electronics

MDPI

Article

Multi-Task Learning Using Gradient Balance and Clipping with an Application in Joint Disparity Estimation and Semantic Segmentation

Yiyou Guo and Chao Wei *

College of Surveying and Geo-Informatics, Tongji University, Shanghai 200092, China; yiyouguo526@tongji.edu.cn
* Correspondence: cwei@tongji.edu.cn

Abstract: In this paper, we propose a novel multi-task learning (MTL) strategy from the gradient optimization view which enables automatically learning the optimal gradient from different tasks. In contrast with current multi-task learning methods which rely on careful network architecture adjustment or elaborate loss functions optimization, the proposed gradient-based MTL is simple and flexible. Specifically, we introduce a multi-task stochastic gradient descent optimization (MTSGD) to learn task-specific and shared representation in the deep neural network. In MTSGD, we decompose the total gradient into multiple task-specific sub-gradients and find the optimal sub-gradient via gradient balance and clipping operations. In this way, the learned network can satisfy the performance of specific task optimization while maintaining the shared representation. We take the joint learning of semantic segmentation and disparity estimation tasks as the exemplar to verify the effectiveness of the proposed method. Extensive experimental results on a large-scale dataset show that our proposed algorithm is superior to the baseline methods by a large margin. Meanwhile, we perform a series of ablation studies to have a deep analysis of gradient descent for MTL.

Keywords: multi-task learning; gradient balance; clipping

Citation: Guo, Y.; Wei, C. Multi-Task Learning Using Gradient Balance and Clipping with an Application in Joint Disparity Estimation and Semantic Segmentation. *Electronics* **2022**, *11*, 1217. https://doi.org/10.3390/electronics11081217

Academic Editor: Jose Eugenio Naranjo

Received: 2 March 2022
Accepted: 7 April 2022
Published: 12 April 2022

1. Introduction

In recent years, due to the powerful representation of deep learning, a deep convolutional neural network (DCNN) basically dominates every task of computer vision, including image classification [1–3], object detection [4–6], semantic segmentation [7–9], etc. In each task, methods based on DCNN incessantly set new records on each benchmark and achieve the state-of-the-art. However, due to the computation limitation or real-world application demand, it is desirable to construct a single network that can handle multiple tasks simultaneously. To this end, many researchers sort to multi-task learning (MTL) [10–13]. Compared to the standard single task learning, current MTL models contain two parts: a shared backbone network for shared feature extraction and a multiple head network in which each head corresponds to one task prediction. To obtain desirable results for each task with one single network, most researchers focus on two principles to boost the performance: firstly, how to design a strong backbone network that can learn both task-specific features and shared features [5,13,14]. The motivation is that the generalizable representation helps to avoid overfitting while task-specific features help the prediction of each head. However, it is difficult to directly learn the generalizable representation and task-specific features explicitly. Secondly, due to the loss functions being used for guiding the backbone network to learn to handle different tasks, how to balance the weights of each loss is vital for MTL. Current MTL methods [15,16] usually need large manual trials or empirical parameter tuning to obtain satisfying results. This procedure relies on the observation of the final performance, which is tedious and time-consuming.

In addition, the tasks handled in MTL are various, especially the domain gap of different tasks may be very large. For instance, the joint learning of disparity estimation [17–19] and semantic segmentation [7–9], which belong to regression and classification, respectively, can be very difficult due to their innate differences. Therefore, how to balance the proportion of loss function of different tasks is a non-trivial problem.

Based on the above observations, we consider if the shared backbone can learn enough mutual features in the shallow part of the network, and the subsequent branches will become less important, which will greatly reduce the parameters and reasoning time. Moreover, the well-learned backbone can provide a great starting point for the subsequent branches to further boost the performance of the network. Additionally, this also brings another advantage: the loss weight is not sensitive to the final performance.

With these aims in mind, this paper focuses on how to sufficiently leverage the mutual feature learning encoder of different tasks. We propose the multi-task stochastic gradient descent optimization (MTSGD). MTSGD can adaptively balance the sub-gradients of different tasks and discard the gradients with opposite sub-gradients. Hence, the backbone network will no longer be biased towards the task with a great value of gradient. Overall, we summarize the contributions of this paper as follows:

- We have a deep analysis of the feature learning in MTL and find that mutual feature learning among the backbone network is important for the final performance. Furthermore, we introduce a novel learning method from the angle of gradient descent to avoid complex network design and elaborate loss weights adjustment.
- We propose an MTSGD method to optimize multi-task learning from the perspective of considering multi-task learning as an optimization problem. We decompose the multiple task gradient into task-specific sub-gradient and leverage the proposed gradient clipping operation to balance the contribution of each sub-gradient.
- We evaluate the proposed method on the challenging MTL case: joint learning of disparity estimation and semantic segmentation. Experiment results on the benchmark dataset validate the effectiveness of the proposed MTSGD.

2. Related Work

Our work is based on some previous work, including semantic segmentation, depth estimation, and multi-task learning. In this part, we will review some representative work.

Semantic Segmentation has been greatly developed in the era of deep learning. FCN [7] is the first to use the fully convolutional neural network for semantic segmentation, and achieve end-to-end semantic segmentation prediction. Unet [20] introduces high and low-level features and uses deconvolution for up-sample operation instead of bilinear interpolation. GCN [21] proposes a Global Convolutional Network to improve the accuracy of classification and location in semantic segmentation. PSPNet [8] uses the pyramid pooling module to aggregate context information from different regions, thereby improving the ability to obtain global context information. The Deeplab [9,22–24] family uses dilated convolution and ASPP to obtain large receptive fields. They greatly improve the quality of semantic segmentation. Recently, there has been work [25] using upsampling methods based on dependency data to obtain multi-scale information and context information, further achieving the state-of-the-art.

Depth Estimation is generally divided into two categories, monocular and binocular. In terms of binocular depth prediction, DispNet [26] constructs a synthetic data set and then utilizes a Unet-structured network for disparity estimation. The CRL [17] further utilizes a cascaded network to predict disparity, and GC-Net [27] uses both 3D convolution and 3D deconvolution to simultaneously convolve disparity and spatial dimensions. PSMNet [18] takes advantage of SPP and multiple hourglass structures for better regressing depth. In monocular prediction, Eigen et al. [28] is the first to use neural networks to perform depth estimation based on a single picture. Liu et al. [29] believe that depth values are continuous, so monocular deep learning is considered as a continuous CRF learning problem. GeoNet [12] uses an additional pose network to provide camera position information and

uses video to train the network to predict monocular depth. CSPN [19] uses the affinity matrix to learn the relationships of the fields around the pixels to better predict the depth.

Multi-task Learning. In general, a network with multiple outputs can be called a multi-tasks model. Mask-rcnn [5] adds an additional instance branch to the object detection. PAD-Net [11] uses multi-model distillation to fuse features between different tasks. PAP [10] introduces the Affinity Learning Layer on each task's branch to learn the cross-task affinity matrix. MultiPoseNet [30] can jointly handle human detection, critical point detection, body segmentation, and pose estimation. The novel allocation algorithm is implemented by the Pose Residual Network (PRN), which receives the results of key points and human detection, and generates accurate poses by assigning key points to human instances. Strezoski et al. [31] propose the task routing, which enables the network to perform a large number of tasks by adopting the conditional feature-wise transformation over the convolutional activations. MT-SFG [14] proposes the stochastic filter groups module to divide the convolution kernels into task-specific feature learning and shared feature learning. UM-Adapt [32] proposes a multi-task framework to perform unsupervised domain adaptation by using cross-task distillation. Multi-task Learning network can perform segmentation and depth prediction simultaneously, which is an important point for many tasks, such as automatic drive, such as MTAN [13] and AdaMT [33]. MTAN [13] uses the attention module to learn the task-specific features from the global features

3. Optimizer for Multi-Task Learning

3.1. Important Observation

As mentioned above, multi-task learning can be deemed as an optimization problem. Let us be given a network $F = (\theta; x)$, where θ denotes the parameters of the network and the x is the input of the network. Network F has multiple outputs $\hat{y_i} \in \{\hat{y_1}, \hat{y_2}, \ldots \hat{y_n}\}$ corresponding to different task $t_i \in \{t_1, t_2, \ldots, t_n\}$. Each task has its own loss $L_i \in \{L_1, L_2, \ldots, L_n\}$. The total loss can be calculated as $L_{total} = w_i \cdot L_i + l_2$, where w_i denotes the weight of the loss of each task and l_2 denotes the L2 regularization term. The proportion of each w determines the performance of each task. It is very difficult to search the optimal combination of each w. Hence, we try to solve this dilemma from a different perspective. Generally speaking, during the backpropagation of the training scheme, at the level of the parameter, each sub-loss will contribute its gradient to the total gradient, which is $\Delta = \Delta_1 + \Delta_2 + \ldots + \Delta_n$, a summation of different gradients of different tasks. There are two problems here if one of the tasks has a large scale gradient, even many times larger than others, this simple summation operation will result in the total gradient Δ favoring the gradient of one task, making $\Delta \approx max(\Delta_1, \Delta_2, \ldots, \Delta_n)$. This will lead to poor performance of other tasks.

In Figure 1, the more intensive contour lines mean a large value of gradients. Therefore, the subgradient of task A is much larger than the other; then, the direction of parameters update based on the total gradient will be biased towards task A.

As the failure examples are shown in Figure 2, the performance of disparity is much better than the semantics due to the sub-task of disparity having a much larger gradient. Another problem is that the direction of gradients from different tasks can be quite varied, which will result in the total gradient not being approached in any of the tasks. The closer the gradient direction from different tasks is, the more likely the network is to learn the mutual feature of multiple tasks. On the contrary, in extreme cases, if the gradient direction is opposite, the direction of the total gradient will be biased toward the one that has the largest gradient value. The proposed gradient-based MTL introduces a multi-tasks stochastic gradient descent optimization (MTSGD) to learn task-specific and shared representation using gradient balance and clipping operations. In joint learning, disparity estimation and semantic segmentation can preserve structural information and suppress the noisy of other objects well. In this way, our MTL model can reduce confusion and increase discriminability, obtaining competitive performance.

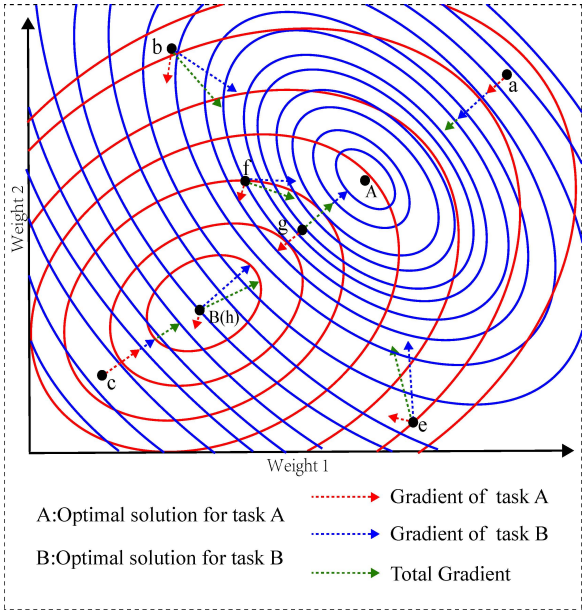

Figure 1. Illustration of the difficulty of multi-task learning. The axis of x and y denotes the different parameters. The red lines and blue lines denote the contour line of the loss of task B and A, respectively. The sparsity of the contour lines indicates the value of the gradient of tasks A and B, respectively.

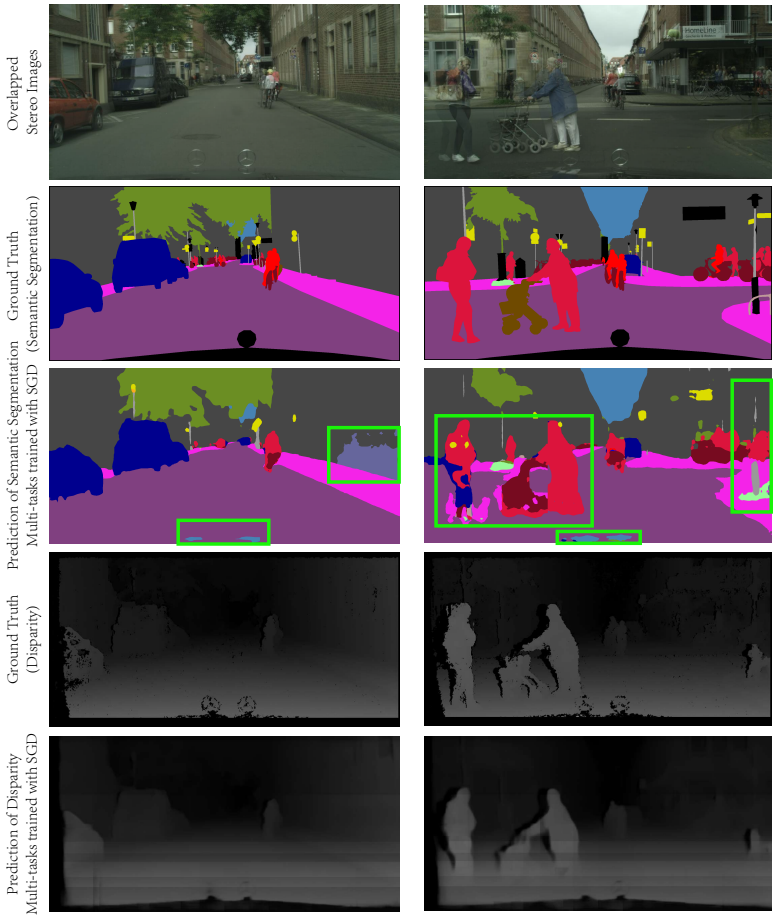

Figure 2. Failure cases of joint learning of semantic segmentation and disparity using the original SGD optimization algorithm. Because the sub-task of disparity has a much larger gradient during training, the network tends to perform a better disparity but with a very poor semantic segmentation.

As shown in Figure 3, when the weight of the loss function of semantic segmentation and disparity estimation to the total loss function is set to 1:1, we visualized their respective gradient distribution in 500 backpropagations during one epoch. We can find that the gradient value of one task is much larger than another task, which will make the network parameters update bias towards the task with the larger gradient. Faced with the problem of the unbalanced gradient distribution, we can try to adjust the combination of the weights of the loss function between different tasks. However, the process of finding such an optimal combination will consume a lot of time and resources, which is often unrealistic. On the contrary, in this work, we attempt to solve this problem directly from the perspective of optimization rather than adjusting the proportion of the weight of different losses between different tasks. We call our method as multi-task stochastic gradient descent (MTSGD). We only make some simple modifications to the original stochastic gradient descent with momentum algorithm (for simplicity, we call it SGD in the rest of the paper) but to obtain a high-performance improvement.

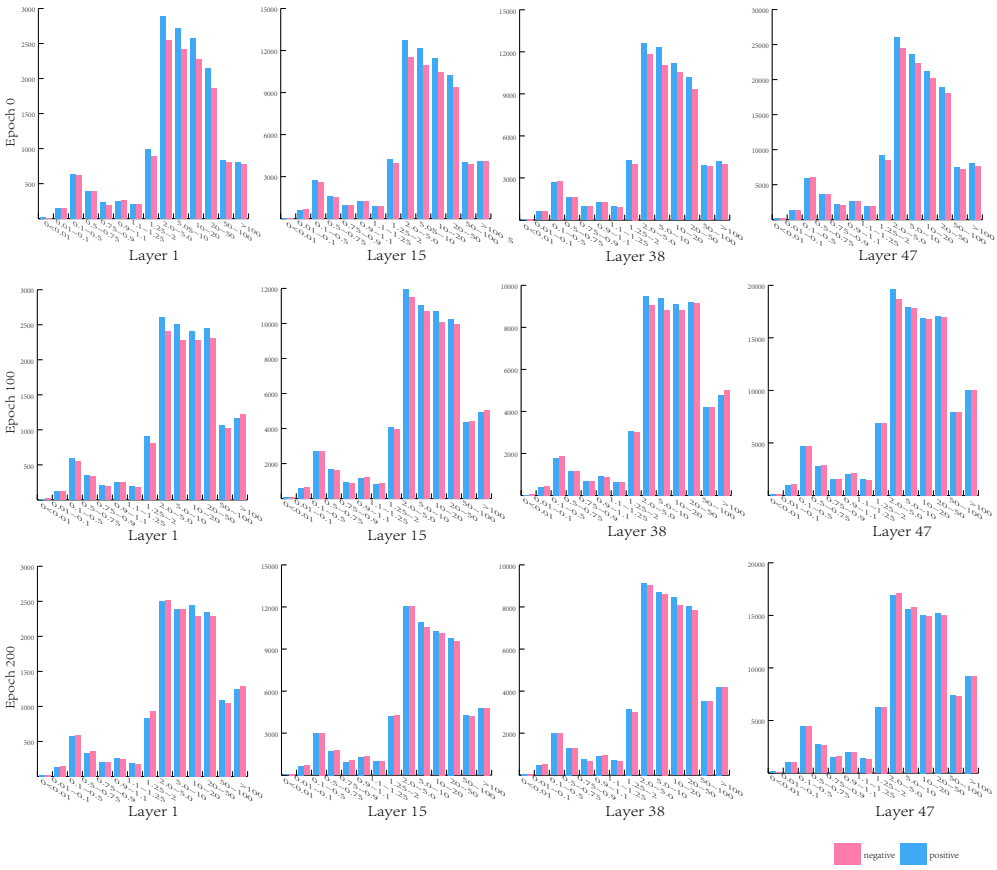

Figure 3. Sub-gradient distribution of semantic segmentation and disparity estimation. The data are gathered from 500 backpropagations of different layers in ResNet50 [2] of different epochs. The x-axis denotes the sub-gradient ratio which is the gradient of disparity: gradient of segmentation. In addition, the y-axis denotes the numbers of the ratio. 'Positive' and 'Negative' denote the sign of the value of ratio equation, and positive indicates the two sub-gradient in the same direction and negative indicates the opposite direction. The value of the x-axis indicates the scale of the difference of the two sub-gradients, 100 denotes that the gradient of disparity is 100 times of segmentation's and 0.01 denotes 0.01 times. Hence, to some extent, the values of 100 and 0.01 are equivalent.

3.2. Multi-Task Stochastic Gradient Descent

The proposed method is based on the SGD, which can be described as follows:

$$\begin{aligned} v &= \rho \cdot v + (1 - \rho) \cdot \Delta \\ w &= w - lr \cdot v - decay \cdot w \end{aligned} \tag{1}$$

where ρ is the momentum term, lr denotes the learning rate, decay denotes the learning rate, and decay denotes the L2 regularization term. In multi-task learning, total gradient Δ can be represented by the summation of multiple sub-gradients:

$$\Delta = \Delta_1 + \Delta_2 + \ldots + \Delta_n \tag{2}$$

where Δ_i denotes the sub-gradient of task i. Given a four-dimensional convolutional kernel w with the dimension of $(N \cdot C \cdot H \cdot W)$, instead of directly using the original gradient value of each task to update the parameters, we use the scaled gradient value. First, we calculate the direction vector of each sub-gradient:

$$e_i = \frac{\Delta_i}{\|\Delta_i\|_2}, \|\Delta_i\|_2 \in \mathbb{R}^{N \cdot C} \tag{3}$$

where $\|\Delta_i\|_2$ denotes the 2-norm of the calculated sub-gradient. Then, we measure the similarity of the directions of each of the direction vectors using element-wise multiplication:

$$c = e_1 \cdot e_2 \cdot \ldots \cdot e_n, \quad -1 \leqslant c \leqslant 1 \quad c \in \mathbb{R}^{N \cdot C} \tag{4}$$

where c denotes the similarity of the directions of each sub-gradient, –1 is the exact opposite direction, and 1 is the same direction. Meanwhile, we discard the gradient of the similarity of direction under a certain threshold:

$$m = max(threshold, c) \tag{5}$$

where m is a boolean mask with the value of 0 or 1 that masks out the gradient which does not satisfy the above condition. Thus, the total gradient will only contain the most similar sub-gradients. We hope that the network updates the parameters only under the circumstance in which the sub-gradient from different tasks share the same direction in each iteration. Because the gradient of these directions is very different, the updating direction of the network parameter may not move in the optimal direction or the direction of only a certain task. However, it is not enough to keep the total gradient in the most similar direction because tasks with a large scale gradient will drown out other tasks with small scale gradients. Thus, the network needs to balance the scale of the sub-gradients to the same order of magnitude:

$$s = min(\|\Delta_1\|_2, \|\Delta_2\|_2, \ldots, \|\Delta_n\|_2), \quad s \in \mathbb{R}^{N \cdot C} \tag{6}$$

where Δ_i denotes the sub-gradient of task i. We obtain the scaling factor w based on the value of the 2-norm of each sub-gradient, and we choose the smallest one as the scale factor. Then, we perform element-wise multiplication between the scale factor and the direction vectors. Hence, the sub-gradient should be in the same order magnitude:

$$g_i = s \cdot e_i \tag{7}$$

where s is the scale factor, and e_i denotes the direction vector of the sub-gradient of task i. The scaled sub-gradients will be summed up together to obtain the total gradient. Then, we conduct the element-wise multiplication between the total gradient and the similarity mask:

$$\begin{aligned} v &= \rho \cdot v + s \cdot (s_1 + s_2 + \ldots + s_n) \\ w &= w - lr \cdot v \end{aligned} \tag{8}$$

where ρ is the momentum term and lr is the learning rate. In short, we choose Equation (8) for the proposed gradient update instead of Equation (1) used in the traditional SGD.

4. Experiment

In this section, we will conduct a large number of experiments to study the effectiveness of our method, and we will verify the effectiveness of our method on the Cityscapes [34] dataset.

4.1. Overall Network Architecture

In this work, we adopt a classic encoder–decoder network as our multi-task network, and we choose ResNet50 [2] and SPP [35] with a dilated network [22,36] as our encoder. As shown in Figure 4, we aim to make the encoder network learn the mutual feature of multiple tasks as much as possible. In addition, to further reduce the performance improvement due to the task-specific well-designed decoder and demonstrate the effectiveness of our approach, we do not use a decoder with a very deep branch. A very deep task-specific decoder branch introduces redundant parameters. In our case, we use the simplest possible decoder, and each task-specific decoder contains only one convolutional layer for channel reduction and a bilinear interpolation for the up-sample operation to obtain the final prediction. Specifically, for the 19 class classification semantic segmentation task of Cityscapes [34], we directly perform a 1×1 convolutional layer to map the feature map from the encoder to a 19-channel feature map corresponding to 19 categories, and then directly up-sample the feature map to the original resolution for final prediction. For the disparity estimation, the same procedure is followed to map a one channel feature for final prediction. In this case, all tasks will share most of the parameters of the network, and each task will have only negligible task-specific parameters for channel reduction. Therefore, we can assume that the performance of each task will heavily rely on the mutual feature learned from the encoder, rather than on the task-specific branch.

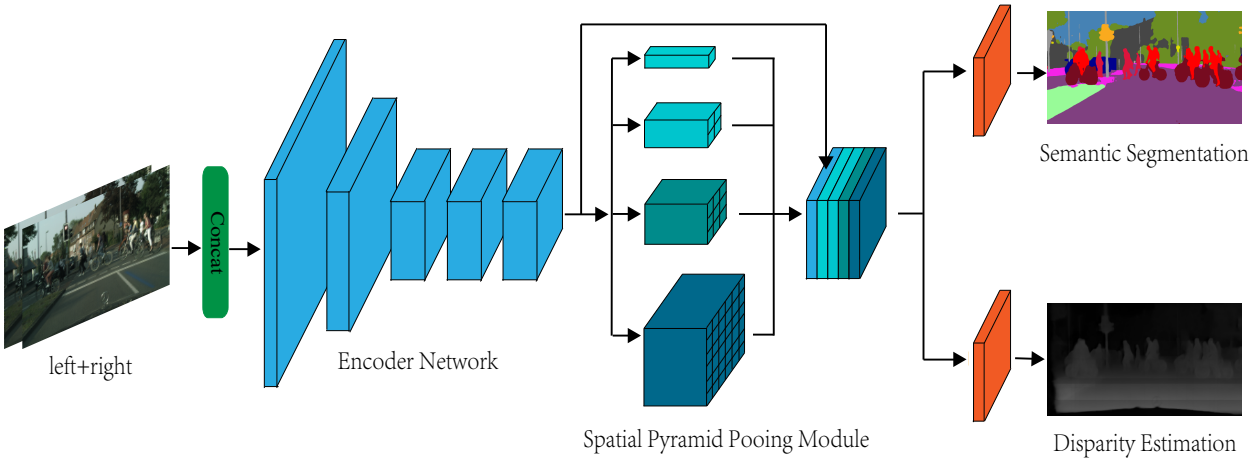

Figure 4. Overall network architecture. The network follows the classic encoder–decoder architecture. ResNet [2] with dilated convolution and the SPP module together serve as the encoder. The encoder directly concatenates the stereo images along the channel dimension and takes them as input. Beyond the encoder, each decoder only consists of a 1×1 convolution layer and a bilinear interpolation up-sampling layer for final prediction.

4.2. Experimental Settings

Dataset and Data Augmentation.We use Cityscapes [34] dataset to verify the effectiveness of our method. Cityscapes [34] is a dataset oriented to traffic scenes, which contains 5000 sets of finely annotated ground truth data and about 20,000 sets of coarsely annotated ground truth data. The data were derived from the road scenes in 50 different cities in Europe. The Cityscapes dataset provides the ground truth of semantic segmentation, instance segmentation, and disparity. Regarding semantic segmentation, it provides a 19 category semantic segmentation mask. Among the 5000 sets of the finely annotated data, there are

2975 training sets, 1525 validation sets, and 500 test sets, respectively. In this work, we mainly use the fine set of semantic segmentation and disparity ground truth for training.

Evaluation Metrics. To evaluate the performance of the semantic segmentation and disparity estimation, we use the standard Jaccard Index also called intersection-over-union (mIoU) as the metric index of semantic segmentation, which can be denoted as:

$$mIoU = \frac{1}{k+1} \sum_{i=0}^{k} \frac{p_{ij}}{\sum_{j=0}^{k} p_{ij} + \sum_{j=0}^{k} p_{ij} - p_{ii}} \tag{9}$$

where i denotes the groundtruth category, j denotes the predicted category, and k denotes the number of the categories. In addition, we use average endpoint error (AEPE) as the metric index of the disparity estimation, which can be denoted as:

$$AEPE = \frac{1}{N} \cdot \sum |y - \hat{y}| \tag{10}$$

where y is the groundtruth value and $\hat{y}$ is the predicted value.

4.3. Experimental Results

Our network architecture is based on ResNet50 [2] with ImageNet [1] pre-trained weights and SPP with dilated networks. The baseline networks use the original minibatch stochastic gradient descent with momentum, which is set to 0.9. Weight decay is set to 0.0001. For Cityscapes [34], we set the batch size to 16, patch size to 512 × 512, and the initial learning rate to 0.01, respectively. The performance of mIou and loss with different training epochs is shown in Figure 5. The oiou can be improved and maintained with the proper setting of epoch. Therefore, we trained the network 200 epochs with a fine set of Cityscapes [34] and adopted a step decay learning rate policy during training. At the time of the 100th and 150th epoch, the learning rate is decreased, multiplied by 0.1. We use the validation set to evaluate the performance of our proposed method.

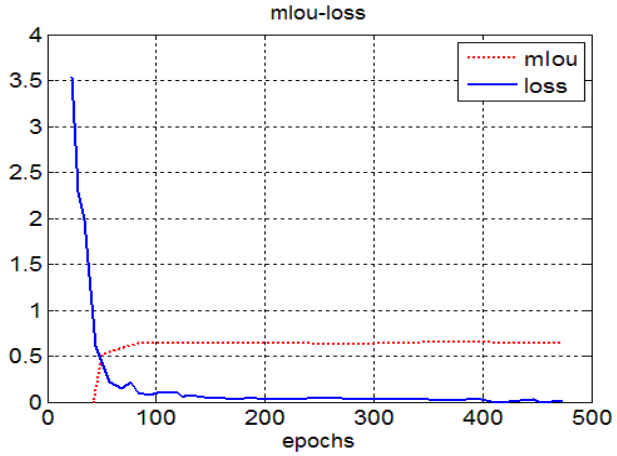

Figure 5. mIou-loss performance with different epochs.

We evaluate our MTSGD on the Cityscapes dataset with state-of-the-art methods and the performances are listed in Table 1. These are MC-CNN, STAN, MTAN, and AdaMt are the newly proposed MTL models, and Joint + SGD is our baseline method. The mIou value reflects that our network has the best Segmentation performance (65.0%) with a relatively large margin (2.47% improvement) compared to other comparison networks. Specifically, the proposed Joint + MTSGD without regularization obtains a significant increase in segmentation with the slightly worse performance of disparity estimation. Moreover, visual improvement on Cityscapes [34] is shown in Figure 6.

Table 1. Performance comparison with representative methods. 'Joint + SGD' denotes the joint learning of semantic segmentation and disparity estimation with the original SGD algorithm, 'Joint + MTSGD with regularization' denotes the model that is trained with our proposed MTSGD optimization algorithm and L2 regularization, and 'Joint + MTSGD without regularization' denotes the model that is trained with our proposed MTSGD optimization algorithm but without L2 regularization.

Methods	Segmentation (mIoU)	Disparity (AEPE)
MC-CNN [37]	-	3.41
Joint + SGD	44.4	3.98
STAN [13]	51.9	-
MTAN [13]	53.40	-
AdaMT [33]	62.53	-
Joint + MTSGD with regularization	62.0	4.26
Joint + MTSGD without regularization	65.0	3.91

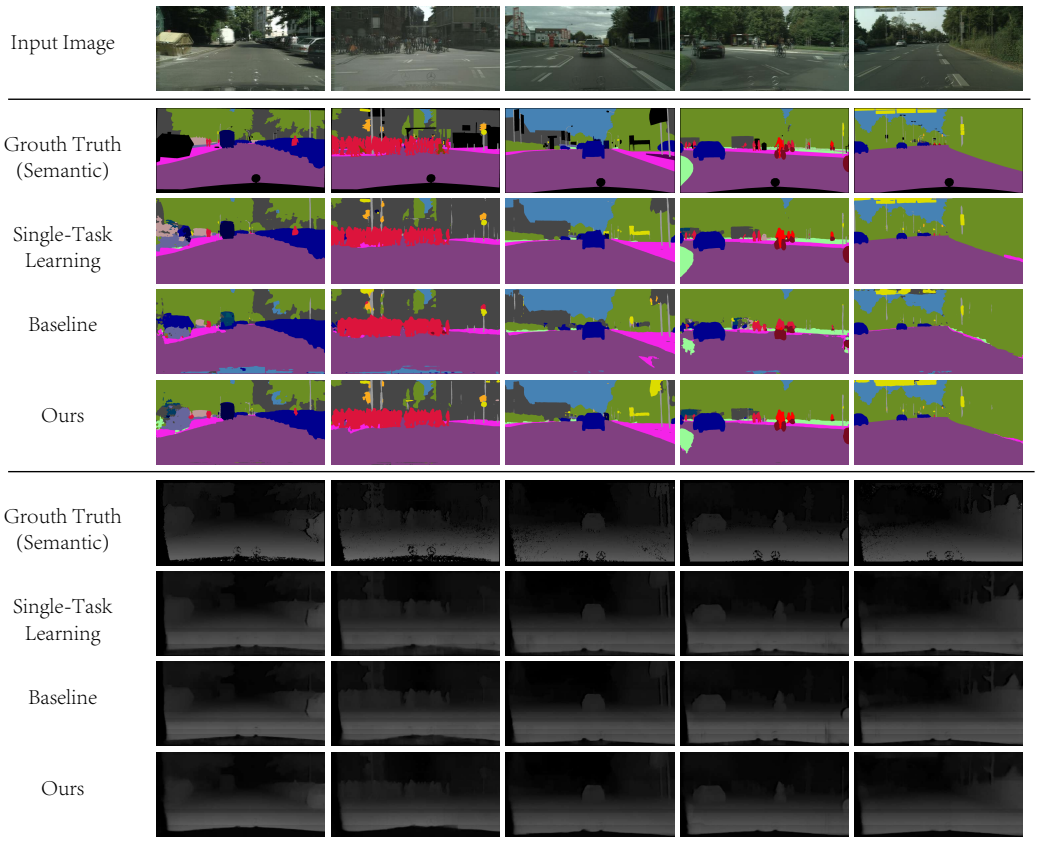

Figure 6. Visual improvement on Cityscapes [34].

4.4. Ablation Studies

Ablation Study for Gradient Balance and Clipping. To prove that the imbalanced sub-gradient will cause the updating of the parameters of the networks in favor of the task with a larger gradient, we experiment with whether the gradient balance will relieve the above issue. We set up three baseline models: the semantic segmentation only model, disparity only model, and joint model of semantic segmentation and disparity, respectively. All of the three baseline models are trained with the original stochastic gradient descent with the momentum algorithm. In addition, we set the weights of the loss function of the joint model to 1:1 regarding semantic segmentation and disparity. Then, we replace the SGD with our proposed MTSGD and set the gradient clipping rate to -1, which means that

there is no gradient clipping. As Table 2 shows, the balanced sub-gradient of two tasks can bring significant performance improvements, especially the performance of the semantic segmentation task. It is interesting to note that the the performance of the disparity task of the joint learning model is even better than the disparity only model. It suggests that some tasks can benefit from other tasks during multi-task learning. Furthermore, when the sub-gradient direction similarity is lower than a certain threshold, the performance can be further improved. Figure 7 shows 2D projections of the performance profile for Disparity Estimation and Semantic Segmentation. The bottom-right is better.

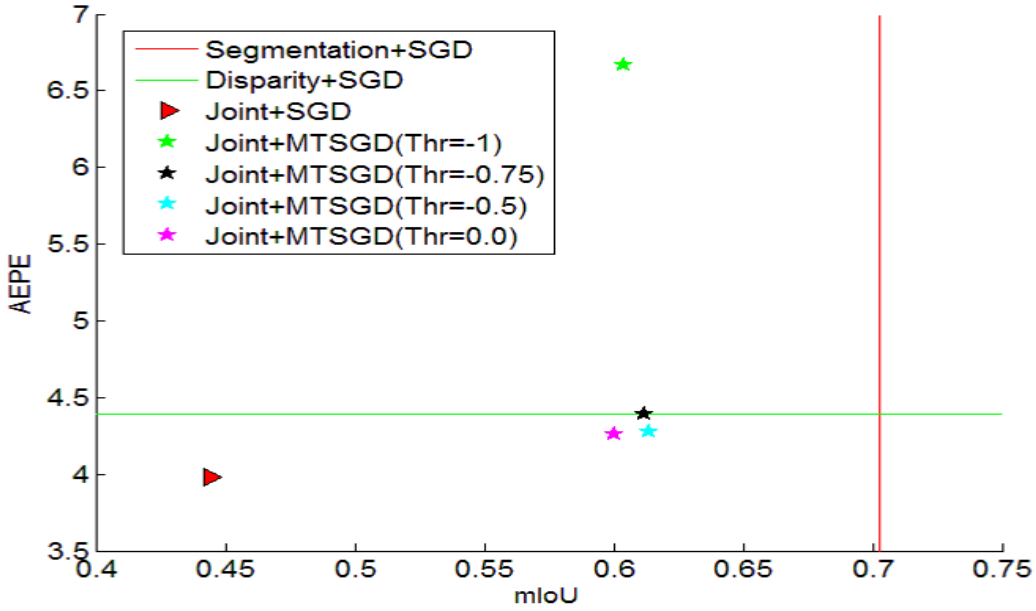

Figure 7. Performance profile for Disparity Estimation–Semantic Segmentation.

Table 2. Investigation of gradients balance and gradient clipping for MTSGD. 'Segmentation + SGD' and 'Disparity + SGD' denote a single task model of semantic segmentation and disparity estimation, respectively. Both of them are trained using the original SGD optimization algorithm. 'Joint + SGD' denotes the joint learning of semantic segmentation and disparity estimation with the original SGD algorithm, and 'Joint + MTSGD' denotes the model that is trained with our proposed MTSGD optimization algorithm. 'Threshold' denotes the threshold for gradient clipping based on the similarity of the sub-gradient of different tasks.

Methods	**Tasks Weight**		**Threshold**	**Segmentation (mIoU)**	**Disparity (AEPE)**
	Segmentation	**Disparity**			
Segmentaion + SGD	1.0	0.0	-	70.3	-
Disparity + SGD	0.0	1.0	-	-	4.39
Joint + SGD	0.5	0.5	-	44.4	3.98
Joint + MTSGD	0.5	0.5	−1.0	60.4	4.67
Joint + MTSGD	0.5	0.5	−0.75	61.2	4.39
Joint + MTSGD	0.5	0.5	−0.5	61.4	4.28
Joint + MTSGD	0.5	0.5	0.0	62.0	4.26

Ablation Study for Weight Decay. Generally speaking, the regularization technique is a powerful tool to prevent network overfitting during network training by limiting the value

of each parameter. The classic L2 regularization term is the most common regularization technique in deep learning. During training, with the L2 regularization term, the total loss function is altered to:

$$total_{loss} = loss + \frac{1}{2}\alpha \cdot w^2 \tag{11}$$

where α denotes the weight decay, and w is the value of parameters of the network. To investigate whether the regularization term still works in multi-tasks learning, we conduct an experiment that removes the L2 regularization term during training. As shown in Table 3, interestingly, we found that the performance of the model on the validation set improved significantly. One possible explanation is that, during propagation, the value of the parameters needs to be subtracted by themselves to prevent some parameters with a large value because the parameters with large values are easily overfitted to some accidental pattern. However, in multi-task learning, there are existing multiple tasks to compete the resource to update the parameters. During training, the total gradient is often not consistent with the same direction, which is likely to result in small value parameters. In this situation, the network tends to be underfit, and the extra regularization technique may backfire and undermine the performance of the network.

Table 3. Investigation of weight decay for MTSGD. 'Weight Decay' denotes the L2 regularization term of weight decay, and 0 indicates that there is no regularization term during training.

Methods	Weight Decay	Threshold	Seg (mIoU)	Disp (AEPE)
MTSGD	0.0	−1.0	62.6	4.13
MTSGD	0.0	−0.75	63.2	4.10
MTSGD	0.0	−0.5	64.3	4.05
MTSGD	0.0	0.0	65.0	3.91
MTSGD	0.0001	−1.0	60.4	4.67
MTSGD	0.0001	−0.75	61.2	4.39
MTSGD	0.0001	−0.5	61.4	4.28
MTSGD	0.0001	0.0	62.0	4.26

Ablation Study for Groups for Gradient Clipping. As we all know that each parameter (convolutional kernel) in the neural network can be regarded as a high dimensional vector, and the gradient represents the increment of the parameters in a backpropagation. When we calculate the direction vector of the gradient with a dimension of $(c \cdot k \cdot k)$, the calculated dimension is a key step. Hence, inspired by the group convolution [38–40], we try to calculate in two ways: one way is to calculate the direction vectors on each channel of each convolutional kernel, which indicates that a kernel with dimensional of $(c \cdot k \cdot k)$ results in c direction vectors. Another is based on the whole convolutional kernel as the calculating unit that obtains a single direction vector of a convolutional kernel with a dimensional of $(c \cdot k \cdot k)$. As shown in Table 4, we find that the prior way to calculate the direction vector can bring more performance improvement when it is used as the calculating unit. This may be due to higher dimensional vectors that can bring more stable gradient updates, as Table 4 shows.

4.5. Discussion

There is unbalanced gradient distribution in MTL, and the network parameters update is biased towards the task with the larger gradient. Its performance relies heavily on how to balance the proportion of loss function of different tasks. From the perspective of optimization, our MTL model learns task-specific and shared representation in the deep neural network via gradient balance and clipping operations. In this way, our MTSGD method can obtain the satisfying performance of specific task optimization while maintaining the shared representation. Moreover, the extra regularization technique may not have a positive effect on MTL.

Table 4. Investigation of the level of direction vector for MTSGD. The 'channel' denotes the direction vector that is calculated based on each channel of the convolutional kernel of each convolution kernel. In addition, 'filter' denotes that the direction vector is calculated based on the whole parameter

Methods	Level	Threshold	Seg (mIoU)	Disp (AEPE)
MTSGD	channel	−1.0	60.4	4.67
MTSGD	channel	−0.75	61.2	4.39
MTSGD	channel	−0.5	61.4	4.28
MTSGD	channel	0.0	62.0	4.26
MTSGD	filter	−1.0	59.7	5.32
MTSGD	filter	−0.75	59.8	5.27
MTSGD	filter	−0.5	60.0	5.12
MTSGD	filter	0.0	60.4	4.98

5. Conclusions

In recent years, deep learning has made great progress in various tasks in the area of computer vision, but there is still a lot of potential and problems to be solved in the field of multi-task learning. In this paper, we discuss some challenges that hinder the performance of multi-task learning. The main problem is that the various distribution of gradients between different tasks, which is caused by different tasks, has different optimal solution spaces. This usually makes the model biased towards one task or converge to a poor local optimal situation. The prior works try to appropriately allocate the weight of the loss of a different task or use a cumbersome decoder to recover the lost performance caused by the weak encoder that can not provide enough mutual features. The method we proposed allows the encoder to better learn the relationship between tasks. The experimental results demonstrate the effectiveness of our method. It is worth noting that our method does not conflict with the prior works. Through our method, the encoder has learned many more mutual features, that is, it provides a better starting point for the subsequent decoder network. Hence, the decoder can better focus on learning the task-specific feature.

At the same time, there remain some opening questions and potential for future research. The network architecture adopted in this work is relatively simple, and a better network structure may be able to further improve the performance. In addition, to some extent, the number of the well-learned parameters that have learned the mutual features between different tasks indicates the effectiveness of the network. How to reuse the less learned parameters will be a very interesting and meaningful problem.

Author Contributions: Y.G. proposed the methodology, data calculation, and result analysis. C.W. contributed to the data collection and discussion. All authors have read and agreed to the published version of the manuscript.

Funding: This paper was supported by the National Natural Science Foundation of China under Grant No. 41801246.

Acknowledgments: The authors would like to thank the anonymous reviewers for their constructive comments and suggestions, which strengthened this paper a lot.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; Berg, A.C.; Fei-Fei, L. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [CrossRef]
2. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
3. Hu, J.; Shen, L.; Sun, G. Squeeze-and-Excitation Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 18–23 June 2018; pp. 7132–7141.
4. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *Adv. Neural Inf. Process. Syst.* **2015**, *1*, 91–99. [CrossRef] [PubMed]

5. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
6. Lin, T.; Dollar, P.; Girshick, R.B.; He, K.; Hariharan, B.; Belongie, S.J. Feature Pyramid Networks for Object Detection. In Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, 21–26 July 2017; pp. 936–944.
7. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3431–3440.
8. Zhao, H.; Shi, J.; Qi, X.; Wang, X.; Jia, J. Pyramid scene parsing network. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2881–2890.
9. Chen, L.C.; Papandreou, G.; Schroff, F.; Adam, H. Rethinking atrous convolution for semantic image segmentation. *arXiv* **2017**, arXiv:1706.05587.
10. Zhang, Z.; Cui, Z.; Xu, C.; Yan, Y.; Sebe, N.; Yang, J. Pattern-Affinitive Propagation across Depth, Surface Normal and Semantic Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–17 June 2019; pp. 4106–4115.
11. Xu, D.; Ouyang, W.; Wang, X.; Sebe, N. Pad-net: Multi-tasks guided prediction-and-distillation network for simultaneous depth estimation and scene parsing. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 28–23 June 2018, pp. 675–684.
12. Yin, Z.; Shi, J. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 28–23 June 2018; pp. 1983–1992.
13. Liu, S.; Johns, E.; Davison, A.J. End-to-end multi-task learning with attention. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 1871–1880.
14. Bragman, F.J.; Tanno, R.; Ourselin, S.; Alexander, D.C.; Cardoso, J. Stochastic Filter Groups for Multi-Task CNNs: Learning Specialist and Generalist Convolution Kernels. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October 27–2 November 2019.
15. Guo, M.; Haque, A.; Huang, D.A.; Yeung, S.; Fei-Fei, L. Dynamic task prioritization for multitask learning. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 270–287.
16. Chen, Z.; Badrinarayanan, V.; Lee, C.Y.; Rabinovich, A. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. *arXiv* **2017**, arXiv:1711.02257.
17. Pang, J.; Sun, W.; Ren, J.S.; Yang, C.; Yan, Q. Cascade residual learning: A two-stage convolutional neural network for stereo matching. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 887–895.
18. Chang, J.R.; Chen, Y.S. Pyramid stereo matching network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 28–23 June 2018; pp. 5410–5418.
19. Cheng, X.; Wang, P.; Yang, R. Depth estimation via affinity learned with convolutional spatial propagation network. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 103–119.
20. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer Assisted Intervention*; Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F., Eds.; Springer: Cham, Switzerland, 2015; pp. 234–241.
21. Peng, C.; Zhang, X.; Yu, G.; Luo, G.; Sun, J. Large Kernel Matters–Improve Semantic Segmentation by Global Convolutional Network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4353–4361.
22. Chen, L.C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A.L. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv* **2014**, arXiv:1412.7062.
23. Chen, L.C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A.L. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 834–848. [CrossRef] [PubMed]
24. Chen, L.C.; Zhu, Y.; Papandreou, G.; Schroff, F.; Adam, H. Encoder-decoder with atrous separable convolution for semantic image segmentation. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 801–818.
25. Tian, Z.; He, T.; Shen, C.; Yan, Y. Decoders Matter for Semantic Segmentation: Data-Dependent Decoding Enables Flexible Feature Aggregation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 3126–3135.
26. Mayer, N.; Ilg, E.; Hausser, P.; Fischer, P.; Cremers, D.; Dosovitskiy, A.; Brox, T. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 4040–4048.
27. Kendall, A.; Martirosyan, H.; Dasgupta, S.; Henry, P.; Kennedy, R.; Bachrach, A.; Bry, A. End-to-end learning of geometry and context for deep stereo regression. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 66–75.
28. Eigen, D.; Puhrsch, C.; Fergus, R. Depth map prediction from a single image using a multi-scale deep network. *Adv. Neural Inf. Process. Syst.* **2014**, 2366–2374.

29. Liu, F.; Shen, C.; Lin, G.; Reid, I. Learning depth from single monocular images using deep convolutional neural fields. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *38*, 2024–2039. [CrossRef] [PubMed]
30. Kocabas, M.; Karagoz, S.; Akbas, E. Multiposenet: Fast multi-person pose estimation using pose residual network. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 417–433.
31. Strezoski, G.; Noord, N.v.; Worring, M. Many Task Learning With Task Routing. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Seoul, Korea, 27–28 October 2019.
32. Kundu, J.N.; Lakkakula, N.; Babu, R.V. UM-Adapt: Unsupervised Multi-Task Adaptation Using Adversarial Cross-Task Distillation. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Long Beach, CA, USA, 15–20 June 2019.
33. Jha, A.; Kumar, A.; Banerjee, B.; Chaudhuri, S. Adamt-net: An adaptive weight learning based multi-task learning model for scene understanding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 3027–3035.
34. Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; Schiele, B. The Cityscapes Dataset for Semantic Urban Scene Understanding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016.
35. Liu, W.; Rabinovich, A.; Berg, A.C. Parsenet: Looking wider to see better. *arXiv* **2015**, arXiv:1506.04579.
36. Yu, F.; Koltun, V. Multi-scale context aggregation by dilated convolutions. *arXiv* **2015**, arXiv:1511.07122.
37. Zbontar, J.; LeCun, Y. MStereo matching by training a convolutional neural network to compare image patches. *J. Mach. Learn. Res.* **2016**, *17*, 2287–2318.
38. Chollet, F. Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE Conference on Computer vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1251–1258.
39. Xie, S.; Girshick, R.; Dollár, P.; Tu, Z.; He, K. Aggregated residual transformations for deep neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1492–1500.
40. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A.A. Inception-v4, inception-resnet and the impact of residual connections on learning. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017.