Supplementary Material: Multi-Loss Weighting with Coefficient of Variations

Rick Groenendijk[1] Sezer Karaoglu[3,1] Theo Gevers[1,3] Thomas Mensink[2,1]
[1] University of Amsterdam [2] Google Research [3] 3DUniversum
{r.w.groenendijk, th.gevers}@uva.nl s.karaoglu@3duniversum.com mensink@google.com

1. Implementation Details

1.1. Depth

Depth estimation can be realised by using a left-right image pair attempting to reconstruct one from the other. When frames are rectified, images can be mapped to another image using a disparity map. The resulting disparity map can be used to estimate depth by means of the camera parameters. In their work [9] propose to estimate depth without explicit depth ground truth supervision, but purely using reconstruction. A model receives the left image I^l as an input, and is tasked with predicting left-to-right disparity d^r and right-to-left disparity d^l, which are used to generate reconstructed right image $\hat{I}^r$ and reconstructed left image $\hat{I}^l$ respectively.

The quality of the reconstructed images is evaluated by means of a number of loss functions which form the full training objective. Each loss function has distinct optimization properties. The following loss functions are combined in line with [5]:

- The L1 loss that minimises the per-pixel distance:

$$\mathcal{L}^l_{L1} = \frac{1}{\mathrm{N}} \sum_{i,j} \|\mathrm{I}^l_{ij} - \hat{\mathrm{I}}^l_{ij}\| \tag{1}$$

- The Structural Similarity Index Measure (SSIM) that measures the perceived quality of the reconstruction [15]:

$$\mathcal{L}^l_S = \frac{1}{\mathrm{N}} \sum_{i,j} \frac{(1 - \mathrm{SSIM}(\mathrm{I}^l_{ij}, \hat{\mathrm{I}}^l_{ij}))}{2} \tag{2}$$

$$\mathrm{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}, \tag{3}$$

  where illumination μ and signal contrast σ are computed around centre pixels x and y. Parameters c_1, c_2 are set close to zero for numerical stability.

- The Left-Right Consistency Loss (LR) that enforces the predicted left and right disparity maps to be consistent:

$$\mathcal{L}^l_{lr} = \frac{1}{\mathrm{N}} \sum_{i,j} |d^l_{ij} - d^r_{i(j+d^l_{ij})}| \tag{4}$$

- The Disparity Smoothness Loss (DISP) that allows changes in disparities only at image edges:

$$\mathcal{L}^l_{disp} = \frac{1}{\mathrm{N}} \sum_{i,j} |\partial_x d^l_{ij}| e^{(-\|\partial_x \mathrm{I}^l_{ij}\|)} + |\partial_y d^l_{ij}| e^{(-\|\partial_y \mathrm{I}^l_{ij}\|)} \tag{5}$$

  This loss is scaled by $\frac{1}{2^s}$ at loss scale *s*, because disparity values are related to image width.

An encoder-decoder architecture is used to compress the image representation and upsample from it. In total disparity maps are regressed at four scales. At each scale all loss functions are evaluated for left and right disparity maps. The full objective combines losses from all scales *s*:

$$\mathcal{L}_{full} = \sum_{s=0}^{3} \sum_{d \in \{d_{left}, d_{right}\}} \mathcal{L}_{(s,d)} \tag{6}$$

$$\mathcal{L}_{(s,d)} = \alpha_{L1}\, \mathcal{L}_{L1} + \alpha_S\, \mathcal{L}_S + \alpha_{lr}\, \mathcal{L}_{lr} + \alpha_{disp}\, \frac{1}{2^s} \mathcal{L}_{disp}, \tag{7}$$

where each of the αs are the loss weights that should be automatically weighted using one of the loss weighting schemes.

The full pipeline is implemented in PyTorch [13][1]. For all implemented methods and datasets, images are downsampled to a resolution of 256x512 and fed to an encoder-decoder network that uses using batch normalisation [8] and Exponential Linear Units (ELUs) [3]. The encoder is a ResNet50 [7]; the decoder alternates bilinear interpolation up-sampling and convolutional layers [5] and outputs disparities at a single scale following the ablation study of [6]. For consistency with the multi-task methods [9, 2], the decoder is extended with two more convolutional layers. Models are trained for 30 epochs on KITTI with a batch size of 8 using an Adam optimiser with a constant learning rate of *1e-4*, unless otherwise stated. Models are trained for 100 epochs on CityScapes with a batch size of 8 using an Adam optimiser with a plateau scheduling learning rate that starts at of *1e-4*, and is halved twice at epochs 30 and 40 respectively [5].

[1] Code is made publicly available at https://github.com/rickgroen/cov-weighting

For KITTI, 8 losses are used, in line with the conclusions of [6]. For CityScapes, we use 8 losses at 4 scales, for a total of 32 losses. Data augmentation is performed in online fashion throughout the training procedure in line with [6]. Like [5] disparities are post-processed during inference and warped to depth estimates which are used for evaluation. For CityScapes evaluation is done directly on disparities, as in [14, 9, 10].

1.2. Semantics

For semantic segmentation the implementation by the authors of EncNet[2] [16] is adapted and augmented with all loss weighing methods. The full network follows an encoder-decoder scheme with dilated convolutions [1] to make predictions at 1/8 resolution. The predictions are then up-sampled to their original resolution. The encoder network is a ResNet50 [7] network with dilated convolutions [1] at the third and fourth network layers. The full network uses batch normalisation [8] and Rectified Linear Units (ReLUs) [12]. For the decoders, there is one Encoding Context Module that is attached to the final layer of the encoder, and one FCN head attached to the penultimate layer of the decoder. For optimisation, an SGD optimiser is used with polynomial learning rate scheduling with a starting learning rate of *1e-4* [16]. During training, images are cropped to 384x384 image patches and used as input to the network. Using a batch size of 8, the network can be trained on a single NVIDIA Titan GTX. Unless otherwise stated the method outlined by [16] is closely followed. The full network is pre-trained on ImageNet [4] and then trained for 40 epochs on PASCAL Context [11]. For quantitative evaluation, Pixel Accuracy (pACC) and Mean Intersection over Union (mIoU) are used, as in [16]. Background pixels are ignored during evaluation.

References

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

[2] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *ICML*, 2018.

[3] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[5] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2017.

[2]Implementation at `https://github.com/zhanghang1989/PyTorch-Encoding`

[6] Rick Groenendijk, Sezer Karaoglu, Theo Gevers, and Thomas Mensink. On the benefit of adversarial training for monocular depth estimation. *Computer Vision and Image Understanding*, 190:102848, 2020.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[8] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[9] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*, 2018.

[10] Shikun Liu, Edward Johns, and Andrew J Davison. End-to-end multi-task learning with attention. In *CVPR*, 2019.

[11] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *CVPR*, 2014.

[12] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

[13] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[14] Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *NeurIPS*, 2018.

[15] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 2004.

[16] Hang Zhang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. Context encoding for semantic segmentation. In *CVPR*, 2018.