# Supplementary Materials

## 1. EVALUATION

Full depth metrics tables for KITTI and Eigen splits are shown in Table 1 and Table 2

## 2. MASKS ANALYSIS.

Figure 1 shows the number of features removed per layer and the original number of filters for two pruned models optimizing only task loss or both task and sparsity loss. Figure 1a shows VGG layers with masking optimized only with $L_{task}$ and interestingly the most removed features are in the deeper layers in the decoder. This aligns with PyDnet pyramidal design, where they predict the depth at different levels using shallow or lightweight decoders. The observation still holds as we apply $L_{sp}$ (Figure 1b) on the final loss where layers with the same number of filters are removed more from the decoder part rather than the encoder part. This observation gives more insight on building encoder-decoder models where usually similar number of weights of the encoder part is used for the decoder which we argue this does not need to be the case. To the best of our knowledge, there is no previous work discussing the encoder-decoder choice of design.

## 3. WEIGHTS SPARSITY VS MASKS SPARSITY.

We evaluated our method compared to training with $\ell$1-norm regularizer on the convolutional filters weights. Not only sparsifying the weights will still require extra step based on a feature importance criteria (e.g remove all weights with norm $\leq$ threshold) for the weights to be pruned and then retraining, but also it does not have some favorable properties as in our method. $\ell$1-loss on all the weights do not differentiate between weights in early layers and those on later layers. However in our $L_{mask}$ from Eq. 2, the layers with larger number of filters contribute more to the loss. It makes sense to try to compress more in wide layers as filters tend to be more redundant than thin layers. Although, a downside to this property is as training progresses and maximum compression rate for a layer is reached, the loss does not adapt to the fact that these wide layers are thinner and might have less redundant features now. This only affect the maximum possible compression rate that can be reached rather than accuracy. As a sanity check we trained VGG with $\ell$1-regularization, we set regularization weight $\lambda$ = 0.005 and found the network having hard time to converge with D1 all error reaching 50.282 on KITTI split.

## 4. REFERENCES


[1] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *CVPR*, 2017, vol. 2, p. 7.

[2] Matteo Poggi, Filippo Aleotti, Fabio Tosi, and Stefano Mattoccia, "Towards real-time unsupervised monocular depth estimation on cpu," *arXiv preprint arXiv:1806.11430*, 2018.

[3] Andreas Geiger, Philip Lenz, and Raquel Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 3354–3361.

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

[5] David Eigen and Rob Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2650–2658.

[6] Fayao Liu, Chunhua Shen, Guosheng Lin, and Ian D Reid, "Learning depth from single monocular images using deep convolutional neural fields.," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 10, pp. 2024–2039, 2016.

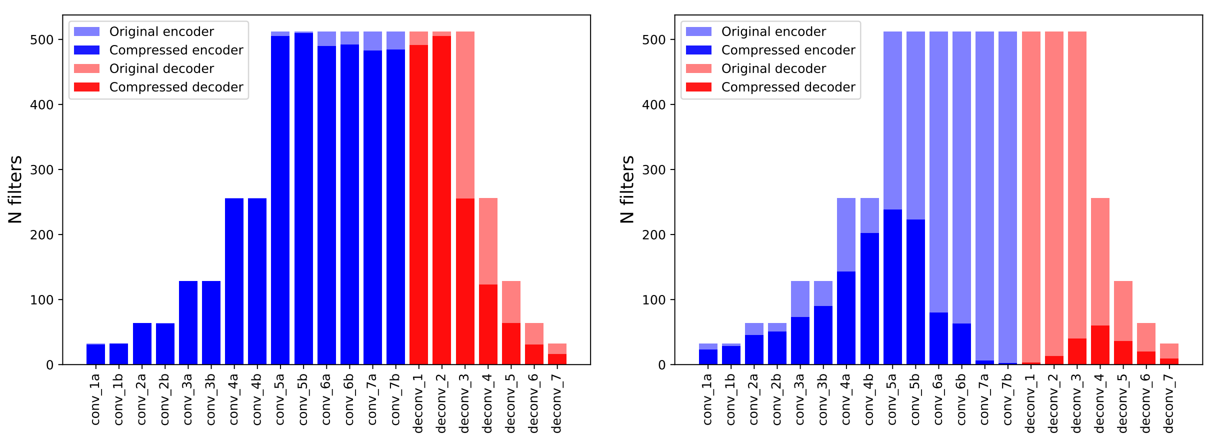
[7] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe, "Unsupervised learning of depth and ego-motion from video," in *CVPR*, 2017, vol. 2, p. 7.

| Ours | | Lower is better | | | | | Higher is better | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Method | Dataset | Abs Rel | Sq Rel | RMS | $RMS_{log}$ | D1 all | $\delta < 1.25$ | $\delta < 1.25^2$ | $\delta < 1.25^3$ | Params |
| LRC + Deep3D [1] | K | 0.151 | 1.312 | 6.344 | 0.239 | 59.64 | 0.781 | 0.931 | 0.976 | 31.6M |
| LRC + VGG [1] | K | 0.124 | 1.388 | 6.125 | 0.217 | 30.272 | 0.841 | 0.936 | 0.975 | 31.6M |
| VGG + $L_{total}$ | K | 0.1313 | 1.5296 | 6.488 | 0.230 | 32.183 $\uparrow$ 1.9 | 0.826 | 0.927 | 0.970 | 7.5M $\downarrow$ 76.1% |
| LRC + Resnet50 [1] | K | 0.1139 | 1.2661 | 5.784 | 0.204 | 28.459 | 0.853 | 0.947 | 0.979 | 58.4M |
| PyD-Net [2] | K | 0.1393 | 1.4720 | 6.570 | 0.240 | 38.478 | 0.805 | 0.919 | 0.967 | 1.9M |
| LRC + VGG [1] | CS + K | 0.104 | 1.070 | 5.417 | 0.188 | 25.523 | 0.875 | 0.956 | 0.983 | 31.6M |
| VGG + $L_{task}$ | CS + K | 0.1026 | 1.0471 | 5.366 | 0.187 | 24.939 $\downarrow$ 1.33 | 0.874 | 0.957 | 0.983 | 30.2M $\downarrow$ 4.2% |
| VGG + $L_{total}$ | CS + K | 0.1101 | 1.1562 | 5.688 | 0.196 | 26.861 $\uparrow$ 1.3 | 0.865 | 0.950 | 0.980 | 4.3M $\downarrow$ 86.1% |
| LRC + VGG pp* [1] | CS + K | 0.100 | 0.934 | 5.141 | 0.178 | 25.077 | 0.878 | 0.961 | 0.986 | 31.6M 2x forward |
| LRC + Resnet50 [1] | CS + K | 0.1009 | 1.0315 | 5.360 | 0.184 | 24.504 | 0.878 | 0.959 | 0.983 | 58.4M |

**Table 1**: Comparison of different models on KITTI 2015 stereo split. In dataset, K is [3] and CS is Cityscapes [4]. Our models prune more than 76% of the original model with maximum 1.9% drop in accuracy. D1-all score represents the percentage of pixels having a disparity error larger than 3. *pp is post-processing done by [1] but requires two forward pass. Suffix $L_x$ in our method indicates the training loss used.

| | Ours | | Lower is better | | | | Higher is better | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Method | Supervised | Dataset | Abs Rel | Sq Rel | RMS | $RMS_{log}$ | $\delta < 1.25$ | $\delta < 1.25^2$ | $\delta < 1.25^3$ | Params |
| Eigen et al. [5] | Yes | K | 0.203 | 1.548 | 6.307 | 0.282 | 0.702 | 0.890 | 0.958 | 54.2M |
| Liu et al. [6] | Yes | K | 0.201 | 1.584 | 6.471 | 0.273 | 0.680 | 0.898 | 0.967 | 40.0M |
| Zhou et al. [7] | No | K | 0.208 | 1.768 | 6.856 | 0.283 | 0.678 | 0.885 | 0.957 | 34.2M |
| LRC + VGG [1] | No | K | 0.148 | 1.344 | 5.927 | 0.247 | 0.803 | 0.922 | 0.964 | 31.6M |
| VGG + $L_{total}$ | No | K | 0.1356 | 1.3625 | 5.891 | 0.236 | 0.827 | 0.927 | 0.965 | 5.7M $\downarrow$ 81.8% |
| PyD-Net | No | K | 0.163 | 1.399 | 6.253 | 0.262 | 0.759 | 0.911 | 0.961 | 1.9M |
| Zhou et al. [7] | No | CS+K | 0.198 | 1.836 | 6.565 | 0.275 | 0.718 | 0.901 | 0.960 | 34.2M |
| LRC + VGG [1] | No | CS+K | 0.124 | 1.076 | 5.311 | 0.219 | 0.847 | 0.942 | 0.973 | 31.6 |
| VGG + $L_{task}$ | No | CS+K | 0.124 — | 1.0775 | 5.280 $\downarrow$ 0.03 | 0.219 | 0.848 | 0.942 | 0.973 | 30.8M $\downarrow$ 2% |
| VGG + $L_{total}$ | No | CS+K | 0.1452 $\uparrow$ 0.02 | 1.4024 | 5.835 $\uparrow$ 0.524 | 0.239 | 0.815 | 0.927 | 0.967 | 5.9M $\downarrow$ 81.1% |
| LRC + ResNet50 pp* [1] | No | CS+K | 0.114 | 0.898 | 4.935 | 0.206 | 0.861 | 0.949 | 0.976 | 58.4M 2x forward |
| PyD-Net [2] | No | CS+K | 0.148 | 1.316 | 5.929 | 0.244 | 0.800 | 0.925 | 0.967 | 1.9M |

**Table 2**: Comparison on Eigen split. In dataset, K indicates training on [3] and CS indicates Cityscapes [4]. Our models compress more than 80% the original model with small drop in accuracy. *pp post-processing done by [1] but requires two forward passes.




(a) $L_{task}$

(b) $L_{task} + L_{mask}$

**Fig. 1**: VGG model with masking with different losses. Blue bars are the encoder part and red bars are the decoder part. Deeper features in the encoder are more prone to removal due to high redundancy. Decoder layers of the same number as the encoder are also more prone to removal which supports multiple other proposed engineered architectures.