SAFE MULTI-TASK LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

In recent years, Multi-Task Learning (MTL) attracts much attention due to its good performance in many applications. However, many existing MTL models cannot guarantee that its performance is no worse than its single-task counterpart on each task. Though this phenomenon has been empirically observed by some works, little work aims to handle the resulting problem, which is formally defined as *negative sharing* in this paper. To achieve *safe multi-task learning* where no *negative sharing* occurs, we propose a Safe Multi-Task Learning (SMTL) model, which consists of a public encoder shared by all the tasks, private encoders, gates, and private decoders. Specifically, each task has a private encoder, a gate, and a private decoder, where the gate is to learn how to combine the private encoder and public encoder for the downstream private decoder. To reduce the storage cost during the inference stage, a lite version of SMTL is proposed to allow the gate to choose either the public encoder or the corresponding private encoder. Moreover, we propose a variant of SMTL to place all the gates after decoders of all the tasks. Experiments on several benchmark datasets demonstrate the effectiveness of the proposed methods.

1 INTRODUCTION

Multi-Task Learning (MTL) (Caruana, 1997; Zhang & Yang, 2021), which aims to improve the generalization performance of multiple learning tasks by sharing knowledge among those tasks, has attracted much attention in recent years. Compared with single-task learning, it not only improves the performance but also reduces the training and inference time. Though MTL has demonstrated its usefulness in many applications, MTL cannot guarantee to improve the performance of all the tasks when compared with single-task learning. As empirically observed in (Lee et al., 2016; Guo et al., 2020; Sun et al., 2020; Standley et al., 2020), when learning on multiple tasks together, each of some existing MTL models can achieve better performance on some of the tasks than a single-task model but perform worse on the other tasks. Such phenomenon is defined as the *negative sharing* phenomenon here, which is similar to the 'negative transfer' phenomenon (Wang et al., 2019) in transfer learning (Yang et al., 2020) but with some difference in the ways of knowledge transfer and sharing in those two learning paradigms as discussed later. One reason for the occurrence of *negative sharing* is that there are unrelated tasks among tasks in investigation, making jointly learning these tasks impair the performance of some tasks.

To the best of our knowledge, there is little work to study the *negative sharing* problem for MTL. In this paper, we firstly give a formal definition for *negative sharing* occurred in MTL. Then we formally define an ideal and also basic situation of MTL, *safe multi-task learning*, where an MTL model performs no worse than its single-task counterpart on each task. Hence, *safe multi-task learning* means that there is no *negative sharing* occurred. According to the definition of MTL (Caruana, 1997; Zhang & Yang, 2021), we can see that every MTL model should achieve *safe multi-task learning*. Otherwise, single-task learning is more preferred than MTL, since an unsafe MTL model may bring the risk of worsening the generalization performance of some or even all the tasks. Moreover, we formally define η*-partially safe multi-task learning* as a loose version of *safe multi-task learning* to allow the MTL model to perform worse than its single-task counterpart on tasks with a proportion no larger than η.

To achieve *safe multi-task learning*, we propose a Safe Multi-Task Learning (SMTL) model. Specifically, given m learning tasks, the SMTL model consists of a public encoder shared by all the tasks, m private encoders for the m tasks, m gates for the m tasks, and m private decoders for the m

tasks. Hence each task has a private encoder, a gate, and a private decoder. The gate of each task is responsible of learning how to linearly combine the public encoder and the corresponding private encoder for the downstream private decoder. To reduce the storage cost during the inference stage, we propose a lite version of SMTL via the Gumbel-softmax trick (Jang et al., 2017; Maddison et al., 2017) to enforce each gate to choose either the public encoder or private encoder. Moreover, to study the impact of different locations of the gates, we propose variants of the SMTL model, which place the gates after the decoders. Furthermore, we analyze the SMTL model from the perspectives of generalization bound and optimization. Experiments on several MTL benchmark datasets demonstrate the effectiveness of the proposed methods.

The main contributions of this paper are summarized as follows.

- We provide formal definitions for MTL, including *negative sharing*, *safe multi-task learning*, and η*-partially safe multi-task learning*.
- To achieve *safe multi-task learning*, we propose a simple and effective SMTL model. Built on the SMTL model, we propose its variants.
- We conduct extensive experiments to demonstrate the superiority of the proposed methods over state-of-the-art methods.

2 Related Work

MTL has been extensively studied in recent years (Evgeniou & Pontil, 2004; Zhang & Yeung, 2010; Kumar & Daume III, 2012; Zhang et al., 2021; Guo et al., 2021). How to design a good network architecture for MTL is an important issue. The most popular model is the multi-head hard sharing architecture (Caruana, 1997; Zhang et al., 2014; Long & Wang, 2015; Liu et al., 2015; Ruder et al., 2019), which shares the first several layers among all the tasks and allow the subsequent layers to be specific to different tasks. Then, to better handle task relationships, different MTL architectures have been proposed. For example, Misra et al. (2016) propose a cross-stitch network to learn to linearly combine hidden representations of different tasks. Liu et al. (2019) propose a Multi-Task Attention Network (MTAN), which consists of a shared network and an attention module for each task so that both shared and task-specific feature representations can be learned via the attention mechanism. Gao et al. (2019) propose a Neural Discriminative Dimensionality Reduction (NDDR) layer to enable automatic feature fusing at every layer from different tasks. Sun et al. (2020) propose an Adaptive Sharing (AdaShare) method to learn the sharing pattern through a task-specific policy that selectively chooses which layers to be executed for each task. Guo et al. (2020) propose an algorithm to learn where to share or branch within a network for MTL. Cui et al. (2021) propose an Adaptive Feature Aggregation (AFA) layer, where a dynamic aggregation mechanism is designed to allow each task to adaptively determine the degree of the knowledge sharing between tasks. All the existing works do not study how to achieve *safe multi-task learning*, which is the focus of this paper.

3 Safe Multi-Task Learning

In this section, we first formally define some terminologies for MTL. Then we introduce the proposed SMTL method. Moreover, we propose some variants of the SMTL model. Finally, we provide some analyses for SMTL.

3.1 Definitions

Definition 1 (Negative Sharing)**.** *For an MTL model which is trained on multiple learning tasks jointly, if its generalization performance on some tasks is inferior to the generalization performance of the corresponding single-task model which is trained on each task separately, then* negative sharing *occurs.*

Remark 1. Negative sharing *occurs when some tasks are totally or partially irrelevant to other tasks. In this case, manually enforcing all the tasks to have some forms of sharing will impair the performance of some or even all the tasks. In Definition 1, the MTL model and the single-task model usually have similar architectures as totally different architectures may bring additional confounding factors. Moreover,* negative sharing *is similar to negative transfer (Wang et al., 2019)*

in transfer learning (Yang et al., 2020). However, knowledge transfer in transfer learning is directed as it is from a source domain to a target domain, while knowledge sharing in MTL is among all the tasks, making it undirected. From this perspective, negative sharing *is different from negative transfer.*

Definition 2 (Safe Multi-Task Learning)**.** *When no* negative sharing *occurs for an MTL model on a dataset, this MTL model is said to achieve* safe multi-task learning *on this dataset.*

The ideal situation for an MTL model is to achieve *safe multi-task learning*. However, not all the MTL methods can achieve *safe multi-task learning* and hence we define *η-partially safe multi-task learning*, which can be viewed as a loose version of *safe multi-task learning*.

Definition 3 (η-Partially Safe Multi-Task Learning)**.** *Given multiple learning tasks in a dataset, η-*partially safe multi-task learning *($0 \leq \eta \leq 100$) indicates that on about η percentage of tasks, the generalization performance of an MTL model is no worse than that of its single-task counterpart.*

When η equals 100, *η-partially safe multi-task learning* becomes *safe multi-task learning*. When η is equal to 0, the MTL model performs worse than its single-task counterpart in all the tasks.

3.2 SMTL

With m learning tasks $\{\mathcal{T}_i\}_{i=1}^m$, our goal is to design a model that can achieve *safe multi-task learning*. To achieve this goal, we propose the SMTL model, which is introduced in the following.

Without loss of generality, we consider the case that different tasks share the input data or equivalently each data point has an output for each task. As shown in the left figure of Figure 1, the SMTL model can be divided into four parts: a public encoder f_S shared by all the tasks, m private encoders $\{f_t\}_{t=1}^m$ for m tasks, m gates $\{g_t\}_{t=1}^m$ for m tasks, and m private decoders $\{h_t\}_{t=1}^m$ for m tasks. For task t, its model consists of the public encoder f_S, the private encoder f_t, the gate g_t, and the private decoder h_t, where f_S and f_t are combined by g_t. Specifically, given a data point $\mathbf{x}$, the gate g_t in task t receives two inputs: $f_S(\mathbf{x})$ and $f_t(\mathbf{x})$, and outputs $g_t(f_S(\mathbf{x}), f_t(\mathbf{x}))$, which is fed into h_t to obtain the final prediction $h_t(g_t(f_S(\mathbf{x}), f_t(\mathbf{x})))$, which is used to define a loss for $\mathbf{x}$.

Here the gate g_t is to determine the contributions of f_S and f_t. Ideally, when task t is unrelated to other tasks, g_t should choose f_t only. On another extreme where all the tasks have the same data distribution, all the tasks should use the same model and hence g_t should choose f_S only. On cases between the two extremes, g_t can combine f_S and f_t in proportion. To achieve the aforementioned effects, we use a simple convex combination function for g_t as

$$g_t(f_S(\mathbf{x}), f_t(\mathbf{x})) = \alpha_t f_S(\mathbf{x}) + (1 - \alpha_t) f_t(\mathbf{x}), \tag{1}$$

where $\alpha_t \in [0, 1]$ defines the weight of $f_S(\mathbf{x})$ and is a learnable parameter. When α_t equals 0, only the private encoder f_t will be used, which corresponds to the unrelated case. When α_t is equal to 1, only the public encoder f_S will be used, which is corresponding to the case that all the tasks follow the same distribution. When α_t is between 0 and 1, f_S and f_t are combined with proportions α_t and $1 - \alpha_t$, respectively, where α_t can be adaptively learned to minimize the training loss on task t. Thus, the entire objective function of SMTL is formulated as

$$\min_{\Theta \in \mathcal{C}} \frac{1}{mn} \sum_{i=1}^{n} \sum_{t=1}^{m} \mathcal{L}_t(\mathbf{y}_t^i, h_t(g_t(f_S(\mathbf{x}_i), f_t(\mathbf{x}_i)))), \tag{2}$$

where $\mathbf{x}_i$ denotes the ith data point, $\mathbf{y}_t^i$ denotes the label of $\mathbf{x}_i$ in task t, n denotes the total number of data points in the training dataset, Θ includes all the parameters in f_S, $\{f_t\}_{t=1}^m$, $\{g_t\}_{t=1}^m$, and $\{h_t\}_{t=1}^m$, $\mathcal{C} = \{\Theta | 0 \leq \alpha_t \leq 1, \forall t\}$ denotes the feasible set for Θ, and $\mathcal{L}_t$ denotes the loss function for task t (e.g., the pixel-wise cross-entropy loss for semantic segmentation, L_1 loss for depth estimation, and element-wise dot product loss for surface normal prediction).

To see why the proposed SMTL model could achieve *safe multi-task learning*, we compare SMTL and the corresponding Single-Task Learning (STL) model which consists of a private encoder f_t and a private decoder h_t. It is easy to see that SMTL can reduce to the STL model for some or even all the tasks by making the gates of those tasks choose the corresponding private encoders (i.e., setting α_t's of those tasks to 0). So if a task is unrelated to other tasks, the SMTL model can use the gating mechanism to separate this outlier task from other tasks, which may help achieve *safe multi-task learning*. Moreover, we can show that the training loss of the SMTL model is no larger than the average of that of the STL model on each task. To see that, it is easy to show that the STL model for

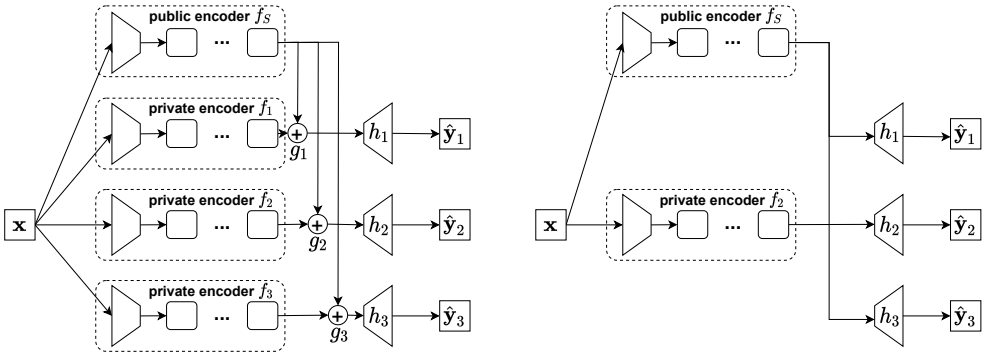

Figure 1: Illustration of the SMTL and L-SMTL models. **Left figure**: Pipeline for the SMTL model, which is identical to the training phase of L-SMTL. For task t, $\mathbf{x}$ is first fed into both the public encoder f_S and private encoder f_t, then it is through the gate g_t to obtain the combined feature representation, and finally it is through the private decoder h_t to obtain the output $\hat{\mathbf{y}}_t$. The number of tasks is set to three for illustration. **Right figure**: Test phase for L-SMTL. After finishing the training process of L-SMTL, g_t can choose which encoder (i.e., the public encoder f_S or private encoder f_t) is used for each task. In this way, at the test process, only the chosen encoders need to be saved, which could reduce the number of parameters and speedup the inference. In this illustration, task 1 and task 3 choose the public encoder, while task 2 goes through its private encoder.

task t can be represented as $h_t(g_t^0(\emptyset, f_t(\mathbf{x})))$, where g_t^0 denotes the gate of task t with α_t as 0 and $\emptyset$ denotes a null network. As g_t^0 is a feasible gate for SMTL, after sufficient training, we probably have

$$\min_{\Theta \in \mathcal{C}} \frac{1}{mn} \sum_{i=1}^{n} \sum_{t=1}^{m} \mathcal{L}_t(\mathbf{y}_t^i, h_t(g_t(f_S(\mathbf{x}_i), f_t(\mathbf{x}_i)))) \leq \min_{\Theta'} \frac{1}{mn} \sum_{i=1}^{n} \sum_{t=1}^{m} \mathcal{L}_t(\mathbf{y}_t^i, h_t(g_t^0(\emptyset, f_t(\mathbf{x}_i)))),$$

where Θ' includes parameters in $\{f_t\}_{t=1}^m$ and $\{h_t\}_{t=1}^m$ in STL models for all the tasks. This inequality shows the benefit of the proposed SMTL model. Even though the training loss is not a tight estimation of the generalization loss, we think that it is an indicator to reflect the generalization performance of the two models. In Section 4.4, we study to use a bi-level formulation to learn the gates as hyperparameters on a validation set and its performance is comparable with the SMTL model based on problem (2), which proves the usefulness of the indication of the training loss.

Similarly, we can show that the training loss of the SMTL model is no larger than that of the DMTL model (a.k.a. the multi-head hard sharing network) which consists of a shared encoder by all the tasks and private decoders for m tasks. To see this, the DMTL model for task t can be represented as $h_t(g_t^1(f_S(\mathbf{x}), \emptyset))$, where g_t^1 denotes the gate of task t with α_t as 1. As g_t^1 is a feasible gate for SMTL, it is easy to get that the training loss of the SMTL model after sufficient training is lower than that of the DTML model, which could be one reason for the phenomenon that SMTL outperforms DMTL in our experiments.

3.3 LITE SMTL

The SMTL model is computationally efficient during both the training and inference stages, but it requires to keep all the private encoders as well as the public encoder at the inference stage. In some applications with limited storage resource such as edge computing, we hope to eliminate some private encoders that have small contributions to the corresponding tasks to reduce the storage cost. To achieve that, we propose a lite version of SMTL called L-SMTL. Specifically, the L-SMTL model will enforce each gate to choose either the public encoder f_S or the corresponding private encoder, which is equivalent to forcing each α_t to be either 1 or 0. Thus, as shown in the right figure of Figure 1, at the inference stage, the unchosen private encoders can be thrown away and we do not need to store parameters in them. However, the objective function of the L-SMTL model, which is similar to problem (2) by replacing the constraint $\alpha_t \in [0, 1]$ with $\alpha_t \in \{0, 1\}$, is non-differentiable as each α_t is binary valued.

To optimize the non-differentiable objective function in L-SMTL, we adopt the Gumbel-softmax trick (Jang et al., 2017; Maddison et al., 2017). Specifically, if task t chooses the public encoder

with the probability α_t, it can be cast as sampling $\tilde{\alpha}_t$ from a Bernoulli distribution $\mathcal{B}(p_{t,1})$ with probability $p_{t,1} = \alpha_t$ to assign $\tilde{\alpha}_t$ the value of 1 (i.e., using the public encoder for task t) and with probability $p_{t,0} = 1 - \alpha_t$ to assign $\tilde{\alpha}_t$ the value of 0 (i.e., using the private encoder for task t). However, since the sampling process is still non-differentiable, the Gumbel-softmax trick is used to reparameterize α_t. We first use an equivalent formulation for sampling $\tilde{\alpha}_t$ based on the Gumbel-Max trick (Gumbel, 1948) as

$$\tilde{\alpha}_t = \arg\max_{k\in\{0,1\}}(b_k + \log p_{t,k}), \tag{3}$$

where b_k ($k = 0, 1$) is drawn from a Gumbel distribution $\mathrm{Gumbel}(0, 1)$ independently (i.e., $b_k = -\log(-\log(u))$ where u is sampled from a uniform distribution $\mathcal{U}(0, 1)$). Then we use the softmax function to approximate the $\arg\max$ function and define

$$\hat{\alpha}_t = \frac{\exp((b_1 + \log p_{t,1})/\tau)}{\sum_{k\in\{0,1\}} \exp((b_k + \log p_{t,0})/\tau)}, \tag{4}$$

where τ is a temperature parameter. It is easy to show that $\hat{\alpha}_t = \tilde{\alpha}_t$ when $\tau \to 0$ and $\hat{\alpha}_t = \frac{1}{2}$ when $\tau \to \infty$. Thus, we use a small value of τ to make a sharp distribution for $\hat{\alpha}_t$. In implementations as shown in (Jang et al., 2017), the $\arg\max$ function in Eq. (3) is used in the forward pass and the softmax function in Eq. (4) is used in the backward pass to approximate true gradients.

3.4 Variant of SMTL and L-SMTL

To study the impact of the position of gates in SMTL, we introduce a variant of SMTL called SMTL$_c$, where all the gates are placed after all the decoders. Similarly, the SMTL$_c$ model also has a lite version called L-SMTL$_c$. An illustration for the SMTL$_c$ and L-SMTL$_c$ models is shown in Figure 2 in Appendix C.

Specifically, the SMTL$_c$ model can be divided into five parts, including a public encoder f_S shared by all the m tasks, m public decoders $\{h_{S,t}\}_{t=1}^m$ for the m tasks, m private encoders $\{f_t\}_{t=1}^m$ for the m tasks, m private decoders $\{h_t\}_{t=1}^m$ for the m tasks, and m gates $\{g_t\}_{t=1}^m$ for the m tasks. For task t, a data point $\mathbf{x}$ is fed into the public encoder f_S and the public decoder $h_{S,t}$ to get an output o_S, and it is also fed into the private encoder f_t and the private decoder h_t to get another output o_t. Then the gate g_t will adaptively combine the two outputs to obtain the final output, i.e., $\hat{y_t} = \alpha_t o_S + (1 - \alpha_t)o_t$.

Similar to the SMTL model, when task t is unrelated to other tasks, ideally the SMTL$_c$ model can only choose the private encoder f_t and the private decoder h_t with a zero α_t. Hence, the SMTL$_c$ model can achieve *safe multi-task learning* when the *negative sharing* occurs. Moreover, when $0 < \alpha_t < 1$, the combination of the public component consisting of the public encoder and decoder and the private component consisting of the private encoder and decoder may act in a way similar to ensemble learning, which may help improve the generalization performance. Moreover, similar to the SMTL model, the training loss of SMTL$_c$ could be lower than those of the STL and DMTL models. Built on the SMTL$_c$ model, the L-SMTL$_c$ model approximately learns binary-valued $\{\alpha_t\}$ via the Gumbel-softmax trick.

3.5 Analysis

We analyze the generalization bound for the SMTL method. We consider a general case that different tasks can have different data distributions. The probability measure for the data distribution in task t is denoted by μ_t and the data in all the tasks take the form of $(\bar{\mathbf{X}}, \bar{\mathbf{Y}}) \sim \prod_{t=1}^m (\mu_t)^n$, where $\bar{\mathbf{X}} = (\mathbf{X}_1, \ldots, \mathbf{X}_m)$, $\mathbf{X}_t = (\mathbf{x}_t^1, \ldots, \mathbf{x}_t^n)$ and $\bar{\mathbf{Y}}$ denotes the label of $\bar{\mathbf{X}}$. Here we consider the encoders $f_1, \ldots, f_m, f_S : \mathcal{X} \to \mathbb{R}^P$ as mapping functions and define $\varphi(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}), f_S(\mathbf{x})) : \mathcal{X} \to \mathbb{R}^{P(m+1)}$. The functions φ and h_t are assumed to be chosen from hypothesis classes $\mathcal{F}$ and $\mathcal{H}$, respectively. Note that we only use f_t and f_S for the representation in task t. For the ease of analysis, we define a weight vector $\beta_t \in \mathcal{M}_t$ for task t, where $\mathcal{M}_t = \{\beta_t \in \mathbb{R}_+^{m+1} \mid \beta_{t,t} + \beta_{t,m+1} = 1, \beta_{t,i} = 0 \text{ if } i \neq t \text{ and } i \neq m+1\}$, and $\beta_{t,i}$ represents the ith entry of β_t. Thus, for given α_t, the corresponding β_t satisfies $\beta_{t,t} = 1 - \alpha_t$ and $\beta_{t,m+1} = \alpha_t$. Therefore, the risk of task t in SMTL can be written as the expectation $\mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mu_t}[\mathcal{L}_t(\mathbf{y}, h_t(\beta_t^T \varphi(\mathbf{x}))]$. We denote by $\mathcal{E}$ the average expected risk of all the tasks. Then, the minimal risk is defined as

$$\mathcal{E}^* = \min_{h_t\in\mathcal{H},\beta_t\in\mathcal{M}_t,\varphi\in\mathcal{F}} \mathcal{E} = \min_{h_t\in\mathcal{H},\beta_t\in\mathcal{M}_t,\varphi\in\mathcal{F}} \frac{1}{m}\sum_{t=1}^m \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mu_t} \mathcal{L}_t(\mathbf{y}, h_t(\beta_t^T \varphi(\mathbf{x}))).$$

Then we obtain a generalization bound for the SMTL method with the proof in Appendix A.

Theorem 1 (Generalization bound). *Assume that $\mathcal{L}_t(\cdot,\cdot) \in [0,1]$ for $t = 1,\ldots,m$ is 1-Lipschitz w.r.t the second argument, and the function φ in $\mathcal{F}$ is M-Lipschitz continuous. Then for $(\bar{\mathbf{X}}, \bar{\mathbf{Y}}) \sim \prod_{t=1}^{m}(\mu_t)^n$, with probability at least $1-\delta$, we have*

$$\mathcal{E} - \mathcal{E}^* \leq \frac{C_1 M G(\mathcal{F}(\bar{\mathbf{X}})) + \min_{z\in\mathcal{F}(\bar{\mathbf{X}})} G(\mathcal{H}'(z))}{mn} + \frac{2QC_2 \sup_{\varphi\in\mathcal{F}} \|\varphi(\bar{\mathbf{X}})\|}{n\sqrt{m}} + \sqrt{\frac{8\ln(4/\delta)}{mn}}, \quad (5)$$

where C_1 and C_2 are two constants, $\mathcal{F}(\bar{\mathbf{X}}) = \{(\varphi(\mathbf{x}_t^i)) : \varphi \in \mathcal{F}\}$, $\|\cdot\|$ denotes the ℓ_2 norm, $\mathcal{H}' = \{z \in \mathbb{R}^{mnP(m+1)} \mapsto h_t(\beta_t^T z) : h_t \in \mathcal{H}, \beta_t \in \mathcal{M}_t\}$, $G(\cdot)$ denotes the Gaussian average, Q_t is defined as

$$Q_t = \sup_{z\neq\tilde{z}\in\mathbb{R}^{nP(m+1)}} \frac{1}{\|z-\tilde{z}\|}\mathbb{E}\sup_{h\in\mathcal{H},\beta\in\mathcal{M}_t}\sum_{i=1}^{n}\gamma_i(h(\beta^T z_i) - h(\beta^T \tilde{z}_i)),$$

$Q = \max_{1\leq t\leq m} Q_t$, and γ is a vector of independent standard normal variables.

In the generalization bound (5), the first term of the right-hand side can be regarded as the cost of estimating the feature map φ, the second term corresponds to the cost of estimating task-specific functions β_t and h_t, and the third term defines the confidence of the bound. The convergence rate of this bound is $O(\frac{1}{\sqrt{mn}})$, which is as tight as typical generalization bounds for MTL such as (Maurer et al., 2016). Moreover, in Appendix B, we discuss some necessary condition for the optimal α_t being 0 or 1.

4 Experiments

4.1 Experimental Setup

Experiments are conducted on four MTL benchmark datasets, including CityScapes (Cordts et al., 2016), NYUv2 (Silberman et al., 2012), PASCAL-Context (Mottaghi et al., 2014), and Taskonomy (Zamir et al., 2018). Detailed introductions of the four datasets are put in Appendix D.1.

The baseline methods in comparison include the Single-Task Learning (**STL**) that trains each task separately, the DMTL model which adopts the multi-head hard sharing architecture, **Cross-stitch network** (Misra et al., 2016), **MTAN** (Liu et al., 2019), **NDDR-CNN** (Gao et al., 2019), **AdaShare** (Sun et al., 2020), and **AFA** (Cui et al., 2021). For fair comparison, we use the same backbone for all the models in comparison.

For each task in the benchmark datasets, we use one or more evaluation metrics to thoroughly evaluate the performance. The detailed introduction of each evaluation metric is put in Appendix D.3. To better show the comparison between each method and STL, we report the relative performance of each method over STL in terms of the jth evaluation metric on task t as $\Delta_{t,j} = (-1)^{p_{t,j}}(\mathrm{M}_{t,j} - \mathrm{STL}_{t,j})$, where for a method M, $\mathrm{M}_{t,j}$ denotes its performance in terms of the jth evaluation metric for task t, $\mathrm{STL}_{t,j}$ is defined similarly, $p_{t,j}$ equals 1 if a lower value represents a better performance in terms of the jth metric in task t and 0 otherwise. So positive relative performance indicates better performance than STL, which is shown in green in the following tables, while worse performance corresponding to negative relative performance is shown in red. The overall relative improvement of a method M over STL is defined as $\Delta_I = \frac{1}{m}\sum_{t=1}^{m}\frac{1}{m_t}\sum_{j=1}^{m_t}\frac{\Delta_{t,j}}{\mathrm{STL}_{t,j}}$, where m_t denotes the number of evaluation metrics in task t. Moreover, to measure the safeness of each method, the estimation $\hat{\eta}$ of η in the definition of *η-partially safe multi-task learning* (i.e., Definition 3) for a method M is computed as $\hat{\eta} = \frac{1}{m}\sum_{t=1}^{m}\frac{1}{m_t}\sum_{j=1}^{m_t}\delta(\Delta_{t,j}) \times 100$, where $\delta(x)$ is a step function that outputs 0 when $x < 0$ and otherwise 1.

We use the Deeplab-ResNet (Chen et al., 2017) with atrous convolutions, a popular architecture for pixel-wise prediction tasks, as encoders and the ASPP architecture (Chen et al., 2017) as decoders. We adopt the ResNet-50 for the CityScapes and NYUv2 datasets to implement the the Deeplab-ResNet, and use the smaller ResNet-18 for the larger PASCAL-Context and Taskonomy datasets for training efficiency. We use the cross-entropy loss for the semantic segmentation, human parts segmentation and saliency estimation tasks, the cosine similarity loss for the surface normal prediction task, and the L1 loss for other tasks. For optimization, we use the Adam method (Kingma & Ba, 2014) with the learning rate as 10^{-4}. All the experiments are conducted on Tesla V100 GPUs.

4.2 EXPERIMENTAL RESULTS

Tables 1-4 show the performance of all the models in comparison on the four benchmark datasets. On the CityScapes dataset, the proposed SMTL, L-SMTL, L-SMTL$_c$, and some baseline methods (i.e., Cross-stitch, MTAN, and NDDR-CNN) all achieve *safe multi-task learning* (i.e., $\hat{\eta} = 100$), which indicates that their performance is better than that of the STL model in all tasks. In addition, the proposed SMTL model achieves the best overall relative improvement Δ_I, which demonstrates its effectiveness. On the NYUv2 and PASCAL-Context datasets, none of the baselines can achieve *safe multi-task learning*, but the proposed methods (i.e., SMTL, SMTL$_c$, and L-SMTL$_c$) can achieve that, which again shows the effectiveness of the proposed methods. Though the proposed L-SMTL method does not achieve *safe multi-task learning* on these two datasets, it still achieves a better $\hat{\eta}$ than baseline methods on the NYUv2 dataset and a comparable $\hat{\eta}$ on the PASCAL-Context dataset. On the PASCAL-Context dataset, the overall relative improvement of all the baselines are negative, while all the proposed methods achieves positive Δ_I's, which shows the superiority of the proposed methods. On the Taskonomy dataset, only the Cross-stitch network and the proposed SMTL, SMTL$_c$, and L-SMTL$_c$ methods can achieve *safe multi-task learning*. According to results shown in Table 4, we can see that the AFA method achieves the best Δ_I because it has the largest improvements on the keypoint detection and edge detection tasks, but it does not achieve *safe multi-task learning*, while the proposed methods can achieve that.

Table 1: Performance of various models on the CityScapes validation dataset. ↑ (↓) indicates the higher (lower) the result, the better the performance. The green color indicates that the corresponding method performs better than the STL method and the red color indicates oppositely.

Method	Segmentation		Depth		Δ_I ↑	$\hat{\eta}$ ↑
	mIoU ↑	Pix Acc ↑	Abs Err ↓	Rel Err↓		
STL	67.48	91.00	0.0139	46.2507		
DMTL	−0.08	−0.08	−0.0003	+0.8245	−0.0014	25
Cross-stitch	+0.53	+0.29	+0.0004	+1.8261	+0.0198	100
MTAN	+1.49	+0.59	+0.0003	+2.4999	+0.0260	100
NDDR-CNN	+0.54	+0.25	+0.0002	+1.3845	+0.0138	100
AdaShare	+0.68	+0.20	−0.0005	−20.651	−0.1175	50
AFA	+1.44	+0.52	−0.0019	−0.9136	−0.0323	50
SMTL	+1.50	+0.62	+0.0005	**+3.3886**	**+0.0346**	100
L-SMTL	+1.17	+0.51	+0.0001	+3.2747	+0.0252	100
SMTL$_c$	**+1.62**	**+0.63**	**+0.0008**	−1.9361	+0.0117	75
L-SMTL$_c$	+1.18	+0.38	**+0.0008**	+1.5060	+0.0279	100

For the proposed methods, we can see that the lite versions (i.e., L-SMTL and L-SMTL$_c$) perform comparable with SMTL and SMTL$_c$, respectively, which suggests that the elimination strategy works well on the four datasets. By comparing SMTL with SMTL$_c$, it seems that the performance is not so sensitive to the two positions of the gates, and similar observations hold for the L-SMTL and L-SMTL$_c$ methods. Moreover, the proposed SMTL and L-SMTL$_c$ methods can achieve *safe multi-task learning* on all the four datasets, while the proposed L-SMTL and SMTL$_c$ methods fail to achieve this. This observation may suggest the SMTL and L-SMTL$_c$ methods are more preferred than others.

Table 2: Performance of various models on the NYUv2 validation dataset. ↑ (↓) indicates the higher (lower) the result, the better the performance. The green color indicates that the corresponding method performs better than the STL method and the red color indicates oppositely.

Method	Segmentation		Depth		Surface Normal					Δ_I ↑	$\hat{\eta}$
					Angle Distance ↓		Within $t°$↑				
	mIoU ↑	Pix Acc ↑	Abs Err ↓	Rel Err↓	Mean	Median	11.25	22.5	30		
STL	53.11	75.20	0.3957	0.1632	22.26	15.49	38.61	64.43	74.69		
DMTL	+0.52	+0.16	+0.0104	+0.0032	−1.34	−1.57	−3.31	−3.42	−2.69	−0.0127	67
Cross-stitch	+0.43	+0.09	+0.0082	+0.0052	−0.31	−0.52	−0.82	−0.93	−0.77	+0.0041	67
MTAN	**+1.03**	**+0.82**	**+0.0176**	**+0.0083**	−0.51	−0.90	−1.80	−1.58	−1.16	+0.0098	67
NDDR-CNN	+0.73	+0.03	+0.0086	+0.0072	−0.34	−0.58	−0.94	−1.00	−0.77	+0.0065	67
AdaShare	−7.72	−5.42	−0.0508	−0.0213	−2.26	−2.08	−4.38	−4.40	−3.99	−0.1107	0
AFA	−1.57	−1.29	−0.0073	−0.0060	−1.97	−1.91	−3.54	−4.21	−3.73	−0.0449	0
SMTL	+0.16	+0.28	+0.0071	+0.0042	+0.26	+0.00	+0.03	+0.46	+0.53	+0.0102	100
L-SMTL	+0.12	+0.10	+0.0078	+0.0035	+0.39	−0.06	−0.25	+0.55	+0.67	+0.0091	85
SMTL$_c$	+0.09	+0.02	+0.0091	+0.0045	+0.26	+0.10	+0.30	+0.62	+0.59	+0.0117	100
L-SMTL$_c$	+0.46	+0.21	+0.0132	+0.0081	**+0.51**	**+0.28**	**+0.83**	**+1.09**	**+0.90**	**+0.0218**	100

4.3 ANALYSIS ON LEARNED $\{\alpha_t\}$

We record the learned $\{\alpha_t\}$ of the proposed models in Table 5. According to results for the SMTL and SMTL$_c$ methods, we can see that some α_t's are closed to 0.5, which means in those cases, the public encoder and the private encoder are both important to the entire model. Thus, only using the public encoder (i.e., DMTL) and only using the private encoder (i.e., STL) cannot achieve good

Table 3: Performance of various models on the PASCAL-Context validation dataset. ↑ (↓) indicates the higher (lower) the result, the better the performance. The green color indicates that the corresponding method performs better than the STL method and the red color indicates oppositely.

Method	Segmentation	Human Parts	Saliency		Surface Normal					Δ_I ↑	$\hat{\eta}$ ↑
					Angle Distance		Within t°				
	mIoU↑	mIoU↑	mIoU↑	maxF↑	Mean ↓	RMSE ↓	11.25 ↑	22.5 ↑	30 ↑		
STL	65.14	58.58	65.02	77.47	15.94	24.87	48.42	80.79	90.03		
DMTL	−0.37	−0.67	−0.92	−0.51	−1.73	−1.29	−6.43	−4.86	−3.02	−0.0262	0
Cross-stitch	−0.17	+0.05	−0.56	−0.40	−0.62	−0.45	−2.36	−2.68	−1.04	−0.0096	25
MTAN	−0.58	+0.50	−0.45	−0.23	−1.20	−0.89	−4.58	−3.31	−2.04	−0.0147	25
NDDR-CNN	+0.14	+0.60	+0.07	+0.00	−0.37	−0.24	−1.50	−0.94	−0.59	−0.0008	75
AdaShare	−12.7	−7.30	−3.65	−2.50	−1.68	−1.23	−6.46	−4.66	−2.83	−0.1017	0
AFA	**+2.12**	**+2.11**	−1.95	−3.96	−1.63	−1.28	−5.68	−4.42	−2.88	−0.0108	50
SMTL	+0.01	+1.05	+0.20	+0.13	+0.26	+0.22	+1.08	+0.74	+0.39	+0.0082	100
L-SMTL	+0.29	+1.39	−0.80	−0.16	+0.34	**+0.40**	+0.95	+1.18	**+0.81**	+0.0093	75
SMTL$_c$	+0.88	+1.43	+0.22	+0.12	+0.36	+0.32	+1.32	+1.08	+0.61	**+0.0142**	100
L-SMTL$_c$	+0.23	+0.82	**+0.61**	**+0.57**	**+0.53**	+0.38	**+2.43**	**+1.36**	+0.65	+0.0126	100

Table 4: Performance of various models on the Taskonomy validation dataset. ↑ (↓) indicates the higher (lower) the result, the better the performance. The green color indicates that the corresponding method performs better than the STL method and the red color indicates oppositely.

Method	Segmentation		Depth		Keypoints	Edges	Surface Normal					Δ_I ↑	$\hat{\eta}$ ↑
							Angle Distance ↓		Within t°↑				
	mIoU ↑	Pix Acc ↑	Abs Err ↓	Rel Err↓	Abs Err ↓	Abs Err ↓	Mean	Median	11.25	22.5	30		
STL	65.42	97.63	0.0072	0.0117	0.1103	0.1349	10.39	4.19	73.67	86.21	90.52		
DMTL	−0.74	−0.07	−0.0026	−0.0043	−0.0022	−0.0025	−1.25	−0.88	−3.43	−1.93	−1.44	−0.0983	0
Cross-stitch	+5.87	+0.67	**+0.0005**	**+0.0009**	+0.0100	+0.0019	+1.50	+0.17	+4.21	+3.81	+3.00	+0.0580	100
MTAN	+6.34	**+0.69**	−0.0011	−0.0019	−0.0248	−0.0111	+0.99	−0.18	+2.30	+2.89	+2.41	−0.0767	36
NDDR-CNN	**+6.64**	**+0.69**	+0.0002	−0.0028	+0.0133	+0.0043	+1.86	**+0.47**	+4.78	+4.18	+3.32	+0.0378	90
AdaShare	+4.90	+0.50	−0.0045	−0.0073	+0.0087	−0.0001	−1.17	−1.93	−3.93	−0.37	−0.12	−0.1265	40
AFA	+5.84	+0.57	−0.0016	−0.0027	**+0.0465**	**+0.0524**	+1.08	−0.17	+3.00	+3.27	+2.68	**+0.1331**	76
SMTL	+4.99	+0.58	+0.0003	+0.0006	+0.0158	+0.0045	+1.64	+0.29	+4.62	+3.95	+3.06	+0.0676	100
L-SMTL	+4.37	+0.53	−0.0006	−0.0011	+0.0129	+0.0026	+1.87	+0.19	+5.02	+4.49	+3.57	+0.0321	80
SMTL$_c$	+6.24	+0.68	+0.0001	+0.0002	+0.0171	+0.0047	+1.70	+0.28	+4.74	+4.14	+3.25	+0.0665	100
L-SMTL$_c$	+4.75	+0.54	**+0.0005**	**+0.0009**	+0.0173	+0.0040	**+2.02**	+0.42	**+5.38**	**+4.62**	**+3.67**	+0.0782	100

Table 5: $\{\alpha_t\}$ learned on four MTL datasets. 'SS' stands for the semantic segmentation task, 'DE' denotes the depth estimation task, 'SNP' is for the surface normal prediction task, 'HPS' corresponds to the human parts segmentation task, 'SE' stands for the saliency estimation task, 'KD' stands for the keypoint detection task, and 'ED' denotes the edge detection task.

Method	CityScapes		NYUv2			PASCAL-Context				Taskonomy				
	SS	DE	SS	DE	SNP	SS	HPS	SE	SNP	SE	DE	KD	ED	SNP
SMTL	0.5002	0.4960	0.4383	0.5188	0.1997	0.4739	0.5529	0.3701	0.2304	0.4886	0.4565	0.4504	0.4578	0.2584
L-SMTL	0.4782	0.4823	0.4475	0.4402	0.3745	0.4982	0.5142	0.4964	0.4277	0.4749	0.4472	0.4079	0.4068	0.4135
SMTL$_c$	0.4896	0.4891	0.3779	0.5320	0.4299	0.4619	0.8729	0.3823	0.4652	0.3988	0.5163	0.4952	0.5208	0.4336
L-SMTL$_c$	0.4972	0.4995	0.5155	0.5242	0.3836	0.5427	0.5826	0.4964	0.3757	0.4878	0.4669	0.3959	0.4102	0.3784

performance, while the proposed models can take the advantages of these two methods to achieve better performance in most cases. Moreover, some of the learned α_t's have relatively small values (i.e., values smaller than 0.3), which are shown in box in Table 5. These small values indicate that for the surface normal prediction task on the NYUv2, PASCAL-Context, and Taskonomy datasets, the public encoder is relatively unimportant, which may imply that the surface normal prediction task is not strongly related to other tasks on these datasets. On the other hand, this observation may explain why DMTL is much worse than STL and why the proposed SMTL method has good performance on these datasets (refer to Tables 2-4).

For the L-SMTL and L-SMTL$_c$ methods, when the learned α_t is larger than 0.5, the corresponding task t will choose to use the public component and otherwise choose the corresponding private component at the inference stage. According to Table 5, we can see that in some cases, all the tasks will choose private components on some datasets (i.e., both methods on the CityScapes and Taskonomy datasets, and L-SMTL on the NYUv2 dataset) and other cases are mixed in that some tasks choose the public component and other tasks choose private components. Interestingly, for both methods, the surface normal prediction task chooses to use private components on the NYUv2, PASCAL-Context, and Taskonomy datasets, which corresponds to the small values for α_t in the SMTL method.

4.4 EXPERIMENTS ON BI-LEVEL FORMULATION

In this section, we use a bi-level formulation for the SMTL model to study the effects of different formulations. Specifically, the original training dataset is divided into two parts, including a training set with n_{tr} data points and a validation set with n_{val} data points. The training set is used to learn parameters in the public encoder, m private encoders, and m private decoders, which corresponds to the lower-level subproblem in the following problem (6). Parameters in the m gates are viewed as hyperparameters and the validation set is used to learn them, which is corresponding to the upper-level subproblem in problem (6). The objective function of SMTL under the bi-level formulation is formulated as

$$\begin{aligned}&\min_{\{g_t\}} \frac{1}{mn_{val}} \sum_{i=1}^{n_{val}} \sum_{t=1}^{m} \mathcal{L}_t(\tilde{\mathbf{y}}_t^i, h_t^*(g_t(f_S^*(\tilde{\mathbf{x}}_i), f_t^*(\tilde{\mathbf{x}}_i))))\\ &\text{s.t. } f_S^*, \{f_t^*\}, \{h_t^*\} = \underset{f_S, \{f_t\}, \{h_t\}}{\arg\min} \frac{1}{mn_{tr}} \sum_{i=1}^{n_{tr}} \sum_{t=1}^{m} \mathcal{L}_t(\bar{\mathbf{y}}_t^i, h_t(g_t(f_S(\bar{\mathbf{x}}_i), f_t(\bar{\mathbf{x}}_i))))),\end{aligned} \tag{6}$$

where $\bar{\mathbf{x}}_i$ denotes the ith data point in the training set, $\bar{\mathbf{y}}_t^i$ denotes the corresponding label of $\bar{\mathbf{x}}_i$ in task t, and $\tilde{\mathbf{x}}_i$ as well as $\tilde{\mathbf{y}}_t^i$ is defined similarly in the validation set. Then we conduct experiments on the CityScapes and NYUv2 datasets to compare the performance of SMTL under different formulations. According to experimental results shown in Table 6, we can see that, on the CityScapes dataset, the learned $\{\alpha_t\}$ by the bi-level formulation (i.e., problem (6)) is similar to that by the single-level formulation (i.e., problem (2)), thus the performance has no much difference. However, on the NYUv2 dataset, SMTL with the bi-level formulation learns a large α_t for the surface normal prediction task, which makes its performance inferior to SMTL with the single-level formulation in some tasks (i.e., depth estimation and surface normal prediction). One reason is that the surface normal prediction task is not so strongly related to other tasks that learning them together may impair not only its own performance but also the performance of other tasks. Moreover, it is well known that the complexity of solving a bi-level optimization problem is much higher than that of solving the corresponding single-level optimization problem and so experiments on larger datasets (i.e., PASCAL-Context and Taskonomy) based on problem (6) are too computational demanding to be conducted. Hence, the single-level formulation of the SMTL model (i.e., problem (2)) is preferred as it is both effective and efficient. For other variants of SMTL, we have similar observations so that we do not report the results for them.

Table 6: Performance and learned α_t of SMTL on the CityScapes and NYUv2 validation datasets, where values in the normal font correspond to the performance of SMTL through the single-level formulation (i.e., problem (2)) and values in the italic font are those through the bi-level formulation (i.e., problem (6)). $\uparrow$ ($\downarrow$) indicates the higher (lower) the result, the better the performance.

Dataset		Segmentation		Depth		Surface Normal				
		mIoU ↑	Pix Acc ↑	Abs Err ↓	Rel Err↓	Angle Distance ↓		Within $t°$↑		
						Mean	Median	11.25	22.5	30
CityScapes	Performace	68.98	91.62	0.0134	42.8621	-	-	-	-	-
		68.98	*91.63*	*0.0136*	*43.4661*	-	-	-	-	-
	Learned α_t	0.5002		0.4960				-		
		0.7019		*0.5327*				-		
NYUv2	Performace	53.27	75.48	0.3886	0.1590	22.00	15.49	38.64	64.89	75.22
		53.40	*75.40*	*0.4088*	*0.1612*	*22.76*	*16.47*	*36.63*	*62.62*	*73.43*
	Learned α_t	0.4383		0.5188				0.1997		
		0.6288		*0.5468*				*0.5825*		

5 CONCLUSION

In this paper, to study the problem of *safe multi-task learning*, we propose a simple and effective SMTL method that can automatically learn to combine encoders via a gating mechanism. To reduce the storage cost, we design lite SMTL by learning a binary gate. Furthermore, we study to place the gates after the decoders. Extensive evaluations demonstrate the effectiveness of the proposed methods. In the future work, we are interested in identifying the location for the gates via neural architecture search.

ETHICS STATEMENT

We have read the Code of Ethics and ensure that this work follows it. No human subject is involved in this work and all the datasets used in experiments are publicly available such that they do not contain any personally identifiable information or offensive content. Hence, this work has no potentially negative societal impact.

REPRODUCIBILITY STATEMENT

For theoretic results, assumptions used have been fully stated and the complete proof is included in the Appendix. We include all the code and instructions in the supplementary material so that this work can be reproduced. The implementation details are provided in Section 4.1 and Appendix.

REFERENCES

Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

R. Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4): 834–848, 2017.

Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3213–3223, 2016.

Chaoran Cui, Zhen Shen, Jin Huang, Meng Chen, Mingliang Xu, Meng Wang, and Yilong Yin. Adaptive feature aggregation in deep multi-task convolutional neural networks. *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.

Theodoros Evgeniou and Massimiliano Pontil. Regularized multi–task learning. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 109–117, 2004.

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.

Emil Julius Gumbel. *Statistical theory of extreme values and some practical applications: a series of lectures*, volume 33. US Government Printing Office, 1948.

Pengsheng Guo, Chen-Yu Lee, and Daniel Ulbricht. Learning to branch for multi-task learning. In *International Conference on Machine Learning*, pp. 3854–3863. PMLR, 2020.

Pengxin Guo, Chang Deng, Linjie Xu, Xiaonan Huang, and Yu Zhang. Deep multi-task augmented feature learning via hierarchical graph neural network. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 538–553. Springer, 2021.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *Proceedings of the 5th International Conference on Learning Representations*, 2017.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Abhishek Kumar and Hal Daume III. Learning task grouping and overlap in multi-task learning. *arXiv preprint arXiv:1206.6417*, 2012.

Michel Ledoux and Michel Talagrand. *Probability in Banach Spaces: isoperimetry and processes*. Springer Science & Business Media, 2013.

Giwoong Lee, Eunho Yang, and Sung Hwang. Asymmetric multi-task learning based on task relatedness and loss. In *International conference on machine learning*, pp. 230–238. PMLR, 2016.

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

Wu Liu, Tao Mei, Yongdong Zhang, Cherry Che, and Jiebo Luo. Multi-task deep visual-semantic embedding for video thumbnail selection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3707–3715, 2015.

Mingsheng Long and Jianmin Wang. Learning multiple tasks with deep relationship networks. *arXiv preprint arXiv:1506.02117*, 2(1), 2015.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *Proceedings of the 5th International Conference on Learning Representations*, 2017.

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.

Andreas Maurer. A chain rule for the expected suprema of gaussian processes. *Theoretical Computer Science*, 650:109–122, 2016.

Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. The benefit of multitask representation learning. *Journal of Machine Learning Research*, 17(81):1–32, 2016.

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.

Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. Latent multi-task architecture learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 4822–4829, 2019.

Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *European conference on computer vision*, pp. 746–760. Springer, 2012.

Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Which tasks should be learned together in multi-task learning? In *International Conference on Machine Learning*, pp. 9120–9132. PMLR, 2020.

Ximeng Sun, Rameswar Panda, Rogério Feris, and Kate Saenko. Adashare: Learning what to share for efficient deep multi-task learning. In *Proceedings of the 33rd Advances in Neural Information Processing Systems*, 2020.

Zirui Wang, Zihang Dai, Barnabás Póczos, and Jaime Carbonell. Characterizing and avoiding negative transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11293–11302, 2019.

Qiang Yang, Yu Zhang, Wenyuan Dai, and Sinno Jialin Pan. *Transfer learning*. Cambridge University Press, 2020.

Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3712–3722, 2018.

Yi Zhang, Yu Zhang, and Wei Wang. Multi-task learning via generalized tensor trace norm. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 2254–2262, 2021.

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

Yu Zhang and Dit-Yan Yeung. A convex formulation for learning task relationships in multi-task learning. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, pp. 733–742, 2010.

Zhanpeng Zhang, Ping Luo, Chen Change Loy, and Xiaoou Tang. Facial landmark detection by deep multi-task learning. In *Proceedings of the 13th European Conference on Computer Vision*, pp. 94–108, 2014.

APPENDIX

A PROOF OF THEOREM 1

In this section, we analyze the generalization bound for the SMTL method.

To analyze the generalization bound for SMTL, we introduce a useful theorem in terms of Gaussian averages (Bartlett & Mendelson, 2002; Maurer et al., 2016).

Theorem 2. *Let $\mathcal{G}$ be a class of functions $\Psi : \mathcal{X} \to [0,1]^T$, and $\mu_1, ..., \mu_m$ be the probability measure on $\mathcal{X}$ with $\bar{\mathbf{X}} = (\mathbf{X}_1, ..., \mathbf{X}_m) \sim \prod_{t=1}^m (\mu_t)^n$ where $\mathbf{X}_t = (\mathbf{x}_t^1, \ldots, \mathbf{x}_t^n)$. Let Z be the random set $\{(\Psi_t(\mathbf{x}_t^i)) : \Psi \in \mathcal{G}\}$ and γ be a vector of independent standard normal variables. Then for all $\Psi \in \mathcal{G}$, with probability at least $1-\delta$ in $\bar{\mathbf{X}}$, we have*

$$\frac{1}{m}\sum_t \left(\mathbb{E}_{\mathbf{x}\sim\mu_t}[\Psi_t(\mathbf{x})] - \frac{1}{n}\sum_i \Psi_t(\mathbf{x}_t^i) \right) \leq \frac{\sqrt{2\pi}G(Z)}{mn} + \sqrt{\frac{9(\ln(2/\delta))}{2mn}},$$

where $G(Z) = \mathbb{E}[\sup_{z\in Z} \langle \gamma, z \rangle]$ is the Gaussian average of the set Z.

Based on Theorem 2, in the following section, we give the proof of Theorem 1.

Proof. By Theorem 2, with probability at least $1-\delta$ in $(\bar{\mathbf{X}}, \bar{\mathbf{Y}}) \sim \prod_{t=1}^m (\mu_t)^n$, for all $h_t \in \mathcal{H}$, $\alpha_t \in \mathcal{M}_t$ and $\varphi \in \mathcal{F}$, we have

$$\mathcal{E} - \frac{1}{mn}\sum_{ti} \mathcal{L}_t(\mathbf{y}_t^i, h_t(\beta_t^T \varphi(\mathbf{x}_t^i))) \leq \frac{\sqrt{2\pi}G(S)}{mn} + \sqrt{\frac{9(\ln(2/\delta))}{2mn}}, \tag{7}$$

where $S = \{(\mathcal{L}_t(y_t^i, h_t(\beta_t^T \varphi(\mathbf{x}_t^i)))) : h_t \in \mathcal{H}, \alpha_t \in \mathcal{M}_t, \varphi \in \mathcal{F}\}$ and $G(S)$ is the Gaussian average of the set S. Then by the Lipschitz property of $\mathcal{L}_t$ and Slepian's Lemma (Ledoux & Talagrand, 2013), we have $G(S) \leq G(S')$, where $S' = \{(h_t(\beta_t^T \varphi(\mathbf{x}_t^i))) : h_t \in \mathcal{H}, \alpha_t \in \mathcal{M}_t, \varphi \in \mathcal{F}\}$.

Note that the input data $\bar{\mathbf{X}} \in \mathcal{X}^{mn}$, and hence $\mathcal{F}(\bar{\mathbf{X}}) \subseteq \mathbb{R}^{mnP(m+1)}$ is defined as $\mathcal{F}(\bar{\mathbf{x}}) = \{(\varphi(\mathbf{x}_t^i)) : \varphi \in \mathcal{F}\}$. We define a class of functions $\mathcal{H}' = \{z \in \mathbb{R}^{mnP(m+1)} \mapsto h_t(\beta_t^T z) : h_t \in \mathcal{H}, \beta_t \in \mathcal{M}_t\}$. Then we have $S' = \mathcal{H}'(\mathcal{F}(\bar{\mathbf{X}}))$. By using Theorem 2 in (Maurer, 2016), we obtain the following inequality

$$G(S') \leq C_1 L(\mathcal{H}') G(\mathcal{F}(\bar{\mathbf{X}})) + C_2 D(\mathcal{F}(\bar{\mathbf{X}})) Q(\mathcal{H}') + \min_{z\in\mathcal{F}(\bar{\mathbf{X}})} G(\mathcal{H}'(z))$$

where C_1 and C_2 are two constants, $L(\mathcal{H}')$ represent the Lipschitz constant of the functions in $\mathcal{H}'$, $D(\mathcal{F}(\bar{\mathbf{x}})) = 2\sup_{\varphi\in\mathcal{F}} \|\varphi(\bar{\mathbf{X}})\|$ denotes the Euclidean diameter of the set $\mathcal{F}(\bar{\mathbf{X}})$, and

$$Q(\mathcal{H}') = \sup_{z\neq\tilde{z}\in\mathbb{R}^{mnP(m+1)}} \frac{1}{\|z-\tilde{z}\|} \mathbb{E} \sup_{\psi\in\mathcal{H}'} \langle \gamma, \psi(z) - \psi(\tilde{z}) \rangle .$$

Let $z, \tilde{z} \in \mathbb{R}^{mnP(m+1)}$, where $z = (z_t^i)$ with $z_t^i \in \mathbb{R}^{P(m+1)}$ and $\tilde{z} = (\tilde{z}_t^i)$ with $\tilde{z}_t^i \in \mathbb{R}^{P(m+1)}$. Then for any functions $\psi \in \mathcal{H}'$, we have

$$\begin{aligned}
\mathbb{E} \sup_{\psi\in\mathcal{H}'} \langle \gamma, \psi(z) - \psi(\tilde{z}) \rangle &= \mathbb{E} \sup_{h_t\in\mathcal{H}, \beta_t\in\mathcal{M}_t} \sum_{ti} \left\langle \gamma_{ti}, h_t(\beta_t^T z_t^i) - h_t(\beta_t^T \tilde{z}_t^i) \right\rangle \\
&= \sum_{t=1}^m \mathbb{E} \sup_{h\in\mathcal{H}, \beta_t\in\mathcal{M}_t} \sum_{i=1}^n \gamma_i (h(\beta_t^T z_t^i) - h(\beta_t^T \tilde{z}_t^i)) \\
&\leq \sqrt{m} \left(\sum_{t=1}^m \left(\mathbb{E} \sup_{h\in\mathcal{H}, \beta_t\in\mathcal{M}_t} \sum_{i=1}^n \gamma_i (h(\beta_t^T z_t^i) - h(\beta_t^T \tilde{z}_t^i)) \right)^2 \right)^{1/2} \\
&\leq \sqrt{m} \left(\sum_{t=1}^m Q_{\max}^2 \sum_{i=1}^n \|z_t^i - \tilde{z}_t^i\|^2 \right)^{1/2} \\
&\leq \sqrt{m} Q_{\max} \|z - \tilde{z}\|,
\end{aligned}$$

where $Q_{\max} = \max_{1\le t\le m} Q_t$. Therefore, $Q(\mathcal{H}') \le \sqrt{m} Q_{\max}$. Moreover, we have

$$\begin{aligned}\|\psi(z) - \psi(\tilde{z})\|^2 &= \sum_{ti} (h_t(\beta_t^T z_t^i) - h_t(\beta_t^T \tilde{z}_t^i)) \\ &\le M^2 \sum_{ti} \|\beta_t^T z_t^i - \beta_t^T \tilde{z}_t^i\|^2 \le M^2 \|\beta_t^T\|^2 \|z - \tilde{z}\|^2\end{aligned}$$

where the first inequality is due to the Lipschitz property and the second inequality is due to the Cauchy-Schwarz inequality. Note that $\|\beta_t\| \le 1$, hence $L(\mathcal{H}') \le M$. Then, we have

$$G(S) \le G(S') \le C_1 M G(\mathcal{F}(\bar{\mathbf{X}})) + 2C_2 \sqrt{m} Q_{\max} \sup_{\varphi\in\mathcal{F}} \|\varphi(\bar{\mathbf{X}})\| + \min_{z\in\mathcal{F}(\bar{\mathbf{X}})} G(\mathcal{H}'(z)). \tag{8}$$

Let $\varphi^*, h_1^*, ..., h_m^*, \beta_1^*, ..., \beta_m^*$ be the minimizer in $\mathcal{E}^*$, then we have

$$\begin{aligned}\mathcal{E} - \mathcal{E}^* = &\left(\mathcal{E} - \frac{1}{mn}\sum_{ti} \mathcal{L}_t(\mathbf{y}_t^i, h_t(\beta_t^T \varphi(\mathbf{x}_t^i)))\right) + \left(\frac{1}{mn}\sum_{ti} \mathcal{L}_t(\mathbf{y}_t^i, h_t^*(\beta_t^{*T} \varphi^*(\mathbf{x}_t^i))) - \mathcal{E}^*\right) \\ &+ \left(\frac{1}{mn}\sum_{ti} \mathcal{L}_t(\mathbf{y}_t^i, h_t(\beta_t^T \varphi(\mathbf{x}_t^i))) - \frac{1}{mn}\sum_{ti} \mathcal{L}_t(\mathbf{y}_t^i, h_t^*(\beta_t^{*T} \varphi^*(\mathbf{x}_t^i)))\right),\end{aligned} \tag{9}$$

where the first term can be bounded by substituting inequality (8) into (7) and the second term can be regarded as mn random variables $\mathcal{L}_t(\mathbf{y}_t^i, h_t^*(\beta_t^{*T} \varphi^*(\mathbf{x}_t^i)))$ with values in $[0, 1]$. By using Hoeffding's inequality, with probability at least $1 - \delta$, we have

$$\frac{1}{mn}\sum_{ti} \mathcal{L}_t(\mathbf{y}_t^i, h_t^*(\beta_t^{*T} \varphi^*(\mathbf{x}_t^i))) - \mathcal{E}^* \le \sqrt{\frac{\ln(1/\delta)}{2mn}}.$$

The last term is non-positive due to the definition of minimizers. Therefore, we have

$$\mathcal{E} - \mathcal{E}^* \le \frac{C_1 M G(\mathcal{F}(\bar{\mathbf{X}})) + \min_{z\in\mathcal{F}(\bar{\mathbf{X}})} G(\mathcal{H}'(z))}{mn} + \frac{2QC_2 \sup_{\varphi\in\mathcal{F}} \|\varphi(\bar{\mathbf{X}})\|}{n\sqrt{m}} + \sqrt{\frac{8\ln(4/\delta)}{mn}}.$$

□

B Necessary Condition for Optimal α_t

We analyze the conditions that model parameters except $\{\alpha_t\}$ satisfy when the optimal α_t equals 0 or 1 for each task. By taking the cross-entropy loss as an example, we have following result.

Theorem 3. *When $\mathcal{L}_t$ is the cross-entropy loss of task t that is formulated as $\mathcal{L}_t = -\frac{1}{n}\sum_{i=1}^n (\mathbf{y}_t^i)^T \log h_t(\alpha_t f_S(\mathbf{x}_i) + (1-\alpha_t) f_t(\mathbf{x}_i))$ where $\mathbf{y}_t^i$ denotes the one-hot label vector, if α_t^* is a local minimum of $\mathcal{L}_t$ over α_t, then we have*

$$\begin{aligned}\sum_{i=1}^n \frac{(\mathbf{y}_t^i)^T f_t(\mathbf{x}_i)(f_S(\mathbf{x}_i) - f_t(\mathbf{x}_i))}{h_t(f_t(\mathbf{x}_i))} \le 0, \quad \textit{if} \quad \alpha_t^* = 0, \\ \sum_{i=1}^n \frac{(\mathbf{y}_t^i)^T f_S(\mathbf{x}_i)(f_S(\mathbf{x}_i) - f_t(\mathbf{x}_i))}{h_t(f_S(\mathbf{x}_i))} \ge 0, \quad \textit{if} \quad \alpha_t^* = 1,\end{aligned}$$

Proof. Fix α_t and define the function $H_t(\epsilon) = \mathcal{L}_t(\alpha_t^* + \epsilon(\alpha_t - \alpha_t^*))$, which is continuously differentiable in an open interval containing $[0, 1]$. By using the chain rule to differentiate H_t, we have

$$0 \le \lim_{\epsilon\to 0} \frac{\mathcal{L}_t(\alpha_t^* + \epsilon(\alpha_t - \alpha_t^*)) - \mathcal{L}_t(\alpha_t^*)}{\epsilon} = \frac{dH_t(0)}{d\epsilon} = \frac{\partial \mathcal{L}_t(\alpha_t^*)}{\partial \alpha_t}(\alpha_t - \alpha_t^*)$$

where the inequality follows from the assumption that α_t^* is a local minimum.

If $\alpha_t^* = 0$, then $\alpha_t - \alpha_t^* \ge 0$. To satisfy $\frac{\partial \mathcal{L}_t(\alpha_t^*)}{\partial \alpha_t}(\alpha_t - \alpha_t^*) \ge 0$, we have $\frac{\partial \mathcal{L}_t(\alpha_t^*)}{\partial \alpha_t} \ge 0$.

If $\alpha_t^* = 1$, then $\alpha_t - \alpha_t^* \le 0$. To satisfy $\frac{\partial \mathcal{L}_t(\alpha_t^*)}{\partial \alpha_t}(\alpha_t - \alpha_t^*) \ge 0$, we have $\frac{\partial \mathcal{L}_t(\alpha_t^*)}{\partial \alpha_t} \le 0$.

Based on the formulation of $\mathcal{L}_t$, we can compute $\frac{\partial \mathcal{L}_t(\alpha_t^*)}{\partial \alpha_t}$ as $\frac{\partial \mathcal{L}_t(\alpha_t^*)}{\partial \alpha_t} = -\frac{1}{n}\sum_{i=1}^{n} \frac{(\mathbf{y}_t^i)^T(\alpha_t^* f_S(\mathbf{x}_i)+(1-\alpha_t^*)f_t(\mathbf{x}_i))(f_S(\mathbf{x}_i)-f_t(\mathbf{x}_i))}{h_t(\alpha_t^* f_S(\mathbf{x}_i)+(1-\alpha_t^*)(f_t(\mathbf{x}_i)))}$.

Then we have

$$\sum_{i=1}^{n} \frac{(\mathbf{y}_t^i)^T f_t(\mathbf{x}_i)(f_S(\mathbf{x}_i) - f_t(\mathbf{x}_i))}{h_t(f_t(\mathbf{x}_i))} \leq 0, \quad \text{if} \quad \alpha_t^* = 0,$$
$$\sum_{i=1}^{n} \frac{(\mathbf{y}_t^i)^T f_S(\mathbf{x}_i)(f_S(\mathbf{x}_i) - f_t(\mathbf{x}_i))}{h_t(f_S(\mathbf{x}_i))} \geq 0, \quad \text{if} \quad \alpha_t^* = 1,$$

in which we reach the conclusion. □

Theorem 3 provide a necessary condition for the optimal α_t being 0 or 1. Such analysis can easily be extended to other loss functions such as the square loss, and we omit it due to similar proofs.

C Illustration of the SMTL$_c$ and L-SMTL$_c$ Models

Similar to the SMTL and L-SMTL models, an illustration of the SMTL$_c$ and L-SMTL$_c$ models is shown in Figure 2.

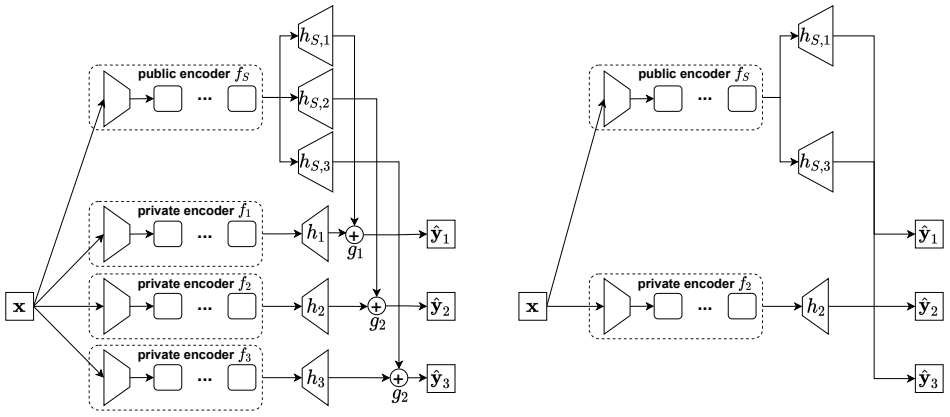

Figure 2: Illustration of the SMTL$_c$ and L-SMTL$_c$ models. **Left figure**: Pipeline for the SMTL$_c$ model, which is identical to the training phase of L-SMTL$_c$. For task t, $\mathbf{x}$ is first fed into the public encoder f_S and the public decoder $h_{S,t}$ to get an output o_S, and it is also fed into the private encoder f_t and the private decoder h_t to get another output o_t. Then it is through the gate to obtain the combined output, i.e., $\hat{\mathbf{y}}_t = \alpha_t o_S + (1-\alpha_t)o_t$. The number of tasks is set to three for illustration. **Right figure**: Test phase for L-SMTL$_c$. After finishing the training process of L-SMTL$_c$, g_t can choose which component (i.e., the public component f_S and $h_{S,t}$ or the private component f_t and h_t) is used for each task. In this way, at the test process, only the chosen components need to be saved, which could reduce the number of parameters and speedup the inference. In this illustration, task 1 and task 3 choose the public component, while task 2 goes through the private component.

D Experimental Setup

D.1 Details of Datasets

CityScapes. The CityScapes dataset (Cordts et al., 2016) consists of high resolution outside street-view images, which contains $2,975$ images for training and 500 images for validation. This dataset contains 19 classes for pixel-wise semantic segmentation, together with ground-truth inverse depth labels. By following (Liu et al., 2019), we evaluate the performance on the 7-class semantic segmentation and depth estimation tasks. All training and validation images are resized to 128×256.

NYUv2. The NYUv2 dataset (Silberman et al., 2012) consisting of RGB-D indoor scene images has 795 images for training and 654 images for validation. We evaluate the performance on three learning tasks: 13-class semantic segmentation, depth estimation, and surface normal prediction. By following (Liu et al., 2019), all the training and validation images were resized to 288×384.

PASCAL-Context. The PASCAL-Context dataset (Mottaghi et al., 2014) is an annotation extension of the PASCAL VOC 2010 challenge and it contains $4,998$ images for training and $5,105$ images for validation. We evaluate the performance on four learning tasks: 21-class semantic segmentation, 7-class human parts segmentation, saliency estimation, and surface normal estimation, where the last two tasks are generated by (Maninis et al., 2019).

Taskonomy. The Taskonomy dataset (Zamir et al., 2018) which contains over 4.5 million indoor images from over $5,000$ buildings with 26 tasks. By following (Standley et al., 2020), we sample five learning tasks, including 17-class semantic segmentation, depth estimation, keypoint detection, edge detection, and surface normal prediction. Furthermore, we select 5 building images (i.e., "allensville", "collierville", "mifflinburg", "noxapater", and "onaga") from the standard tiny benchmark as our dataset, which contains $13,286$ images for training and $3,794$ images for validation.

D.2 Settings of Batch Size

The settings of the batch size for all the models on different datasets are shown in Table 7.

Table 7: Settings of batch size for all the models on the four datasets.

Dataset	STL	DMTL	Cross-stitch	MTAN	NDDR-CNN	AdaShare	AFA	SMTL	L-SMTL	SMTL$_c$	L-SMTL$_c$
CityScapes	180	180	100	80	80	120	150	70	70	70	70
NYUv2	8	4	4	4	4	4	4	4	4	4	4
PASCAL-Context	40	40	24	20	18	32	8	18	18	15	15
Taskonomy	230	230	120	130	100	180	40	100	100	90	90

D.3 Evaluation Metrics

On the PASCAL-Context dataset, by following (Maninis et al., 2019), the semantic segmentation is evaluated by the mean Intersection over Union (mIoU) and on the other three datasets, by following (Sun et al., 2020), this task is additionally evaluated by the Pixel Accuracy (Pix Acc). For the depth estimation task, the absolute error (Abs Err) and relative error (Rel Err) are used as the evaluation metrics. For the surface normal prediction task, the mean and median angle distances between the prediction and ground truth of all pixels are used as measures. For this task, the percentage of pixels, whose prediction is within the angles of 11.25°, 22.5°, and $30^\circ$to the ground truth, is used as another measure. For the keypoint detection and edge detection tasks, the absolute error (Abs Err) is used as the evaluation metric. For the human parts segmentation task, the mIoU is used as the measure. For the saliency estimation task, the mIoU and max F-measure (maxF) are adopted as the evaluation metrics.

E Comparison of Training Time

The training time per epoch for all the models on the CityScapes dataset is recorded in Table 8. According to the results, the training time of the proposed SMTL and L-SMTL methods is comparable with all the baseline methods, implying that the proposed SMTL and L-SMTL methods are as efficient as baseline methods. The training time of the proposed SMTL$_c$ and L-SMTL$_c$ methods is a bit longer due to additional public decoders introduced.

Table 8: Training time per epoch for all the models on the CityScapes dataset.

	DMTL	Cross-stitch	MTAN	NDDR-CNN	AdaShare	AFA	SMTL	L-SMTL	SMTL$_c$	L-SMTL$_c$
Time (s)	122	150	153	175	141	58	175	175	272	272