since $\theta^{(t)}$ is generated by our MoCoGrad algorithm, we can get the update rule by

$$\theta_i^{(t+1)} = \theta_i^{(t)} - \mu_t \left(g_{i,t} + \lambda_t \sum_{k=1}^{K} \frac{\|g_{k,i,t}\|}{\|m_{k,i}^{(t-1)}\|} m_{k,i}^{(t-1)} \right). \quad (19)$$

For the sake of simplicity, we define $r_t = \sum_{k=1}^{K} \frac{\|g_{k,i,t}\|}{\|m_{k,i}^{(t-1)}\|} m_{k,i}^{(t-1)}$ and expand Eq. (19) by

$$\theta_i^{(t+1)} - \theta_i^* = \theta_i^{(t)} - \theta_i^* - \mu_t(g_{i,t} + \lambda_t r_t). \quad (20)$$

For further bounding this inequality Eq. (18), we need to calculate $(\theta_i^{(t)} - \theta_i^*)^2$ and re-arrange terms to construct the following equality.

$$E_0 = \underbrace{\frac{(\theta_i^{(t)} - \theta_i^*)^2 - (\theta_i^{(t+1)} - \theta_i^*)^2}{2\mu_t}}_{E_1} \underbrace{-\lambda_t r_t(\theta_i^{(t)} - \theta_i^*)}_{E_2} + \underbrace{\mu_t/2(g_{i,t} + \lambda_t r_t)^2}_{E_3} \quad (21)$$

With **Assumption 1**, we bound term E_1 in the following manner:

$$E_1 \le \frac{(\theta_i^{(1)} - \theta_i^*)^2}{2\mu_1} + \sum_{t=2}^{T} D_i^2(\frac{1}{2\mu_t} - \frac{1}{2\mu_{t-1}}) \le \frac{D_i^2}{2\mu_T}. \quad (22)$$

According to Cauchy-Schwarz and Young's inequality, the E_2 can be bounded as follows:

$$E_2 \le \sum_{t=1}^{T} \lambda_t \sum_{k=1}^{K} \frac{\|g_{k,i,t}\|}{\|m_{k,i}^{(t-1)}\|} m_{k,i}^{(t-1)} D_i \le \sum_{t=1}^{T} K G_i D_i \lambda_t. \quad (23)$$

And the E_3 can be bounded as follows:

$$E_3 \le \sum_{t=1}^{T} \left(\|g_{i,t}\| + \left\| \lambda_t \sum_{k=1}^{K} \frac{\|g_{k,i,t}\|}{\|m_{k,i}^{(t-1)}\|} \|m_{k,i}^{(t-1)}\| \right\| \right) \le \sum_{t=1}^{T} \frac{\mu_t}{2}(1+K\lambda_t)^2 D_i^2 \quad (24)$$

Finally, we submit Eq. (21), Eq. (22), Eq. (23) and Eq. (24) back into Eq. (18), we then obtain that

$$R(T) \le \sum_{i=1}^{d} \frac{D_i^2}{2\mu_T} + K\sum_{t=1}^{T}\sum_{i=1}^{d} \lambda_t G_i D_i + \sum_{t=1}^{T}\sum_{i=1}^{d} \frac{\mu_t}{2}(1+k\lambda_t)^2 G_i^2$$

Thus, the proof of **Theorem 3** has completed. □

Based on **Theorem 3**, the following result falls as an immediate corollary of the convergence rate of MoCoGrad.

Corollary 1. *Suppose $\mu_t = \mu/t^p$, $\lambda_t = \lambda/t^p$ for all $t \in [T]$ in **Theorem 3**, then we have*

$$R(T) \le 2D^2\frac{t^p}{\mu} + KGD\int_{t=1}^{T} \frac{\lambda}{t^p} dt + \frac{G^2}{2}\int_{t=1}^{T} (\frac{\mu}{t^p} + K^2\lambda^2\frac{\mu}{t^{3p}} + 2K\lambda\frac{\mu}{t^{2p}})dt \quad (25)$$

Then,

$$R(T)/T = \mathcal{O}(T^{\max(p, 1-p, 1-3p)})/T \quad (26)$$

when $p = 1/2$, the $\mu_t = \mu/\sqrt{t}$, we can get $\mathcal{O}(R(T)/T) = \mathcal{O}(T^{1/2})$ and $R(T)/T \xrightarrow{T\to\infty} 0$.

Since existing optimization methods, like SGD [46], Adam [31], Adagrad [47], have the same optimization upper $\mathcal{O}(T^{1/2})$ with our proposed MoCoGrad, the convergence rate of MoCoGrad is acceptable in practice.

V. Experiments

We evaluate whether MoCoGrad could benefit the performance of MTL models on a variety of MTL challenges, including recommendation system tasks, quantum chemistry molecule graph regression tasks, and computer vision tasks (i.e., image recognition and scene understanding).

A. Datasets

Experiments are conducted on six widely-used real-world MTL benchmark datasets:

- `AliExpress dataset` [48]: The AliExpress dataset[2] is collected from real-world traffic logs of the search system in AliExpress. This dataset is collected from 5 countries with more than 100M records and has two prediction tasks: Click-Through Rate (CTR) and Click-Through&Conversion Rate prediction (CTCVR). We select 4 countries, Spain (ES), French (FR), Netherlands (NL), and America (US), and construct 4x2 tasks learning scenarios. We randomly split 90% data in the time sequence for training while the rest 10% for testing.
- `MovieLens dataset` [49]: This is a movie recommendation dataset[3] which contains 10M ratings from 71567 users on 10,681 movies from Jan. 1996 to Dec. 2008 in a total of 18 different genres. We select 9 genres: Crime, Documentary, Fantasy, FilmNoir, Horror, Mystery, Thriller, War, and Western. In this paper, following [10], we treat the rating regression for movies in each selected genre as different tasks and construct a multi-task regression dataset that contains 9 regression tasks. We randomly split the whole dataset with 80% for training, 10% for validation, and the remaining 10% for testing.
- `QM9 dataset` [50]: The QM9 dataset is a widely used benchmark for graph neural networks, which consists of $\sim 130K$ molecules represented as graphs. Following [24], we select 11 properties of molecules QM9 dataset and construct 11 regression tasks. We use 110K molecules for training, 10K for validation, and the rest 10K for testing. This is a multi-input MTL dataset.
- `NYUv2 dataset` [51]: The NYUv2 dataset is an indoor scene understanding dataset, which consists of 1449 images and 464 diverse indoor scenes with detailed annotations. We construct a three-task learning scenario: the Depth Prediction, Semantic Segmentation, and Surface Normal Estimation tasks. We adopt 13-class annotation for Semantic Segmentation and follow the common train/val splits: 795 images for training and 654 images for validation.
- `CityScape dataset` [52]: The CityScapes dataset focuses on the semantic understanding of urban street scenes and contains 3,600 high-resolution street-view images. We construct a two-task learning scenario using Depth Prediction and Semantic Segmentation tasks. We follow the standard dataset splitting, 2,975 images for training, 125 for validation, and 500 for testing, respectively.
- `Office-Home dataset` [53]: The Office-Home dataset contains four domains: Art (artistic images), Clipart (collection of clipart images), Product (images of objects without background), and Real-World (images of objects captured with a regular camera). It has 15,500 images in total, and each domain contains 65 categories. Based on it, we treat each domain as a 65-way classification problem. Following [54], we randomly divide the dataset into training, validation, and testing sets, 60%, 20%, and 20%, respectively. Note that, each task in Office-Home dataset has its own input data. This is a multi-input MTL dataset.

[2]https://tianchi.aliyun.com/dataset/74690
[3]https://files.grouplens.org/datasets/movielens/ml-10m.zip

B. Compared methods

To better illustrate the effectiveness of our proposed MoCoGrad, we compare MoCoGrad with ten state-of-the-art multi-task learning methods, especially those that attempt to mitigate task conflicts and improve the ability of MTL generalization.

- `STL`: Single-task learning (**STL**) trains individual tasks with an independent model.
- `DWA` [26]: Dynamic weight average (**DWA**) is a simple adaptive weighting method that can leverage the task loss rates over time to set task weights adaptively.
- `MGDA` [19]: This method casts multi-task learning as a multi-objective optimization problem and attempts to optimize the upper bound of multi-objective loss by searching a convex combination of task gradients in a convex hull. The solutions generated by MGDA prove to be a Pareto optimal solution.
- `PCGrad` [23]: PCGrad is a simple yet general approach for avoiding the interference between task gradients, via manipulating the conflicting gradients. Specifically, when gradient conflicts occur, PCGrad removes conflicting components of each gradient by projecting a task's gradient onto the normal plane of the gradient of any other task, and then repeats this process over all tasks.
- `GradDrop` [41]: Gradient Sign Dropout (GradDrop) is a probabilistic masking procedure. Specifically, GradDrop algorithmically selects one sign based on gradient distribution at the current iteration, and then masks out all task gradient values that have the opposite sign.
- `GradVac` [9]: Gradient Vaccine (GradVac) is a gradient manipulation MTL method that avoids task conflicts by manipulating conflicting gradients. GradVac sets a different task conflicts objective by considering more complex inter-task relatedness, so as to adaptively align task gradients.
- `CAGrad` [8]: Conflict-Averse Gradient descent (CAGrad) method is proposed to solve the problem that average gradient direction can harm specific tasks' performance in multi-task learning. CAGrad is a gradient-based optimization method that searches for a convex combination of all

task losses by maximizing the worst local improvement of individual tasks nearby the average gradient.

- `IMTL` [55]: IMTL proposes an end-to-end learning framework that can optimize the scaling factors to make equal projections of the aggregated gradients onto individual tasks and weigh task losses to simultaneously keep all task losses at a comparable scale. It balances the gradients and losses of all tasks simultaneously and is a fair learning approach for jointly training tasks.
- `RLW` [54]: Random Loss Weighting (RLW) proposes to balance all training tasks by sampling task weights from a normal distribution and then minimizing the aggregated loss weights based on sampled task weights.
- `Nash-MTL` [24]: Nash-MTL regards the gradient aggregation process at each iteration as a bargaining game, making task gradients negotiate to achieve a balance. Specifically, Nash-MTL models the multi-task learning problem as a bargaining problem and obtains the Nash bargaining solution, which is used to scale all task losses.

C. Evaluation Metrics

Due to the difference in different datasets, we use different evaluation metrics to measure the performance of comparison methods.

- Recommendation system tasks. For recommendation system tasks from `AliExpress` dataset, such as CTR and CTCVR tasks, we follow [48] and use `AUC` (the area under the curve) to evaluate the goodness of order by ranking all the products with the predicted CTR and CTCVR.
- Regression tasks. For regression tasks from `MovieLens` and `QM9` datasets, we use mean absolute error (`MAE`) and root mean square error (`RMSE`) to evaluate the prediction accuracy of comparison methods.
- Scene understanding tasks on `NYUv2` and `CityScape` datasets. Following [17], [54], we use the mean intersection over union (`mIoU`) and pixel accuracy (`PixAcc`) to evaluate the semantic segmentation task and use absolute error (`Abs Err`) and relative error (`Rel Err`) to evaluate the depth prediction task. For the surface normal estimation task, we use the mean and median angle distances of all the pixels (`Rel Mean` and `Median`) and the percentage of pixels that are within the angles of $11.25°$, $22.5°$, $30°$ for evaluation.
- Image recognition tasks on `Office-Home` dataset. We use the average accuracy (`Avg ACC`) to evaluate the performance of all comparison methods.

In addition, due to the difference in scale between different per-task metrics, we define a common metric to evaluate each task. We use the average of relative improvement of MTL methods based on task conflict intensity (TCI) on each metric of each task to evaluate the multi-task performance [24], [54], defined as $\mathbf{\Delta}_{\boldsymbol{M}}$,

$$\mathbf{\Delta}_{\boldsymbol{M}} = \frac{1}{K}\sum\nolimits_{k=1}^{K} \frac{(-1)^{s_k}(M_{m,k} - M_{b,k})}{M_{b,k}}, \qquad (27)$$

where $M_{b,k}$ and $M_{m,k}$ denote the value of metric M_k obtained by the single-task learning method and multi-task learning methods, respectively. s_k is set to 0 if a higher value denotes better performance for a metric M_k and otherwise 1.

D. Implementation Details

All of our experiments are conducted on the Workstations of V100 with 32GB memory and Ubuntu 16.04 Server system and implemented by PyTorch 3.7.9 based on LibMTL [56], which is an open-source multi-task learning package. For a fair comparison, we adopt the hard-parameters sharing network architecture and set similar parameters for all the models. Moreover, we also set the same task-shared and -specific networks for all the models on the same tasks. For the `AliExpress` dataset, we use an embedding layer followed by two-layer MLP as task-shared layers. For the `QM9` dataset, we use graph convolutional neural networks as task-shared layers, and Adam optimizer with learning-rate set to 0.003. For the `MovieLens` dataset, we implement the BST model [57] as task-shared layers. For the `NYUv2` and `CityScapes` datasets, we use ResNet-50 network as task-shared layers and the Atrous Spatial Pyramid Pooling module [58] as task-specific layers with hidden dimensionality set to 2048. For the `Office-Home` dataset, we use ResNet-18 network as task-shared layers. We use Adam optimizer with a learning-rate set to 0.0001 and one-layer MLP as per-task task-specific layers for all datasets. Experiments are executed for ten runs, and the average results are reported.

E. Results of Multi-task Recommendation System

We evaluate MoCoGrad on predicting 2 tasks over 4 scenarios from `AliExpress` dataset, and the results of `AUC` are reported in Table I where the best `AUC` scores are bold. As well, the relative performance improvement of all MTL methods is displayed. Obviously, our proposed MoCoGrad outperforms both single CTR and CTCVR prediction methods and multi-task learning methods across four different scenarios in 8 out of 9 metrics, and obtains a 0.48% improvement over the state-of-the-art methods. This should be attributed to several factors. First, utilizing the momentum information of the task, MoCoGrad can capture an adequate level of informativeness that accurately reflects the current task gradient at each iteration, not easily influenced by noisy data. Second, leveraging momentum information can facilitate modeling the relationship between tasks at each iteration, so as to provide enough information to help MoCoGrad better alleviate the dilemma of task conflicts. More concretely, some gradient manipulation-based methods, such as PCGrad, GradDrop, and GradVac, perform poorly in CTR and CTCVR prediction tasks. The major reason may be that they can not accurately capture the true task gradient of each task at each iteration, influenced by the noisy data from some mini-batch data. The Nash-MTL method obtains the worst overall performance, although it has comparable performance in the CTR task. The massive gap in performance between different tasks comes

TABLE I
PERFORMANCE (AUC) ON ALIEXPRESS DATASET (2 X 4 TASKS IN TOTAL).

Methods	ES		FR		NL		US		$\Delta_M \uparrow$
	CTR	CTCVR	CTR	CTCVR	CTR	CTCVR	CTR	CTCVR	
STL	0.7266	0.8855	0.7259	0.8737	0.7222	0.859	0.7061	0.8637	0.0%
DWA	0.7238	0.8847	0.7045	0.8669	0.7236	0.8576	0.6965	0.8744	-0.54%
MGDA	0.723	0.8782	0.7238	0.8691	0.7209	0.8581	0.7032	0.8759	-0.18%
PCGrad	0.7275	0.867	0.7152	0.8753	0.724	0.8542	0.6978	0.8721	-0.47%
GradDrop	0.7307	0.8705	0.7165	0.8759	0.7251	0.8464	0.6877	0.8744	-0.58%
GradVac	0.7299	0.8593	0.7054	0.8755	0.7233	0.8455	0.7039	0.8739	-0.71%
CAGrad	0.7256	0.8722	0.716	0.877	0.7237	0.8586	0.6967	0.8726	-0.35%
IMTL	0.7253	0.8677	0.7246	0.8639	0.7225	0.8563	0.6883	**0.8788**	-0.57%
RLW	0.7275	0.8846	0.7238	0.8717	0.7266	0.8562	0.6992	0.8757	+0.02%
Nash-MTL	0.7286	0.8546	0.7145	0.8672	0.7245	0.8445	0.6878	0.8692	-1.11%
MoCoGrad	**0.7316**	**0.8870**	**0.7244**	**0.8783**	**0.7278**	**0.8614**	**0.7073**	0.8764	**+0.48%**

TABLE II
PERFORMANCE (AVERAGE MAE AND RMSE) ON QM9 DATASET (11 TASKS) AND MOVIELENS DATASET (9 TASKS).

Methods	QM9		MovieLens	
	Avg MAE $\downarrow$	$\Delta_M \uparrow$	Avg RMSE$\downarrow$	$\Delta_M \uparrow$
STL	0.7474	+0.00%	0.9009	+0.00%
DWA	0.6979	+20.49%	0.8841	+1.57%
MGDA	0.6813	+21.41%	0.8841	+1.56%
PCGrad	0.7514	+20.58%	0.8859	+1.36%
GradDrop	0.646	+24.02%	0.8862	+1.38%
GradVac	0.684	+24.56%	0.8826	+1.76%
CAGrad	0.7975	+21.36%	0.8867	+1.34%
IMTL	0.6372	+19.12%	0.8808	+1.89%
RLW	0.7961	+22.62%	0.8909	+0.75%
Nash-MTL	0.6744	+27.85%	0.9049	-0.50%
MoCoGrad	**0.5864**	**+32.30%**	**0.8721**	**+2.93%**

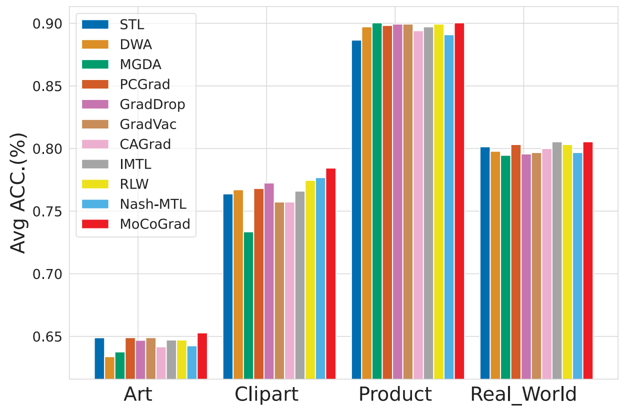


Fig. 5. The comparison performance of our proposed MoCoGrad with other baselines over four tasks on Office-Home dataset.

from task conflicts, and Nash-MTL can not deal with conflicts well.

F. Results of Multi-task Regression Tasks

We conduct several experiments to evaluate the performance of the proposed MoCoGrad on two multi-task regression datasets, QM9 and MovieLens dataset. The results of average MAE and RMSE metrics (lower value indicates better) are reported in Table II, where the best results are highlighted in bold type. It is observed that MoCoGrad significantly outperforms other baselines throughout all datasets, especially in the QM9 dataset that contains 11 tasks. More specifically, our MoCoGrad obtains 0.5846 in Avg MAE metric, which achieves a 4.45% improvement over state-of-the-art baselines on the QM9 dataset. Due to the MovieLens dataset that contains 9 tasks, with the number of tasks decreasing, the degree of task conflict between tasks becomes less intense, and consequently the performance of all comparison methods slightly deteriorates. It is no surprise that our MoCoGrad outperforms the baselines in the QM9 dataset more apparently than in the MovieLens dataset, since MoCoGrad indeed mitigates the task conflicts, especially when the number of joint training tasks is large. This demonstrates that our MoCoGrad is still effective in dealing with multi-task regression tasks, and it can better mitigate task conflicts, so as to improve the overall performance of MTL models.

G. Results of Scene Understanding Tasks

We evaluate the effectiveness of our proposed MoCoGrad in scene understanding tasks and report the performance of all comparison methods on NYUv2 and CityScapes datasets in Table III and Table IV. Obviously, our MoCoGrad achieves the best performance in 7 out of 10 metrics and obtains a 9.65% improvement over a single learning method in the NYUv2 dataset. The performance of MoCoGrad is well-balanced across tasks. We find similar patterns as on NYUv2 dataset in Table IV, but CityScapes dataset is more complex and larger. It is observed that our MoCoGrad achieves the best performance across all tasks. Existing gradient manipulation-based methods, such as PCGrad and GradDrop, perform poorly since they can not accurately identify the task gradients in each iteration under complex datasets, leading to degraded performance in Depth prediction task. This implies that our MoCoGrad can better mitigate task conflicts and improve the performance of all tasks and demonstrates the effectiveness of

TABLE III
PERFORMANCE ON NYUv2 DATASET WITH THREE TASKS: SEMANTIC SEGMENTATION, SURFACE NORMAL ESTIMATION, AND DEPTH PREDICTION.

Methods	Segmentation		Depth		Surface Normal					$\Delta_M \uparrow$
					Angle Distance		Within t°			
	mIoU↑	PixAcc↑	Abs Err↓	Rel Err↓	Rel Mean↓	Median↓	11.25° ↑	22.5° ↑	30° ↑	
STL	0.5292	0.7437	0.4822	0.1669	23.1671	17.2378	0.2793	0.5137	0.6988	+0.00%
DWA	0.5323	0.7537	0.3814	**0.1569**	23.8444	17.3143	0.3468	0.6024	0.7143	+7.68%
MGDA	0.5041	0.7213	0.4104	0.1688	22.4232	16.9118	0.3469	0.6092	0.7137	+6.23%
PCGrad	0.5383	0.7537	0.382	0.1615	23.4739	17.0915	0.3508	0.6113	0.7219	+8.28%
GradDrop	0.5367	0.7522	0.3839	0.1590	23.6725	16.9916	0.3529	0.6091	0.7187	+8.30%
GradVac	0.5404	0.7543	0.3858	0.1637	23.5048	16.9492	0.3517	0.6113	0.7218	+8.21%
CAGrad	0.5388	0.7519	0.3936	0.1575	23.5877	17.9514	0.3522	0.6137	0.7016	+7.44%
IMTL	0.5422	0.7531	0.3972	0.1677	23.5643	17.0286	0.3411	0.6022	0.7197	+6.97%
RLW	0.5405	0.7546	0.3827	0.1599	23.4271	17.7078	0.3481	**0.6152**	0.7201	+8.00%
Nash-MTL	0.5329	0.7477	0.3832	0.1685	**22.3121**	16.9411	0.3486	0.6067	0.7214	+8.04%
MoCoGrad	**0.5444**	**0.7566**	**0.3810**	0.1579	22.3779	**16.8209**	**0.3544**	0.6141	**0.7238**	**+9.65%**

TABLE IV
EXPERIMENTAL RESULTS ON CITYSCAPES DATASET (2 TASKS IN TOTAL).

Methods	Segmentation		Depth		$\Delta_M \uparrow$
	mIoU ↑	PixAcc ↑	Abs Err↓	Rel Err↓	
STL	0.7401	0.9316	0.0125	20.777	+0.00%
DWA	0.7522	0.9337	0.0108	18.6485	+6.43%
MGDA	0.7352	0.9275	0.0101	21.1459	+4.08%
PCGrad	0.7512	0.9348	0.0115	21.5973	+1.47%
GradDrop	0.7518	0.9342	0.0112	22.1326	+1.43%
GradVac	0.7571	0.9356	0.0109	19.0876	+5.91%
CAGrad	0.7528	0.9342	0.0114	18.2485	+5.74%
IMTL	0.7439	0.9310	0.0115	18.9272	+4.34%
RLW	0.7450	0.9296	0.0118	22.3387	-0.37%
Nash-MTL	0.7499	0.9346	0.0102	18.6321	+7.59%
MoCoGrad	**0.7568**	**0.9375**	**0.0106**	**16.2842**	**+9.93%**

MoCoGrad in several real-world scene understanding scenarios.

H. *Results of Image Recognition Tasks*

To verify the effectiveness of our proposed MoCoGrad in image recognition tasks, we conduct a comparison experiment on the `Office-Home` dataset and show the results of comparison methods across four tasks in Fig. 5. It is observed that our MoCoGrad achieves the best performance compared with ten comparison baselines and the performance of MoCoGrad is well balanced across all four tasks. We can see some methods, such as MGDA and CAGrad, perform worse than the single learning method and obtain imbalanced performance across different tasks. The major reason is that these methods can not handle task conflicts between different tasks during training. This also demonstrates the effectiveness of our proposed MoCoGrad, which can mitigate task conflicts well and achieve better-balanced performance across all tasks.

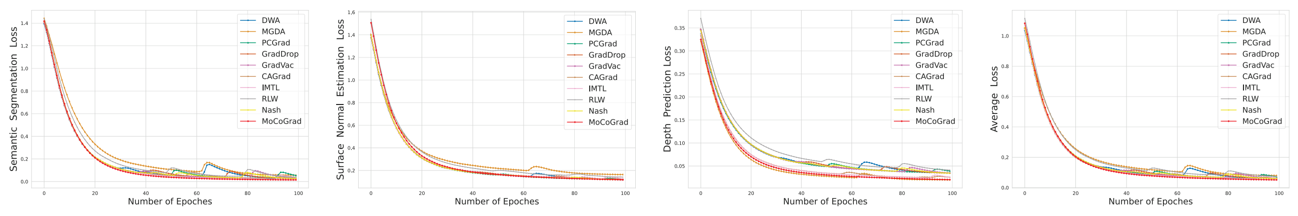
VI. ANALYSIS

A. *Empirical Analysis on Convergence*

In the theoretical analysis part of Section IV-C, we theoretically prove the convergence of our proposed MoCoGrad and analyze the convergence rate. Furthermore, we conduct extensive experiments on the `NYUv2` dataset to empirically analyze the convergence of MoCoGrad. Fig. 6 shows the training loss curves of Depth Prediction, Semantic Segmentation, and Surface Normal Estimation tasks and the average loss values of three tasks. It is clear that the loss value of our MoCoGrad is constantly decreasing throughout the training process, and our MoCoGrad converges to a lower average loss value on three tasks compared to other MTL methods. More specifically, the loss curves of some methods, such as DWA, MGDA, CAGrad, Nash-MTL, and IMTL, fluctuate with the increasing number of epochs, which illustrates that they can not handle task conflicts throughout the training process. Fig. 6(d) shows the training loss curves of average loss values of all three tasks. We can see that some methods, such as PCGrad, GradVac, and GradDrop, can not converge to the optimal objective value under the same training epochs. This implies that the convergence rate of these methods is slower than our proposed MoCoGrad. The major reason is that they do not handle task conflicts well at each iteration, leading to degraded MTL performance. Therefore, compared with other baselines, our MoCoGrad can help MTL models converge to the minimizer of objective functions and has a faster convergence speed.

B. *Effects of Different Architectures*

We conduct experiments to illustrate that our MoCoGrad can be seamlessly incorporated into other MTL architectures while obtaining improved performance. We select five MTL architectures, i.e., hard-parameters sharing (HPS), Cross_stitch network [25], Multi-Task Attention Network (MTAN) [26], MMoE [18], and Customized Gate Control network (CGC) [27] and use the performance of single task learning method as the baseline to measure the relative improvement Δ_M of our MoCoGrad with different MTL architectures on `CityScapes` dataset shown in Fig. 7. Obviously, our MoCoGrad method outperforms the single task learning method under all the five architectures. The combination of our MoCoGrad and MTAN, or Cross_stitch, or MMoE or CGC outperforms MoCoGrad with HPS, which illustrates the effectiveness of the combination of MoCoGrad and more



(a) Semantic segmentation task. (b) Depth prediction task. (c) Surface normal estimation task. (d) Average performance.

Fig. 6. Learning curve of comparison methods about three learning tasks on NYUv2 dataset.

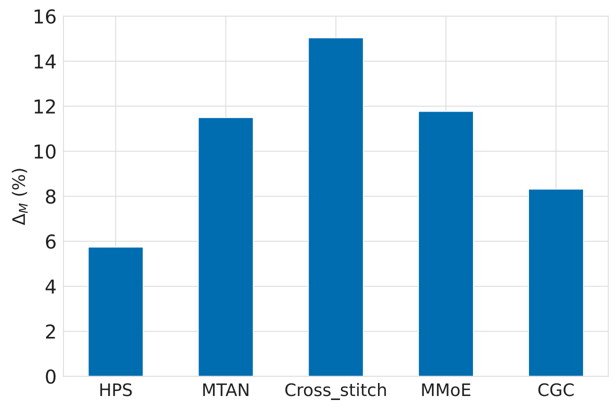


Fig. 7. The results of proposed MoCoGrad with different architectures on `CityScapes` dataset.

complex architectures. Moreover, it implies the potential of the proposed MoCoGrad in selecting appropriate MTL architectures.

C. Analysis on Backward Time

Most SOTA MTL methods require performing K backward-propagation procedures to calculate the K task gradients at each optimization step. This process is typically more expensive, and here we replace task gradients with feature-level gradients to speed up the optimization process. We conduct experiments to compare the backward propagation time per iteration in seconds of our proposed MoCoGrad with other baseline methods at each optimization process. We train these MTL methods with the same experimental setting for a fair comparison. Fig. 8 shows the backward time of comparison methods on the `AliExpress` dataset. The Nash-MTL method needs more time for backward-propagation, because more time is needed to solve the Nash equilibrium at each iteration. Moreover, we can observe that the proposed MoCoGrad has a comparable backward time with GradVac and PCGrad at each iteration, demonstrating that our MoCoGrad can easily be applied in practice.

D. Parameters Analysis

The hyper-parameter λ is used to control the degree of calibration of conflicting task gradient. To investigate the impact of λ, we conduct experiments on the `Office-Home`

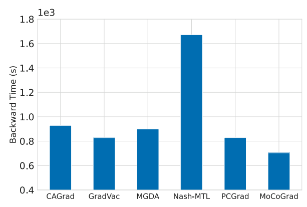


Fig. 8. The backward time of our proposed MoCoGrad with other baselines.

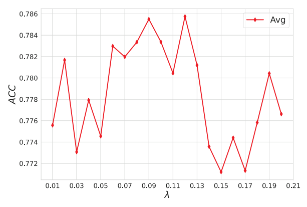


Fig. 9. The impact of different values of different λ on `Office-Home` dataset.

dataset, and the average results of all tasks are shown in Fig. 9. Noting that MoCoGrad with $\lambda = 0.12$ performs overall better than other values on the `Office-Home` dataset. The MoCoGrad with a larger value of $\lambda > 0.13$ or lower value of $\lambda < 0.06$, performs unsatisfactorily overall all tasks. One possible reason is that larger or lower λ can cause excessive pruning of conflicting gradients or insufficient pruning of conflicting gradients by proposed MoCoGrad. Therefore, that can be pretty detrimental to some tasks' performance and degrade the generalization capability of MTL models.

VII. Conclusions

In this paper, we propose a novel multi-task learning method for mitigating task conflicts and improving the performance of all tasks simultaneously, namely MoCoGrad. MoCoGrad leverages the momentum information of tasks to calibrate the gradients of conflicting tasks, which avoids the gradient of tasks being significantly deviated from the normal value due to the influence of noisy data from each mini-batch. We also provide a theoretical analysis of the convergence and convergence rate of our MoCoGrad. Sufficient experiments on six public MTL benchmarks demonstrate the effectiveness and superiority of our approach compared to all comparison baselines.

VIII. Acknowledgment

This work was supported by the National Natural Science Foundation of China (No. 62076079), and The Guangdong Major Project of Basic and Applied Basic Research(No.2019B030302002), and The Major Key Project of PCL(Grant No. PCL2022A03), and Shenzhen Science and Technology Program (Grant No. ZDSYS20210623091809029).

References

[1] J. Chen, B. Zhu, C. Ngo, T. Chua, and Y. Jiang, "A study of multi-task and region-wise deep learning for food ingredient recognition," *IEEE Trans. Image Process.*, vol. 30, pp. 1514–1526, 2021.

[2] M. Yang, W. Huang, W. Tu, Q. Qu, Y. Shen, and K. Lei, "Multitask learning and reinforcement learning for personalized dialog generation: An empirical study," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 32, no. 1, pp. 49–62, 2021.

[3] Q. Liao, H. Chai, H. Han, X. Zhang, X. Wang, W. Xia, and Y. Ding, "An integrated multi-task model for fake news detection," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 11, pp. 5154–5165, 2022.

[4] L. Wang, X. Xu, K. Ouyang, H. Duan, Y. Lu, and H. Zheng, "Self-supervised dual-channel attentive network for session-based social recommendation," in *38th IEEE International Conference on Data Engineering, ICDE , Kuala Lumpur, Malaysia, May 9-12*. IEEE, 2022, pp. 2034–2045.

[5] L. Luo, Y. Li, B. Gao, S. Tang, S. Wang, J. Li, T. Zhu, J. Liu, Z. Li, and S. Pan, "Mamdr: A model agnostic learning framework for multi-domain recommendation," in *39th IEEE International Conference on Data Engineering, ICDE*. IEEE, 2023.

[6] M. Teichmann, M. Weber, J. M. Zöllner, R. Cipolla, and R. Urtasun, "Multinet: Real-time joint semantic reasoning for autonomous driving," in *IEEE Intelligent Vehicles Symposium, IV, Changshu, Suzhou, China, June 26-30*. IEEE, 2018, pp. 1013–1020.

[7] R. Caruana, "Multitask learning," *Machine learning*, vol. 28, no. 1, pp. 41–75, 1997.

[8] B. Liu, X. Liu, X. Jin, P. Stone, and Q. Liu, "Conflict-averse gradient descent for multi-task learning," in *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems, NeurIPS, December 6-14, virtual*, 2021, pp. 18 878–18 890.

[9] Z. Wang, Y. Tsvetkov, O. Firat, and Y. Cao, "Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models," in *9th International Conference on Learning Representations, ICLR, Virtual Event, Austria, May 3-7*. OpenReview.net, 2021.

[10] Z. Hu, Z. Zhao, X. Yi, T. Yao, L. Hong, Y. Sun, and E. H. Chi, "Improving multi-task generalization via regularizing spurious correlation," in *NeurIPS*, 2022.

[11] S. Wang, Y. Li, H. Li, T. Zhu, Z. Li, and W. Ou, "Multi-task learning with calibrated mixture of insightful experts," in *38th IEEE International Conference on Data Engineering, ICDE*. IEEE, 2022, pp. 3307–3319.

[12] H. Chai, S. Tang, J. Cui, Y. Ding, B. Fang, and Q. Liao, "Improving multi-task stance detection with multi-task interaction network," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP, Abu Dhabi, United Arab Emirates, December 7-11*. Association for Computational Linguistics, 2022, pp. 2990–3000.

[13] H. Chai, J. Cui, Y. Wang, M. Zhang, B. Fang, and Q. Liao, "Improving gradient trade-offs between tasks in multi-task text classification," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL*. Association for Computational Linguistics, 2023.

[14] S. Zhai, B. Liu, D. Yang, and Y. Xiao, "Group buying recommendation model based on multi-task learning," in *39th IEEE International Conference on Data Engineering, ICDE*, 2023.

[15] X. Han, X. Zhu, L. Yu, L. Zhang, Y. Song, and T. Xiang, "Fame-vil: Multi-tasking vision-language model for heterogeneous fashion tasks," *CoRR*, vol. abs/2303.02483, 2023.

[16] M. Xu, L. Jiang, C. Li, Z. Wang, and X. Tao, "Viewport-based CNN: A multi-task approach for assessing 360° video quality," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 4, pp. 2198–2215, 2022.

[17] S. Vandenhende, S. Georgoulis, W. V. Gansbeke, M. Proesmans, D. Dai, and L. V. Gool, "Multi-task learning for dense prediction tasks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 7, pp. 3614–3633, 2022.

[18] J. Ma, Z. Zhao, X. Yi, J. Chen, L. Hong, and E. H. Chi, "Modeling task relationships in multi-task learning with multi-gate mixture-of-experts," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, Y. Guo and F. Farooq, Eds. ACM, 2018, pp. 1930–1939.

[19] O. Sener and V. Koltun, "Multi-task learning as multi-objective optimization," in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems, NeurIPS, December 3-8, Montréal, Canada*, S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., 2018, pp. 525–536.

[20] X. Lin, H. Zhen, Z. Li, Q. Zhang, and S. Kwong, "Pareto multi-task learning," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems, NeurIPS*, 2019, pp. 12 037–12 047.

[21] D. Mahapatra and V. Rajan, "Multi-task learning with user preferences: Gradient descent with controlled ascent in pareto optimization," in *Proceedings of the 37th International Conference on Machine Learning, ICML*, vol. 119. PMLR, 2020, pp. 6597–6607.

[22] P. Ma, T. Du, and W. Matusik, "Efficient continuous pareto exploration in multi-task learning," in *Proceedings of the 37th International Conference on Machine Learning, ICML*, vol. 119. PMLR, 2020, pp. 6522–6531.

[23] T. Yu, S. Kumar, A. Gupta, S. Levine, K. Hausman, and C. Finn, "Gradient surgery for multi-task learning," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems, NeurIPS, December 6-12, virtual*, 2020.

[24] A. Navon, A. Shamsian, I. Achituve, H. Maron, K. Kawaguchi, G. Chechik, and E. Fetaya, "Multi-task learning as a bargaining game," in *International Conference on Machine Learning, ICML , 17-23 July, Baltimore, Maryland, USA*, ser. Proceedings of Machine Learning Research, vol. 162. PMLR, 2022, pp. 16 428–16 446.

[25] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert, "Cross-stitch networks for multi-task learning," in *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Las Vegas, NV, USA, June 27-30*. IEEE Computer Society, 2016, pp. 3994–4003.

[26] S. Liu, E. Johns, and A. J. Davison, "End-to-end multi-task learning with attention," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Long Beach, CA, USA, June 16-20*. Computer Vision Foundation / IEEE, 2019, pp. 1871–1880.

[27] H. Tang, J. Liu, M. Zhao, and X. Gong, "Progressive layered extraction (PLE): A novel multi-task learning (MTL) model for personalized recommendations," in *Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, September 22-26*. ACM, 2020, pp. 269–278.

[28] H. Chai, Z. Yin, Y. Ding, L. Liu, B. Fang, and Q. Liao, "A model-agnostic approach to mitigate gradient interference for multi-task learning," *IEEE Transactions on Cybernetics*, 2022.

[29] N. Qian, "On the momentum term in gradient descent learning algorithms," *Neural Networks*, vol. 12, no. 1, pp. 145–151, 1999.

[30] T. Tieleman, G. Hinton *et al.*, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural networks for machine learning*, vol. 4, no. 2, pp. 26–31, 2012.

[31] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR, San Diego, CA, USA, May 7-9, Conference Track Proceedings*, 2015.

[32] Y. Zhang and Q. Yang, "A survey on multi-task learning," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 12, pp. 5586–5609, 2022.

[33] T. Evgeniou and M. Pontil, "Regularized multi–task learning," in *Proceedings of the Tenth International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22-25*. ACM, 2004, pp. 109–117.

[34] A. Argyriou, T. Evgeniou, and M. Pontil, "Convex multi-task feature learning," *Mach. Learn.*, vol. 73, no. 3, pp. 243–272, 2008.

[35] G. Bai and L. Zhao, "Saliency-regularized deep multi-task learning," in *The 28th Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, August 14 - 18*. ACM, 2022, pp. 15–25.

[36] I. Kokkinos, "Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Honolulu, HI, USA, July 21-26*. IEEE Computer Society, 2017, pp. 5454–5463.

[37] M. Long, Z. Cao, J. Wang, and P. S. Yu, "Learning multiple tasks with multilinear relationship networks," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems, December 4-9, Long Beach, CA, USA*, 2017, pp. 1594–1603.

[38] A. Kendall, Y. Gal, and R. Cipolla, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Salt Lake City, UT, USA, June 18-22*. Computer Vision Foundation / IEEE Computer Society, 2018, pp. 7482–7491.

[39] Y. Lu, A. Kumar, S. Zhai, Y. Cheng, T. Javidi, and R. S. Feris, "Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Honolulu, HI, USA, July 21-26*. IEEE Computer Society, 2017, pp. 1131–1140.

[40] S. Vandenhende, S. Georgoulis, L. V. Gool, and B. D. Brabandere, "Branched multi-task networks: Deciding what layers to share," in *31st British Machine Vision Conference, BMVC, Virtual Event, UK, September 7-10*. BMVA Press, 2020.

[41] 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," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems, December 6-12, virtual*, 2020.

[42] X. Sun, R. Panda, R. Feris, and K. Saenko, "Adashare: Learning what to share for efficient deep multi-task learning," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems, NeurIPS, December 6-12, virtual*, 2020.

[43] H. Li, Y. Wang, Z. Lyu, and J. Shi, "Multi-task learning for recommendation over heterogeneous information network," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 2, pp. 789–802, 2022.

[44] Z. Chen, V. Badrinarayanan, C. Lee, and A. Rabinovich, "Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks," in *Proceedings of the 35th International Conference on Machine Learning, ICML, Stockholmsmässan, Stockholm, Sweden, July 10-15*, ser. Proceedings of Machine Learning Research, vol. 80. PMLR, 2018, pp. 793–802.

[45] J. Bernstein, Y. Wang, K. Azizzadenesheli, and A. Anandkumar, "SIGNSGD: compressed optimisation for non-convex problems," in *Proceedings of the 35th International Conference on Machine Learning, ICML, Stockholmsmässan, Stockholm, Sweden, July 10-15*, ser. Proceedings of Machine Learning Research, vol. 80. PMLR, 2018, pp. 559–568.

[46] H. Robbins and S. Monro, "A stochastic approximation method," *The annals of mathematical statistics*, pp. 400–407, 1951.

[47] J. C. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *J. Mach. Learn. Res.*, vol. 12, pp. 2121–2159, 2011.

[48] P. Li, R. Li, Q. Da, A. Zeng, and L. Zhang, "Improving multi-scenario learning to rank in e-commerce by exploiting task relationships in the label space," in *The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23*. ACM, 2020, pp. 2605–2612.

[49] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Trans. Interact. Intell. Syst.*, vol. 5, no. 4, pp. 19:1–19:19, 2016.

[50] R. Ramakrishnan, P. O. Dral, M. Rupp, and O. A. Von Lilienfeld, "Quantum chemistry structures and properties of 134 kilo molecules," *Scientific data*, vol. 1, no. 1, pp. 1–7, 2014.

[51] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *12th European Conference on Computer Vision, Florence, Italy, October 7-13, Proceedings, Part V*, vol. 7576. Springer, 2012, pp. 746–760.

[52] 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 *IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Las Vegas, NV, USA, June 27-30*. IEEE Computer Society, 2016, pp. 3213–3223.

[53] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, "Deep hashing network for unsupervised domain adaptation," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR , Honolulu, HI, USA, July 21-26*. IEEE Computer Society, 2017, pp. 5385–5394.

[54] B. Lin, F. YE, Y. Zhang, and I. Tsang, "Reasonable effectiveness of random weighting: A litmus test for multi-task learning," *Transactions on Machine Learning Research*, 2022. [Online]. Available: https://openreview.net/forum?id=jjtFD8A1Wx

[55] L. Liu, Y. Li, Z. Kuang, J. Xue, Y. Chen, W. Yang, Q. Liao, and W. Zhang, "Towards impartial multi-task learning," in *9th International Conference on Learning Representations, ICLR, Virtual Event, Austria, May 3-7*. OpenReview.net, 2021.

[56] B. Lin and Y. Zhang, "LibMTL: A python library for multi-task learning," *arXiv preprint arXiv:2203.14338*, 2022.

[57] Q. Chen, H. Zhao, W. Li, P. Huang, and W. Ou, "Behavior sequence transformer for e-commerce recommendation in alibaba," *CoRR*, vol. abs/1905.06874, 2019. [Online]. Available: http://arxiv.org/abs/1905.06874

[58] L. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Computer Vision - ECCV - 15th European Conference, Munich, Germany, September 8-14, Proceedings, Part VII*, ser. Lecture Notes in Computer Science, vol. 11211. Springer, 2018, pp. 833–851.