Cross-Task Consistency Learning Framework for Multi-Task Learning

Akihiro Nakano, Shi Chen, and Kazuyuki Demachi

***Abstract*—Multi-task learning (MTL) is an active field in deep learning in which we train a model to jointly learn multiple tasks by exploiting relationships between the tasks. It has been shown that MTL helps the model share the learned features between tasks and enhance predictions compared to when learning each task independently. We propose a new learning framework for 2-task MTL problem that uses the predictions of one task as inputs to another network to predict the other task. We define two new loss terms inspired by cycle-consistency loss and contrastive learning, *alignment loss* and *cross-task consistency loss*. Both losses are designed to enforce the model to align the predictions of multiple tasks so that the model predicts consistently. We theoretically prove that both losses help the model learn more efficiently and that cross-task consistency loss is better in terms of alignment with the straight-forward predictions. Experimental results also show that our proposed model achieves significant performance on the benchmark Cityscapes and NYU dataset.**

***Index Terms*—Consistency learning, deep neural network, multitask learning (MTL), task relationship.**

I. Introduction

DEEP learning has made significant progress in the last decade, improving and enhancing its ability to classify, predict, and understand inputs from various modals. Although its performance are now excelling the skills of humans in certain domains, one disadvantage of deep learning models is that its application is limited to single-task problems. While humans are capable of handling multiple tasks simultaneously, common deep learning models require different models to be trained for each task. Therefore, recent works have focused on developing a multi-task learning model, a model optimized for multiple tasks.

Multi-Task Learning (MTL) is a method to train a model to learn multiple tasks jointly. Humans are able to process multiple tasks at the same time and combine the information for a more semantic task. For example, when we look at an image, we are able to recognize objects, estimate depth, infer the scene, and predict what will happen next. One reason we can process and integrate multiple sensory tasks is because they are closely related to each other. MTL aims to exploit this "relationship" in a way such that machines can utilize it. In most MTL frameworks, features are divided into task-common and task-specific features. This method of exploiting task relationships by partially sharing the learned features has effectively improved performance. Another merit of MTL is the reduction of parameters. Compared to normal single task learning (STL), MTL requires fewer parameters because part of the model is shared between the tasks.

Previous works have experimented numerous patterns of network architecture. Some works have proposed a method to explicitly enforce the model to capture relationships between tasks using additional units or matrices [1]–[3]. Other works have simply divided the features into task-common features and task-specific features so that it learns the relationships implicitly and focused on balancing the losses between multiple tasks [4]–[8]. Some works in computer vision field have utilized the multiple views of a single scene in a self-supervised fashion [9], [10].

In this work, we define and propose a new framework called *cross-task consistency learning framework* for 2-task MTL problem inspired by CycleGAN's [11] cycle-consistency loss and contrastive learning using multi-views [12], [13]. We introduce two loss terms, *alignment loss* and *cross-task consistency loss*, aimed to maintain consistency between the predictions. We then prove an inequality relationship between alignment loss and cross-task consistency loss and show that not only cross-task consistency loss is superior to alignment loss in approximating the straight-forward predictions but also that both terms are upper-bounded by a small value.

Specifically, we demonstrate our method in learning semantic segmentation and depth estimation task in the computer vision field. We use these two tasks because (i) scene segmentation and depth perception are related to each other [14], and (ii) the encoder-decoder architecture which has shown significant performance for both tasks is suited to capture task-common and task-specific features. Our architecture, which we name as XTasC-Net (Cross-Task Consistency Network), consists of two modules: first, the input image is feeded to an encoder-decoder architecture to produce predictions of the two tasks (direct predictions). Then, we prepare two separate encoder-decoder architecture networks (Task-Transfer Networks, TTNets) that takes in the direct predictions as inputs and outputs the prediction for the other task (task-transferred predictions). The proposed alignment loss and cross-task consistency loss utilizes the task-transferred predictions so that some of the information from the other task is learned through the scope of the TTNets. The experimental results on Cityscapes [15] and NYU [16] dataset shows that our model shows competitive performance with less number of parameters.

A. Nakano is with Department of Systems Innovation, Faculty of Engineering, The University of Tokyo, Tokyo, Japan (e-mail: nakano.akihiro@weblab.t.u-tokyo.ac.jp).

S. Chen, and K.Demachi are with Department of Nuclear Engineering and Management, School of Engineering, The University of Tokyo, Tokyo, Japan (e-mail: shichen@g.ecc.u-tokyo.ac.jp; demachi@n.t.u-tokyo.ac.jp)

In summary, we make mainly two contributions. First, we present a principled multi-task learning framework for 2-task MTL problem by relating to cycle-consistency and contrastive learning. We theoretically derive our loss term using implicit latent variable models (LVMs) and prove an inequality relationship regarding the proposed two loss terms. Secondly, we propose a new architecture called XTasC-Net for semantic segmentation and depth estimation task.

II. Related Work

MTL has been one of the key approaches in improving generalization and learning efficiency by using information of *related* tasks [17], [18]. A common challenge of MTL is the formulation of task relationships. Some works have approached using regularization methods [19] or by defining a convex optimization problem to estimate task relationships [20]–[22]. Zamir *et al.* [23] built a taxonomical structure between multiple vision-related tasks. Furthermore, Zhou *et al.* [24] used adversarial networks to learn the task relationships.

In the computer vision field, some works explored modeling task relationships explicitly in their model architecture. For example, Long *et al.* [1] developed Deep Relationship Networks which places a matrix prior between the fully-connected layers of each task so that the model learns the task relationship. Misra *et al.* [2] proposed Cross-stitch Networks which uses cross-stitch units, a module that learns how much sharing is needed between each respective layer of the networks. Yang *et al.* [3] used min-max optimization to make the model learn both task-specific and task-common features.

On the contrary, other works have experimented enforcing the model to learn task relationships implicitly. There have been mainly three approaches: using knowledge distillation, balancing multi-task loss function, and utilizing stereo views of the data.

One approach has been to use knowledge distillation [25] by preparing two phases of training to enhance performance [26]–[30]. These models are trained on several tasks jointly in the first phase. Then, using the trained weights from the first phase, the features are combined through another network to distill its information to make the final predictions. For example, Xu *et al.* [26] proposed PAD-Net which experiments concatenation of the features or applying an attention map when passing features of other tasks for a certain task. Zhang *et al.* [28] and Vandenhende *et al.* [30] utilized self-attention modules in each task's network. Li et al.'s [29] model used knowledge distillation by adding a loss term between the pretrained STL network and MTL model's weights.

Another approach is to train a single model by balancing the loss functions of each task [4]–[8], [31]–[33]. Kendall *et al.* [6] proposed uncertainty weights, a quantity that can be seen as the relative confidence between tasks. Other methods [5], [7] have used loss magnitude to leverage the losses. On the other hand, Liu *et al.* [4] introduced DWA (dynamic weight averaging) which uses relative loss reduction and Yu *et al.* [8] implemented PCGrad, an algorithm that fixes contradicting gradients in the shared architecture.

Finally, some works for MTL including depth estimation task has utilized the multiple views of the data [9], [10]. Using the advantage that some dataset were captured using a stereo camera, these methods have applied the photometric reconstruction loss originally proposed by MonoDepth [34], [35].

III. Method

We use the following notations: a common network architecture in recent works consists of two modules, a shared network $f_{W_E} : \mathcal{X} \to \tilde{\mathcal{X}}$ followed by T individual networks $f_{W_t} : \tilde{\mathcal{X}} \to \mathbb{R}^{d_t}$ for each task where W_E, W_t are the weights of each networks for $t \in [T]$. Since the number of tasks is limited to $T = 2$, we refer the two tasks as y and z. Further, for simplicity, we regard the $\tilde{x} = f_{W_E}(x)$ as our input and denote this as x.

Our key assumption is that there exists some mappings $\mathcal{F}_\theta : \mathcal{Z} \to \mathcal{Y}$ and $\mathcal{G}_\phi : \mathcal{Y} \to \mathcal{Z}$ that describes the likelihood between the tasks,

$$\begin{aligned} y \sim p_\theta(y|z) \Leftrightarrow y = \mathcal{F}_\theta(\varepsilon; z), \qquad \varepsilon \sim p(\varepsilon) \\ z \sim p_\phi(z|y) \Leftrightarrow z = \mathcal{G}_\phi(\varepsilon'; y), \qquad \varepsilon' \sim p(\varepsilon') \end{aligned} \tag{1}$$

where $\varepsilon, \varepsilon'$ are noise.

Since y and z are both outputs from a neural network given an input x, we can further denote this using f_{W_t} as,

$$\begin{aligned} y = \mathcal{F}_\theta(\varepsilon; f_{W_2}(x)) \\ z = \mathcal{G}_\phi(\varepsilon'; f_{W_1}(x)) \end{aligned} \tag{2}$$

Using this notation, we propose *cross-task consistency loss*, which is defined as,

$$\begin{aligned} \ell^{\text{XTC}}_{2\to1} = \|\mathcal{F}_\theta(f_{W_2}(x)) - f_{W_1}(x)\|_2^2 \\ \ell^{\text{XTC}}_{1\to2} = \|\mathcal{G}_\phi(f_{W_1}(x)) - f_{W_2}(x)\|_2^2 \end{aligned} \tag{3}$$

where $\|\cdot\|_2$ indicates l_2 norm. The loss takes the difference between the outputs of task-specific networks, $f_{W_1}(x), f_{W_2}(x)$, and those outputs that are transferred to predict the other task, $\mathcal{G}_\phi(f_{W_1}(x)), \mathcal{F}_\theta(f_{W_2}(x))$. Below, we refer the former as *direct predictions* and the latter as *task-transferred predictions*.

The intuition of cross-task consistency loss is to pass partial information of task-specific features to other tasks using the assumed mappings, $\mathcal{F}_\theta, \mathcal{G}_\phi$. Although the shared network helps the model learn these task-common features, we expect some of the task-specific features learned in each task-specific network can also be utilized for the other task.

In the following sections, we derive our proposed loss term from [13], [36]. First, in section III-A, we derive a similar loss term which we name as alignment loss from Tiao *et al.*'s [36] proof of CycleGAN based on implicit latent variable models (LVMs). Then, in section III-B, we will prove that cross-task consistency loss is better than alignment loss based on Tosh et al.'s proof [13].

A. Alignment Loss

Fig. 1 describes the diagrams of CycleGAN and our cross-task consistency learning framework. Similar to how CycleGAN has a bidirectional mapping between the input y (real images) and the output z (generated images), our framework also a bidirectional mapping between the two tasks' outputs.

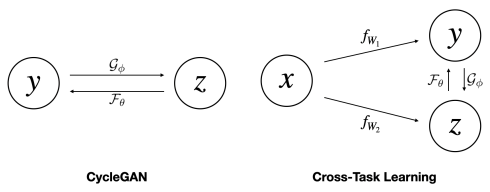

Fig. 1. Diagrams of CycleGAN and our cross-task consistency learning framework.

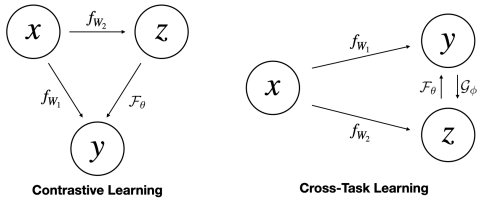

Fig. 2. Diagrams of contrastive learning using multi-views and our cross-task consistency learning framework.

Using LVMs, we can describe the joint probability of observing y and z, the two tasks, as the product of the prior distribution and the likelihood. Since we model the likelihoods using (1), we can express the joint probability as,

$$\begin{aligned} p_\theta(y,z) &= p_\theta(y|z)p(z) \\ q_\phi(y,z) &= q_\phi(z|y)q(y) \end{aligned} \tag{4}$$

where $p(\cdot), q(\cdot)$ denotes probability. We further use Tial *et al.*'s implicit LVMs, in which we replace the prior distributions with implicit distributions $p^*(u)$. Implicit distributions are distributions given by a finite collection of data, $U^* = \{u_i^*\}_{i=1}^N$, i.e. $u_i^* \sim p^*(u)$, for $u \in \{y, z\}$. Therefore, (4) can be written as,

$$\begin{aligned} p_\theta(y,z) &= p_\theta(y|z)p^*(z) \propto p_\theta(y|z)p^*(x) \\ q_\phi(y,z) &= q_\phi(z|y)q^*(y) \propto q_\phi(z|y)q^*(x) \end{aligned} \tag{5}$$

since both y and z are conditioned on input x.

Now, regardless of using parameters θ or ϕ, the joint distribution should be equivalent. Therefore, we consider minimizing the statistical distance between $p_\theta(y,z)$ and $q_\phi(y,z)$ using *symmetric* KL divergence, $\mathrm{KL}_{\mathrm{SYMM}}[p_\theta(y,z)\|q_\phi(y,z)]$, where

$$\mathrm{KL}_{\mathrm{SYMM}}[p\|q] = \mathrm{KL}[p\|q] + \mathrm{KL}[q\|p] \tag{6}$$

Then, since minimizing the KL divergence is equivalent to maximizing the likelihood in (5), we can derive the loss as follows;

Assume Gaussian noise, i.e., $\varepsilon \sim \mathcal{N}(0, \sigma_1^2 I)$, $\varepsilon' \sim \mathcal{N}(0, \sigma_2^2 I)$. Then,

$$\begin{aligned} y &\sim \mathcal{N}(\mathcal{F}_\theta(z), \sigma_1^2 I) \\ z &\sim \mathcal{N}(\mathcal{G}_\phi(y), \sigma_2^2 I) \end{aligned}$$

Therefore using maximum likelihood estimation,

$$\begin{aligned} \max p_\theta(y,z) \iff & \min \mathbb{E}_{p^*(z)p_\theta(y|z)}[-\log p_\theta(y|z)] \\ \therefore \quad & \mathbb{E}_{p^*(z)p_\theta(y|z)}[-\log p_\theta(y|z)] \\ &= \frac{1}{2\sigma_1^2}\mathbb{E}_{p^*(z)p_\theta(y|z)}[\|y - \mathcal{F}_\theta(z)\|_2^2] \\ &\quad + \frac{D}{2}\log 2\pi\sigma_1^2 \\ &= \gamma_1 \mathbb{E}_{p^*(z)p_\theta(y|z)}[\|y - \mathcal{F}_\theta(z)\|_2^2] + \delta_1 \\ &\propto \mathbb{E}_{p^*(z)p_\theta(y|z)}[\|y - \mathcal{F}_\theta(z)\|_2^2] \\ &\propto \mathbb{E}_{p^*(x)p_\theta(y|z)}[\|y - \mathcal{F}_\theta(f_{W_2}(x))\|_2^2] \end{aligned} \tag{7}$$

where $\gamma_1 = \frac{1}{2\sigma_1^2}$ and $\delta_1 = \frac{D}{2}\log\frac{\pi}{\gamma_1}$. A similar thing can be said for maximizing $q_\phi(y,z)$. Hence,

$$\begin{aligned} \ell_{2\to1}^{\mathrm{ALIGN}} &= \|y - \mathcal{F}_\theta(f_{W_2}(x))\|_2^2 \\ \ell_{1\to2}^{\mathrm{ALIGN}} &= \|z - \mathcal{G}_\phi(f_{W_1}(x))\|_2^2 \end{aligned} \tag{8}$$

are the *alignment losses*. However, we introduce another loss, cross-task consistency loss, which is defined as (3), by replacing the first term of each equation with the direct predictions. In the following section, we explain the merits of using cross-task consistency loss over alignment loss.

B. Cross-Task Consistency Loss

Our learning framework can also be related with contrastive learning using multi-views (Fig. 2). Methods such as SimCLR [12] learns in a self-supervised fashion by creating additional inputs that are correlated with the original input. Whereas contrastive learning uses the redundancy between the inputs x and the generated images z, our framework has redundancy between the two tasks.

Below, we use the following notations to express direct predictors, alignment (ALIGN) predictors, and cross-task consistency (XTC) predictors,

$$\begin{aligned} \text{direct}: \ & x \mapsto \mathbb{E}[Y|X=x] \\ \text{ALIGN}: \ & x \mapsto \mathbb{E}[Y|\mathbb{E}[Z|X=x]] \\ \text{XTC}: \ & x \mapsto \mathbb{E}[\mathbb{E}[Y|X=x]|\mathbb{E}[Z|X=x]] \end{aligned} \tag{9}$$

Furthermore, without proof, we can state that the quantity,

$$\xi_Y = \mathbb{E}[(Y - \mathbb{E}[Y|X])^2] \tag{10}$$

is small.

By assuming that there exists some relationship between y and z, predicting y by first predicting z from x should, intuitively, achieve similar predictions as directly predicting y from x.

The following proposition tells us that not only does this strategy work but also using the direct predictions as the target works better.

Proposition. *Let X, Y, Z be random variables. Then, cross-task self-consistency loss yields smaller expected error between direct loss compared to cross-task consistency loss, i.e.,*

$$\begin{aligned} 0 &= \mathbb{E}[(\mathbb{E}[\mathbb{E}[Y|X]|\mathbb{E}[Z|X]] - \mathbb{E}[Y|X])^2] \\ &\le \mathbb{E}[(\mathbb{E}[Y|\mathbb{E}[Z|X]] - \mathbb{E}[Y|X])^2] \\ &\le \xi_Y \end{aligned}$$

This holds true for case where Y and Z are switched.

See Appendix A for proof.

The proposition implies (1) cross-task consistency loss yields a smaller expected difference between direct loss than alignment loss, and (2) both losses are upper-bounded by some small value. Our proposed loss can be seen as learning one task through the scope of the other task. As the model is composed

of a shared network and task-specific networks, cross-task consistency loss forces the model to efficiently pass the information of one task to the other and hold consistency between its predictions. Furthermore, both cross-task consistency loss and alignment loss are upper-bounded by ξ_Y, which tells us that cross-task terms are competitive with direct predictions.

IV. XTasC-Net

A. Model Architecture

Based on our cross-task consistency learning framework, we propose an original neural network model named *XTasC-Net* (Cross-Task Consistency Network) to conduct the experiments. Fig. 3 shows our model which is built from 3 modules.

XTasC-Net is an encoder-decoder architecture with separate, individual decoders for each task. The encoder and decoder module is connected also with skip-connections, similar to U-Net [37]. The output of each decoder, the direct predictions, are then fed into separate networks, which we refer as *TTNet* (Task-Transfer Networks), similar to [38]. The outputs of TTNet are the task-transferred predictions.

We choose ResNet [39], a commonly used network for image encoding, as the backbone of the encoder. There are 5 types of ResNet, namely ResNet18, ResNet34, ResNet50, ResNet101, ResNet152. After comparing results the results, we decided to use ResNet34 which resulted in best performance with the least number of parameters (see Appendix B).

The decoder consists of 5 blocks and 1 convolutional layer. Based on U-Net, each decoder block takes in the concatenation of feature maps of the previous block and the respective encoder block. It then processes through 3 layers of convolutional network with kernel size 3×3 followed by batch normalization and ReLU activation function. The number of channels are all set to 128. At the end of each block, the feature maps are upsampled by 2. The output of the 5th block is then fed into a 1×1 convolutional layer so that the output results in the required dimensions for a given task. For segmentation task, the number of classes to classify is the dimension, while for depth estimation, the dimension is 1.

TTNets are also designed similarly to U-Net with 3 blocks of contracting path and 3 blocks of expansive path. In the contracting path, each block consists of 2 repeating applications of 3×3 convolutional layer followed by ReLU activation function. At the end of each block, the feature maps are downsampled by a 2×2 max-pooling layer with stride 2. The number of channels are, in order, 64, 128, 256. On the other side, the expansive path is built symmetrically with 2 convolutional layers with the same kernel size followed by ReLU activation layer. At the end of each block, the feature maps are upsampled by 2. The output of the final block is then fed into a 1×1 convolutional layer similar to the decoder.

B. Loss Functions

As mentioned in section III, our proposed method is to add the cross-task consistency loss to the loss function. Below, we denote y_1, y_2 as the target value/class we wish to predict, $\hat{y}_1, \hat{y}_2$ as the direct predictions, and $\hat{y}_{2\to1}, \hat{y}_{1\to2}$ as the task-transferred predictions for task 1 and 2 respectively. We prepare 2 types of loss functions: ℓ_1, ℓ_2 are the losses for direct predictions and $\ell_{2\to1}, \ell_{1\to2}$ are the cross-task consistency losses. Using these notations, the loss of task t, $\mathcal{L}_t$, can be denoted as the weighted average of the 2 losses,

$$\mathcal{L}_t = ((1-\lambda_t)\ell_t(\hat{y}_t, y_t) + \lambda_t \ell_{t\to s}(\hat{y}_{t\to s}, \hat{y}_t)) \tag{11}$$

where $\lambda_t \in [0, 1]$ is the weight and s is the other task.

Let task 1 be the segmentation task and task 2 be the depth estimation task.

Following the example of numerous previous works, we use cross-entropy loss as the loss function for the segmentation task. Given an $H \times W$ output, the cross entropy loss for direct predictions is expressed as,

$$\ell_1(\hat{y}_1, y_1) = -\frac{1}{HW}\sum_{i=0}^{H-1}\sum_{j=0}^{W-1} y_1(i,j)\log \hat{y}_1(i,j) \tag{12}$$

For task-transferred predictions, the loss is expressed as,

$$\ell_{2\to1}(\hat{y}_{2\to1}, \hat{y}_1) = -\frac{1}{HW}\sum_{i=0}^{H-1}\sum_{j=0}^{W-1} \hat{y}_1(i,j)\log \hat{y}_{2\to1}(i,j) \tag{13}$$

We do not backpropagate the loss via direct prediction for cross-task consistency loss because the target we are trying to minimize against is the direct prediction, the output of the other task's decoder.

For depth estimation task, we use l_1 norm distance as the loss function. Given an $H \times W$ output, the depth loss for direct predictions is written as,

$$\ell_2(\hat{y}_2, y_2) = \frac{1}{HW}\sum_{i=0}^{H-1}\sum_{j=0}^{W-1} |\hat{y}_2(i,j) - y_2(i,j)| \tag{14}$$

for all pixels with valid depth values. During training, we mask the invalid pixels so that its loss is not backpropagated through the network. We also use l_1 norm for task-transferred predictions as well.

$$\ell_{1\to2}(\hat{y}_{1\to2}, \hat{y}_2) = \frac{1}{HW}\sum_{i=0}^{H-1}\sum_{j=0}^{W-1} |\hat{y}_{1\to2}(i,j) - \hat{y}_2(i,j)| \tag{15}$$

Similar to task-transferred prediction of segmentation task, the loss is not backpropagated via the direct prediction.

Overall, the loss $\mathcal{L}_{\text{TOTAL}}$ we wish to minimize is the weighted sum of the 2 tasks,

$$\mathcal{L}_{\text{TOTAL}} = \sum_{t=1}^{2} \omega_t \mathcal{L}_t \tag{16}$$

where ω_1, ω_2 are the weights. For ω_1, ω_2, we experiment using equal weights, uncertainty weights [6], and GradNorm [7].

V. Experiments

A. Datasets

We consider 2 datasets, Cityscapes [15] and NYU [16] dataset to validate our proposed method.

Cityscapes dataset is a collection of diverse urban street scenes gathered using a stereo camera. It is provided with 19-class and 7-class segmentation labels, and we use the 7-class

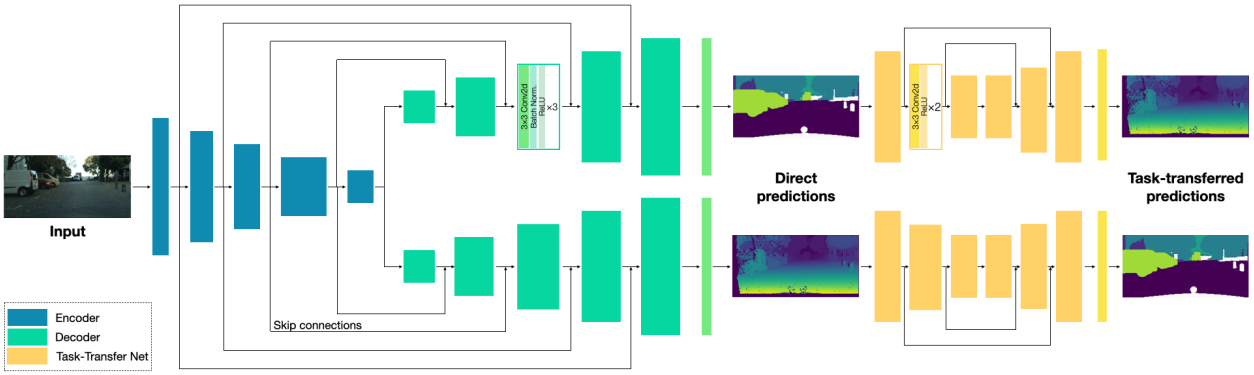

Fig. 3. Architecture overview of XTasC-Net.

version so that it can be compared with previous works. The images' resolution is 1024×2048 but we resize to 128×256 to speed up training.

NYU dataset is a dataset composed of a wide variety of indoor scenes recorded by an RGB camera and depth cameras using Microsoft Kinect with 13-class segmentation labels. The resolution of the images are 480×640 but we resize to 288×384 to speed up training.

For both datasets, we apply normalization, random horizontal flipped with a probability of 0.5, and random scaled cropping with scales chosen randomly from $[1.0, 1.2, 1.5]$. We mask out invalid pixels during training, such as void classes of segmentation task and pixels with depth 0 (incorrectly calculated depth) for depth estimation task. Following previous works, for depth of Cityscapes, we use the inverse disparity values as the target because the raw disparity values range from 0 to infinity (ex. sky) and training to predict such infinite values lead to poor generalization.

B. Evaluation Metrics

We use the following metrics to evaluate the performance. For segmentation task, we use mean Intersection-over-Union (mIoU) and pixel accuracy (Pix. Acc.). For depth estimation task, we use absolute error (Abs. Err.) and absolute relative error (Rel. Err.).

Furthermore, following [30], [40], we compute performance improvement Δ_m of a model m against the baseline model b as the average percentage points' gain of $|M|$ evaluation metrics,

$$\Delta_m = \frac{1}{|M|} \sum_{i \in [M]} (-1)^{l_i} \frac{M_{m,i} - M_{b,i}}{M_{b,i}} \tag{17}$$

where $l_i = 1$ if a lower value means better performance for measure M_i of task i and 0 otherwise.

C. Training Protocols

For all datasets, we use Adam optimizer with an initial learning rate of 0.0001. For Cityscapes dataset, the learning rate is halved every 80 epochs during training. We train the model for 250 epochs with batch size 8. For NYU dataset, we halve the learning rate every 60 epochs during training. We train the model for 100 epochs with batch size 6. We choose λ_1, λ_2 as 0.01 for Cityscapes and 0.0001 for NYU dataset.

We conduct all experiments using Pytorch. Our codes are available at https://github.com/akarimoon/xtask_mt.

D. Baseline Models

We compare the result of our XTasC-Net with 2 models using the same architecture but without TTNet (by setting $\lambda_1 = \lambda_2 = 0$), one that learns each task independently and the other that learns jointly. We refer to these models as Base ST-Net and Base MT-Net, respectively.

E. Results

1) Results on Cityscapes dataset: First, we compare our results against the 2 baseline models with several weighting methods in Table I.

Compared to Base ST-Net with equal weights, we observe that XTasC-Net with uncertainty weights or GradNorm leads to improvement in all four evaluation metrics. Between the two weighting methods, we find that the model with uncertainty weighting excels especially for depth estimation task. Furthermore, under uncertainty weighting regime, XTasC-Net has better overall scores than Base MT-Net (performance: 1.44 $\rightarrow$ 1.58).

Overall, we can observe that our XTasC-Net succeeds in achieving higher results compared to Base MT-Net and Base ST-Net. These results show that task-transferred predictions do not interfere with direct predictions but exploit the task relationships to improve both tasks' predictions.

Table II shows the results for all methods.

We show the number of parameters in the table to show that our XTasC-Net achieves the best result with fewer parameters. The results show that our model outperforms previous works on most evaluation metrics with a sufficiently fewer amount of parameters.

For segmentation task, we observe great improvement on mIoU metric. We think that this is due to the difference of using attention modules or not. Previous works such as [4], [5], [8], [29] have used attention modules in their networks, enabling the model to "look" at the entire image in the training phase. On the other hand, since our model only uses convolutional layers, the model can only learn from pixels nearby. Intuitively, this leads to higher mIoU while attention modules improve pixel accuracy.

TABLE I
ABLATION RESULTS OF DIFFERENT MODELS AND WEIGHTING METHODS ON CITYSCAPES VALIDATION SET FOR 7-CLASS SEMANTIC SEGMENTATION AND DEPTH ESTIMATION TASK. BEST RESULTS ARE IN **BOLD**.

Model	Weighting	Segmentation (Higher Better)		Depth (Lower Better)		Performance (Higher Better)
		mIoU	Pix Acc	Abs Err	Rel Err	Δ_m
Base ST-Net	Equal weights	66.40	93.48	0.0124	19.78	0.00
Base MT-Net	Equal weights	65.86	93.25	0.0129	20.47	-1.50
	Uncert. weights [6]	**66.84**	**93.57**	0.0122	19.61	1.44
XTasC-Net	Equal weights	66.32	93.51	0.0126	21.19	-1.55
	Uncert. weights [6]	66.51	93.56	**0.0122**	**19.40**	**1.58**
	GradNorm [7]	66.48	93.52	0.0124	19.58	0.93

TABLE II
RESULTS OF MULTI-TASK LEARNING ON CITYSCAPES VALIDATION SET FOR 7-CLASS SEMANTIC SEGMENTATION AND DEPTH ESTIMATION TASK. #P SHOWS THE NUMBER OF PARAMETERS OF THE MODEL. *Italic* REPRESENTS ESTIMATED VALUES. BEST RESULTS ARE IN **BOLD** AND SECOND BEST ARE UNDERLINED. * EQUAL WEIGHTS. † UNCERTAINTY WEIGHTS. ‡ GRADIENT-BASED WEIGHT LEARNING.

Model	#P. $[\times 10^7]$	Segmentation (Higher Better)		Depth (Lower Better)	
		mIoU	Pix Acc	Abs Err	Rel Err
STAN [4]	12.52	51.90	90.87	0.0145	27.46
Dense† [41]	14.96	51.89	91.22	0.0134	25.36
Cross-Stitch† [2]	≈ 8.24	50.31	90.43	0.0152	31.36
MTAN* [4]	4.12	53.04	91.11	0.0144	33.63
PCGrad* [8]	4.12	53.59	91.45	0.0171	31.34
KD4MTL* [29]	4.12	52.71	91.54	0.0139	27.33
AdaMT-Net‡ [5]	*4.91*	62.53	**94.16**	0.0125	22.23
XTasC-Net† (Ours)	3.15	**66.51**	93.56	**0.0122**	**19.40**

The result shows that our method reduces error both in terms of absolute error and relative error for the depth estimation task. Intuitively, learning from the segmentation task's predictions motivates the model to output different depth ranges for each class. We think that this leads to depth predictions with more contrast and hence higher accuracy.

Our qualitative results are shown in Fig. 4.

2) Results on NYU dataset: We compare our results against the 2 baseline models in Table III.

Again, we observe a slight drop in performance for segmentation task while by using uncertainty weighting or GradNorm method, XTasC-Net improves depth estimation scores. For both equal weighting and uncertainty weighting, the results show that XTasC-Net improves overall performance.

Table IV shows the result for all methods. We use the scores using uncertainty weights for comparison because this weighting method resulted in higher performance for both Cityscapes and NYU dataset.

As shown in the table, our model outperforms all previous works and achieves state of the art for both tasks.

For the segmentation task, we again observe a larger improvement for mIoU compared to pixel accuracy. As described in section V-E1, we can infer that this is due to the models' characteristics. Using only convolutional layers without any attention modules, we achieve a 7.87 point improvement of mIoU and a smaller gain of 2.67 points for pixel accuracy. Our model shows significant performance for depth estimation tasks as well (Abs Err: 0.6002 → 0.5954, Abs Rel Err: 0.2547 → 0.2235).

VI. CONCLUSION

In this work, we made mainly two contributions. First, we proposed a new model architecture called XTasC-Net (Cross-Task Consistency Network) that achieved state-of-the-art results for most evaluation metrics on two benchmark datasets, Cityscapes and NYU dataset. We also showed that our model is parameter-efficient. Secondly, we introduced a new loss term, cross-task consistency loss, for the 2-task MTL setting.

Cross-task consistency loss is a term that takes the loss between the transferred prediction with the prediction of the other task. This loss motivated the model to output consistent predictions for both tasks by transferring information of one task to the other. We showed both theoretically and empirically the effect of cross-task consistency loss. Although our proposed loss term is limited to 2-task MTL problems, it efficiently uses the task relationship for performance improvement.

While we only explored the case of semantic segmentation and depth estimation task, we expect our proposed framework can be applied to any MTL settings if there exists some relationship between the tasks. In the field of computer vision, many tasks are related to each other [23]. With the introduction of datasets aimed for MTL [23], [42], we think that our

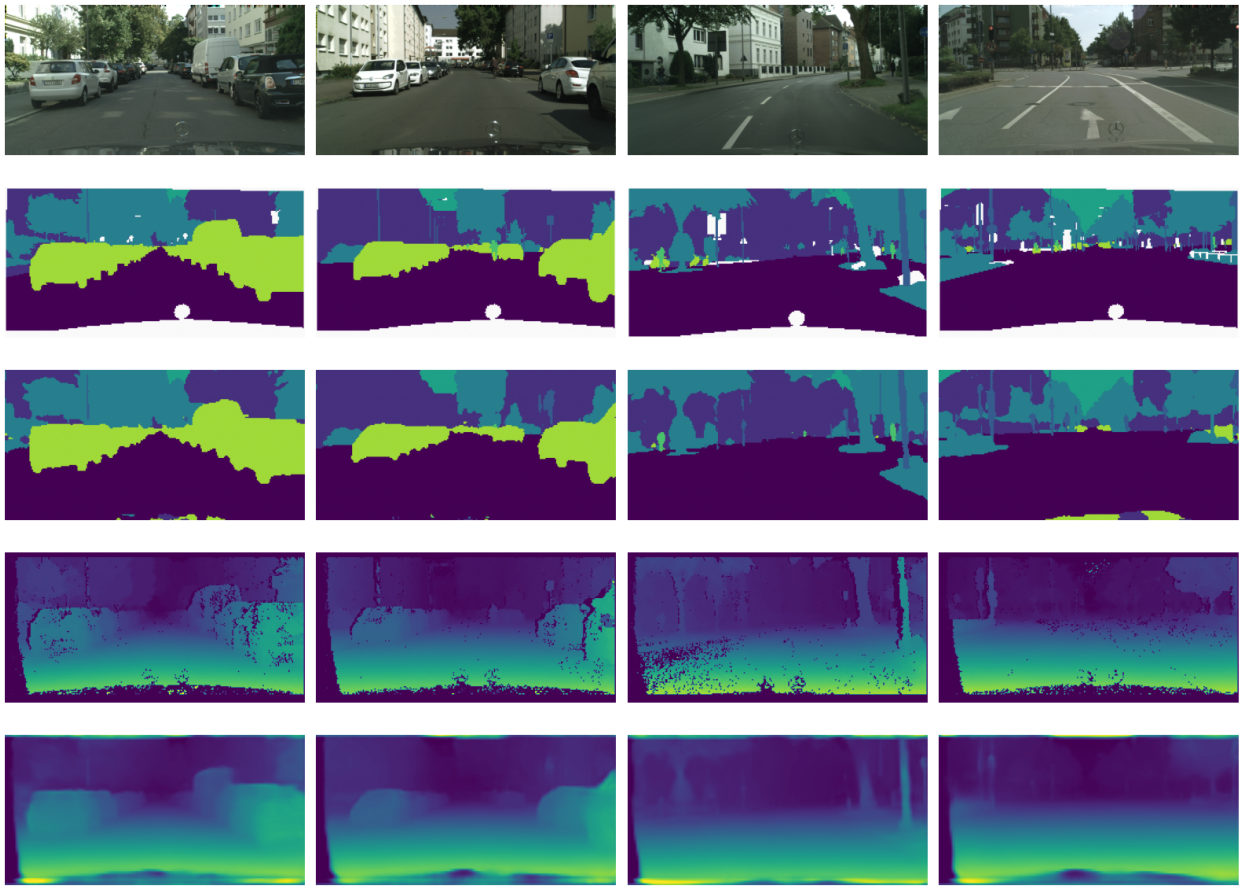

Fig. 4. Qualitative results on Cityscapes validation set. From top to bottom: input image, ground truth segmentation, predicted segmentation, ground truth depth, and predicted depth.

TABLE III
ABLATION RESULTS OF DIFFERENT WEIGHTING METHODS ON NYU TEST SET FOR 13-CLASS SEMANTIC SEGMENTATION AND DEPTH ESTIMATION TASK. BEST RESULTS ARE IN **BOLD**.

Model	Weighting	Segmentation (Higher Better)		Depth (Lower Better)		Performance (Higher Better)
		mIoU	Pix Acc	Abs Err	Rel Err	Δ_m
Base ST-Net	Equal weights	30.62	62.30	0.6451	0.2443	0.00
Base MT-Net	Equal weights	30.42	62.89	0.6384	0.2326	1.53
	Uncert. weights [6]	**31.02**	**63.58**	0.6103	0.2284	3.82
XTasC-Net	Equal weights	30.65	63.13	0.6319	0.2237	2.98
	Uncert. weights [6]	30.31	63.02	**0.5954**	0.2235	4.09
	GradNorm [7]	30.71	63.44	0.6002	**0.2222**	**4.53**

framework will help the model learn and exploit the task relationships. Furthermore, if the framework can be extended to an arbitrary number of tasks, it will help our understanding of task relationships.

APPENDIX A
PROOF OF THE PROPOSITION

Proof: By the law of total expectation, the first part of the equation is,

$$\begin{aligned}&\mathbb{E}[(\mathbb{E}[\mathbb{E}[Y|X]|\mathbb{E}[Z|X]] - \mathbb{E}[Y|X])^2] \\ &= \mathbb{E}[(\mathbb{E}[Y|X] - \mathbb{E}[Y|X])^2] \\ &= 0\end{aligned}$$

The first inequality of the equation holds because,

$$\begin{aligned}&\mathbb{E}[(\mathbb{E}[Y|\mathbb{E}[Z|X]] - \mathbb{E}[Y|X])^2] \\ &= \mathbb{E}[(\mathbb{E}[Y|\mathbb{E}[Z|X]] - \mathbb{E}[\mathbb{E}[Y|X]|\mathbb{E}[Z|X]])^2] \\ &= \mathbb{E}[(\mathbb{E}[Y - \mathbb{E}[Y|X]|\mathbb{E}[Z|X]])^2] \\ &\geq 0\end{aligned}$$

Furthermore, the second inequality holds from Jensen's inequality,

$$\mathbb{E}[(\mathbb{E}[Y|\mathbb{E}[Z|X]] - \mathbb{E}[Y|X])^2]$$

TABLE IV
RESULTS OF MULTI-TASK LEARNING ON NYU TEST SET FOR 13-CLASS SEMANTIC SEGMENTATION AND DEPTH ESTIMATION TASK. BEST RESULTS ARE IN **BOLD** AND SECOND BEST ARE UNDERLINED. * EQUAL WEIGHTS. † UNCERTAINTY WEIGHTS. ‡ GRADIENT-BASED WEIGHT LEARNING. ⋆ DWA.

Model	Segmentation (Higher Better)		Depth (Lower Better)	
	mIoU	Pix Acc	Abs Err	Rel Err
STAN [4]	16.65	55.07	0.6935	0.2891
Dense [41]	17.22	55.59	0.6002	0.2654
Cross-Stitch [2]	17.01	53.99	0.6095	0.2671
MTAN⋆ [4]	20.10	53.73	0.6417	0.2758
PCGrad† [8]	21.29	54.07	0.6705	0.3000
KD4MTL* [29]	22.44	57.32	0.6003	0.2601
AdaMT-Net‡ [5]	20.61	58.91	0.6136	0.2547
XTasC-Net†(Ours)	**30.31**	**63.02**	**0.5954**	**0.2235**

$$
\begin{aligned}
&= \mathbb{E}[(\mathbb{E}[Y - \mathbb{E}[Y|X]|\mathbb{E}[Z|X]])^2] \\
&\leq \mathbb{E}[\mathbb{E}[(Y - \mathbb{E}[Y|X])^2|\mathbb{E}[Z|X]]] \\
&= \mathbb{E}[(Y - \mathbb{E}[Y|X])^2] = \xi_Y
\end{aligned}
$$

■

APPENDIX B
EFFECTS OF DIFFERENT ENCODERS

As explained in section IV, we use ResNet as our encoder. ResNet has several variations of depth, and different works have used different encoders as their encoder. Therefore, we also evaluate the change of performance between 3 types of ResNet, ResNet18, 34, and 50 (Table V, VI). We evaluate ResNet18 because it has fewer parameters compared to ResNet34. We also evaluate ResNet50 because it has more parameters but no more than the other models used for comparison.

TABLE V
ABLATION RESULTS OF DIFFERENT ENCODERS ON CITYSCAPES VALIDATION SET FOR 7-CLASS SEMANTIC SEGMENTATION AND DEPTH ESTIMATION TASK. #P SHOWS THE NUMBER OF PARAMETERS OF THE MODEL. * USE WEIGHTS PRETRAINED ON IMAGENET.

ResNet	#P. $[\times 10^7]$	Segmentation (Higher Better)		Depth (Lower Better)	
		mIoU	Pix Acc	Abs Err	Rel Err
18	2.14	66.15	93.47	0.0124	19.28
34	3.15	66.51	93.56	0.0122	19.40
34*	3.15	68.71	94.23	0.0111	18.48
50	4.04	66.12	93.50	0.0124	19.98

The results show that ResNet34 is the best encoder for both datasets resulting in the best scores for most evaluation metrics. Naturally, using a deeper encoder motivates the model to learn more contextual features and improve performance. However, for our model, the results show no such improvement by using deeper ResNet. The results also confirm that using pretrained weights of ResNet34 leads to better performance. Although the results in section V-E use the scores achieved without using pretrained weights for fair comparison, one should consider using the weights in applications.

TABLE VI
ABLATION RESULTS OF DIFFERENT ENCODERS ON NYU TEST SET FOR 13-CLASS SEMANTIC SEGMENTATION AND DEPTH ESTIMATION TASK. #P SHOWS THE NUMBER OF PARAMETERS OF THE MODEL. * USE WEIGHTS PRETRAINED ON IMAGENET.

ResBet	#P. $[\times 10^7]$	Segmentation (Higher Better)		Depth (Lower Better)	
		mIoU	Pix Acc	Abs Err	Rel Err
18	2.14	29.72	61.44	0.6174	0.2314
34	3.15	30.31	63.02	0.5954	0.2235
34*	3.15	44.75	75.81	0.4851	0.1835
50	4.04	28.63	61.66	0.6115	0.2287

REFERENCES

[1] M. Long, Z. Cao, J. Wang, and P. S. Yu, "Learning multiple tasks with multilinear relationship networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 30, 2017, pp. 1594–1603.

[2] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert, "Cross-stitch networks for multi-task learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 3994–4003.

[3] P. Yang, Q. Tan, J. Ye, H. Tong, and J. He, "Deep multi-task learning with adversarial-and-cooperative nets," in *Proc. 28th Int. Joint Conf. Artif. Intell. (IJCAI)*, 7 2019, pp. 4078–4084.

[4] S. Liu, E. Johns, and A. J. Davison, "End-to-end multi-task learning with attention," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 1871–1880.

[5] A. Jha, A. Kumar, B. Banerjee, and S. Chaudhuri, "Adamt-net: An adaptive weight learning based multi-task learning model for scene understanding," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, 2020, pp. 3027–3035.

[6] A. Kendall, Y. Gal, and R. Cipolla, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 7482–7491.

[7] Z. Chen, V. Badrinarayanan, C.-Y. Lee, and A. Rabinovich, "GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks," in *Proc. 35th Int. Conf. Mach. Learn. (ICML)*, 2018, pp. 794–803.

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

[9] P. Chen, A. H. Liu, Y. Liu, and Y. F. Wang, "Towards scene understanding: Unsupervised monocular depth estimation with semantic-aware representation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 2619–2627.

[10] M. Klingner, J.-A. Termöhlen, J. Mikolajczyk, and T. Fingscheidt, "Self-supervised monocular depth estimation: Solving the dynamic object problem by semantic guidance," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Cham: Springer, 2020, pp. 582–600.

[11] J. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, 2017, pp. 2242–2251.

[12] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. 37th Int. Conf. Mach. Learn. (ICML)*, 2020, pp. 1597–1607.

[13] C. Tosh, A. Krishnamurthy, and D. Hsu, "Contrastive learning, multi-view redundancy, and linear models," 2020. [Online]. Available: http://arxiv.org/abs/2008.10150

[14] J. Burge, C. C. Fowlkes, and M. S. Banks, "Natural-scene statistics predict how the figure–ground cue of convexity affects human depth perception," in *J. Neuroscience*, vol. 30, 2010, pp. 7269–7280.

[15] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 3213–3223.

[16] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Cham: Springer, 2012, pp. 746–760.

[17] R. Caruana, "Multitask learning," in *Learning to Learn*, S. Thrun and L. Pratt, Eds. Boston, MA: Springer, 1998, ch. 5, pp. 95–133.

[18] S. Ruder, "An overview of multi-task learning in deep neural networks," 2017. [Online]. Available: http://arxiv.org/abs/1706.05098

[19] C. A. Micchelli and M. Pontil, "Kernels for multi–task learning," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2004, pp. 921–928.

[20] C. Ciliberto, Y. Mroueh, T. Poggio, and L. Rosasco, "Convex learning of multiple tasks and their structure," in *Proc. 32nd Int. Conf. Mach. Learn. (ICML)*, 2015, p. 1548–1557.

[21] Y. Zhang and D.-Y. Yeung, "A convex formulation for learning task relationships in multi-task learning," in *Proc. 26th Conf. Uncert. Artif. Intell. (UAI)*, 2010, p. 733–742.

[22] C. Ciliberto, A. Rudi, L. Rosasco, and M. Pontil, "Consistent multitask learning with nonlinear output relations," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 30, 2017, pp. 1986–1996.

[23] A. R. Zamir, A. Sax, W. Shen, L. Guibas, J. Malik, and S. Savarese, "Taskonomy: Disentangling task transfer learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 3712–3722.

[24] F. Zhou, C. Shui, M. Abbasi, L. E. Robitaille, B. Wang, and C. Gagné, "Task similarity estimation through adversarial multitask neural network," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 2, pp. 466–480, 2021.

[25] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *NIPS Deep Learning and Representation Learning Workshop*, 2015. [Online]. Available: http://arxiv.org/abs/1503.02531

[26] D. Xu, W. Ouyang, X. Wang, and N. Sebe, "Pad-net: Multi-tasks guided prediction-and-distillation network for simultaneous depth estimation and scene parsing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 675–684.

[27] S. Vandenhende, S. Georgoulis, B. D. Brabandere, and L. Van Gool, "Branched multi-task networks: Deciding what layers to share," 2019. [Online]. Available: http://arxiv.org/abs/1904.02920

[28] Z. Zhang, Z. Cui, C. Xu, Y. Yan, N. Sebe, and J. Yang, "Pattern-affinitive propagation across depth, surface normal and semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 4106–4115.

[29] W.-H. Li and H. Bilen, "Knowledge distillation for multi-task learning," in *Proc. Eur. Conf. Comput. Vis. Workshops (ECCVW)*. Cham: Springer, 2020, pp. 163–176.

[30] S. Vandenhende, S. Georgoulis, and L. Van Gool, "Mti-net: Multi-scale task interaction networks for multi-task learning," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Cham: Springer, 2020, pp. 527–543.

[31] A. Mousavian, H. Pirsiavash, and J. Košecká, "Joint semantic segmentation and depth estimation with deep convolutional networks," in *Proc. 4th Int. Conf. 3D Vis.*, 2016, pp. 611–619.

[32] Z. Zhang, Z. Cui, C. Xu, Z. Jie, X. Li, and J. Yang, "Joint task-recursive learning for semantic segmentation and depth estimation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, September 2018, pp. 235–251.

[33] O. Sener and V. Koltun, "Multi-task learning as multi-objective optimization," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2018, pp. 525–536.

[34] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017.

[35] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow, "Digging into self-supervised monocular depth prediction," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, 2019.

[36] L. C. Tiao, E. V. Bonilla, and F. Ramos, "Cycle-consistent adversarial learning as approximate bayesian inference," 2018. [Online]. Available: http://arxiv.org/abs/1806.01771

[37] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *Med. Image Comput. Comput.-Assisted Intervention (MICCAI)*, pp. 234–241, 2015.

[38] A. R. Zamir, A. Sax, N. Cheerla, R. Suri, Z. Cao, J. Malik, and L. J. Guibas, "Robust learning through cross-task consistency," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 11 194–11 203.

[39] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 770–778.

[40] K. K. Maninis, I. Radosavovic, and I. Kokkinos, "Attentive single-tasking of multiple tasks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, p. 1851–1860.

[41] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, p. 4700–4708.

[42] M. Roberts and N. Paczan, "Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding," 2020. [Online]. Available: http://arxiv.org/abs/2011.02523

Akihiro Nakano received the B.E. degree in systems innovation from the University of Tokyo, Tokyo, Japan, in 2021.
He is currently pursuing the master's degree at Department of Technology Management for Innovation, the University of Tokyo, Tokyo, Japan. His current research interests include multitask learning, multimodal learning, and reinforcement learning.

Shi Chen received the M.S. and Ph.D. degrees in nuclear engineering and management from the University of Tokyo, Tokyo, Japan, in 2017 and 2020, respectively.
He is currently a project researcher with Department of Nuclear Engineering and Management, School of Engineering, The University of Tokyo, Tokyo, Japan. His current research interests include nuclear safety & security, computer vision, pattern recognition, and deep learning.

Kazuyuki Demachi was born in Sendai, Miyagi, Japan in 1970. He received the B.S. in nuclear engineering in 1992, M.S. in quantum system engineering in 1994 and Ph.D. degree in Quantum System Engineering from the University of Tokyo, Tokyo, Japan.
From 1997 to 1999, he was an assistant professor with the Nuclear Engineering Research Laboratory (NERL) of Department of Engineering, the University of Tokyo. Since 2000, he has been a lecturer with the NERL. Since 2001, he has been an associate professor of NERL. NERL was reorganized to Nuclear Professional School, School of Engineering. His research interests include nuclear security technology and nuclear maintenance engineering.