Auxiliary Tasks in Multi-task Learning
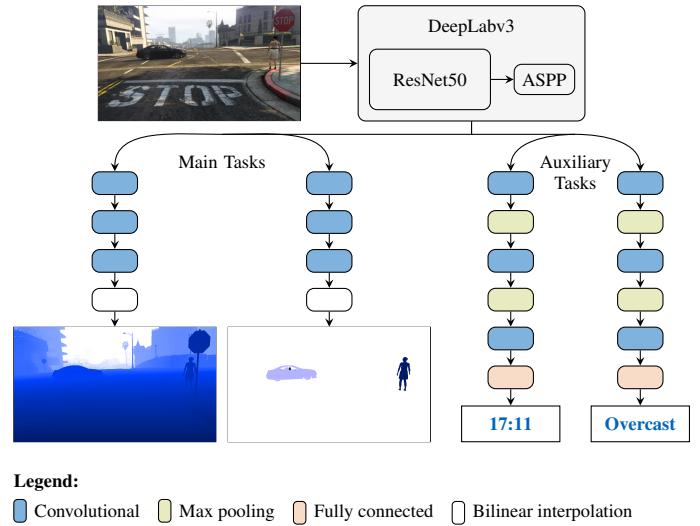
Lukas Liebel   Marco Körner

Computer Vision Research Group, Chair of Remote Sensing Technology
Technical University of Munich, Germany
{lukas.liebel, marco.koerner}@tum.de

Multi-task convolutional neural networks (CNNs) have shown impressive results for certain combinations of tasks, such as single-image depth estimation (SIDE) and semantic segmentation. This is achieved by pushing the network towards learning a robust representation that generalizes well to different atomic tasks. We extend this concept by adding auxiliary tasks, which are of minor relevance for the application, to the set of learned tasks. As a kind of additional regularization, they are expected to boost the performance of the ultimately desired main tasks. To study the proposed approach, we picked vision-based road scene understanding (RSU) as an exemplary application. Since multi-task learning requires specialized datasets, particularly when using extensive sets of tasks, we provide a multi-modal dataset for multi-task RSU, called synMT. More than $2.5 \cdot 10^5$ synthetic images, annotated with 21 different labels, were acquired from the video game Grand Theft Auto V (GTA V). Our proposed deep multi-task CNN architecture was trained on various combination of tasks using synMT. The experiments confirmed that auxiliary tasks can indeed boost network performance, both in terms of final results and training time.

1 Introduction

Various applications require solving several *atomic tasks* from the *computer vision* domain using a single image as input. Such basic tasks often include *single-image depth estimation (SIDE)* [9, 10, 19, 20], *semantic segmentation* [1, 4, 6, 21], or *image classification* [14, 30, 31]. While each task is traditionally tackled individually, close connections between them exist. Exploiting those by solving them jointly can increase the performance of each individual task and save time during training and inference. *Multi-task* learning, *i.e.*, the concept of learning several outputs from a single input simultaneously, was applied to numerous tasks and techniques, including *artificial neural network* architectures with hard parameter sharing, within the last decades [2, 3, 29]. *Convolutional neural network (CNN)* architectures for multi-task learning were proven successful, especially for the combination of *object detection* and semantic segmentation [13, 34], as well as SIDE and semantic segmentation [9, 16]. In order to combine the contributing single-task *loss functions* to a final multi-task loss, subject to optimization, Kendall et al. [16] very recently proposed an approach for learning a weighting between them, in order to address their unequal properties and behaviors.

Building upon these concepts, we introduce *auxiliary tasks* to multi-task setups. Apart from the *main tasks*, which are the ultimately required output for an application, auxiliary tasks serve solely for learning a rich and robust common representation of an image. By choosing tasks that are easy to learn and support the main tasks, this helps to boost the system during the training

Figure 1: Schematic overview of a first implementation of the proposed concept for the utilization of auxiliary tasks in multi-task learning applied to *vision-based road scene understanding (RSU)*.

stage. A similar concept was proposed by Romera-Paredes et al. [26]. Contrary to their notion of *principal* and auxiliary task, we define auxiliary tasks as *seemingly* unrelated tasks that in fact base on similar features as the main tasks. We, therefore, neither explicitly model them as different groups, nor impose any penalties to encourage orthogonal representations.

In order to analyze our concept, we have chosen vision-based

RSU as an exemplary application. *Advanced driver assistance systems (ADAS)* and *autonomous vehicles* need to gather information about the surrounding of the vehicle, in order to be able to safely guide the driver or vehicle itself through complex traffic scenes. Apart from traffic signs and lane markings, typical components of such scenes that need to be considered are other road users, *i.e.*, primarily vehicles and pedestrians. When it comes to decisionmaking, additional important local or global parameters will certainly be taken into account. Those include, *i.a.*, object distances and positions, or the current time of day [22] and weather conditions [28, 33]. The application of vision-based methodology seems natural in the context of RSU, since the rules of the road and signage were designed for humans who mainly rely on visual inspection in this regard. Figure 1 shows an overview of our network architecture and illustrates the concept of auxiliary tasks, with SIDE and semantic segmentation serving as main tasks and the estimation of the time of day and weather conditions as auxiliary tasks.

Deep learning techniques, which represent the state of the art in virtually all of the aforementioned sub-tasks of RSU, heavily rely on the availability of large amounts of training data. The availability of specialized datasets, such as KITTI [12] and Cityscapes [7], facilitated CNN-based visual RSU. For multi-task learning, annotated datasets with various labels are required. Manual annotation of training data is, however, a time-consuming and hence expensive task. Moreover, it is hard to ensure that all possibly occurring phenomena and their numerous combinations are covered by the training data. Some phenomena cannot be covered by real data at all. Typical example for such scenes in the context of *autonomous driving* are traffic accidents, which are of great interest for solving this task but luckily happen rarely. Intentionally causing those situations is often only possible at great expenses or even outright impossible.

As the development of techniques for creating near photorealistic *synthetic images* in the field of *computer graphics* rapidly advances, a growing interest in *synthetic training data* arose in recent years. Simulations allow for the acquisition of virtually unlimited amounts of automatically labeled training data with relatively little effort. In their pioneering work, Richter et al. [24] proposed a method for the propagation of manual annotations to following frames of a sequence using additional information extracted from intermediate rendering steps of the popular video game *Grand Theft Auto V (GTA V)*. Following publications proposed methods for the fully-automatic generation of training data from video games [15, 25] and custom simulators based on game engines [23] or procedural models [32]. Widespread synthetic datasets for visual RSU include Virtual KITTI [11], SYNTHIA [27], and CARLA [8].

Based on the current state of the art, we discuss the concept of introducing auxiliary tasks to multi-task setups in the remainder of this paper. Our main contributions are: i.) A novel concept for boosting training and final performance of multi-task CNNs by utilizing seemingly unrelated auxiliary tasks for regularization, ii.) a first implementation and study of this concept in the context of vision-based multi-task RSU, iii.) a synthetic dataset,

called *synMT*, for RSU containing a multitude of multi-modal ground-truth labels[1], and iv.) a multi-task CNN architecture building upon the work of Chen et al. [4] and Kendall et al. [16].

2 Introducing Auxiliary Tasks to Multi-task Learning

The general idea of multi-task learning is to find a common representation in the earlier layers of the network, while the individual tasks $\tau \in \mathcal{T}$ are solved in their respective single-task branches in the later stages of the network. This is most commonly realized as an *encoder-decoder* structure, in which each atomic task represents a specialized *decoder* to the representation provided by the common *encoder*. While each type of label $\boldsymbol{y}_{\mathcal{T}} = (y_{\tau_1}, y_{\tau_2}, \dots) \in \mathcal{Y}_{\mathcal{T}}$ favors the learning of certain features in the common part, some of them can be exploited by other tasks as well. This structure can thus help to boost the performance of the atomic tasks.

We extend this concept by adding auxiliary tasks to $\mathcal{T}$. We refer to auxiliary tasks as atomic tasks that are of minor interest or even irrelevant for the application. Despite being seemingly unrelated, they are expected to assist in finding a rich and robust representation of the input data $\boldsymbol{x}$ in the common part, from which the ultimately desired main tasks profit. Auxiliary tasks should be chosen, such that they are easy to be learned and use labels that can be obtained with low effort, *e.g.*, global descriptions of the scene. By forcing the network to generalize to even more tasks, learning auxiliary tasks restricts the parameter space during optimization. Since, by design, auxiliary tasks are simple, robust, and uncorrelated with the main tasks to a certain extent, they can thus be regarded as a regularizer.

In order to be able to optimize a multi-task network for the learnable parameters $\boldsymbol{\omega}_{\mathcal{T}} = (\boldsymbol{\theta}_{\mathcal{T}})$, which in this case only consist of the network parameters $\boldsymbol{\theta}_{\mathcal{T}}$, a multi-task loss function $\mathrm{L}_{\mathcal{T}}(\boldsymbol{x}, \boldsymbol{y}_{\mathcal{T}}, \boldsymbol{y}'_{\mathcal{T}}; \boldsymbol{\omega}_{\mathcal{T}})$, comparing ground-truth labels $\boldsymbol{y}_{\mathcal{T}}$ to predictions $\boldsymbol{y}'_{\mathcal{T}}$, has to be designed. The final multi-task loss, subject to optimization, is a combination of the single-task losses. Since each of the contributing single-task loss functions $\mathrm{L}_{\tau}(\boldsymbol{x}, \boldsymbol{y}_{\tau}, \boldsymbol{y}'_{\tau}; \boldsymbol{\omega}_{\tau})$ may behave differently, weighting each with a factor c_{τ} is essential. This yields a combined loss function

$$\mathrm{L}_{\mathrm{comb}}(\boldsymbol{x}, \boldsymbol{y}_{\mathcal{T}}, \boldsymbol{y}'_{\mathcal{T}}; \boldsymbol{\omega}_{\mathcal{T}}) = \sum_{\tau \in \mathcal{T}} \mathrm{L}_{\tau}(\boldsymbol{x}, y_{\tau}, y'_{\tau}; \boldsymbol{\omega}_{\tau}) \cdot c_{\tau} \quad . \quad (1)$$

Instead of manually tuning c_{τ} to account for the differing variances and offsets amongst the single-task losses, the coefficients can be added to the learnable network parameters $\boldsymbol{\omega}_{\mathcal{T}} = (\boldsymbol{\theta}_{\mathcal{T}}, \boldsymbol{c}_{\mathcal{T}})$. This, however, requires augmenting $\mathrm{L}_{\mathrm{comb}}$ with a regularization term $\mathrm{R}(c_{\tau})$ to avoid trivial solutions. We adapt the regularization term $\mathrm{R}_{\log}(c_{\tau}) = \log(c_{\tau}^2)$ suggested by Kendall et al. [16] slightly to $\mathrm{R}_{\mathrm{pos}}(c_{\tau}) = \ln(1 + c_{\tau}^2)$ in order to enforce positive regularization values. Thus, decreasing c_{τ} to $c_{\tau}^2 < 1$ no longer yields negative loss values. We consider

---

[1] The synMT dataset and source code utilized to acquire it are publicly available at http://www.lmf.bgu.tum.de/en/synmt

these considerations in our final multi-task loss

$$L_{\mathcal{T}}(x, y_{\mathcal{T}}, y'_{\mathcal{T}}; \omega_{\mathcal{T}}) = \sum_{\tau \in \mathcal{T}} \frac{1}{2 \cdot c_\tau^2} \cdot L_\tau(x, y_\tau, y'_\tau; \omega_\tau) \\ + \ln\left(1 + c_\tau^2\right) \quad . \tag{2}$$

3 Experiments and Results

For an experimental evaluation of the approach presented in Section 2, we chose vision-based RSU as an exemplary application. In RSU, various tasks have to be solved simultaneously, making it a prime example for multi-task approaches. Multi-task training requires several different annotations for each image, especially when additional labels for auxiliary tasks are needed, and an overall large number of samples. Since no suitable dataset was available, we compiled a dataset, called synMT, for our experiments ourselves. This dataset will be made available to the public and is thus described in detail in the following section. Utilizing synMT, we conducted experiments on multi-task training with auxiliary tasks. A description of the used network architecture and the conducted experiments is given in Section 3.2, followed by a discussion of the results in Section 3.3.

3.1 A Synthetic Dataset for Multi-task Road Scene Understanding

Datasets for RSU, such as KITTI [12] and Cityscapes [7] exist but provide few annotations per image. Often, manual annotation is not feasible without major effort. In contrast to this, the simulation of traffic scenes enables fully-automatic labeling but is a sophisticated task which requires extensive knowledge from the field of computer graphics. The video game industry invests heavily in creating photorealistic renderings and dynamic simulations, especially for games from the *open world* genre, which allow the player to freely explore a virtual world. Making use of techniques developed in this field, synthetic datasets for RSU have been presented [8, 11, 24, 25, 27]. Not only does simulation allow to freely set all parameters of the scene, it also facilitates the creation of error-free labels.

GTA V is the latest title in a series of open world video games that provide the player with a realistic environment set in modern-time USA. We decided upon using GTA V as the source for our dataset, primarily because of its state-of-the-art graphics and the availability of prior work regarding the extraction of annotations. The game is set on a large island with urban areas, countryside, and highways. Figure 2 shows a map of the accessible area of GTA V, which outranks other simulation sceneries, such as the virtual city model of SYNTHIA [27], by far. The player is able to take part in the simulated traffic with various vehicles.

In order to acquire data in the most realistic way possible, we set up a simulator consisting of a gaming PC and likewise periphery. The computer features an NVIDIA GeForce GTX 1080 GPU, complemented by an Intel i7-7700 CPU and a PCIe NVMe SSD. This allows running state-of-the-art video games

(a) GTA V (Images: Rockstar Games)

(b) SYNTHIA (Image: Ros et al. [27])

Figure 2: (a) The accessible area in GTA V with a map of the whole island (left) and a detail of the main city, given as a simulated satellite image (right) in comparison to (b) the city area of SYNTHIA [27].

in high resolution and quality while recording data with a high framerate simultaneously. To enhance the driving experience and thus enable our drivers to steer their car as naturalistic as possible, we equipped our machine with a Logitech G920 force-feedback steering wheel and pedals. We instructed the drivers to obey the rules of the road and adapt to the simulated traffic.

The acquired ground-truth labels can be divided into two groups according to the used method for acquisition. Pixel-wise depth maps and semantic masks, as required for tasks such as SIDE and semantic segmentation, were extracted from the respective buffers using DirectX functions. Labels for all other tasks were obtained by utilizing the Script Hook V[2] library, which enables access to metadata about the current scene. Images and labels were captured with a resolution of 1028×720 px at a framerate of 15 fps. Table 1 shows an overview of the recorded data. Note that for the experiments conducted for this work, we only made use of the depth maps, pixel-accurate masks for cars and pedestrians, the timestamp, as well as the weather conditions. The numerous and various images and ground-truth annotations provided by synMT are a precious data source for expanding the presented multi-task approach. They can furthermore serve as training data for entirely different tasks, such as video frame prediction.

In order to obtain disjoint subsets of samples for training and testing, the map area was split into non-overlapping areas. A 2D-histogram over the position of all collected samples with

---

[2]Alexander Blade, http://www.dev-c.com/gtav/scripthookv, 2018.

Table 1: Contents of the proposed synthetic dataset synMT for multi-task RSU, acquired in the video game GTA V, illustrated by a single example. The dataset consists of more than $2.5 \cdot 10^5$ annotated images with a resolution of $1028 \times 720\,\mathrm{px}$, sampled at 15 fps.
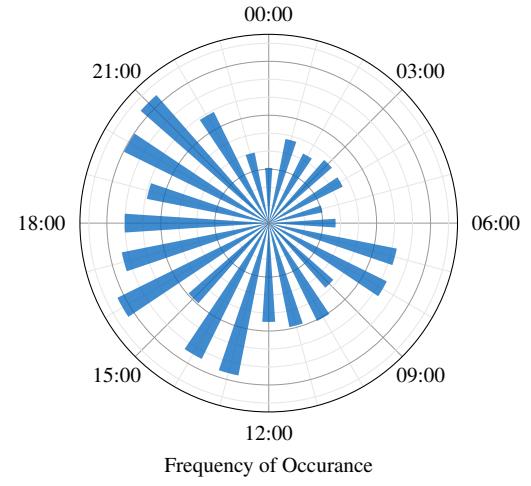

(a) RGB	(b) Depth Map	(c) Semantic Masks

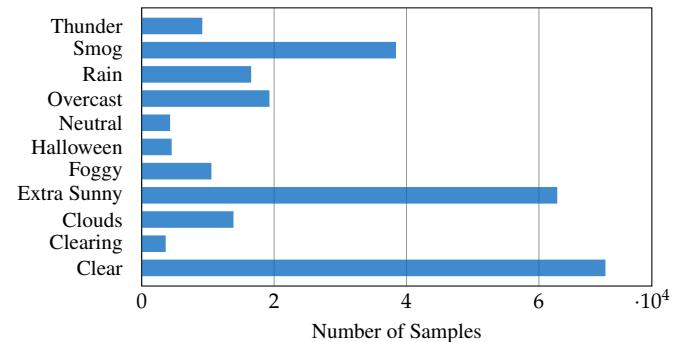| Vehicle:		Car	Time of Day:		09:11:21	Steering:		$-67.5°$
Model:		Asea	Weather:		Clear	Throttle:		0.08
Location:	-1049.59	(N)	Rain Level:		0	Brake:		0
	-779.87	(E)	Snow Level:		0	Gear:		2nd
	18.76	(h)	Wind Speed:	16.95 $\frac{\mathrm{km}}{\mathrm{h}}$		Speed:		24.16 $\frac{\mathrm{km}}{\mathrm{h}}$
Heading:	2.34	(Pitch)	Wind Direction:	0.0	(N)	Velocity Vector:		-5.35 (Pitch)
	-1.21	(Roll)		-1.0	(E)			4.06 (Roll)
	54.55	(Yaw)		0.0	(h)			2.62 (Yaw)
Past Events:		315 s ago (Driven in Wrong Lane),		10 815.02 s ago (Crash, Car),		89 018.60 s ago (Crash, Pedestrian)		

(d) Other Labels

a bin size of approximately $65 \times 65\,\mathrm{m}^2$ was computed to determine areas with available samples. A test set was compiled by randomly selecting 100 bins from the dataset and choosing a subset of samples from them. The test areas were buffered with a width of approximately 65 m in order to clearly separate test and training areas. Remaining samples, which overlap with neither the test areas nor their respective buffers, were kept for training. The final subsets for training and testing consist of $2.5 \cdot 10^5$ and 10^3 samples, respectively. Figure 3 shows the distribution of the two auxiliary labels used for the experiments described in Section 3.3 for the training set. Note that the subset sampled from the test areas was designed to show similar statistical characteristics.

3.2 Network Architecture and Training

In order to study the proposed approach, we chose the commonly used tasks SIDE and semantic segmentation as our main tasks. By studying the clues on which the predictions for the main tasks are based upon, tasks that encourage the network to learn features that may alleviate possible pitfalls can be found. Methods for SIDE have been found to largely rely on image gradients as features [18]. The time of day, especially different lighting conditions, distort them severely. By adding a *regression* for the current *time of day* as an auxiliary task, we push the network towards learning to differentiate between gradients caused by geometry and lighting. Methods for semantic segmentation need to clearly separate between actual objects that are expected to be labeled as such, and noise, *i.a.*, in the form of raindrops, snow, or mirror images. Furthermore, objects may change appearance depending on the visibility conditions. We, therefore, added the *classification* of *weather conditions* as a second auxiliary task to force the network to explicitly learn

(a) Time of Day

(b) Weather Conditions

Figure 3: Distribution of samples over (a) time and (b) weather conditions in the training set of synMT.

Table 3: Performance on the synMT test set after optimization in different single- and multi-task configurations.

τ_1 – Semantic Segmentation MIoU	τ_2 – Depth Estimation RMSE of r(d) (in m)	τ_3 – Time Estimation RMSCTD (in min)	τ_4 – Weather Classification Accuracy
74.28%	—	—	—
—	0.0826	—	—
—	—	83	—
—	—	—	79.90%
72.83%	0.0671	—	—
71.85%	0.1132	436	—
65.58%	0.05704	—	70.15%
70.42%	0.1793	110	69.82%

features that are useful for identifying this influence. Both tasks were optimized by single-task training in preliminary experiments and could be proven to converge to good results after moderate training time.

Since the encoder part of the network primarily serves as a feature extractor, pre-training is feasible and reasonable. State-of-the-art CNN architectures for *image recognition*, such as *residual networks (ResNets)* [14], are well suited as a basis for this part. The goal of image recognition, however, is to solve a global classification task. Hence, a modification is necessary in order to obtain pixel-wise outputs as desired for semantic segmentation and SIDE. The concept of *fully-convolutional networks (FCNs)* [21] enables transformation. One instance of such architectures is *DeepLabv3* [4], which uses a ResNet as its feature extractor and employs *atrous spatial pyramid pooling (ASPP)* [5] to retain spatial information at different scales. We utilized a re-implementation of this architecture with a pre-trained ResNet50 and replaced the final 1×1 *convolutional* layer with task-specific decoders, as suggested by Kendall et al. [16]. An overview of the architecture is shown in Figure 1.

For each of the SIDE and semantic segmentation decoders, we employ two 3×3 convolutional layers with *rectified linear unit (ReLU)* activations, followed by a 1×1 convolution and bilinear interpolation. The weather classification branch consists of a ReLU-activated 5×5 convolutional layer, two 3×3 *max pooling* layers with stride 3, and a 3×3 convolutional layer with ReLU activation between them, followed by a 1×1 convolution and a *fully connected* layer with 11 neurons, corresponding to the number of classes for this label. The time regression branch is identical to the weather classification branch except for the first max pooling layer, with a size of 5×5 and a stride of 5. As a scalar output is expected, the final fully connected layer features only one neuron.

For each atomic task $\tau_i \in \mathcal{T}$, an appropriate loss function, computed using the corresponding ground-truth label y_τ, was selected or designed. The commonly used combination of *softmax* and *cross entropy* functions was employed for the pixel-wise and scalar classification tasks. The pixel-wise and scalar regression tasks were assessed by modified *mean squared error (MSE)* functions. For the analysis of traffic scenes, the relative distance of closer objects is much more relevant than that of objects far away from the vehicle. Therefore, we scaled the ground-truth

depth non-linearly, such that instead of the absolute distance d from the sensor, we estimated a scaled range

$$\text{r}(d) = 1 - \frac{\log(d)}{\log(1000)} \quad . \tag{3}$$

By applying this mapping, relevant depth values in the region of 1 m to 1 km were logarithmically scaled to $[0, 1]$. We additionally clipped all values with $d < 1\,\text{m}$ and $d > 1000\,\text{m}$ to the minimum of $\text{r}(d) = 0$ and maximum of $\text{r}(d) = 1$, respectively. By optimizing the MSE of $\text{r}(d)$, the network is encouraged to predict closer ranges with high precision while tolerating large errors in further distances.

When calculating the difference between a predicted point in time t' and the respective ground-truth t, each given as the minute of the day, the periodicity of the values has to be considered. We calculated a *squared cyclic time difference*

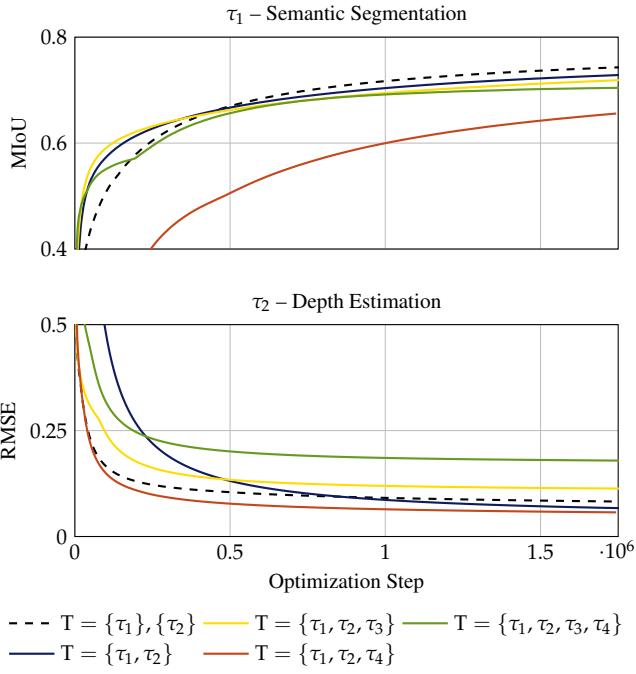$$\text{sctd}(t, t') = \min\{(t - t')^2, (t - t' + 1440)^2, \\ (t - t' - 1440)^2\} \tag{4}$$

and minimized the single-task loss

$$\text{L}_{\text{time}}(x, y_{\text{time}}, y'_{\text{time}}; \omega_{\text{time}}) = \frac{\text{sctd}(y_{\text{time}}, y'_{\text{time}})}{b} \cdot 10^{-5} \tag{5}$$

given a *mini batch* of size b. Note that scaling the values by 10^{-5} brings them down to more common loss values and was solely done for convenience regarding the implementation.

Training was conducted using an implementation of the network architecture described in Section 2 and synMT, introduced in Section 3.1. In each experiment, a different combination of tasks was considered. Namely the four selected individual tasks as single-task reference ($\mathcal{T} = \{\tau_1\}, \{\tau_2\}, \{\tau_3\}, \{\tau_4\}$), the combination of all four ($\mathcal{T} = \{\tau_1, \tau_2, \tau_3, \tau_4\}$), the well-established combination of SIDE and semantic segmentation ($\mathcal{T} = \{\tau_1, \tau_2\}$) as well as this traditional combination, enhanced by either weather classification or time regression ($\mathcal{T} = \{\tau_1, \tau_2, \tau_3\}$ and $\mathcal{T} = \{\tau_1, \tau_2, \tau_4\}$).

The loss weighting variables c_τ were initialized with a value of $|\mathcal{T}|^{-1} = 0.25$ for all tasks τ in each experiment. The spatial pyramid pooling rates were set to 1, 2, and 4, yielding a final output stride of 16 before interpolation. Since the deep
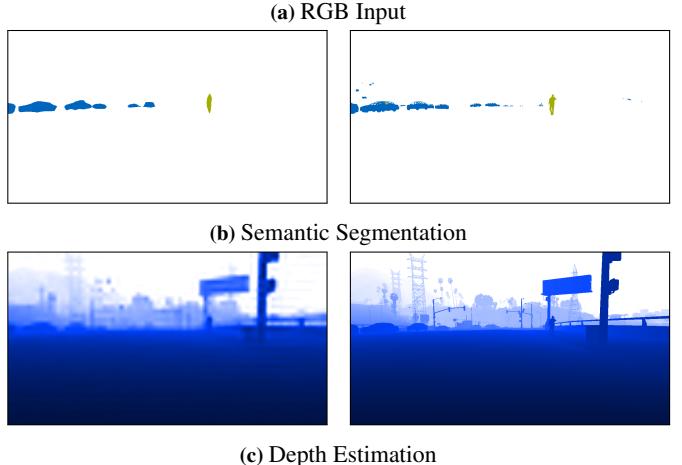
Figure 4: Performance for intermediate snapshots of the network parameters, showing the influence of multi-task learning as a regularization measure by yielding faster convergence. Optimization was conducted for $1.75 \cdot 10^6$ iterations (28 epochs) in 12 days on an NVIDIA DGX-1v GPU server.

(a) RGB Input

(b) Semantic Segmentation

(c) Depth Estimation

Figure 5: Prediction results (left) and ground truth (right) for (b) semantic segmentation and (c) depth estimation, derived from an RGB input image (a). The predictions illustrate the good performance of the network, despite the large output stride of 16.

multi-task network in conjunction with large input images is demanding in GPU memory, a small batch size of $b = 4$ was utilized for optimization with a maximum *learning rate* of $\alpha = 10^{-5}$ for the *Adam* optimizer [17].

We performed optimization on an NVIDIA DGX-1v server, equipped with eight NVIDIA Tesla V100 (16 GB) GPUs, for approximately 12 days. Using the NVIDIA DGX-1v *(Volta)* over an NVIDIA DGX-1 *(Pascal)* decreased the computation time per iteration by approximately 20%. Note that since no parallelization was implemented, each GPU was used for a different experiment. Hence, optimization of the network could be conducted on a single NVIDIA Tesla V100 in roughly the same time, or an NVIDIA Tesla P100 and even NVIDIA Titan X (Pascal) with a minor increase in training time.

3.3 Results and Discussion

The performance of the network was evaluated using the test set and an interpretable error metric for each task, *i.e.*, the *root mean squared error (RMSE)* for SIDE, *mean intersection over union (MIoU)* for semantic segmentation, accuracy for weather prediction, and the *root mean squared cyclic time difference (RMSCTD)* for time regression. A summary of the results is given in Table 3, which shows the single-task baseline in the first part and the multi-task results for different $\mathcal{T}$ in the second part. As expected, the auxiliary tasks achieve high results overall with the best performance in the single-task settings. Since they were solely added to support the network optimization during the training phase, the results are not further discussed here. Note that the single notably higher value for τ_3 in $\mathcal{T} = \{\tau_1, \tau_2, \tau_3\}$

was caused by a high weighting coefficient c_{τ_3} that diverged in the early training stages and could not be recovered from an unfavorable but apparently stable state. In our experiments, we observed that the coefficients did not necessarily converge to similar values at each re-initialization, as reported by Kendall et al. [16].

While semantic segmentation, in general, did not profit from any multi-task setup in our experiments, the best depth prediction was achieved when optimizing for $\mathcal{T} = \{\tau_1, \tau_2, \tau_4\}$, *i.e.*, adding weather classification to the main tasks. The results for $\mathcal{T} = \{\tau_1, \tau_2, \tau_3, \tau_4\}$, *i.e.*, both main tasks and auxiliary tasks, suggest furthermore that it is possible to learn a common representation that generalizes to four different tasks. The achieved moderate performance, however, reveals that $\mathcal{T}$, especially the auxiliary tasks, has to be chosen carefully. Further experiments should be conducted in order to find auxiliary tasks that support the main tasks desired for a selected application more effectively. However, our first experiment showed that even seemingly unrelated auxiliary tasks can improve the results of main tasks.

In order to study additional benefits of an extended multi-task setup, we evaluated the network performance at intermediate optimization states. Figure 4 shows the results of the main tasks for each $\mathcal{T}$. As expected, auxiliary tasks serve as regularization and thus facilitate faster convergence. This became especially apparent for SIDE, where $\mathcal{T} = \{\tau_1, \tau_2, \tau_3\}, \{\tau_1, \tau_2, \tau_4\}$ converged considerably faster than the traditional multi-task setup

$\mathcal{T} = \{\tau_1, \tau_2\}$. For semantic segmentation, the single-task setup yielded best results but converged slowly.

Samples from synMT with predictions and gound-truth are shown in Figure 5. The influence of the high output stride, which requires interpolation with a factor of 16, is clearly visible in the pixel-wise outputs of the main tasks.

4 Conclusion

We presented a new concept for improving multi-task learning by adding seemingly unrelated auxiliary tasks to the set of learned tasks in order to improve the performance of the ultimately desired main tasks. Auxiliary tasks should be reasonably easy to learn and utilize annotations that require little effort to acquire. Due to their properties, they can, thus, serve as a regularization measure and improve performance, robustness, and training speed of the network.

To study our proposed approach, we applied the concept to vision-based RSU using our new synthetic multi-task dataset synMT and an extended state-of-the-art CNN architecture. The results showed that auxiliary tasks can indeed boost the performance of the main tasks. Several combinations of tasks were studied and the reported influence on the results can serve as a starting point for the evaluation of application-specific multi-task systems. Since the correlation between pairs and sets of main and auxiliary task is expected to be alike in synthetic and real environments, drawn conclusions can be applied to the design of real-world data acquisition campaigns and networks.

Acknowledgements

References

[1] V. Badrinarayanan, A. Kendall, and R. Cipolla. SegNet: A deep convolutional encoder-decoder architecture for image segmentation. *Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495, 2017.

[2] R. Caruana. Multitask learning: A knowledge-based source of inductive bias. In *International Conference on Machine Learning*, pages 41–48, 1993.

[3] R. Caruana. *Learning to learn*, chapter Multitask Learning, pages 95–133. 1998.

[4] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv:1706.05587*, 2017.

[5] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2018.

[6] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. *arXiv:1802.02611*, 2018.

[7] 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 *Conference on Computer Vision and Pattern Recognition*, pages 3213–3223, 2016.

[8] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. CARLA: An open urban driving simulator. In *Annual Conference on Robot Learning*, pages 1–16, 2017.

[9] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *International Conference on Computer Vision*, pages 2650–2658, 2015.

[10] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *International Conference on Neural Information Processing Systems*, pages 2366–2374, 2014.

[11] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig. Virtual worlds as proxy for multi-object tracking analysis. In *Conference on Computer Vision and Pattern Recognition*, pages 4340–4349, 2016.

[12] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The KITTI dataset. *International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[13] R. Girshick. Fast R-CNN. In *International Conference on Computer Vision*, pages 1440–1448, 2015.

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[15] M. Johnson-Roberson, C. Barto, R. Mehta, S. N. Sridhar, K. Rosaen, and R. Vasudevan. Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? In *International Conference on Robotics and Automation*, pages 746–753, 2017.

[16] A. Kendall, Y. Gal, and R. Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Conference on Computer Vision and Pattern Recognition*, 2018. (to appear).

[17] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, pages 1–15, 2015.

[18] T. Koch, L. Liebel, F. Fraundorfer, and M. Körner. Evaluation of CNN-based single-image depth estimation methods. *arXiv:1805.01328*, 2018.

[19] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab. Deeper depth prediction with fully convolutional residual networks. In *International Conference on 3D Vision*, pages 239–248, 2016.

[20] F. Liu, C. Shen, and G. Lin. Deep convolutional neural fields for depth estimation from a single image. In *Conference on Computer Vision and Pattern Recognition*, pages 5162–5170, 2015.

[21] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[22] W.-C. Ma, S. Wang, M. A. Brubaker, S. Fidler, and R. Urtasun. Find your way by observing the sun and other semantic cues. In *International Conference on Robotics and Automation*, pages 6292–6299, 2017.

[23] M. Müller, V. Casser, J. Lahoud, N. Smith, and B. Ghanem. Sim4CV: A photo-realistic simulator for computer vision applications. *International Journal of Computer Vision*, pages 1–18, 2018.

[24] S. R. Richter, V. Vineet, S. Roth, and V. Koltun. Playing for data: Ground truth from computer games. In *European Conference on Computer Vision*, pages 102–118, 2016.

[25] S. R. Richter, Z. Hayder, and V. Koltun. Playing for benchmarks. In *International Conference on Computer Vision*, pages 2213–2222, 2017.

[26] B. Romera-Paredes, A. Argyriou, N. Berthouze, and M. Pontil. Exploiting unrelated tasks in multi-task learning. In *International Conference on Artificial Intelligence and Statistics*, pages 951–959, 2012.

[27] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Conference on Computer Vision and Pattern Recognition*, pages 3234–3243, 2016.

[28] M. Roser and F. Moosmann. Classification of weather situations on single color images. In *Intelligent Vehicles Symposium*, pages 798–803, 2008.

[29] S. Ruder. An overview of multi-task learning in deep neural networks. *arXiv:1706.05098*, 2017.

[30] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014.

[31] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[32] A. Tsirikoglou, J. Kronander, M. Wrenninge, and J. Unger. Procedural modeling and physically based rendering for synthetic data generation in automotive applications. *arXiv:1710.06270*, 2017.

[33] X. Yan, Y. Luo, and X. Zheng. Weather recognition based on images captured by vision system in vehicle. In *Advances in Neural Networks – ISNN 2009*, pages 390–398, 2009.

[34] J. Yao, S. Fidler, and R. Urtasun. Describing the scene as a whole: Joint object detection, scene classification and semantic segmentation. In *Conference on Computer Vision and Pattern Recognition*, pages 702–709, 2012.