DenseNet with pre-activated deconvolution for estimating depth map from single image

Conference Paper · September 2017

CITATIONS
0

READS
673

5 authors, including:

Saurav Sharma
National Institute of Technology Rourkela
6 PUBLICATIONS **15** CITATIONS
SEE PROFILE

Ram Prasad Padhy
National Institute of Technology Rourkela
12 PUBLICATIONS **35** CITATIONS
SEE PROFILE

Nabarun Goswami
The University of Tokyo
6 PUBLICATIONS **101** CITATIONS
SEE PROFILE

Pankaj K Sa
National Institute of Technology Rourkela
109 PUBLICATIONS **681** CITATIONS
SEE PROFILE

DenseNet with pre-activated deconvolution for estimating depth map from single image

Saurav Sharma[1]
sauravsharma.nitr@gmail.com
Ram Prasad Padhy[1]
ramprasad.nitr@gmail.com
Suman Kumar Choudhury[1]
sumanchoudhury.nitr@gmail.com
Nabarun Goswami[2]
nabarun.goswami@sony.com
Pankaj Kumar Sa[1]
pankajksa@nitrkl.ac.in

[1] Department of Computer Science and Engineering
National Institute of Technology
Rourkela, India

[2] Sony India Software Centre Private Limited
Bengaluru, India

Abstract

In this article, an attempt has been made to estimate the depth map of a scene from a single RGB image. We propose a fully convolutional architecture to learn the multi-valued ambiguous mapping between the monocular images and their corresponding depth maps. The densely connected convolutional network, DenseNet, is employed to maximize the information flow along the deep framework. We introduce an efficient strategy to learn the feature map up-sampling by appending a network of pre-activated deconvolutional layers to the existing DenseNet. The unified architecture, comprising the above two components, is trained end-to-end without any further post-processing. Exhaustive simulation on benchmark dataset not only reveals the superiority of the proposed model over the current state-of-the-art but also requires significantly less number of training samples for convergence.

1 Introduction

Depth estimation from a single image has been an ongoing research in computer vision community. It has a broad range of applications in various vision-centric tasks such as, robotic perception, autonomous navigation, augmented reality, 3D scene construction, advanced driver assistance system, and so forth. Most of the existing methods follow the stereo aided vision, using a pair of cameras, to estimate the depth map. Even humans are facilitated with binocular vision. In contrast to this, monocular depth prediction is an ill-posed problem due to the inherent ambiguous mapping between single RGB image and its desired depth map. Few recently reported articles apply several additional cues to estimate the depth from a single camera view. Structure from motion [27] exploits the camera motion information to approximate the position of the camera in the world co-ordinate over discrete temporal intervals, and then apply the triangulation geometry to estimate depth

between the consecutive FoV. Zhang *et al*. [29] observe the variation in light illumination over time to formulate a model for depth prediction. Another work exploits the focus cue to devise a focal stack aligning algorithm for depth measurement. All these methods, however, perform in constrained environments, and therefore cannot always be deployed for real world applications.

The deep convolutional neural networks (DCNN), especially designed for vision related applications, have gained significant popularity over the last few years. In particular, the CNN learns the complex mapping between the input and output space in an hierarchical fashion through multiple intermediate layers. More is the level of deepness in a CNN, stronger is its ability to learn the underlying pattern. However, It may so happen that the input information, over a large number of deeper layers, may get either vanished or washed out when it reaches to the end. Few articles have been reported to address this degradation problem in the recent past. Highway Networks [26] and ResNets [6] facilitate identity mapping through by-passing the information flow from each intermediate layer to more than one subsequent layers. Stochastic depth [9] proposed a variant of ResNets, where the intermediate layers are randomly dropped during training to let the network learn a more robust representation. The FractalNets [14] present a methodology to combine multiple parallel layer sequences with varying number of convolutional blocks per sequence to build a rich deeper network; again, a number of shortcut (by-pass) connections are made between multiple parallel layers. To facilitate maximum information flow along the deep network, the DenseNet [8] architecture connects the output of each intermediate layer to all its subsequent layers and so on.

In this article, we present a fully convolutional deep architecture to estimate the depth from a single camera view. The DenseNet model is employed to extract the deep convolutional features. Alongside, an efficient way of up-sampling strategy is proposed to improve the spatial extent of the low resolution feature map. The proposed model along with few competitive methods have been simulated on a dataset of benchmark monocular images. An analysis has been made to choose the appropriate optimization function between the RMSE loss and the berHu loss [20, 30] for the task of depth prediction.

The rest of this article is enumerated as follows. An overview on existing monocular depth estimation methods is given in Section 2. The proposed deep framework is elaborated in Section 3. Simulation details alongside the obtained results are analyzed in Section 4. Concluding remarks are summarized in Section 5.

2 Related Work

There has been a decent amount of research work carried out in the field of vision-based depth estimation using both monocular and stereo approaches. The stereo based methods deploy a pair of cameras along one common plane and apply the triangulation geometry to obtain the depth using various cues such as, stereopsis, disparity, eye-convergence, and so forth. However, depth estimation from monocular images possesses many challenges owing to its ill-posed behavior in the absence of local and global information.

Preliminary work on monocular depth estimation usually apply various hand-engineered features [7], where coarse geometric characteristics of a scene are learned by drawing various assumptions about the 3D plane. Saxena *et al*. [22] used the Markov random field (MRF) to extract both local and global features from monocular RGB images to build a system called “Make3D" [23] for depth prediction. Instead of directly estimating the depth, Liu

et al. [16] utilized the semantic labels of an image to construct the 3D geometry of the scene. They proposed two different approaches using MRF to perform semantic segmentation based on pixels and super-pixels. Ladicky *et al*. [13] predicted the likelihood depth value for a pixel by performing both semantic segmentation and depth estimation using a classification based approach. Karsch *et al*. [10] in their approach, presented a matching technique on a predefined dataset to extract a set of similar, look-alike images to that of the input image, and warped with SIFT Flow [17] followed by global depth optimization to produce an approximate depth map. Konrad *et al*. [11] fused a number of depth maps obtained for the nearest set of images and applied cross-bilateral filter for smoothening. Liu *et al*. [19] used the Conditional Random Field (CRF) to realize depth estimation as an optimization problem with discrete and continuous potential variables with the assumption that the pixels having similar RGB properties possess similar depth values.

Recent advances in deep learning have led to the usage of various CNN architectures for depth estimation from single images. The deep CNN architectures usually follow the principle of transfer learning; a pre-trained CNN model, on a large dataset such as ImageNet [3], is used as a good initialization of weights upon which the domain specific dataset is trained to fine-tune the model in regard to the desired application. Eigen *et al*. [5] used a two stage architecture of different scales for depth estimation; in the first phase, the CNN model is applied to extract the coarse depth map, which is subsequently fed, alongside the raw input, again to the same CNN model to fine-tune the result. This work is further extended [4] to develop a three stage CNN architecture where the first stage is pretrained on either AlexNet [12] or VGGNet [25] to produce coarse details in predicted depth image, which is refined in further stages. Roy *et al*. [21] proposed a neural regression forest that employ both CNN and an ensemble of regression trees to produce continuous depth maps. Liu *et al*. [18] combined CNNs with probabilistic graphical models and a CRF loss layer to improve the quality of predicted depth images. Wang *et al*. [28] employed a joint CNN with hierarchical CRF layers to perform both semantic segmentation and depth prediction on input RGB images. Chakrabarti *et al*. [2] used convoluational layers to produce distributions of depth derivatives having different order, scale and orientation. They later incorporated all the distributions to produce the final depth estimate.

The hand-crafted methods are either parametric or scene-constrained, and hence cannot be deployed across varying environments. The CNN based models, on the other hand, obviously produce comparatively better result, however, still needs either multi-tier architecture or other means of post-processing to further fine-tune the result. The present work alleviates the above issue by presenting a fully convolutional deep architecture for improved depth estimation; a CNN model is chosen that can maximize the information flow along the deeper network without any means of degradation issues. Additionally, an up-sampling strategy is proposed to improve the spatial resolution of the depth map so as to precisely recognize various objects present at discrete depth levels.

3 Proposed Methodology

This section presents a fully convolutional architecture for monocular depth estimation using single RGB image. The proposed framework is a sequence of two modules. A deep convolutional network is first employed to extract the feature map. It can be realized that the output map of a deep CNN usually possesses very low spatial resolution as compared to that of the input resolution. It works well for any classification task, where the low resolution

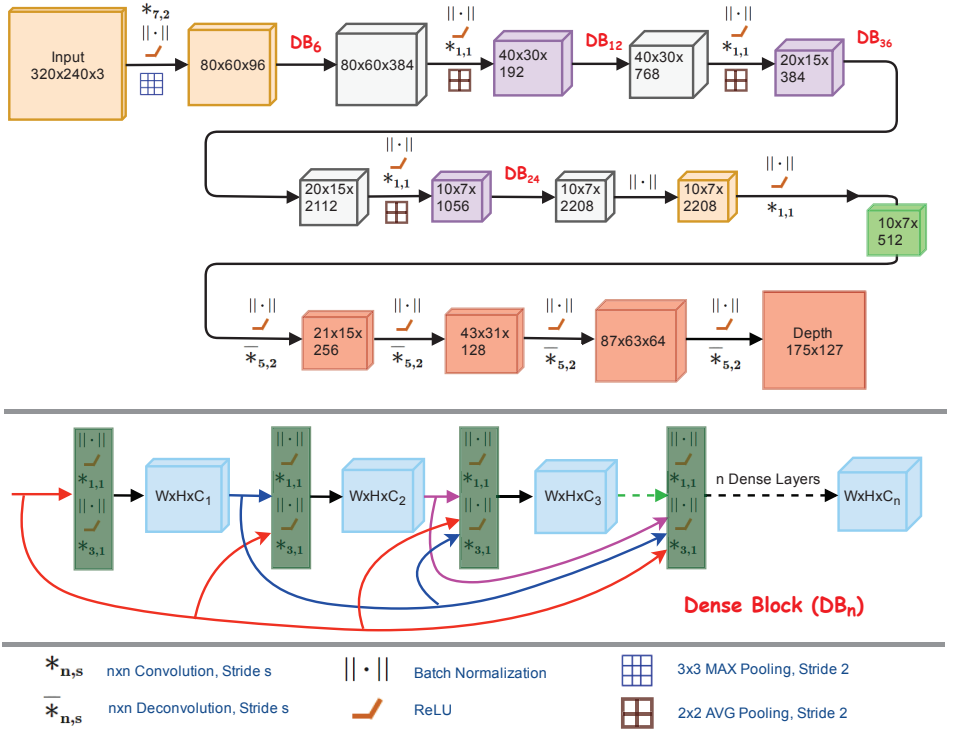

Figure 1: Proposed fully convolutional architecture. Stage (1): DenseNet-161 model accepts input image of size $320 \times 240 \times 3$ and produces a feature map of size $10 \times 7 \times 2208$. Stage (2): a bootleneck layer reduces the number of feature maps: $10 \times 7 \times 2208$ to $10 \times 7 \times 512$ (Green box). Stage (3): a sequence of four deconvolution layer up-samples the spatial resolution of feature map : $10 \times 7 \times 512$ to 175×127 (Deconvolution outputs in red boxes). Bottom row depicts the the schematic diagram of a Dense Block; DB_n represents a Dense Block with n layers.

map is usually fed into either a fully-connected layer or an equivalent convolutional layer to predict the desired object class. In contrast to this, a regression task, such as depth estimation, requires at-least a minimum higher resolution output, where the objects at different depth should be precisely distinguished. Accordingly, we suggest a simple yet efficient mechanism to up-sample the low-resolution images with minimal parameter overhead.

A number of deeper architectures have been reported over the last few years. We embed the Densely Connected Convolutional Networks (DenseNet [8]), to learn the feature representation. Subsequently, few "deconvolution layers" are incorporated to learn the up-samping within the unified framework. The architectural diagram is depicted in Figure 1. All the steps of the proposed architecture are enumerated in the subsequent paragraphs.

DenseNet: The DenseNet model comprises a number of intermediate layers such that each layer accepts inputs from all its preceding layers and forwards its output to all the subsequent layers of the network. Mathematically, the p^{th} layer accepts the feature maps of all its preceding layers and computes a non-linear mapping function $H_p(X)$, given by: $x_p = H_p([x_0, x_1, \cdots, x_{p-1}])$, where $X = [x_0, x_1, \cdots, x_{p-1}]$ represents the concatenation of the

feature maps generated by all its preceding layers $0, 1, \cdots, p-1$. The DenseNet model is arranged as a sequence of multiple dense-blocks as shown in Figure 1. A dense block computes a number of composite mapping functions. Each of these functions performs three consecutive operations: batch normalization followed by a ReLU followed by a 3×3 convolution. Each 3×3 convolutional layer produces a fixed k number of output feature maps, termed as the growth rate of the network. However, it accepts many more inputs owing to the dense connectivity of the model. Therefore, a 1×1 bottleneck convolution layer is employed prior to the 3×3 convolution that reduces the input to $4 \times k$ feature maps only. All the output feature maps within a dense block possess the same spatial resolution. To facilitate more compactness, the concatenated feature map of one dense block is fed into its next dense block via a transition layer that reduces the spatial resolution of the concatenated input map by half of its original size; a transition layer performs a batch normalization, a 1×1 convolution and a 2×2 average pooling. An architecture with such dense connectivity is chosen to facilitate direct connectivity among all the layers and thereby maximizes the information flow along the network. The original paper of DenseNet [8] prepare four different models out of which the DenseNet-161 (K=48) performs superior as compared to others for the ImageNet classification task. The present work also adopts the DenseNet-161 model for the regression task as shown in Figure 1; it takes an input size of $320 \times 240 \times 3$ (width $\times$ height $\times$ # input channels), forwards it though the CNN having four dense blocks, and yields an output size of $10 \times 7 \times 2208$ (width $\times$ height $\times$ # feature maps).

Deconvolution Blocks: The major contribution of our work is the use of pre-activated deconvolution blocks to enhance the spatial resolution of predicted depth images. The low resolution feature map, obtained by the CNN, needs to be up-sampled to precisely identify the objects at discrete depth levels. An easiest way is to incorporate a fully-connected layer to the end of the last convolutional layer of the DenseNet. The present simulation, if embed a fully connected layer, requires more than 3.4 billion parameters to connect a $10 \times 7 \times 2208$ feature map to a 175×127 fc layer. Moreover, the fully connected layer cannot well exploit the local correlation within the neighborhood pixels. In contrast to this, we concatenate a network of pre-activated deconvolution layers towards the end of the DenseNet to learn the up-sampling within a unified network. Each of the deconvolution layer performs three sequential operations in a pre-activated arrangement: $\langle$`Batch Normalization-ReLU-DeConvolution`$\rangle$. ResNets [6] has already shown through experimentation that such a pre-activated arrangement of convolution performs well across a deep CNN network. Even, the DenseNet model also performs the batch normalization and ReLU prior to each 3×3 and 1×1 convolution. In short, the proposed architecture first applies a sequence of BN-ReLU-Conv operations, via DenseNet, to yield a low resolution feature map, and then again append a sequence of BN-ReLU-DeConv operations to upsample the feature map size within the same network. Before the deconvolution layers, the proposed architecture, as shown in Figure 1, first applies a bottleneck layer (1×1 convolution) that reduces the number of feature maps from 2208 (obtained from DenseNet) to 512, while keeping the same spatial resolution. Then, a total of four deconvolution blocks (BN-ReLU-DeConv$_{5\times5,\texttt{stride = 2}}$) is appended to produce the desired depth map. It can be noted that more number of deconvolution blocks may be required depending on the size of the depth map. The present simulation requires only 5.4 million parameters, to be learned, between a $10 \times 7 \times 2208$ DenseNet output map and a 175×127 predicted depth map.

Loss function: The choice of the loss function plays a crucial role in any CNN based optimization task. We simulate the proposed model in equation with minimizing two loss

functions separately that have been extensively used in regression tasks; (i) RMSE loss, (ii) Reverse Huber (berHu loss).

The RMSE loss minimizes the squared-root euclidean norm between the predicted depth map ($\hat{y}$) and the ground-truth map (y), given by,

$$RMSE\,(\hat{y}-y) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2} \tag{1}$$

where $i = 1, 2, \cdots, n$ are the pixel indexes of each image present in one mini-batch.

The berHu loss poses a good trade-off between the L_1 norm and L_2 norm so as to provide higher weights to pixels having higher residual. The berHu loss [20, 30] between the predicted depth map and the ground-truth map can be given as,

$$berHu\,(\hat{y}-y) = \begin{cases} L_1\,(\hat{y}-y) & \text{if } (\hat{y}-y) \in [-t, t] \\ \frac{1}{2t} \times \left(L_2\,(\hat{y}-y) + t^2\right) & \text{otherwise} \end{cases} \tag{2}$$

$$L_1\,(\hat{y}-y) = \sum_{i=1}^{n} |\hat{y}_i - y_i|$$

$$L_2\,(\hat{y}-y) = \sum_{i=1}^{n} (\hat{y}_i - y_i)^2$$

where $i = 1, 2, \cdots, n$ are the pixel indexes of each image present in one mini-batch. It can be observed that Equation 2 is continuous, and moreover, first order differentiable at threshold t that decides the switching between L_1 to L_2. In each step of gradient descent, t is set as the 15% of the maximal per-batch error, given by, $t = 0.15 \times \max(|\hat{y}_i - y_i|)$.

4 Experimental Results

This section provides a detailed analysis of the experiments carried out for the proposed architecture. Our model is implemented using PyTorch[1] and trained on a system having dual Quadro K2200 GPU with 8GB memory. The first stage of our architecture takes the pre-trained DenseNet-161 model, on ImageNet dataset [3], as initialization. The second stage, having one bottleneck convolutional layer and four deconvolution blocks are initialized with weights taken from a normal distribution with 0 mean and 0.02 variance.

NYU Depth Dataset: The proposed model is evaluated on the standard NYU Depth v2 [24] RGB-D image dataset. It comprises images of multiple indoor scenes taken from a Microsoft Kinect depth camera. Moreover, it contains both labeled and raw images for evaluation. The training and test set, of the raw dataset, comprises 249 and 215 indoor scenes respectively; each scene contains a number of RGB images as well as its corresponding depth maps.

We consider a subset of the raw dataset that further needs to be preprocessed in accordance with the depth prediction task. A kinect camera can process a maximum depth of 10 meters, and hence all the ground-truth depth maps are normalized in the range of 0 to 10. Besides, the spatial resolution of the RGB images (640×480) is reduced to 320×240 for our model evaluation. We perform data augmentation to increase the number of training

[1] https://github.com/pytorch/pytorch

samples; in particular, we take a total of 5900 training samples to train our model. Prior to training, all the input images are normalized with the mean and standard deviation of the ImageNet dataset.

Data Augmentation: We resort to standard data augmentation techniques to increase the number of training samples. The input RGB images and corresponding target depth images are subjected to following data augmentation techniques —

- **Scaling:** images are scaled randomly with a factor that lies between 1 to 1.5.
- **Rotation:** images are rotated randomly with an angle that lies between -5° to $+5^\circ$.
- **Flipping:** images are flipped randomly with a probability of 0.5.

Although the augmentation is done randomly, it may be noted that the augmentation value for a specific RGB image and its corresponding depth image is kept same. Data augmentation helps the model to learn efficiently by adding different variations of the same training sample.

Architecture Evaluation: Our model is trained using Stochastic Gradient Descent [1] as the optimization method and berHu, RMSE as the loss functions (Section 3). The training for both the loss functions is done separately and the quantitative results are shown in Table 1. The training is performed with a batch-size of 5 and initial learning rate is set to 0.001. The learning rate is decreased by a factor of 5, when the loss becomes more or remains same for consecutive 5 epochs. It has been observed that berHu loss produces better qualitative predictions as compared to RMSE loss. This can be attributed to the fact that berHu is observed to give better convergence in comparison to RMSE. Also, in terms of training loss, berHu outperforms RMSE. This can be better visualized in Figure 2, which represents the epoch-wise training loss values for both the loss functions. The resolution for the predicted depth image of our proposed model is taken as 175×127. Deconvolution network containing 4 deconvolution layers enhances the depth image from low resolution to high resolution. The prediction is done on the standard NYU Depth v2 labeled test dataset which contains 654 images. The predicted depth images are then compared with ground truth images for quantitative evaluation using standard error metrics defined in previous depth estimation works [4, 5, 13, 15, 18]. Table 1 reports our model performance as compared to other state-of-the-art models. It can be observed that our model outperforms existing state-of-the-art techniques with fewer training samples.

The proposed architechture, including the DenseNet and deconvolution blocks, has a total of 167 parametric layers. Also, our model contains less parameters as compared to state-of-the-art models. The number of parameters in our model (31 million) is far less as comapred to Eigen *et al*. [4] (218 million) for obtaining similar resolution depth maps. Few of the predicted depth images obtained using berHu loss function is delineated in Figure 3. It can be noticed that our model exhibits remarkable visual quality for the predicted depth images.

5 Conclusion

In this article, we present a deeper architecture to estimate the depth map from a single RGB image. The fully convolutional architecture comprises two sequencial modules. The first module adopts the DenseNet model to yield a low-resolution feature map. The second stage embeds a sequence of deconvolution blocks, arranged in pre-activated style, to increase

NYU Depth V2	lower is better				higher is better		
	rel	rms	rms(log)	log_{10}	δ_1	δ_2	δ_3
Karsch *et al.* [10]	0.374	1.12	-	0.134	-	-	-
Ladicky *et al.* [13]	-	-	-	-	0.542	0.829	0.941
Liu *et al.* [19]	0.335	1.06	-	0.127	-	-	-
Li *et al.* [15]	0.232	0.821	-	0.094	0.621	0.886	0.968
Liu *et al.* [18]	0.230	0.824	-	0.095	0.614	0.883	0.971
Wang *et al.* [28]	0.220	0.745	0.262	0.094	0.605	0.890	0.970
Eigen *et al.* [5]	0.215	0.907	0.285	-	0.611	0.887	0.971
Roy and Todorovic [21]	0.187	0.744	-	0.078	-	-	-
Eigen and Fergus [4]	0.158	0.641	0.214	-	0.769	**0.950**	**0.988**
Proposed method (RMSE)	0.159	**0.549**	0.213	0.064	0.791	0.946	0.984
Proposed method (berHu)	**0.153**	**0.549**	**0.208**	**0.062**	**0.799**	**0.950**	0.985

Table 1: Comparative analysis of various methods on NYU Depth V2 labeled test dataset based on standard error metrics [4, 5, 13, 15, 18].

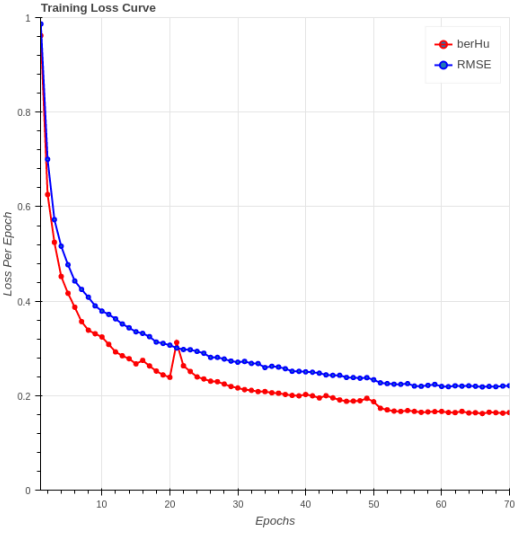

Figure 2: Training Loss Curve for RMSE and berHu loss functions.

the spatial resolution of the depth map so as to precisely visualize various objects present at discrete depth levels. The unified architecture with the above two modules is trained end-to-end with no need of any further post-improvisation module. An analysis has been made to choose the appropriate loss function between the RMSE and berHu for the task of depth estimation. The NYU Depth V2 RGB-D dataset is taken into consideration to evaluate the proposed model, and the results, thus obtained, are compared with few existing state-of-the-art models. It has been observed that the proposed model outperforms the other depth models both qualitatively and quantitatively. The proposed model is three times deeper than the current state-of-the-art and uses 10 times fewer training samples for convergence.

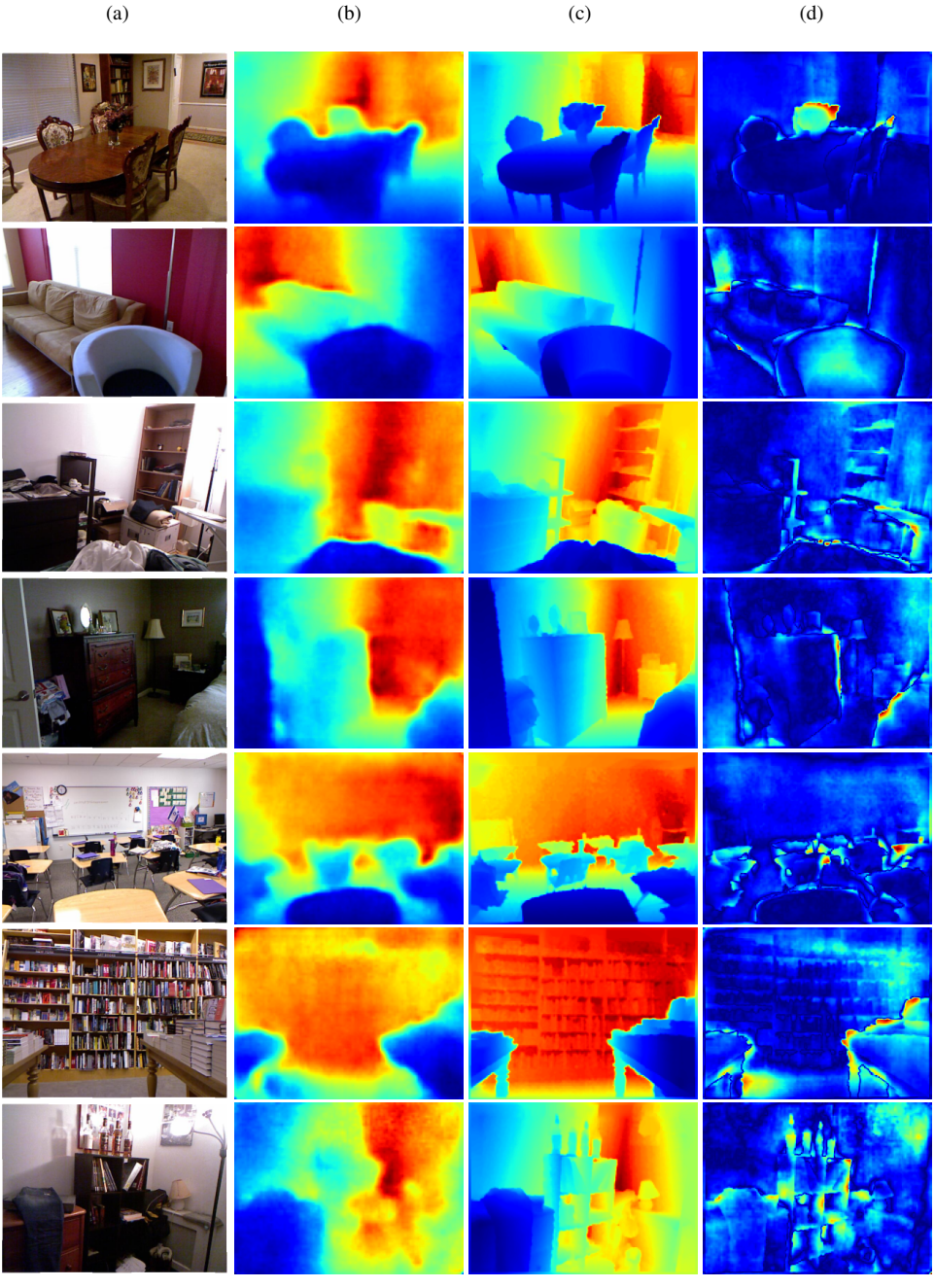

Figure 3: Prediction results of sample images by our proposed architecture on NYU Depth V2 dataset. The figure shows (a) input image (b) predicted depth image (c) ground depth image (d) absolute error map. For better comparison, all the colormaps are scaled equally.

References

[1] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[2] Ayan Chakrabarti, Jingyu Shao, and Greg Shakhnarovich. Depth from a single image by harmonizing overcomplete local network predictions. In *Advances in Neural Information Processing Systems*, pages 2658–2666, 2016.

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[4] 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*, pages 2650–2658, 2015.

[5] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374, 2014.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pages 630–645. Springer, 2016.

[7] Derek Hoiem, Alexei A Efros, and Martial Hebert. Geometric context from a single image. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 654–661. IEEE, 2005.

[8] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. Densely connected convolutional networks. *arXiv preprint arXiv:1608.06993*, 2016.

[9] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European Conference on Computer Vision*, pages 646–661. Springer, 2016.

[10] Kevin Karsch, Ce Liu, and Sing Kang. Depth extraction from video using non-parametric sampling. *Computer Vision–ECCV 2012*, pages 775–788, 2012.

[11] Janusz Konrad, Meng Wang, and Prakash Ishwar. 2d-to-3d image conversion by learning depth from examples. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, pages 16–22. IEEE, 2012.

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[13] Lubor Ladicky, Jianbo Shi, and Marc Pollefeys. Pulling things out of perspective. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 89–96, 2014.

[14] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Fractalnet: Ultra-deep neural networks without residuals. *arXiv preprint arXiv:1605.07648*, 2016.

[15] Bo Li, Chunhua Shen, Yuchao Dai, Anton van den Hengel, and Mingyi He. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1119–1127, 2015.

[16] Beyang Liu, Stephen Gould, and Daphne Koller. Single image depth estimation from predicted semantic labels. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1253–1260. IEEE, 2010.

[17] Ce Liu, Jenny Yuen, and Antonio Torralba. Sift flow: Dense correspondence across scenes and its applications. *IEEE transactions on pattern analysis and machine intelligence*, 33(5):978–994, 2011.

[18] Fayao Liu, Chunhua Shen, and Guosheng Lin. Deep convolutional neural fields for depth estimation from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5162–5170, 2015.

[19] Miaomiao Liu, Mathieu Salzmann, and Xuming He. Discrete-continuous depth estimation from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 716–723, 2014.

[20] Art B Owen. A robust hybrid of lasso and ridge regression. *Contemporary Mathematics*, 443:59–72, 2007.

[21] Anirban Roy and Sinisa Todorovic. Monocular depth estimation using neural regression forest. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5506–5514, 2016.

[22] Ashutosh Saxena, Sung H Chung, and Andrew Y Ng. Learning depth from single monocular images. In *NIPS*, volume 18, pages 1–8, 2005.

[23] Ashutosh Saxena, Min Sun, and Andrew Y Ng. Make3d: Learning 3d scene structure from a single still image. *IEEE transactions on pattern analysis and machine intelligence*, 31(5):824–840, 2009.

[24] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. *Computer Vision–ECCV 2012*, pages 746–760, 2012.

[25] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[26] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

[27] Richard Szeliski. Structure from motion. *Computer Vision*, pages 303–334, 2011.

[28] Peng Wang, Xiaohui Shen, Zhe Lin, Scott Cohen, Brian Price, and Alan L Yuille. Towards unified depth and semantic prediction from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2800–2809, 2015.

[29] Ruo Zhang, Ping-Sing Tsai, James Edwin Cryer, and Mubarak Shah. Shape-from-shading: a survey. *IEEE transactions on pattern analysis and machine intelligence*, 21(8):690–706, 1999.

[30] Laurent Zwald and Sophie Lambert-Lacroix. The berhu penalty and the grouped effect. *arXiv preprint arXiv:1207.6868*, 2012.