A CLOSER LOOK AT LOSS WEIGHTING IN MULTI-TASK LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-Task Learning (MTL) has achieved great success in various fields, however, how to balance different tasks to avoid negative effects is still a key problem. To achieve the task balancing, there exist many works to balance task losses or gradients. In this paper, we unify eight representative task balancing methods from the perspective of loss weighting and provide a consistent experimental comparison. Moreover, we surprisingly find that training a MTL model with random weights sampled from a distribution can achieve comparable performance over state-of-the-art baselines. Based on this finding, we propose a simple yet effective weighting strategy called Random Loss Weighting (RLW), which can be implemented in only one additional line of code over existing works. Theoretically, we analyze the convergence of RLW and reveal that RLW has a higher probability to escape local minima than existing models with fixed task weights, resulting in a better generalization ability. Empirically, we extensively evaluate the proposed RLW method on six image datasets and four multilingual tasks from the XTREME benchmark to show the effectiveness of the proposed RLW strategy when compared with state-of-the-art strategies.

1 INTRODUCTION

Multi-Task Learning (MTL) (Zhang & Yang, 2021; Vandenhende et al., 2021) aims to jointly train several related tasks to improve their generalization performance by leveraging common knowledge among them. Since this learning paradigm can not only significantly reduce the model size and increase the inference speed but also improve the performance, it has been successfully applied in various fields of deep learning, such as Computer Vision (CV) (Vandenhende et al., 2021), Natural Language Processing (NLP) (Chen et al., 2021), reinforcement learning (Zhang & Yang, 2021) and so on. However, when all the tasks are not related enough, which may be reflected via conflicting gradients or dominating gradients (Yu et al., 2020), it is more difficult to train a multi-task model than training them separately because some tasks dominantly influence model parameters, leading to unsatisfactory performance for other tasks. This phenomenon is related to the *task balancing* problem (Vandenhende et al., 2021) in MTL. Recently, several works focus on tackling this issue from an optimization perspective via balancing task losses or gradients.

In this paper, we investigate eight State-Of-The-Art (SOTA) task balancing approaches and unify them as loss weighting strategies. According to the way of generating loss weights, those methods can be divided into three types, including the solving approach such as directly solving a quadratic optimization problem in a multi-objective formulation as weights (Sener & Koltun, 2018), the calculating approach such as projecting conflict gradients (Yu et al., 2020), and the learning approach such as learning weights in a gradient descent manner (Chen et al., 2018b). On the other hand, since there are some discrepancies of the implementation details such as using different backbone networks for training or different metrics for the evaluation among those SOTA weighting methods, leading to inconsistent comparisons, we provide a unified testbed on six CV datasets and four multilingual problems from the XTREME benchmark (Hu et al., 2020) for those SOTA weighting strategies to show a fair comparison.

In addition, inspired by dynamic weighting processes in those SOTA strategies where loss weights vary over training iterations or epochs, we have a sudden whim: *what will happen if a MTL model is trained with random loss weights?* Specifically, in each training iteration, we first sample the loss weights from a distribution with some normalization and then minimize the aggregated loss weighted by the normalized random weights. Surprisingly, this seemingly unreliable method can not only converge but also achieve comparable performance with the SOTA weighting strategies. Based on this observations, we proposed a simple yet effective weighting strategy for MTL, called Random Loss Weighting (RLW). It is very easy to implement RLW by adding only one line of code and this strategy does not incur any additionally computational cost. An implementation example of RLW in PyTorch (Paszke et al., 2019) is shown below.

```python
outputs = model(inputs)
loss = criterion(outputs, labels) # [1, task_num] vector
weight = F.softmax(torch.randn(task_num), dim=-1) # RLW is only this!
loss = torch.sum(loss*weight)
optimizer.zero_grad()
loss.backward()
optimizer.step()
```

To show the effectiveness of RLW, we provide both theoretical analyses and empirical evaluations. Firstly, the objective function of RLW can be considered as a doubly stochastic optimization problem when optimizing by stochastic gradient descent or its variants, where the randomness is from both the mini-batch sampling of the data for each task and the random sampling of loss weights. From this perspective, we give a convergence analysis for RLW. Besides, we can show that RLW has a higher probability to escape local minima when compared with fixing loss weights, resulting in a better generalization performance. Empirically, as described before, we compare RLW with SOTA weighting approaches on six CV datasets and four multilingual problems to show its competitive performance.

In summary, the main contributions of this paper are four-fold.

- We provide a unified testbed on six multi-task computer vision datasets and four multilingual problems from the XTREME benchmark for a fair comparison among eight SOTA weighting methods and the proposed RLW method.
- We propose a simple yet effective RLW strategy, which we think is an ignored baseline in MTL.
- We provide the convergence guarantee and effectiveness analysis for RLW.
- Experiments show that RLW can achieve comparable performance with SOTA weighting methods without bringing any additionally computational cost.

2 Preliminary

Suppose there are T tasks and task t has its corresponding dataset $\mathcal{D}_t$. An MTL model usually contains two parts of parameters: task-sharing parameters θ and task-specific parameters $\{\psi_t\}_{t=1}^T$. For example, in CV, θ usually denotes parameters in the feature extractor shared by all tasks and ψ_t represents the task-specific output module for task t. Let $\ell_t(\cdot;\theta,\psi_t)$ denotes a task-specific loss function for task t. Then the objective function of a MTL model can be formulated as

$$\mathcal{L}_{\text{MTL}} = \sum_{t=1}^{T} \lambda_t \ell_t(\mathcal{D}_t;\theta,\psi_t), \tag{1}$$

where $\ell_t(\mathcal{D}_t;\theta,\psi_t)$ denotes the average loss on $\mathcal{D}_t$ for task t and $\{\lambda_t\}_{t=1}^T$ are task-specific loss weights with a constraint that $\lambda_t \geq 0$ for all t. When minimizing Eq. (1) by Stochastic Gradient Descent (SGD) or its variants, the task-specific parameters $\{\psi_t\}_{t=1}^T$ are simply updated based on the corresponding task gradient $\nabla_{\psi_t}\ell_t(\mathcal{D}_t;\theta,\psi_t)$, while the task-sharing parameters θ should be updated by all the task losses jointly as

$$\theta = \theta - \eta \sum_{t=1}^{T} \lambda_t \nabla_\theta \ell_t(\mathcal{D}_t;\theta,\psi_t), \tag{2}$$

where η is a learning rate. Obviously, for the update of task-sharing parameters θ, the loss weighting (i.e., $\{\lambda_t\}_{t=1}^T$ in Eq. (1)) influences θ via the aggregated gradient essentially and the gradient weighting in Eq. (2) during the backward process has the same effect as the loss weighting when they are using the same weights. Therefore, we can ignore the level on which the weights act and focus on the generation of weights. For simplicity, these two types of weighting are all referred to as loss weighting in the following sections.

Apparently, the most simple method for loss weighting is to set a same weight for every tasks, i.e., without loss of generality, $\lambda_t = \frac{1}{T}$ for all t. This approach is a common baseline in MTL and it is called Equally Weighting (**EW**) in this paper. To tackle the task balancing problem and improve the performance of MTL model, there are several works to study how to generate appropriate weights. In this paper, we investigate eight SOTA weighting strategies, i.e. Gradient Normalization (**GradNorm**) (Chen et al., 2018b), Uncertainty Weights (**UW**) (Kendall et al., 2018), **MGDA** (Sener & Koltun, 2018), Dynamic Weight Average (**DWA**) (Liu et al., 2019a), Projecting Conflicting Gradient (**PCGrad**) (Yu et al., 2020), Gradient sign Dropout (**GradDrop**) (Chen et al., 2020), Impartial Multi-Task Learning (**IMTL**) (Liu et al., 2021), and Gradient Vaccine (**GradVac**) (Wang et al., 2021).

According to different ways of generating loss weights, we categorize those loss weighting strategies into three types: the learning approach, the solving approach, and the calculating approach. Both GradNorm and UW consider the loss weights $\{\lambda_t\}_{t=1}^T$ in Eq. (1) as learnable parameters and explicitly optimize them by gradient descent. MGDA casts MTL as a multi-objective optimization problem and directly solves the loss weights $\{\lambda_t\}_{t=1}^T$ in Eq. (1) by solving a quadratic programming problem. DWA, PCGrad, GradDrop and GradVac directly compute the weights $\{\lambda_t\}_{t=1}^T$ by combining gradients and/or losses of all the tasks. IMTL is a hybrid strategy, which combines the learning and the calculating approaches. We summarize those strategies from the perspective of loss weighting in Table 5 in Appendix A.

We now unify those eight SOTA methods as loss weighting strategies, i.e., generating loss weights $\{\lambda_t\}_{t=1}^T$ in Eq. (1). Noticeably, almost all the existing strategies except EW need to incur intensive computation to generate loss weights in every iteration, such as solving a quadratic optimization problem in MGDA, and operating on high-dimensional gradients in PCGrad, GradDrop, IMTL, and GradVac. Different from those strategies, the proposed RLW strategy generates loss weights in a sampling way, thus it is as efficient as EW without bringing additionally computational costs.

3 THE RLW METHOD

In this section, we introduce the proposed RLW method. The RLW method is a simple loss weighting strategy and it considers the loss weights $\boldsymbol{\lambda} = (\lambda_1, \cdots, \lambda_T) \in \mathbb{R}^T$ as random variables. Formally, the objective function of the RLW method is formulated as

$$\mathcal{L}_{\mathrm{RLW}}(\theta) = \mathbb{E}_{\boldsymbol{\lambda}}\left[\boldsymbol{\lambda}^\top \boldsymbol{\ell}(\boldsymbol{\mathcal{D}};\theta)\right] = \mathbb{E}_{\boldsymbol{\lambda}}[\boldsymbol{\lambda}]^\top \boldsymbol{\ell}(\boldsymbol{\mathcal{D}};\theta), \tag{3}$$

where $\mathbb{E}[\cdot]$ denotes the expectation and $\boldsymbol{\ell}(\boldsymbol{\mathcal{D}};\theta) = (\ell_1(\mathcal{D}_1;\theta), \cdots, \ell_T(\mathcal{D}_T;\theta))$ where we omit the task-specific parameters $\{\psi_t\}_{t=1}^T$ in Eq. (3) for brevity. To guarantee loss weights in $\boldsymbol{\lambda}$ to be non-negative, we can first sample $\tilde{\boldsymbol{\lambda}} = (\tilde{\lambda}_1, \cdots, \tilde{\lambda}_T)$ from any distribution $p(\tilde{\boldsymbol{\lambda}})$ and then normalize $\tilde{\boldsymbol{\lambda}}$ into $\boldsymbol{\lambda}$ via a mapping f, where $f : \mathbb{R}^T \to \Delta^T$ is a normalization function for example softmax function and Δ^T denotes a convex hull in $\mathbb{R}^T$, i.e. $\boldsymbol{\lambda} \in \Delta^T$ means $\sum_{t=1}^T \lambda_t = 1$ and $\lambda_t \geq 0$ for all t. Note that in most cases $p(\boldsymbol{\lambda})$ is different from $p(\tilde{\boldsymbol{\lambda}})$.

In Eq. (3), $p(\boldsymbol{\lambda})$ is usually too complex to compute its expectation $\mathbb{E}_{\boldsymbol{\lambda}}[\boldsymbol{\lambda}]$, thus a stochastic approximation scheme is adopted to minimize Eq. (3). When the mini-batch SGD (Bottou, 1991) or its variants is used to minimize Eq. (3) as most deep learning models did, Eq. (3) can be viewed as a doubly stochastic optimization problem, where the randomness is from both the mini-batch data sampling for each task and the randomly sampling of the loss weights. In the following, we show that the approximated gradient $\nabla_\theta \boldsymbol{\lambda}^\top \boldsymbol{\ell}(\tilde{\boldsymbol{\mathcal{D}}};\theta)$ is an unbiased estimation of the true gradient of $\mathcal{L}_{\mathrm{RLW}}(\theta)$, where $\tilde{\boldsymbol{\mathcal{D}}}$ denotes a mini-batch data sampled from all the tasks. Specifically, as $\tilde{\mathcal{D}}_t$ is a mini-batch data sampled from $\mathcal{D}_t$ to calculate the stochastic gradient $\nabla_\theta \ell_t(\tilde{\mathcal{D}}_t;\theta)$ to approximate the full gradient $\nabla_\theta \ell_t(\mathcal{D}_t;\theta)$ for task t, we have $\mathbb{E}_{\tilde{\boldsymbol{\mathcal{D}}}}[\nabla_\theta \boldsymbol{\ell}(\tilde{\boldsymbol{\mathcal{D}}};\theta)] = \nabla_\theta \boldsymbol{\ell}(\boldsymbol{\mathcal{D}};\theta)$. Therefore, when we further randomly sample a weight vector $\boldsymbol{\lambda}$, we have

$$\mathbb{E}_{\boldsymbol{\lambda}}\left[\mathbb{E}_{\tilde{\boldsymbol{\mathcal{D}}}}[\nabla_\theta \boldsymbol{\lambda}^\top \boldsymbol{\ell}(\tilde{\boldsymbol{\mathcal{D}}};\theta)]\right] = \mathbb{E}_{\boldsymbol{\lambda}}[\boldsymbol{\lambda}]^\top \nabla_\theta \boldsymbol{\ell}(\boldsymbol{\mathcal{D}};\theta) = \nabla_\theta \mathcal{L}_{\mathrm{RLW}}(\theta),$$

which verifies that $\nabla_\theta \boldsymbol{\lambda}^\top \boldsymbol{\ell}(\tilde{\boldsymbol{\mathcal{D}}};\theta)$ is an unbiased estimation.

In practice, it is very easy to implement the RLW method without modifying network architecture or bringing additionally computational costs. Specifically, in each iteration, we first sample $\tilde{\boldsymbol{\lambda}}$ from $p(\tilde{\boldsymbol{\lambda}})$ and normalize it to obtain $\boldsymbol{\lambda}$ via appropriate normalization function f, and then minimize the aggregated loss weighted by $\boldsymbol{\lambda}$. The entire algorithm of RLW (i.e., minimizing Eq. (3)) via SGD is shown in Algorithm 1. Apparently, the only difference between the proposed RLW strategy and the widely used EW strategy is Line 7 in Algorithm 1 and it is very easy to implement with only one line of code.

Algorithm 1 Optimization Algorithm for RLW by SGD

```
Input: numbers of iterations $K$, numbers of tasks $T$, learning rate $\eta$, dataset $\{\mathcal{D}_t\}_{t=1}^T$, weight distribution $p(\tilde{\boldsymbol{\lambda}})$
1: Randomly initialized $\theta_0$;
2: for $k = 1$ to $K$ do
3:     for $t = 1$ to $T$ do
4:         Sample a mini-batch data $\tilde{\mathcal{D}}_t$ from $\mathcal{D}_t$;
5:         Compute loss $\ell_t(\tilde{\mathcal{D}}_t;\theta_k)$;
6:     end for
7:     Sample weights $\tilde{\boldsymbol{\lambda}}$ from $p(\tilde{\boldsymbol{\lambda}})$ and normalize it into $\boldsymbol{\lambda}$ via $f$;    ▷ RLW is only this
8:     $\theta_{k+1} = \theta_k - \eta\nabla_\theta \sum_{t=1}^T \lambda_t \ell_t(\tilde{\mathcal{D}}_t;\theta_k)$;
9: end for
```

In this paper, we use six different distributions for $p(\tilde{\boldsymbol{\lambda}})$ in the proposed RLW method, including uniform distribution between 0 and 1 (denoted by **Uniform**), standard normal distribution (denoted by **Normal**), Dirichlet distribution with $\alpha = 1$ (denoted by **Dirichlet**), Bernoulli distribution with probability $1/2$ (denoted by **Bernoulli**), Bernoulli distribution with probability $1/2$ and a constraint $\sum_{t=1}^T \tilde{\lambda}_t = 1$ (denoted by **constrained Bernoulli**), and normal distribution with a random mean and a random variance both sampling from a uniform distribution $\mathcal{U}(0,1)$ for each task (denoted by **random Normal**). We set f as a function of $f(\tilde{\boldsymbol{\lambda}}) = \tilde{\boldsymbol{\lambda}}/(\sum_{t=1}^T \tilde{\lambda}_t)$ if $p(\tilde{\boldsymbol{\lambda}})$ is the Bernoulli distribution or the constrained Bernoulli distribution and a softmax function for the other types of distribution. When sampling from the first five types of distribution, $\mathbb{E}[\boldsymbol{\lambda}]$ is simply proportional to $(\frac{1}{T},\cdots,\frac{1}{T})$, thus it is fair to compare with the EW strategy. When $p(\tilde{\boldsymbol{\lambda}})$ is a random Normal distribution, it means each $\tilde{\lambda}_t$ is sampled from a normal distribution with random mean and variance, thus it is intractable to compute the expectation for $p(\boldsymbol{\lambda})$ and combining with such distribution can further show the effectiveness of RLW.

When sampling from a Bernoulli distribution, the weights for all tasks are either 0 or 1, i.e., $\lambda_t \in \{0,1\}$ for all t. In this way, just a subset of tasks contributes to updating the task-sharing parameters θ. This manner can be viewed as the mini-batch sampling on the task level. If considering an additional constraint that $\sum_{t=1}^T \tilde{\lambda}_t = 1$, it implies only one task is involved in the update of the task-sharing parameters in each iteration. Although there are some works (Dong et al., 2015; Liu et al., 2015a; Søgaard & Goldberg, 2016; Subramanian et al., 2018; Sanh et al., 2019; Liu et al., 2019b) adopting this strategy to train a MTL model, it is a special case in the proposed RLW strategy and beyond existing works, we also provide theoretical analyses to show the effectiveness of the proposed RLW method.

4 Analysis

As the optimization procedure of the RLW method can be viewed as the doubly stochastic optimization, this strategy increases the randomness compared with the fixed loss weights methods optimizing via SGD (denoted by FW), where EW is a special case. In this section, we focus on analyzing how the extra randomness from the loss weights sampling affects the convergence and effectiveness of RLW compared with FW.

In the case of without misunderstanding, we simply use $\ell_t(\theta)$ instead of $\ell_t(\mathcal{D}_t;\theta)$ to denote the loss function of task t for brevity in this section and Appendix B. For the ease of the analysis, we need to make the following assumption.

Assumption 1. *The loss function $\ell_t(\theta)$ of task t is L_t-Lipschitz continuous w.r.t θ, and that $\mathbb{E}_{\mathcal{D}_t}[\|\ell_t(\mathcal{D}_t;\theta)\|^2] = \sigma_t^2$. Loss weights in $\boldsymbol{\lambda}$ satisfy $\mathbb{E}_{\boldsymbol{\lambda}}[\boldsymbol{\lambda}] = \boldsymbol{\mu}$.*

In the following theorem, we analyze the convergence property of Algorithm 1 for RLW.

Theorem 1. *Suppose the loss function $\ell_t(\theta)$ of task t is c_t-strongly convex. We define $\theta_* = \arg\min_\theta \mathcal{L}_{\mathrm{RLW}}(\theta)$ and denote by θ_k the solution in the k-th iteration. When η, the step size or equivalently the learning rate in SGD, satisfies $\eta \leq 1/2c$, where $c = \min_{1\leq t\leq T}\{c_t\}$, under Assumption 1 we have*

$$\mathbb{E}[\|\theta_k - \theta_*\|^2] \leq (1-2\eta c)^k \|\theta_0 - \theta_*\|^2 + \frac{\eta\kappa}{2c}, \tag{4}$$

where $\kappa = \sum_{t=1}^T \sigma_t^2$. Then for any positive ε, $\mathbb{E}[\|\theta_k - \theta_*\|^2] \leq \varepsilon$ can be achieved after $k = \frac{\kappa}{2\varepsilon c^2}\log\left(\frac{\varepsilon_0}{\varepsilon}\right)$ iterations with $\eta = \frac{\varepsilon c}{\kappa}$, where $\varepsilon_0 = \mathbb{E}[\|\theta_0 - \theta_*\|^2]$.

Theorem 1 shows that the RLW method with the fixed step size has a linear convergence up to a radius around the optimal solution, which is similar to FW according to the property of the standard SGD method (Moulines & Bach, 2011; Needell et al., 2016). Although RLW has a larger κ than FW, i.e., $\kappa_{\mathrm{FW}} = \sum_{t=1}^T \mu_t^2 \cdot \sum_{t=1}^T \sigma_t^2 \leq \kappa$, possibly requiring more iterations for RLW method to reach the same accuracy with FW, our empirical experiments in Appendix C.1 show that this does not cause much impact in practice.

We next analyze the effectiveness of the RLW method from the perspective of stochastic optimization. It is observed that the SGD method can escape sharp local minima and converge to a better solution than Gradient Descent (GD) techniques under various settings with the help of noisy gradients (Hardt et al., 2016; Kleinberg et al., 2018). Inspired by those works, we first provide the following Theorem 2 and then leverage this theorem to show that the extra randomness in the RLW method can help RLW to better escape sharp local minima and achieve a better generalization performance than FW.

For the ease of presentation, we introduction some notations. Here we consider the update step of these stochastic methods as $\theta_{k+1} = \theta_k - \eta(\nabla \boldsymbol{\mu}^\top \boldsymbol{\ell}(\theta_k) + \xi_k)$, where ξ_k is a noise with $\mathbb{E}[\xi_k] = 0$ and $\|\xi_k\|^2 \leq r$. Here r denotes the intensity of the noise. For the analysis, we construct an intermediate sequence $\varphi_k = \theta_k - \eta\nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\theta_k)$. Then we get $\mathbb{E}_{\xi_k}[\varphi_{k+1}] = \varphi_k - \eta\nabla\mathbb{E}_{\xi_k}[\boldsymbol{\mu}^\top\boldsymbol{\ell}(\varphi_k - \eta\xi_k)]$. Therefore, the sequence $\{\varphi_k\}$ can be regarded as an approximation of using GD to minimize the function $\mathbb{E}_{\xi_k}[\boldsymbol{\mu}^\top\boldsymbol{\ell}(\varphi - \eta\xi_k)]$.

Theorem 2. *Suppose $\nabla\ell_t(\theta)$ is M_t-Lipschitz continuous and $\|\xi_k\|^2 \leq r$. If the loss function $\ell_t(\theta)$ of task t is c_t-one point strongly convex w.r.t a local minimum θ_* after convolved with noise ξ, i.e., $\langle \nabla\mathbb{E}_\xi \ell_t(\varphi - \eta\xi), \varphi - \theta_*\rangle \geq c_t\|\varphi - \theta_*\|^2$, then under Assumption 1 we have $\|\varphi_K - \theta_*\|^2 \leq \frac{2\beta}{\rho\delta}$ with probability at least $1-\delta$ after $K = \frac{1}{\rho}\log\left(\frac{\rho\varepsilon_0}{\beta}\right)$ iterations with $\eta \leq \frac{c}{M^2}$, where $\varepsilon_0 = \mathbb{E}[\|\varphi_0 - \theta_*\|^2]$, $c = \min_{1\leq t\leq T}\{c_t\}$, $M = \max_{1\leq t\leq T}\{M_t\}$, $\rho = 2\eta c - \eta^2 M^2$ and $\beta = \eta^2 r^2(1+\eta M)^2$.*

Firstly, Theorem 2 only requires that $\ell_t(\theta)$ is c_t-one point strongly convex w.r.t θ_* after convolved with noise ξ, which is much weak than the convexity assumption and can hold for deep neural networks. Moreover, this theorem implies that for both RLW and FW methods, their solutions have a high probability to get close to a local minimum θ_* depending on the noise ξ. Note that by adding extra noise, the sharp local minimum will disappear and only the flat local minimum with a large diameter will still exist (Kleinberg et al., 2018). On the other hand, those flat local minima may satisfy one point strongly convexity assumption in Theorem 2, thus the diameter of the converged flat local minimum is affected by the noise intensity. Due to the extra randomness from loss weights sampling, the RLW method can provide a stronger noise (i.e. a larger r) than FW (referred to Appendix B.3). Hence RLW can better escape sharp local minima and converge to a flatter local minimum than FW, resulting in better generalization performance.

5 Experiments

In this section, we empirically evaluate the proposed RLW method by conducting experiments on six computer vision datasets (i.e., NYUv2, CityScapes, CelebA, PASCAL-Context, Office-31, and Office-Home) and four multilingual problems from the XTREME benchmark (Hu et al., 2020). *Due to page limit, experimental results of the CityScapes, CelebA, Office-31, Office-Home datasets and two multilingual problems are put in Appendix C.*

5.1 DATASETS

The **NYUv2** dataset (Silberman et al., 2012) is an indoor scene understanding dataset, which consists of video sequences recorded by the RGB and Depth cameras in the Microsoft Kinect. It contains 795 and 654 images with ground-truths for training and validation, respectively. This dataset includes three tasks: 13-class semantic segmentation, depth estimation, and surface normal prediction.

The **PASCAL-Context** dataset (Mottaghi et al., 2014) is an annotation extension of the PASCAL VOC 2010 challenge. It contains 10,103 images, which are divided into two parts: 4,998 for training and 5,105 for validation. We consider two tasks with annotations in this dataset: 21-class semantic segmentation and 7-class human part segmentation. By following (Maninis et al., 2019), we generate two additional tasks, including the saliency estimation and surface normal estimation tasks where their ground-truth labels are computed by the label distillation using pretrained state-of-the-art models (Bansal et al., 2017; Chen et al., 2018a).

The **XTREME benchmark** (Hu et al., 2020) is a large-scale multilingual multi-task benchmark for cross-lingual generalization evaluation, which covers fifty languages and contains nine tasks. We conduct experiments on four tasks containing Named Entity Recognition (NER), Part-Of-Speech (POS) tagging, Natural Language Inference (NIL), and Paraphrase Identification (PI) from this benchmark. On each task, we construct a multilingual problem by choosing the four languages with the largest number of data. The more details are provided in Appendix C.6.

5.2 IMPLEMENTATION DETAILS

The network architecture used adopt the hard-parameter sharing pattern (Caruana, 1993), which shares bottom layers of the network for all tasks and uses separate top layers for each task. Other architectures with more parameter sharing manners are provided in Section 5.7. The implementation details on each dataset are introduced in the following.

For the **NYUv2** dataset, the DeepLabV3+ architecture (Chen et al., 2018a) is used. Specifically, a ResNet-50 network pretrained on the ImageNet dataset with dilated convolutions (Yu et al., 2017) is used as a shared encoder among tasks and the Atrous Spatial Pyramid Pooling (ASPP) (Chen et al., 2018a) module is used as task-specific head for each task. Input images are resized to 288×384. The Adam optimizer (Kingma & Ba, 2015) with the learning rate as 10^{-4} and the weight decay as 10^{-5} is used for training and the batch size is set to 8. We use the cross-entropy loss, L_1 loss and cosine loss as the loss function of the semantic segmentation, depth estimation and surface normal prediction tasks, respectively. For the **PASCAL-Context** dataset, the network architecture is similar to that on NYUv2 dataset with a shallower ResNet-18 network used as the shared encoder due to constraints of computing resources. All input images are resized to 512×512. The Adam optimizer with both the learning rate and weight decay as 10^{-4} is applied for training and the batch size is set to 12. The cross-entropy loss is used for two segmentation tasks and saliency estimation task, while the normal estimation task uses the L_1 loss. For each multilingual problem in the **XTREME** benchmark, a pretrained multilingual BERT (mBERT) model (Devlin et al., 2019) implemented via the open source transformers library (Wolf et al., 2020) is used as the shared encoder among languages and a fully connected layer is used as the language-specific output layer for each language. The Adam optimizer with the learning rate as 2×10^{-5} and the weight decay as 10^{-8} is used for training and the batch size is set to 32. The cross-entropy loss is used for the four multilingual problems.

5.3 EVALUATION METRIC

To measure the performance of MTL models in a scalar metric, for homogeneous MTL problems (e.g., the Office-31 dataset) which contain tasks of the same type such as the classification task, we directly average the performance metrics among tasks. For heterogeneous MTL problems (e.g., the NYUv2 dataset) that contain tasks of different types, by following (Maninis et al., 2019; Vandenhende et al., 2021), we compute the average of the relative improvement over the EW method on each metric of each task as

$$\Delta_{\mathrm{p}} = 100\% \times \frac{1}{T}\sum_{t=1}^{T}\frac{1}{N_t}\sum_{n=1}^{N_t}\frac{(-1)^{p_{t,n}}(M_{t,n} - M_{t,n}^{\mathrm{EW}})}{M_{t,n}^{\mathrm{EW}}},$$

where N_t denotes the number of metrics in task t, $M_{t,n}$ denotes the performance of a task balancing strategy for the nth metric in task t, $M_{t,n}^{\mathrm{EW}}$ is defined similarly for the EW strategy, and $p_{t,n}$ is set to 1 if a higher value indicates better performance for the nth metric in task t and otherwise 0.

5.4 RESULTS ON THE NYUv2 DATASET

The results on the NYUv2 validation dataset are shown in Table 1. It is noticeable that the proposed RLW strategy can achieve comparable performance with SOTA baseline methods. Firstly, RLW with six weight distributions can always outperform EW, which implies that training in a doubly stochastic manner can have a better generalization ability. Secondly, RLW can achieve a balanced improvement on all tasks. That is, RLW has comparable or better performance on each metric in each task when compared with EW, resulting in a large Δ_{p}. Different from RLW, many baseline methods achieve unbalanced performance on all the tasks. For example, IMTL significantly outperforms other methods on the normal prediction task but has unsatisfactory performance on the other two tasks. Hence, this can be one advantage of the proposed RLW strategy since MTL aims to improve the generalization performance of each task as much as possible. Thirdly, RLW with some distributions (i.e., "constrained Bernoulli" and "Normal") can improve the generalization performance by more than 1%, which is significantly better than baseline methods. Even, RLW with the constrained Bernoulli distribution can entirely dominate not only the EW method but also some baseline methods such as GradNorm, UW, DWA, and GradDrop, on each task, which demonstrates the effectiveness of the RLW method.

Moreover, we compare the average time of training one epoch for each loss weighting strategy with the same batch size (i.e., 8) on a single NVIDIA GeForce RTX 3090 GPU. The relative training speed of each method over the EW method is reported as Δ_{t} in Table 1. Noticeably, the proposed RLW strategy is as computationally efficient as EW, while some baseline methods are compute-intensive. For example, PCGrad and GradVac take about twice the time of EW for each epoch because of computing the gradients of parameters. MGDA spends a lot of time to solve a complex quadratic programming problem. Furthermore, Δ_{t} of those baseline methods will increase as the network becomes deeper, while the proposed RLW strategy is architecture-agnostic and always as efficient as EW.

By combining the above analysis, we think that the proposed RLW method is an effective and efficient loss weighting strategy for MTL.

Table 1: Performance on the **NYUv2** validation dataset with three tasks: 13-class semantic segmentation, depth estimation, and surface normal prediction. The best results for each task on each measure are highlighted in **bold**. $\uparrow$ ($\downarrow$) indicates that the higher (lower) the result, the better the performance.

Weighting Strategy	**Segmentation**		**Depth**		**Surface Normal**					$\mathbf{\Delta_p}\uparrow$	$\mathbf{\Delta_t}\downarrow$
					Angle Distance		**Within t°**				
	mIoU↑	**Pix Acc↑**	**Abs Err↓**	**Rel Err↓**	**Mean↓**	**Median↓**	**11.25↑**	**22.5↑**	**30↑**		
EW	53.91	75.56	0.3840	0.1567	23.6338	17.2451	34.94	60.65	71.81	+0.00	×1.00
GradNorm	53.81	75.35	0.3863	0.1556	23.6106	17.2565	34.98	60.58	71.76	−0.06	×1.82
UW	53.15	75.41	0.3817	0.1576	23.6487	17.2040	34.98	60.71	71.80	−0.24	×1.01
MGDA	53.66	75.37	0.3864	0.1610	23.4757	16.9912	35.44	61.17	72.16	−0.35	×2.64
DWA	53.33	75.42	0.3834	0.1556	23.5806	17.1242	35.18	60.88	71.91	+0.07	**×1.00**
PCGrad	53.34	75.43	0.3857	0.1600	23.2293	16.6966	36.09	61.80	72.66	+0.12	×2.10
GradDrop	53.80	75.56	0.3857	0.1587	23.8726	17.1406	35.10	60.72	71.60	−0.33	×1.84
IMTL	52.90	74.88	0.3883	0.1632	**23.0534**	**16.5304**	**36.30**	**62.20**	**73.08**	−0.35	×1.82
GradVac	53.52	75.43	0.3840	0.1559	23.2892	16.8601	35.67	61.53	72.46	+0.48	×2.11
RLW (Uniform)	54.09	75.78	0.3826	0.1563	23.6272	17.2711	34.73	60.67	71.87	+0.17	**×1.00**
RLW (Normal)	54.19	**75.98**	0.3789	0.1570	23.1984	16.7944	35.71	61.74	72.77	+1.02	
RLW (Dirichlet)	53.54	75.45	0.3834	0.1547	23.6392	17.0715	35.28	60.92	71.88	+0.27	
RLW (Bernoulli)	53.72	75.62	0.3850	0.1610	23.1413	16.6591	36.08	61.98	72.86	+0.28	
RLW (constrained Bernoulli)	**54.32**	75.78	**0.3779**	**0.1533**	23.2101	16.9354	35.41	61.44	72.58	**+1.29**	
RLW (random Normal)	54.08	75.77	0.3815	0.1581	23.5598	16.9577	35.53	61.20	72.13	+0.39	

5.5 RESULTS ON THE PASCAL-CONTEXT DATASET

The results on the PASCAL-Context validation dataset are shown in Table 2. The empirical observations are similar to those on the NYUv2 dataset. Specifically, RLW with different distributions can outperform EW, which means RLW has a better generalization ability. Most baseline methods have unsatisfactory performance on this dataset and the best baseline, i.e., MGDA, achieves the largest Δ_{p} of 0.18%. Thus, RLW outperforms many baseline methods. Moreover, RLW with the constrained Bernoulli distribution achieves the highest improvement of 0.46%. Furthermore, RLW does achieve more balanced improvement than some baseline methods. For example, GradNorm performs not so good on the saliency estimation task, leading to the lowest Δ_{p}. Although IMTL significantly improves the performance of the surface normal estimation task, it performs unsatisfactorily on the other tasks especially the semantic segmentation task. Moreover, the relative training speed of each method over the EW method is similar to the NYUv2 dataset and hence we omit it in Table 2.

Table 2: Performance on the **PASCAL-Context** validation dataset with four tasks: 21-class semantic segmentation (abbreviated as SS), 7-class human parts segmentation (abbreviated as HPS), saliency estimation, and surface normal prediction. The best results for each task on each measure are highlighted in **bold**. ↑ (↓) indicates that the higher (lower) the result, the better the performance.

Weighting Strategy	SS	HPS	Saliency		Surface Normal					Δ_{p}↑
					Angle Distance		Within t°			
	mIoU↑	mIoU↑	mIoU↑	maxF↑	Mean↓	RMSE↓	11.25↑	22.5↑	30↑	
EW	64.52	58.71	64.31	77.11	17.6444	26.1634	42.24	75.93	86.97	+0.00
GradNorm	64.13	58.49	61.64	72.46	18.0455	26.4642	41.03	74.90	86.26	−1.94
UW	63.72	59.13	64.47	77.26	17.4962	26.0463	42.64	76.48	87.35	+0.09
MGDA	63.34	58.86	**64.79**	**77.54**	17.3070	25.8584	43.30	77.14	87.79	+0.18
DWA	64.32	58.61	64.30	77.15	17.4065	25.9242	42.72	76.71	87.59	+0.14
PCGrad	63.58	58.68	63.79	76.71	17.2376	25.8572	43.69	77.11	87.70	−0.08
GradDrop	64.04	**59.36**	62.31	73.10	17.3246	25.8966	43.29	76.92	87.63	−0.58
IMTL	62.67	58.35	62.92	73.21	**16.8026**	**25.4852**	**45.02**	**78.49**	**88.65**	−0.81
GradVac	62.99	58.63	64.30	77.15	17.1852	25.7621	43.55	77.41	88.00	−0.10
RLW (Uniform)	63.52	59.03	63.75	76.71	17.0261	25.6528	44.21	77.77	88.24	+0.27
RLW (Normal)	64.14	58.43	64.05	76.86	17.1794	25.7734	43.69	77.31	87.91	+0.16
RLW (Dirichlet)	63.97	58.88	64.30	77.19	17.2147	25.8093	43.67	77.33	87.89	+0.37
RLW (Bernoulli)	64.34	58.35	64.28	77.03	17.3379	25.9016	43.18	76.93	87.66	+0.11
RLW (constrained Bernoulli)	**65.07**	58.52	64.19	76.96	17.3377	25.9005	43.30	77.04	87.69	**+0.46**
RLW (random Normal)	64.09	59.15	63.84	76.76	17.3860	25.9472	42.98	76.73	87.54	+0.16

5.6 Results on the XTREME benchmark

Table 3: Performance on two multilingual problems, i.e. POS and PI from the **XTREME** benchmark. The best results for each language are highlighted in **bold**.

Weighting Strategy	POS (F1 Score)					PI (Accuracy)				
	`en`	`zh`	`te`	`vi`	**Avg**	`en`	`zh`	`de`	`es`	**Avg**
EW	95.02	88.89	91.16	87.11	90.55	94.09	84.59	89.44	90.24	89.59
GradNorm	95.01	88.91	91.88	87.06	90.71	94.39	**85.94**	**90.99**	91.44	90.69
UW	94.89	88.77	90.96	87.12	90.44	93.74	85.44	90.24	91.29	90.18
MGDA	95.08	88.97	92.35	87.12	90.88	94.64	84.99	89.84	90.89	90.09
DWA	95.02	89.03	91.87	**87.27**	90.80	94.69	84.99	89.49	91.44	90.15
PCGrad	94.85	88.42	90.72	86.71	90.18	94.19	85.49	89.09	91.24	90.00
GradDrop	95.08	89.06	90.65	87.17	90.49	94.29	84.44	89.69	90.94	89.84
IMTL	94.87	88.80	92.18	86.72	90.65	94.54	84.79	90.14	90.99	90.12
GradVac	94.87	88.41	90.62	86.47	90.09	94.29	84.94	89.19	90.89	89.83
RLW (Uniform)	95.06	89.00	92.31	86.93	90.83	94.69	85.79	90.29	91.94	90.68
RLW (Normal)	95.01	88.87	**92.86**	86.85	**90.90**	94.39	85.34	90.04	91.84	90.40
RLW (Dirichlet)	**95.16**	88.96	91.64	87.24	90.75	94.24	84.39	89.99	90.99	89.90
RLW (Bernoulli)	95.13	**89.10**	91.13	87.03	90.60	95.09	85.89	90.24	**91.99**	**90.80**
RLW (constrained Bernoulli)	94.98	89.05	92.33	86.87	90.81	94.69	85.49	90.19	90.99	90.34
RLW (random Normal)	95.07	89.00	91.10	87.25	90.60	**94.79**	84.94	89.54	90.99	90.07

We study four multilingual problems from the XTREME benchmark (Hu et al., 2020) and show experimental results of POS and PI in Table 3. *Due to the page limit, the results of NLI and NER are placed in Table 12 in the Appendix.* Different from heterogeneous MTL problems on the NYUv2 and PASCAL-Context datasets, in these multilingual problems, each language has its own input data, which is usually called homogeneous MTL problems (Zhang & Yang, 2021). According to the results in Tables 3, RLW with diverse distributions still outperforms EW in all the two multilingual problems, which further shows the effectiveness of RLW on different type of MTL problem.

Besides, RLW achieves comparable and even better performance than those baseline methods. For example, on the POS multilingual problem, RLW has the highest average F1 score and it achieves the best average accuracy on the PI problem.

5.7 RLW WITH DIFFERENT ARCHITECTURES

The proposed RLW strategy can be seamlessly combined with other MTL network architectures without increasing additional computation cost. To see this, we combine the RLW strategy with three SOTA MTL architectures, i.e., **cross-stitch** network (Misra et al., 2016), Multi-Task Attention Network (**MTAN**) (Liu et al., 2019a), and CNN with Neural Discriminative Dimensionality Reduction layer (**NDDR-CNN**) (Gao et al., 2019) and evaluate all the methods on the NYUv2 dataset. Experimental results when using the MTAN architecture are shown in Table 4. *Due to page limit, the results for the cross-stitch and NDDR-CNN networks are put in Tables 6 and 7 in the Appendix, respectively.*

According to the results, we have some observations. Firstly, compared with the performance when using the DMTL architecture (i.e., the results on Table 1), the performance of each loss weighting strategy with the deeper MTAN architecture is improved, especially on the surface normal estimation task, which is due to the larger capacity of the MTAN. Secondly, the RLW strategy with different distributions can outperform EW, which indicates the effectiveness of RLW with more advanced and deeper architectures. Thirdly, compared with baseline methods, the proposed RLW method can achieve competitive performance. For example, RLW with the random normal distribution can improve over EW by 0.76% and is among the top-3 methods.

Table 4: Performance under the **MTAN** architecture on the **NYUv2** validation dataset with three tasks: 13-class semantic segmentation, depth estimation, and surface normal prediction. The best results for each task on each measure are highlighted in **bold**. ↑ (↓) indicates that the higher (lower) the result, the better the performance.

Weighting Strategy	Segmentation		Depth		Surface Normal					Δ_p↑
					Angle Distance		Within t°			
	mIoU↑	Pix Acc↑	Abs Err↓	Rel Err↓	Mean↓	Median↓	11.25↑	22.5↑	30↑	
EW	53.77	75.79	0.3789	0.1546	22.8344	16.6021	36.37	62.45	73.32	+0.00
GradNorm	54.75	75.89	0.3797	0.1537	22.6295	16.2829	37.09	63.12	73.87	+0.83
UW	**54.80**	**75.97**	0.3767	0.1536	22.6744	16.3050	36.90	63.14	73.87	**+0.95**
MGDA	53.94	**75.97**	0.3788	0.1573	22.6157	16.1395	37.24	63.38	73.94	+0.37
DWA	53.71	75.72	0.3801	0.1539	23.1560	16.8354	35.95	61.85	72.79	−0.39
PCGrad	53.83	75.70	0.3823	0.1568	22.9481	16.3528	37.05	62.76	73.37	−0.16
GradDrop	53.83	75.85	**0.3754**	0.1530	22.7846	16.4156	36.77	62.74	73.51	+0.57
IMTL	53.39	75.20	0.3807	0.1549	**22.2571**	**15.8336**	**38.04**	**64.11**	**74.59**	+0.72
GradVac	54.52	75.68	0.3755	0.1546	22.9389	16.5692	36.48	62.33	73.20	+0.34
RLW (Uniform)	54.27	75.51	0.3820	0.1548	22.9640	16.4375	36.89	62.65	73.29	+0.08
RLW (Normal)	53.70	75.62	0.3791	0.1551	22.8395	16.3328	37.05	62.82	73.44	+0.16
RLW (Dirichlet)	53.36	75.08	0.3778	**0.1514**	22.8803	16.3579	36.99	62.80	73.40	+0.35
RLW (Bernoulli)	53.46	75.74	0.3820	0.1517	22.5642	16.2013	37.21	63.29	73.96	+0.61
RLW (constrained Bernoulli)	54.11	75.47	0.3821	0.1558	22.7969	16.2204	36.93	62.90	73.56	+0.10
RLW (random Normal)	54.10	75.77	0.3802	0.1554	22.4400	16.0336	37.74	63.68	74.17	+0.76

6 CONCLUSIONS

In this paper, we have unified eight state-of-the-art task balancing methods from the loss weighting perspective. Based on randomly sampling task weights from distributions, we propose a simple RLW strategy that can achieve comparable performance with state-of-the-art baselines. We analyze the convergence property of the proposed RLW method and the double stochasticity that can help escape sharp local minima. Finally, we provide a consistent and comparative comparison to show the effectiveness of the proposed RLW method on six computer vision datasets and four multilingual tasks from the XTREME benchmark. In our future studies, we will apply the proposed RLW method to more fields such as reinforcement learning.

Ethics Statement

In this paper, we propose the RLW method, a simple yet effective loss weighting strategy for multi-task learning. This method is a general machine learning algorithm and we do not apply it to any specific applications. Thus there is no any negative ethics impacts in our paper.

Reproducibility Statement

For the experiments in this paper, all the datasets we used are publicly available and implementation details including data splits and hyperparameters (i.e., the learning rate, the weight decay, and the batch size) are provided in Section 5.2 and Appendix C. Besides, the source code is put in the supplementary material. For the theoretical results in this paper, we have stated the full set of assumptions in Section 4 and the complete proofs are provided in Appendix B.

References

Aayush Bansal, Xinlei Chen, Bryan C. Russell, Abhinav Gupta, and Deva Ramanan. Pixelnet: Representation of the pixels, by the pixels, and for the pixels. *arXiv preprint arXiv:1702.06506*, 2017.

Léon Bottou. Stochastic gradient learning in neural networks. *Proceedings of Neuro-Nımes*, 91(8): 12, 1991.

Rich Caruana. Multitask learning: A knowledge-based source of inductive bias. In *Proceedings of the 10th International Conference on Machine Learning*, pp. 41–48, 1993.

Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the 14th European Conference on Computer Vision*, volume 11211, pp. 833–851, 2018a.

Shijie Chen, Yu Zhang, and Qiang Yang. Multi-task learning in natural language processing: An overview. *arXiv preprint arXiv: 2109.09138*, 2021.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *Proceedings of the International Conference on Machine Learning*, pp. 794–803. PMLR, 2018b.

Zhao Chen, Jiquan Ngiam, Yanping Huang, Thang Luong, Henrik Kretzschmar, Yuning Chai, and Dragomir Anguelov. Just pick a sign: Optimizing deep multitask models with gradient sign dropout. In *Proceedings of the 33rd Advances in Neural Information Processing Systems*, 2020.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel R. Bowman, Holger Schwenk, and Veselin Stoyanov. XNLI: evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2475–2485, 2018.

Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3213–3223, 2016.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4171–4186, 2019.

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. Multi-task learning for multiple language translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing*, pp. 1723–1732, 2015.

Yuan Gao, Jiayi Ma, Mingbo Zhao, Wei Liu, and Alan L Yuille. Nddr-cnn: Layerwise feature fusing in multi-task cnns by neural discriminative dimensionality reduction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3205–3214, 2019.

Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *Processing of the International Conference on Machine Learning*, pp. 1225–1234. PMLR, 2016.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 4411–4421. PMLR, 2020.

Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7482–7491, 2018.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, 2015.

Bobby Kleinberg, Yuanzhi Li, and Yang Yuan. An alternative view: When does sgd escape local minima? In *Processing of the International Conference on Machine Learning*, pp. 2698–2707. PMLR, 2018.

Liyang Liu, Yi Li, Zhanghui Kuang, Jing-Hao Xue, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. Towards impartial multi-task learning. In *Proceedings of the 9th International Conference on Learning Representations*, 2021.

Shikun Liu, Edward Johns, and Andrew J. Davison. End-to-end multi-task learning with attention. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1871–1880, 2019a.

Xiaodong Liu, Jianfeng Gao, Xiaodong He, Li Deng, Kevin Duh, and Ye-Yi Wang. Representation learning using multi-task deep neural networks for semantic classification and information retrieval. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 912–921, 2015a.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pp. 4487–4496, 2019b.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision*, 2015b.

Kevis-Kokitsi Maninis, Ilija Radosavovic, and Iasonas Kokkinos. Attentive single-tasking of multiple tasks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1851–1860, 2019.

Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3994–4003, 2016.

Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan L. Yuille. The role of context for object detection and semantic segmentation in the wild. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 891–898, 2014.

Eric Moulines and Francis Bach. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *Processing of the Advances in Neural Information Processing Systems*, volume 24, pp. 451–459, 2011.

Deanna Needell, Nathan Srebro, and Rachel Ward. Stochastic gradient descent, weighted sampling, and the randomized kaczmarz algorithm. *Mathematical Programming*, 155(1-2):549–573, 2016.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajic, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis M. Tyers, and Daniel Zeman. Universal dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pp. 4034–4043, 2020.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pp. 1946–1958, 2017.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Processing of the 32rd Advances in Neural Information Processing Systems*, pp. 8024–8035, 2019.

Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *Proceedings of the 6th European Conference on Computer Vision*, pp. 213–226. Springer, 2010.

Victor Sanh, Thomas Wolf, and Sebastian Ruder. A hierarchical multi-task approach for learning embeddings from semantic tasks. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pp. 6949–6956, 2019.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Proceedings of the 31st Advances in Neural Information Processing Systems*, pp. 525–536, 2018.

Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *Proceedings of the 8th European Conference on Computer Vision*, pp. 746–760, 2012.

Anders Søgaard and Yoav Goldberg. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, 2016.

Sandeep Subramanian, Adam Trischler, Yoshua Bengio, and Christopher J. Pal. Learning general purpose distributed sentence representations via large scale multi-task learning. In *Proceedings of the 6th International Conference on Learning Representations*, 2018.

Simon Vandenhende, Stamatios Georgoulis, Wouter Van Gansbeke, Marc Proesmans, Dengxin Dai, and Luc Van Gool. Multi-task learning for dense prediction tasks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5018–5027, 2017.

Zirui Wang, Yulia Tsvetkov, Orhan Firat, and Yuan Cao. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. In *Proceedings of the 9th International Conference on Learning Representations*, 2021.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1112–1122, 2018.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pp. 38–45, 2020.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. PAWS-X: A cross-lingual adversarial dataset for paraphrase identification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 3685–3690, 2019.

Fisher Yu, Vladlen Koltun, and Thomas A. Funkhouser. Dilated residual networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 636–644, 2017.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Proceedings of the 33rd Advances in Neural Information Processing Systems*, 2020.

Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

APPENDIX

A SUMMARY OF LOSS WEIGHTING STRATEGIES

In this section, we summarize the eight SOTA methods introduced in Section 2 from a perspective of loss weighting. We define $\boldsymbol{G} = [\boldsymbol{g}_1, \cdots, \boldsymbol{g}_T]$ and $\boldsymbol{u}_t = \boldsymbol{g}_t / \|\boldsymbol{g}_t\|$, where $\boldsymbol{g}_t = \nabla_\theta \ell_t(\mathcal{D}_t; \theta)$ denotes the gradient of $\ell_t(\mathcal{D}_t; \theta)$ with respect to θ. Let $\mathbb{R}_+^T$ denote the non-negative subspace in the T-dimensional space $\mathbb{R}^T$, $\mathbb{1}$ denote $(1, \cdots, 1)$, $\mathbb{I}$ denote an identity matrix, $\mathrm{diag}(\boldsymbol{a})$ denote a diagonal matrix with its principal diagonal $\boldsymbol{a}$, $[\cdot]_+$ be the ReLU operation, $\mathrm{sgn}(\cdot)$ represent the sign function, and $\mathcal{I}(\cdot)$ denote the indicator function. The summary of different methods is in Table 5.

Table 5: A summary of SOTA weighting strategies from a perspective of loss weighing. $*$ means whether a convergence analysis (abbreviated as Conv.) is provided in the original paper. † denotes that the corresponding weighting strategy needs not to compute gradients (abbreviated as Not Grad.) for generating loss weights $\boldsymbol{\lambda}$.

Approach	**Strategy**	**Weight $\boldsymbol{\lambda}$** (k-th iteration)	**Conv.***	**Not Grad.**†
	EW	$(\frac{1}{T}, \cdots, \frac{1}{T})$	✓	✓
Learning	GradNorm	$\min_{\boldsymbol{\lambda}} \sum_{t=1}^T \left\|\lambda_t \|\boldsymbol{g}_t\| - g r_t^\alpha\right\|$, where $g = (\sum_{t=1}^T \lambda_t \|\boldsymbol{g}_t\|)/T, r_t = \tilde{\ell}_t / (\frac{1}{T}\sum_{j=1}^T \tilde{\ell}_j)$ $\tilde{\ell}_t = \ell_t/\ell_t^0$, and α is pre-defined	✗	✗
	UW	$\min_{\boldsymbol{\lambda}} (\boldsymbol{\lambda}^\top \boldsymbol{\ell} + \mathbb{1}\log(\boldsymbol{\lambda}^\top)/2),\ \text{s.t.}\ \boldsymbol{\lambda} \in \mathbb{R}_+^T$	✗	✓
	IMTL-L	$\min_{\boldsymbol{\lambda}} (\boldsymbol{\lambda}^\top \boldsymbol{\ell} - \mathbb{1}\log(\boldsymbol{\lambda})^\top),\ \text{s.t.}\ \boldsymbol{\lambda} \in \mathbb{R}_+^T$	✗	✗
Solving	MGDA	$\arg\min_{\boldsymbol{\lambda}} \left\{\|\boldsymbol{\lambda}^\top \boldsymbol{G}\|^2 \mid \boldsymbol{\lambda} \in \Delta^T\right\}$	✓	✗
Calculating	DWA	$T \times \mathrm{softmax}(\boldsymbol{\ell}^{k-1}/\boldsymbol{\ell}^{k-2})$	✗	✓
	PCGrad	$\mathbb{1}(\mathbb{I} + \boldsymbol{C}\boldsymbol{N})$, where $\boldsymbol{N} = \mathrm{diag}(1/\|\boldsymbol{g}_1\|, \cdots, 1/\|\boldsymbol{g}_T\|)$, $\boldsymbol{C} = [C_{pq}]_{T\times T}, C_{pp} = 0$, $C_{pq} = \left[-\big(\boldsymbol{g}_p + \sum_{p'<q} C_{pp'}\boldsymbol{u}_{p'}\big)\boldsymbol{u}_q^\top\right]_+$	✓	✗
	GradDrop	$\mathbb{1}^\top \boldsymbol{s} + (\mathbb{I} - \mathrm{diag}(\boldsymbol{s}))\boldsymbol{M}$, where $\boldsymbol{s} \in [0,1]^T$ is pre-defined, $\boldsymbol{M} = \mathrm{diag}\big(\mathcal{I}(\boldsymbol{d} > \boldsymbol{e})\big)\mathcal{I}(\tilde{\boldsymbol{G}} > 0) +$ $\mathrm{diag}\big(\mathcal{I}(\boldsymbol{d} < \boldsymbol{e})\big)\mathcal{I}(\tilde{\boldsymbol{G}} < 0)$, $\boldsymbol{d} = (1/2)\big(1 + \mathbb{1}\boldsymbol{G}/(\sum_{t=1}^T \|\boldsymbol{g}_t\|)\big)$, $\boldsymbol{e} \sim \mathrm{Uniform}(0,1), \tilde{\boldsymbol{G}} = \mathrm{sgn}(\theta) \odot \boldsymbol{G}$	✗	✗
	IMTL-G	$[1 - \mathbb{1}\boldsymbol{\alpha}_{2:T}^\top, \boldsymbol{\alpha}_{2:T}]$, where $\boldsymbol{\alpha}_{2:T} = \boldsymbol{g}_1 \boldsymbol{N}^\top (\boldsymbol{D}\boldsymbol{N}^\top)^{-1}$, $\boldsymbol{N} = \mathbb{1}^\top \boldsymbol{u}_1 - \boldsymbol{U}_{2:T}, \boldsymbol{D} = \mathbb{1}^\top \boldsymbol{g}_1 - \boldsymbol{G}_{2:T}$	✗	✗
	GradVac	$\mathbb{1}(\mathbb{I} + \boldsymbol{C}\boldsymbol{N})$, where $\boldsymbol{N} = \mathrm{diag}(1/\|\boldsymbol{g}_1\|, \cdots, 1/\|\boldsymbol{g}_T\|)$, $\boldsymbol{C} = [C_{pq}]_{T\times T}, C_{pp} = 0$, $C_{pq} = \mathrm{sgn}(\hat{\phi}_{pq} - \phi_{pq}) * \left[\|\hat{\boldsymbol{g}}_p\| * A(\hat{\phi}_{pq}, \phi_{pq})\right]$, $A(\hat{\phi}_{pq}, \phi_{pq}) = \frac{\hat{\phi}_{pq}\sqrt{1-(\phi_{pq})^2} - \phi_{pq}\sqrt{1-(\hat{\phi}_{pq})^2}}{\sqrt{1-(\hat{\phi}_{pq})^2}}$, $\phi_{pq} = (\hat{\boldsymbol{g}}_p \cdot \boldsymbol{g}_q)/(\|\hat{\boldsymbol{g}}_p\|\|\boldsymbol{g}_q\|)$, $\hat{\boldsymbol{g}}_p = \boldsymbol{g}_p + \sum_{p'<q} C_{pp'}\boldsymbol{u}_{p'}$, $\hat{\phi}_{pq} = (1-\beta)\hat{\phi}_{pq} + \beta\phi_{pq}$, β is pre-defined, and $\hat{\phi}_{pq}$ is initialized as 0	✓	✗
Sampling	RLW (**ours**)	$f(\tilde{\boldsymbol{\lambda}})$, where $\tilde{\boldsymbol{\lambda}} \sim p(\tilde{\boldsymbol{\lambda}})$ and f is a normalization function	✓	✓

B Proof of Section 4

B.1 Proof of Theorem 1

Since ℓ_t is c_t-strongly convex w.r.t θ, for $\mathcal{L}_{\mathrm{RLW}}(\theta) = \boldsymbol{\lambda}^\top \boldsymbol{\ell}(\theta)$, for any two points θ_1 and θ_2 in $\mathbb{R}^d$, we have

$$\begin{aligned}\left\langle \nabla \boldsymbol{\lambda}^\top \boldsymbol{\ell}(\theta_1) - \nabla \boldsymbol{\lambda}^\top \boldsymbol{\ell}(\theta_2), \theta_1 - \theta_2 \right\rangle &= \sum_{t=1}^T \lambda_t \left\langle \nabla \ell_t(\theta_1) - \nabla \ell_t(\theta_2), \theta_1 - \theta_2 \right\rangle \\ &\geq \sum_{t=1}^T c_t \lambda_t \|\theta_1 - \theta_2\|^2. \end{aligned} \tag{5}$$

Since $0 \leq \lambda_t \leq 1$, we have $\sum_{t=1}^T c_t \lambda_t \geq c$, where $c = \min_{1 \leq t \leq T}\{c_t\}$. Then for any λ, $\mathcal{L}_{\mathrm{RLW}}(\theta)$ is c-strongly convex.

With notations in Theorem 1, we have

$$\begin{aligned}\|\theta_{k+1} - \theta_*\|^2 &= \|\theta_k - \theta_* - \eta \nabla \boldsymbol{\lambda}^\top \boldsymbol{\ell}(\tilde{\boldsymbol{\mathcal{D}}}; \theta_k)\|^2 \\ &= \|\theta_k - \theta_*\|^2 - 2\eta \left\langle \theta_k - \theta_*, \nabla \boldsymbol{\lambda}^\top \boldsymbol{\ell}(\tilde{\boldsymbol{\mathcal{D}}}; \theta_k) \right\rangle + \eta^2 \|\nabla \boldsymbol{\lambda}^\top \boldsymbol{\ell}(\tilde{\boldsymbol{\mathcal{D}}}; \theta_k)\|^2. \end{aligned}$$

Note that $\mathbb{E}_{\boldsymbol{\lambda}}\left[\mathbb{E}_{\tilde{\boldsymbol{\mathcal{D}}}}[\nabla \boldsymbol{\lambda}^\top \boldsymbol{\ell}(\tilde{\boldsymbol{\mathcal{D}}}; \theta_k)]\right] = \nabla \boldsymbol{\mu}^\top \boldsymbol{\ell}(\boldsymbol{\mathcal{D}}; \theta_k)$ and

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\lambda}}\left[\mathbb{E}_{\tilde{\boldsymbol{\mathcal{D}}}}[\|\nabla \boldsymbol{\lambda}^\top \boldsymbol{\ell}(\tilde{\boldsymbol{\mathcal{D}}}; \theta_k)\|^2]\right] &\leq \mathbb{E}_{\boldsymbol{\lambda}}\left[\mathbb{E}_{\tilde{\boldsymbol{\mathcal{D}}}}[\|\boldsymbol{\lambda}^\top\|^2 \|\nabla \boldsymbol{\ell}(\tilde{\boldsymbol{\mathcal{D}}}; \theta_k)\|^2]\right] \\ &\leq \mathbb{E}_{\boldsymbol{\lambda}}\Big[\sum_{t=1}^T \lambda_t^2\Big] \cdot \sum_{t=1}^T \sigma_t^2 \\ &\leq \sum_{t=1}^T \sigma_t^2, \end{aligned}$$

where the first inequality is due to the Cauchy-Schwarz inequality and the third inequality is due to $0 \leq \lambda_t \leq 1$. Then, by defining $\kappa = \sum_{t=1}^T \sigma_t^2$, we obtain

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\lambda}}\left[\mathbb{E}_{\tilde{\boldsymbol{\mathcal{D}}}}[\|\theta_{k+1} - \theta_*\|^2]\right] &\leq \|\theta_k - \theta_*\|^2 - 2\eta \left\langle \theta_k - \theta_*, \nabla \boldsymbol{\mu}^\top \boldsymbol{\ell}(\theta_k) \right\rangle + \eta^2 \kappa \\ &\leq (1 - 2\eta c)\|\theta_k - \theta_*\|^2 + \eta^2 \kappa. \end{aligned} \tag{6}$$

If $1 - 2\eta c > 0$, we recursively apply the inequality (6) over the first k iterations and we can obtain

$$\begin{aligned}\mathbb{E}[\|\theta_{k+1} - \theta_*\|^2] &\leq (1 - 2\eta c)^k \|\theta_0 - \theta_*\|^2 + \sum_{j=0}^{k-1} (1 - 2\eta c)^j \eta^2 \kappa \\ &\leq (1 - 2\eta c)^k \|\theta_0 - \theta_*\|^2 + \frac{\eta \kappa}{2c}. \end{aligned}$$

Thus the inequality (4) holds if $\mu \leq \frac{1}{2c}$.

According to inequality (6), the minimal value of a quadratic function $g_\varepsilon(\eta) = (1 - 2\eta c)\varepsilon + \eta^2 \kappa$ is achieved at $\eta_* = \frac{\varepsilon c}{\kappa}$. By setting $\|\theta_0 - \theta_*\|^2 = \varepsilon_0$, we have

$$\begin{aligned}\mathbb{E}[\|\theta_{k+1} - \theta_*\|^2] &\leq g_{\|\theta_k - \theta_*\|^2}(\eta_*) \\ &= (1 - \frac{2\|\theta_k - \theta_*\|^2 c^2}{\kappa})\|\theta_k - \theta_*\|^2 \\ &\leq (1 - \frac{2\varepsilon c^2}{\kappa})\|\theta_k - \theta_*\|^2 \\ &\leq (1 - \frac{2\varepsilon c^2}{\kappa})^k \varepsilon_0. \end{aligned}$$

Then if $\mathbb{E}[\|\theta_{k+1} - \theta_*\|^2] \geq \varepsilon$, we have $\varepsilon \leq (1 - \frac{2\varepsilon c^2}{\kappa})^k \varepsilon_0$. Therefore, $k \leq \frac{\kappa}{2\varepsilon c^2} \log\left(\frac{\varepsilon_0}{\varepsilon}\right)$.

B.2 Proof of Theorem 2

Since $\varphi_k = \theta_k - \eta\nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\theta_k)$ and $\theta_{k+1} = \theta_k - \eta(\nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\theta_k) + \xi_k)$, we have

$$\varphi_{k+1} = \varphi_k - \eta\xi_k - \nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\varphi_k - \eta\xi_k).$$

Since the loss function $\ell_t(\theta)$ of task t is c_t-one point strongly convex w.r.t a given point θ_* after convolved with noise ξ, similar to inequality (5), we have

$$\left\langle \nabla\mathbb{E}_\xi[\boldsymbol{\mu}^\top\boldsymbol{\ell}(\varphi - \eta\xi)], \varphi - \theta_* \right\rangle \geq c\|\varphi - \theta_*\|^2,$$

where $c = \min_{1\leq t\leq T}\{c_t\}$. Since $\nabla\ell_t(\theta)$ is M_t-Lipschitz continuous, for any two points θ_1 and θ_2 in $\mathbb{R}^d$, we have

$$\|\nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\theta_1) - \nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\theta_2)\| = \sum_{t=1}^{T}\mu_t\|\nabla\ell_t(\theta_1) - \nabla\ell_t(\theta_2)\| \leq \sum_{t=1}^{T} M_t\mu_t\|\theta_1 - \theta_2\|. \tag{7}$$

Note that $\sum_{t=1}^T M_t\mu_t \leq M$, where $M = \max_{1\leq t\leq T}\{M_t\}$. Therefore, $\nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\theta)$ is M-Lipschitz continuous. Then we can get

$$\begin{aligned}
\mathbb{E}[\|\varphi_{k+1} - \theta_*\|^2] &= \mathbb{E}[\|\varphi_k - \eta\xi_k - \nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\varphi_k - \eta\xi_k) - \theta_*\|^2] \\
&\leq \mathbb{E}[\|\varphi_k - \theta_*\|^2 + \|\eta\xi_k\|^2 + \|\nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\varphi_k - \eta\xi_k)\|^2 - 2\left\langle\varphi_k - \theta_*, \eta\xi_k\right\rangle \\
&\quad - 2\left\langle\varphi_k - \theta_*, \nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\varphi_k - \eta\xi_k)\right\rangle + 2\left\langle\nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\varphi_k - \eta\xi_k), \eta\xi_k\right\rangle] \\
&\leq \|\varphi_k - \theta_*\|^2 + \eta^2r^2 + \mathbb{E}[\|\nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\varphi_k - \eta\xi_k)\|^2] - 2\eta c\|\varphi_k - \theta_*\|^2 \\
&\quad + 2\mathbb{E}[\left\langle\nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\varphi_k - \eta\xi_k) - \nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\varphi_k), \eta\xi_k\right\rangle] \\
&\leq (1 - 2\eta c)\|\varphi_k - \theta_*\|^2 + \eta^2r^2 + \eta^2\mathbb{E}[\|M(\theta_* - (\varphi_k - \eta\xi_k))\|^2] + 2\eta^3r^2M \\
&\leq (1 - 2\eta c)\|\varphi_k - \theta_*\|^2 + \eta^2r^2 + \eta^2M^2\|\varphi_k - \theta_*\|^2 + \mathbb{E}[\langle\varphi_k - \theta_*, \eta\xi_k\rangle] \\
&\quad + \eta^2M^2\mathbb{E}[\|\eta\xi_k\|^2] + 2\eta^3r^2M \\
&\leq (1 - 2\eta c + \eta^2M^2)\|\varphi_k - \theta_*\|^2 + \eta^2r^2(1 + \eta M)^2,
\end{aligned}$$

where the second inequality is due to the convexity assumption and $\mathbb{E}[\xi_k] = 0$, the third and forth inequalities are due to the Lipschitz continuity. We set $\rho = 2\eta c - \eta^2M^2$ and $\beta = \eta^2r^2(1 + \eta M)^2$. If $\rho \geq 0$, we have $\eta \leq \frac{c}{M^2}$, then we get

$$\begin{aligned}
\mathbb{E}[\|\varphi_{k+1} - \theta_*\|^2] &\leq (1 - \rho)\|\varphi_k - \theta_*\|^2 + \beta \\
&\leq (1 - \rho)^k\|\varphi_0 - \theta_*\|^2 + \sum_{j=0}^{k-1}(1 - \rho)^j\beta \\
&\leq (1 - \rho)^k\|\varphi_0 - \theta_*\|^2 + \frac{\beta}{\rho}.
\end{aligned}$$

So if $k \leq \frac{1}{\rho}\log\left(\frac{\rho\varepsilon_0}{\beta}\right)$, we have $\mathbb{E}[\|\varphi_{k+1} - \theta_*\|^2] \leq \frac{2\beta}{\rho}$. Then by the Markov inequality, with probability at least $1 - \delta$, we have

$$\|\varphi_K - \theta_*\|^2 \leq \frac{2\beta}{\rho\delta}.$$

B.3 Noise upper bound

Suppose the noise produced by the FW method is $\bar{\xi} = \|\nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\tilde{\boldsymbol{\mathcal{D}}};\theta) - \nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\boldsymbol{\mathcal{D}};\theta)\|$ and $\|\bar{\xi}\|^2 \leq R$. The noise produced by the RLW method is $\xi = \|\nabla\boldsymbol{\lambda}^\top\boldsymbol{\ell}(\tilde{\boldsymbol{\mathcal{D}}};\theta) - \nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\boldsymbol{\mathcal{D}};\theta)\|$. We have

$$\begin{aligned}
\|\xi\|^2 &= \|\nabla\boldsymbol{\lambda}^\top\boldsymbol{\ell}(\tilde{\boldsymbol{\mathcal{D}}};\theta) - \nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\tilde{\boldsymbol{\mathcal{D}}};\theta) + \nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\tilde{\boldsymbol{\mathcal{D}}};\theta) - \nabla\boldsymbol{\mu}^\top\boldsymbol{\ell}(\boldsymbol{\mathcal{D}};\theta)\|^2 \\
&= \|(\boldsymbol{\lambda}^\top - \boldsymbol{\mu}^\top)\nabla\boldsymbol{\ell}(\tilde{\boldsymbol{\mathcal{D}}};\theta)\|^2 + 2\left\langle(\boldsymbol{\lambda}^\top - \boldsymbol{\mu}^\top)\boldsymbol{\ell}(\tilde{\boldsymbol{\mathcal{D}}};\theta), \bar{\xi}\right\rangle + \|\bar{\xi}\|^2.
\end{aligned}$$

Because the noise $\bar{\xi}$ can be any direction, there exists a constant $s > 0$ such that $\|\bar{\xi}\|^2 = R$ and $\bar{\xi} = s(\boldsymbol{\lambda}^\top - \boldsymbol{\mu}^\top)\nabla\boldsymbol{\ell}(\tilde{\boldsymbol{\mathcal{D}}};\theta)$. Then, we have $\|\xi\|^2 \leq (1 + 2s)\|\boldsymbol{\lambda} - \boldsymbol{\mu}\|^2\|\nabla\boldsymbol{\ell}(\tilde{\boldsymbol{\mathcal{D}}};\theta)\|^2 + R$. Thus, the norm of the noise provided by the RLW method has a larger least upper bound than FW.

B.4 Convergence with decreasing learning rate

In the following theorem, we show the convergence analysis and generalization bound of the RLW method under the strong convexity case with decreasing step sizes.

Theorem 3. *Suppose the loss function $\ell_t(\theta)$ of task t is c_t-strongly convex and Assumption 1 holds. Then if we choose $\eta_k = \alpha/k$ and $\alpha > 1/2c$, we have*

$$\mathbb{E}[\|\theta_k - \theta_*\|^2] \leq \frac{\max\{2\alpha^2\kappa(2\alpha c - 1)^{-1}, \|\theta_0 - \theta_*\|^2\}}{k},$$

where $\kappa = \sum_{t=1}^T \sigma_t^2$ and $c = \min_{1\leq t\leq T}\{c_t\}$. Moreover, by defining the optimal loss function as $\mathcal{L}_{\mathrm{RLW}}(\theta_*)$, we can obtain the following generalization bound as

$$\mathbb{E}[\|\mathcal{L}_{\mathrm{RLW}}(\theta_k) - \mathcal{L}_{\mathrm{RLW}}(\theta_*)\|] \leq \frac{\max\{\alpha^2 L\kappa(2\alpha c - 1)^{-1}, L\|\theta_0 - \theta_*\|^2\}}{k},$$

where $L = \max_{1\leq t\leq T}\{L_t\}$.

Proof. According to inequality (6), by setting $\eta = \alpha/k$, we have

$$\mathbb{E}[\|\theta_{k+1} - \theta_*\|^2] \leq (1 - \frac{2\alpha c}{k})\|\theta_k - \theta_*\|^2 + \frac{\alpha^2\kappa}{k^2}.$$

Then by induction we can get

$$\mathbb{E}[\|\theta_{k+1} - \theta_*\|^2] \leq \frac{Q}{k},$$

where $Q = \max\{2\alpha^2\kappa(2\alpha c - 1)^{-1}, \|\theta_0 - \theta_*\|^2\}$.

Since $\ell_t(\theta)$ is L_t-Lipschitz continuous, similar to inequality (7), for any two points θ_1 and θ_2 in $\mathbb{R}^d$, we have

$$\|\nabla\mathcal{L}_{\mathrm{RLW}}(\theta_1) - \nabla\mathcal{L}_{\mathrm{RLW}}(\theta_2)\| \leq \sum_{t=1}^T L_t\mu_t\|\theta_1 - \theta_2\|.$$

Therefore, $\mathcal{L}_{\mathrm{RLW}}(\theta)$ is L-Lipschitz continuous, where $L = \max_{1\leq t\leq T}\{L_t\}$. Then we can get

$$\mathbb{E}[\|\mathcal{L}_{\mathrm{RLW}}(\theta_k) - \mathcal{L}_{\mathrm{RLW}}(\theta_*)\|] \leq \frac{L}{2}\mathbb{E}[\|\theta_k - \theta_*\|^2] \leq \frac{LQ}{2k},$$

where we reach the conclusion.

□

C Additional Experimental Results

C.1 Experiments on Convergence

In Figure 1, we empirically compare the convergence speed of the EW and RLW methods in terms of the performance curve on both the NYUv2 and CelebA validation datasets.

On the NYUv2 dataset with three tasks, the performance curves of the RLW method with three different distributions are both consistent with the EW method, which indicates the RLW method has a similar convergence property to the EW method. As the number of tasks increases, i.e., on the CelebA dataset with forty tasks, we find that the RLW method with the Normal and Bernoulli distributions still converges as fast as the EW method, while the RLW method with the constrained Bernoulli distribution converges slower. One reason for this phenomenon is that only one task is used for updating the parameters in each training iteration.

In summary, the proposed RLW strategy combined with all the distributions except the constrained Bernoulli distribution has similar training efficiency to the EW method.

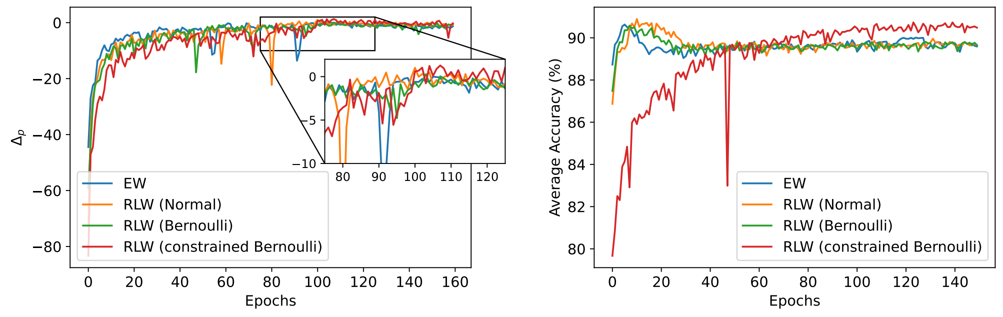

Figure 1: Comparison on the convergence speed of EW and the proposed RLW strategy on the NYUv2 dataset (**Left**) and the CelebA dataset (**Right**), respectively.

C.2 Additional results of RLW on Different Architectures

The results of different loss weighting strategies with the cross-stitch and NDDR-CNN networks are put in Tables 6 and 7, respectively. The empirical observations are similar to those with the MTAN architecture (i.e., Table 4). Firstly, the performance of each loss weighting strategy with the deeper NDDR-CNN architecture is improved on the three tasks, while only the surface normal estimation task can be improved with the cross-stitch network. Secondly, the RLW strategy significantly outperform EW on both architectures. Thirdly, compared with the baseline methods, RLW can achieve comparable or even better performance than all the baseline methods except IMTL on both two architectures. For example, RLW with the constrained Bernoulli distribution has an improvement of 1.79% on the cross-stitch architecture, which is slightly lower than the best performed method, IMTL. Besides, RLW with the Normal distribution is among the top-2 methods based on the NDDR-CNN network.

Table 6: Performance of different loss weighting strategies with the **cross-stitch** architecture on the **NYUv2** validation dataset with three tasks: 13-class semantic segmentation, depth estimation, and surface normal prediction. The best results for each task on each measure are highlighted in **bold**. ↑ (↓) means the higher (lower) the result, the better the performance.

Weighting Strategy	Segmentation		Depth		Surface Normal					Δ_p↑
					Angle Distance		Within t°			
	mIoU↑	Pix Acc↑	Abs Err↓	Rel Err↓	Mean↓	Median↓	11.25↑	22.5↑	30↑	
EW	53.39	75.06	0.3822	0.1595	23.1939	16.6749	35.99	61.79	72.69	+0.00
GradNorm	53.92	75.39	0.3833	0.1547	22.7461	16.2463	37.15	63.02	73.57	+1.42
UW	53.88	75.76	0.3820	0.1587	23.1397	16.4221	36.61	62.30	72.91	+0.71
MGDA	53.33	75.07	0.3877	0.1584	23.1588	16.6327	36.41	61.75	72.57	−0.05
DWA	52.62	75.02	0.3801	0.1563	23.4577	16.7667	36.13	61.51	72.24	+0.02
PCGrad	53.83	75.58	0.3853	0.1568	22.5621	16.0502	37.80	63.43	73.89	+1.45
GradDrop	52.23	74.96	0.3855	0.1583	22.7940	15.9848	37.79	63.25	73.53	+0.56
IMTL	**54.16**	**75.83**	0.3833	0.1571	**22.3801**	15.9137	37.81	**63.97**	**74.46**	**+1.89**
GradVac	52.93	75.22	0.3871	0.1559	22.4499	**15.8148**	**38.31**	63.96	74.21	+1.42
RLW (Uniform)	53.23	75.19	0.3870	0.1576	22.8859	16.3011	37.06	62.55	73.17	+0.53
RLW (Normal)	53.34	75.32	0.3804	0.1577	22.9278	16.2227	37.25	62.68	73.27	+0.95
RLW (Dirichlet)	52.86	75.30	0.3859	0.1604	22.9058	16.0335	37.66	62.98	73.31	+0.47
RLW (Bernoulli)	53.44	75.33	**0.3782**	0.1558	22.8759	16.1796	37.27	62.79	73.29	+1.33
RLW (constrained Bernoulli)	53.25	75.28	0.3819	**0.1530**	22.5622	15.9059	37.80	63.39	73.78	+1.79
RLW (random Normal)	53.22	75.21	0.3861	0.1607	22.6981	15.8673	37.97	63.44	73.73	+0.79

C.3 Results on the CityScapes Dataset

Dataset The CityScapes dataset (Cordts et al., 2016) is a large-scale urban street scenes understanding dataset and it is comprised of a diverse set of stereo video sequences recorded from 50 different cities in fine weather during the daytime. It contains 2,975 and 500 annotated images for training and validation, respectively. This dataset includes two tasks: 7-class semantic segmentation and depth estimation.

Table 7: Performance of different loss weighting strategies with the **NDDR-CNN** architecture on the **NYUv2** validation dataset with three tasks: 13-class semantic segmentation, depth estimation, and surface normal prediction. The best results for each task on each measure are highlighted in **bold**. ↑ (↓) means the higher (lower) the result, the better the performance.

Weighting Strategy	Segmentation		Depth		Surface Normal					Δ_p↑
					Angle Distance		Within t°			
	mIoU↑	Pix Acc↑	Abs Err↓	Rel Err↓	Mean↓	Median↓	11.25↑	22.5↑	30↑	
EW	53.93	75.56	0.3835	0.1566	22.7908	16.1512	37.16	63.14	73.65	+0.00
GradNorm	54.47	75.37	0.3882	0.1551	22.4295	15.9649	37.89	63.76	74.22	+0.51
UW	54.16	75.60	**0.3795**	0.1541	22.6800	16.2123	37.19	63.05	73.66	+0.52
MGDA	53.65	75.19	0.3841	0.1579	22.7391	16.0227	37.48	63.29	73.70	−0.19
DWA	54.24	75.50	0.3835	0.1600	22.6793	16.0649	37.40	63.33	73.83	−0.13
PCGrad	53.83	75.08	0.3867	0.1575	22.5083	15.9023	38.03	63.73	74.10	+0.07
GradDrop	53.82	75.40	0.3841	0.1536	22.8330	16.4121	36.85	62.78	73.36	−0.02
IMTL	**54.49**	**75.68**	0.3808	**0.1531**	**21.9939**	**15.3531**	**39.42**	**65.10**	**75.18**	**+2.00**
GradVac	53.99	75.62	0.3874	0.1546	22.5202	15.9511	37.96	63.68	74.05	+0.47
RLW (Uniform)	53.90	75.47	0.3859	0.1579	22.5236	15.8438	37.94	63.77	74.11	+0.18
RLW (Normal)	53.93	75.31	0.3828	0.1532	22.4460	15.7951	38.04	63.93	74.24	+0.88
RLW (Dirichlet)	53.17	74.88	0.3814	0.1561	22.5713	15.8921	37.88	63.64	73.99	+0.14
RLW (Bernoulli)	54.04	75.13	0.3818	0.1563	22.4328	15.8549	37.99	63.76	74.10	+0.53
RLW (constrained Bernoulli)	53.77	75.24	0.3816	**0.1531**	22.7075	16.0182	37.50	63.24	73.75	+0.49
RLW (random Normal)	53.90	75.60	0.3847	0.1592	22.5007	16.0155	37.82	63.62	74.03	+0.02

Implementation Details The network architecture and optimizer on the CityScapes dataset are the same as those of the NYUv2 dataset. We resize all the images to 128×256 and set the batch size as 64 for training. The Adam optimizer (Kingma & Ba, 2015) with the learning rate as 10^{-4} and the weight decay as 10^{-5} is used. We use the cross-entropy loss and L_1 loss for the semantic segmentation and depth estimation tasks, respectively.

Results According to the results shown in Table 8, we can see that all the loss weighting strategies except the GradDrop method outperform EW. Especially, the PCGrad and MGDA methods can achieve a significant improvement of 2.79% and 2.60% in terms of $\mathbf{\Delta_p}$, respectively. Compared with those baseline methods, the RLW strategy not only outperforms EW by using the six different distributions but also achieves comparable performance with those baseline methods. For example, RLW with the Dirichlet distribution can improve over EW by 1.95% and is among the top-3 methods.

Table 8: Performance on the **CityScapes** validation dataset with two tasks: 7-class semantic segmentation and depth estimation. The best results for each task on each measure are highlighted in **bold**. ↑ (↓) means the higher (lower) the result, the better the performance.

Weighting Strategy	Segmentation		Depth		Δ_p↑
	mIoU↑	Pix Acc↑	Abs Err↓	Rel Err↓	
EW	68.72	91.50	0.0134	46.5974	+0.00
GradNorm	68.70	91.41	0.0133	45.6962	+0.64
UW	68.72	91.48	0.0134	45.0403	+0.83
MGDA	68.36	91.17	0.0123	45.1606	+2.60
DWA	68.62	91.45	0.0132	47.1341	+0.04
PCGrad	**70.16**	**91.93**	0.0128	44.6817	**+2.79**
GradDrop	68.53	91.46	0.0134	47.1396	−0.37
IMTL	68.74	91.50	0.0128	45.8279	+1.54
GradVac	69.89	91.94	**0.0126**	47.2281	+1.70
RLW (Uniform)	68.74	91.54	0.0132	45.2453	+1.12
RLW (Normal)	68.82	91.53	0.0132	46.1333	+0.67
RLW (Dirichlet)	69.37	91.75	0.0132	**44.2218**	+1.95
RLW (Bernoulli)	69.59	91.81	0.0130	47.0400	+0.91
RLW (constrained Bernoulli)	69.49	91.81	0.0128	46.9825	+1.28
RLW (random Normal)	68.59	91.51	0.0132	46.7122	+0.27

C.4 Results on the CelebA Dataset

Dataset The CelebA dataset (Liu et al., 2015b) is a large-scale face attributes dataset with 202,599 face images, each of which has 40 attribute annotations. It is split into three parts: 162,770, 19,867 and 19,962 images for training, validation and test, respectively. This dataset contains 40 tasks and each task is a binary classification problem for one attribute.

Implementation Details We use the ResNet-18 network as a sharing feature extractor and a fully connected layer with two output units as a task-specific head for each task. All the input images are resized to 64×64. The Adam optimizer with the learning rate as 10^{-3} is applied for training and the batch size is set to 512. The cross-entropy loss is used for the 40 tasks.

Results Since the number of tasks is large, we only report the average classification accuracy on the forty tasks in the CelebA dataset and the results are shown in Table 9. The proposed RLW strategy slightly outperforms EW and performs comparable with all the baseline methods.

Table 9: Average classification accuracy (%) of different methods on the **CelebA** dataset with forty tasks. The best results for each task on each measure are highlighted in **bold**.

Weighting Strategy	**Avg Acc**
EW	90.70
GradNorm	90.77
UW	90.84
MGDA	90.73
DWA	90.77
PCGrad	90.85
GradDrop	90.71
IMTL	**90.91**
GradVac	90.75
RLW (Uniform)	90.86
RLW (Normal)	90.73
RLW (Dirichlet)	90.89
RLW (Bernoulli)	90.71
RLW (constrained Bernoulli)	90.81
RLW (random Normal)	90.88

C.5 Results on the Office-31 and Office-Home Datasets

Datasets The Office-31 dataset (Saenko et al., 2010) consists of three domains: Amazon (**A**), DSLR (**D**), and Webcam (**W**), where each domain contains 31 object categories. The Office-31 dataset contains 4,110 labeled images and we randomly split these samples with 60% for training, 20% for validation, and the rest 20% for test. The Office-Home dataset (Venkateswara et al., 2017) has four domains: artistic images (**Ar**), clip art (**Cl**), product images (**Pr**), and real-world images (**Rw**). It has 15,500 labeled images in total and each domain contains 65 classes. We divide the entire data in the same proportion as Office-31. For both two datasets, we consider the multi-class classification problem on each domain as a task. Similar to multilingual tasks from the XTREME benchmark, each task in both Office-31 and Office-Home datasets has its own input images, thus MTL problems in these two datasets are homogeneous MTL problems (Zhang & Yang, 2021).

Implementation Details We use the same configuration for both Office-31 and Office-Home datasets. Specifically, the ResNet-18 network pretrained on the ImageNet dataset is used as a sharing backbone among tasks and a fully connected layer is applied as a task-specific output layer for each task. All the input images are resized to 224×224. We use the Adam optimizer with the learning rate as 10^{-3} and set the batch size to 64 for training. The cross-entropy loss is applied for all tasks of both two datasets.

Table 10: Classification accuracy (%) of different methods on the **Office-31** and **Office-Home** datasets. The best results for each task are highlighted in **bold**.

Weighting Strategy	Office-31				Office-Home				
	A	D	W	Avg	Ar	Cl	Pr	Rw	Avg
EW	87.17	98.36	**99.44**	94.99	68.88	80.93	91.73	81.72	80.81
GradNorm	86.66	99.18	98.89	94.91	68.88	80.49	91.73	**82.70**	80.95
UW	87.35	99.18	98.89	95.13	69.63	**82.55**	91.20	81.08	81.12
MGDA	87.52	99.18	**99.44**	95.38	69.44	79.30	91.63	81.72	80.52
DWA	87.52	99.18	**99.44**	95.38	70.39	79.95	90.36	82.05	80.69
PCGrad	87.00	98.36	98.33	94.56	68.31	80.71	90.57	81.94	80.38
GradDrop	87.17	98.36	98.89	94.80	68.50	81.47	91.41	81.40	80.69
IMTL	87.52	98.36	98.89	94.92	67.93	80.49	**91.94**	82.05	80.60
GradVac	**88.71**	98.36	98.89	95.32	67.93	82.01	91.84	81.72	80.87
RLW (Uniform)	**88.71**	98.36	98.89	95.32	70.01	81.79	90.88	80.97	80.92
RLW (Normal)	88.03	99.18	98.89	95.36	69.44	80.93	90.36	**82.70**	80.86
RLW (Dirichlet)	88.37	99.18	98.89	95.48	70.01	81.69	91.31	81.18	81.05
RLW (Bernoulli)	88.20	99.18	98.89	95.42	68.69	82.23	91.73	81.72	81.09
RLW (constrained Bernoulli)	88.37	**100**	**99.44**	**95.94**	**70.77**	81.69	91.41	82.05	**81.48**
RLW (random Normal)	88.37	98.36	98.89	95.20	68.88	82.12	91.41	81.72	81.03

Results According to the results on Office-31 and Office-Home datasets shown in Table 10, it is noticeable that RLW with six different types of distributions outperforms EW on both datasets. In addition, the RLW method performs better than most baseline methods. Moreover, the RLW method with the constrained Bernoulli distribution achieves the best average classification accuracy on both Office-31 and Office-Home datasets, and this strategy achieve 100% accuracy on the DSLR domain in the Office-31 dataset.

C.6 Additional results on XTREME benchmark

Datasets The datasets used in the NER, POS, and PI tasks are the Wikiann dataset (Pan et al., 2017), Universal Dependency v2.5 treebanks (Nivre et al., 2020), and PAWS-X dataset (Yang et al., 2019), respectively. For the NIL task, the MultiNLI dataset (Williams et al., 2018) and the XNLI dataset (Conneau et al., 2018) are used for training and evaluation, respectively. For the NER, NIL, and PI tasks, the same four languages including English (`en`), Mandarin (`zh`), German (`de`), and Spanish (`es`), are used. For the POS task, Telugu (`te`) and Vietnamese (`vi`) instead of German and Spanish are used. The numbers of data for all the tasks are summarized in Table 11.

Table 11: The numbers of training, validation, and test data for each language in each task from the XTREME benchmark.

	NER	POS	NIL	PI
`en`	20.0K+10.0K+10.0K	6.9K+1.8K+3.2K	392.7K+2.5K+5.0K	49.4K+2.0K+2.0K
`zh`	20.5K+10.3K+10.3K	4.0K+0.5K+2.9K	392.7K+2.5K+5.0K	49.4K+2.0K+2.0K
`de`	20.0K+10.0K+10.0K	-	392.7K+2.5K+5.0K	49.4K+2.0K+2.0K
`es`	20.0K+10.0K+10.0K	-	392.7K+2.5K+5.0K	49.4K+2.0K+2.0K
`te`	-	1.0K+0.1K+0.1K	-	-
`vi`	-	1.4K+0.8K+0.8K	-	-

Results The results on the NLI and NER multilingual tasks are shown in Table 12. The empirical observations are similar to those on the POS and PI tasks shown in Table 3. Specifically, the RLW method with all the distributions outperforms EW in the two multilingual problems. In addition, RLW achieves comparable and even better performance than those baseline methods. For example, on the NLI multilingual problem, RLW achieve the highest average accuracy.

Table 12: Performance on two multilingual problems, i.e., NLI and NER from the **XTREME benchmark**. The best results for each language are highlighted in **bold**.

Weighting Strategy	NLI (Accuracy)					NER (F1 Score)				
	`en`	`zh`	`de`	`es`	**Avg**	`en`	`zh`	`de`	`es`	**Avg**
EW	82.05	77.04	77.80	78.48	78.84	85.21	83.61	90.73	92.89	88.11
GradNorm	83.33	77.34	78.20	79.54	79.60	85.70	84.16	91.13	93.17	88.54
UW	82.93	77.58	77.96	**79.72**	79.55	85.59	84.42	91.14	93.24	88.60
MGDA	83.11	77.32	77.84	79.02	79.32	85.68	84.13	90.99	93.28	88.52
DWA	83.17	77.52	78.06	79.56	79.58	85.55	84.42	90.91	93.22	88.53
PCGrad	82.55	77.06	**78.84**	79.24	79.42	84.69	82.26	90.17	92.72	87.46
GradDrop	82.15	76.10	77.36	78.22	78.46	**85.92**	**84.45**	91.06	93.29	**88.68**
IMTL	82.63	77.78	76.56	79.28	79.06	84.97	83.92	90.82	92.79	88.12
GradVac	81.39	76.74	76.94	78.50	78.39	84.38	82.54	90.03	92.29	87.31
RLW (Uniform)	83.21	**78.14**	78.04	79.58	**79.74**	85.48	84.18	90.92	**93.30**	88.47
RLW (Normal)	82.63	76.36	77.26	79.24	78.87	85.25	83.79	**91.24**	**93.30**	88.40
RLW (Dirichlet)	82.99	76.92	77.40	78.56	78.97	85.31	83.97	91.06	93.18	88.38
RLW (Bernoulli)	83.11	77.52	78.10	78.84	79.39	85.59	83.84	91.09	93.01	88.38
RLW (constrained Bernoulli)	82.69	76.54	77.74	78.70	78.92	85.56	83.31	90.98	92.86	88.18
RLW (random Normal)	**83.65**	77.72	77.78	79.16	79.58	85.15	84.09	90.94	93.22	88.35

C.7 Results of different backbones on NYUv2 dataset

Those loss weighting strategies are agnostic to both architectures (or sharing pattern) and backbone networks. Tables 13 and 14 show consistent results when applying the SOTA methods and the proposed RLW strategy on different backbone networks, i.e., ResNet-18 and ResNet-101, respectively.

Compared the results of the Tables 13, 1 (or 15), and 14, we can find that with the depth of the backbone network increasing, the performance of each loss weighting strategy is improved. In addition, the proposed RLW method can always outperform EW on all backbones. On the other hand, the performance of those SOTA methods could degrade on deeper backbones. For example, the best relative improvement (i.e., Δ_{p}) of SOTA methods is 1.58%, 0.48%, and 0.37% for ResNet-18, ResNet-50, and ResNet-101 backbones, respectively. Compared with those baseline strategies, the RLW method can always achieve a significant improvement and even outperform the SOTA baselines on all the backbones.

Table 13: Performance on the **NYUv2** validation dataset using **ResNet-18** as the backbone network. The best results for each task on each measure are highlighted in **bold**. ↑ (↓) indicates that the higher (lower) the result, the better the performance.

Weighting Strategy	Segmentation		Depth		Surface Normal					Δ_{p}↑	$\Delta_{\mathrm{t}}^{\mathrm{d}}$↓	Δ_{t}↓
					Angle Distance		Within t°					
	mIoU↑	Pix Acc↑	Abs Err↓	Rel Err↓	Mean↓	Median↓	11.25↑	22.5↑	30↑			
EW	47.65	70.71	0.4165	0.1740	26.2454	19.4920	31.53	55.66	66.87	+0.00	+0.000	×1.00
GradNorm	48.11	71.21	0.4200	0.1694	26.4314	20.0192	30.63	54.66	66.08	−0.03	+18.16	×1.76
UW	48.16	71.41	0.4172	0.1729	26.1624	19.4585	31.65	55.74	66.97	+0.49	+0.273	×1.01
MGDA	47.79	71.03	0.4180	0.1745	26.1139	19.4272	31.71	55.80	66.98	+0.13	+39.53	×2.65
DWA	48.42	71.53	0.4174	0.1746	26.2161	19.4966	31.50	55.69	66.92	+0.37	+0.074	×1.00
PCGrad	48.35	71.50	0.4185	0.1761	25.8142	19.1808	31.95	56.37	67.60	+0.61	+26.80	×2.12
GradDrop	47.78	70.84	0.4169	0.1740	26.1840	19.5099	31.65	55.65	66.86	+0.09	+18.32	×1.77
IMTL	47.38	70.53	0.4157	**0.1677**	25.6736	18.6073	**32.96**	**57.35**	**68.25**	+1.58	+18.21	×1.76
GradVac	48.24	71.21	0.4146	0.1718	25.8110	19.1442	32.03	56.41	67.56	+1.10	+27.30	×2.14
RLW (Uniform)	48.14	71.27	0.4173	0.1747	25.9891	19.4086	31.53	55.87	67.17	+0.35	+0.033	×1.00
RLW (Normal)	**48.65**	71.55	**0.4103**	0.1727	25.8828	19.0478	32.12	56.57	67.59	+1.47	+0.051	
RLW (Dirichlet)	47.79	71.29	0.4144	0.1752	25.8251	19.0171	32.24	56.63	67.70	+0.77	+0.062	
RLW (Bernoulli)	48.51	**71.62**	0.4121	0.1715	**25.5254**	18.8868	32.53	56.89	68.01	**+1.79**	−0.033	
RLW (constrained Bernoulli)	48.12	71.49	0.4116	0.1731	25.5469	**18.8112**	32.53	57.01	68.09	+1.54	−0.067	
RLW (random Normal)	47.79	71.10	0.4160	0.1760	25.7642	19.1556	31.93	56.39	67.66	+0.46	+0.041	

Except the relative training speed Δ_{t}, we also provide the difference of training speed for one epoch between each method and the EW method as $\Delta_{\mathrm{t}}^{\mathrm{d}}$ in Tables 13, 14, and 15 to better show the computational efficiency of the proposed RLW strategy. It can be observed that most of the SOTA methods, especially MGDA, PCGrad, and GradVac, need more training time (i.e., $\Delta_{\mathrm{t}}^{\mathrm{d}}$) when the backbone becomes deeper. One main reason is due to the computation of all task gradients. Compared with those baseline methods, the proposed RLW strategy brings negligible and backbone-agnostic computational cost, which indicates RLW is as efficient as EW. Even, RLW with Bernoulli and constrained

Bernoulli distributions are even more efficient than EW since only some tasks are involved in the loss computation. Moreover, although DWA and UW are efficient since DWA computes loss weights via the loss ratio and UW updates loss weights by gradient descent methods, those two methods do not perform well in all cases. For example, the performance of DWA is 1.02% lower than EW on the NYUv2 dataset with the ResNet-101 backbone (in Table 14). Differently, the proposed RLW strategy can not only keep the efficiency but also outperform EW and achieve comparable performance with SOTA methods in all cases.

Table 14: Performance on the **NYUv2** validation dataset using **ResNet-101** as the backbone network. The best results for each task on each measure are highlighted in **bold**. ↑ (↓) indicates that the higher (lower) the result, the better the performance.

Weighting Strategy	Segmentation		Depth		Surface Normal					Δ_p↑	Δ_t^d↓	Δ_t↓
					Angle Distance		Within t°					
	mIoU↑	Pix Acc↑	Abs Err↓	Rel Err↓	Mean↓	Median↓	11.25↑	22.5↑	30↑			
EW	54.85	76.14	0.3720	0.1523	23.1482	16.4630	36.40	62.43	73.03	+0.00	+0.000	×1.00
GradNorm	54.62	**76.48**	0.3804	0.1556	23.0503	16.6280	36.15	62.03	72.97	−0.87	+51.83	×1.66
UW	**55.58**	76.46	0.3733	0.1540	23.0585	16.5488	36.22	62.22	73.05	−0.01	+0.318	×1.01
MGDA	55.35	76.18	0.3682	0.1536	22.8771	16.4448	36.43	62.55	73.48	+0.33	+129.6	×2.66
DWA	55.12	75.95	0.3759	0.1540	23.5170	16.9624	35.10	61.47	72.49	−1.02	+0.063	×1.00
PCGrad	54.57	76.15	0.3754	0.1519	22.8305	16.3208	36.68	62.75	73.48	+0.08	+87.58	×2.12
GradDrop	54.95	76.01	0.3718	0.1523	23.0108	16.5162	36.34	62.31	73.14	+0.02	+52.94	×1.68
IMTL	54.44	75.87	0.3736	0.1525	**22.6043**	**16.0620**	37.22	**63.30**	**73.98**	+0.37	+51.64	×1.66
GradVac	54.90	76.41	0.3709	0.1523	22.8708	16.3755	36.65	62.66	73.38	+0.34	+88.40	×2.13
RLW (Uniform)	55.16	76.41	0.3721	**0.1501**	22.9222	16.4800	36.42	62.41	73.26	+0.47	+0.034	
RLW (Normal)	55.02	76.15	0.3703	0.1539	22.7994	16.2493	36.82	62.86	73.54	+0.31	+0.046	
RLW (Dirichlet)	54.99	76.36	0.3697	0.1539	22.7090	16.2848	36.82	62.80	73.57	+0.38	+0.059	×1.00
RLW (Bernoulli)	55.37	76.39	0.3733	0.1527	22.8259	16.3255	36.73	62.72	73.43	+0.39	−0.525	
RLW (constrained Bernoulli)	55.13	76.14	0.3703	0.1516	22.8667	16.4130	36.52	62.54	73.36	+0.40	−0.624	
RLW (random Normal)	54.94	76.46	**0.3678**	0.1509	22.7614	16.1379	**37.27**	63.02	73.62	**+0.96**	+0.066	

C.8 Learning to combine the distributions for RLW

In this section, we proposed a method, called RLW-L2C, to learn a combination of those six different distributions for RLW. Specifically, in each iteration, we firstly sample loss weights from each of six distributions, then sum those weights elementwisely weighted by learnable distribution weights, and finally normalize the aggregated weights by the softmax function. The results of RLW-L2C on the NYUv2 dataset are shown in Table 15. Although the performance of RLW-L2C is slightly inferior to the best performance of RLW with single distribution (i.e. 0.09% performance degradation), this method still significantly outperforms those SOTA baselines and it can be adopted to any datasets and architectures by adaptively learning how to combine different distributions. Moreover, RLW-L2C is computationally efficient by only increasing about 0.01% computational cost over EW. Thus, those empirical results show that RLW-L2C is an adaptive, effective, and efficient loss weighting strategy for MTL.

Table 15: Performance on the **NYUv2** validation dataset with three tasks: 13-class semantic segmentation, depth estimation, and surface normal prediction. The best results for each task on each measure are highlighted in **bold**. ↑ (↓) indicates that the higher (lower) the result, the better the performance.

Weighting Strategy	Segmentation		Depth		Surface Normal					Δ_p↑	Δ_t^d↓	Δ_t↓
					Angle Distance		Within t°					
	mIoU↑	Pix Acc↑	Abs Err↓	Rel Err↓	Mean↓	Median↓	11.25↑	22.5↑	30↑			
EW	53.91	75.56	0.3840	0.1567	23.6338	17.2451	34.94	60.65	71.81	+0.00	+0.000	×1.00
GradNorm	53.81	75.35	0.3863	0.1556	23.6106	17.2565	34.98	60.58	71.76	−0.06	+51.88	×1.82
UW	53.15	75.41	0.3817	0.1576	23.6487	17.2040	34.98	60.71	71.80	−0.24	+0.318	×1.01
MGDA	53.66	75.37	0.3864	0.1610	23.4757	16.9912	35.44	61.17	72.16	−0.35	+100.8	×2.64
DWA	53.33	75.42	0.3834	0.1556	23.5806	17.1242	35.18	60.88	71.91	+0.07	+0.051	×1.00
PCGrad	53.34	75.43	0.3857	0.1600	23.2293	16.6966	36.09	61.80	72.66	+0.12	+68.42	×2.10
GradDrop	53.80	75.56	0.3857	0.1587	23.8726	17.1406	35.10	60.72	71.60	−0.33	+52.70	×1.84
IMTL	52.90	74.88	0.3883	0.1632	**23.0534**	**16.5304**	**36.30**	**62.20**	**73.08**	−0.35	+51.59	×1.82
GradVac	53.52	75.43	0.3840	0.1559	23.2892	16.8601	35.67	61.53	72.46	+0.48	+65.56	×2.11
RLW (Uniform)	54.09	75.78	0.3826	0.1563	23.6272	17.2711	34.73	60.67	71.87	+0.17	+0.089	
RLW (Normal)	54.19	**75.98**	0.3789	0.1570	23.1984	16.7944	35.71	61.74	72.77	+1.02	+0.067	
RLW (Dirichlet)	53.54	75.45	0.3834	0.1547	23.6392	17.0715	35.28	60.92	71.88	+0.27	+0.044	×1.00
RLW (Bernoulli)	53.72	75.62	0.3850	0.1610	23.1413	16.6591	36.08	61.98	72.86	+0.28	−0.394	
RLW (constrained Bernoulli)	**54.32**	75.78	**0.3779**	**0.1533**	23.2101	16.9354	35.41	61.44	72.58	**+1.29**	−0.457	
RLW (random Normal)	54.08	75.77	0.3815	0.1581	23.5598	16.9577	35.53	61.20	72.13	+0.39	+0.030	
RLW-L2C	54.24	75.76	0.3837	0.1515	23.3497	16.8673	35.57	61.44	72.43	+1.20	+0.355	×1.01