Deep Learning for Monocular Depth Prediction

Yue Kuang
Columbia University
yk2951@columbia.edu

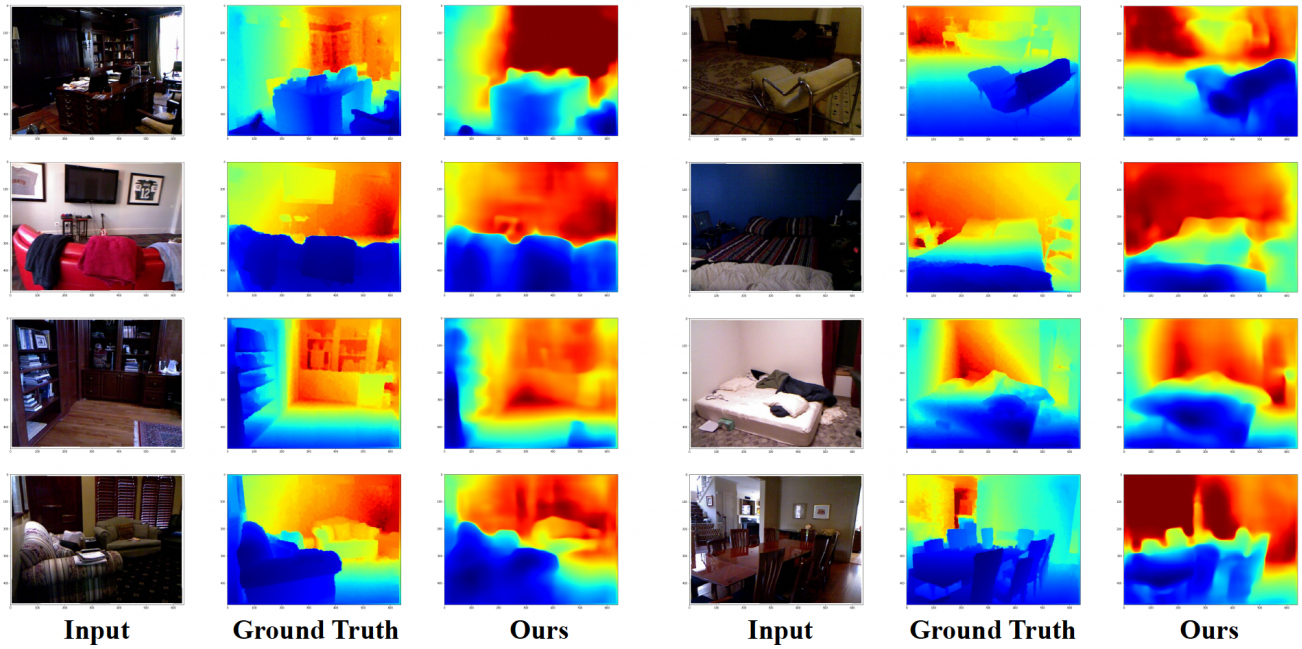

Figure 1: The monocular depth prediction experiment results for our model on our test set from NYU-Depth V2 [9].

1. Introduction[1]

Depth prediction is the task of estimating the depth from a single RGB image. It requires taking an input image and predicting the distance to the camera plane at each pixel. Accurate depth prediction from images is a fundamental task in many applications including scene understanding, reconstruction, self-driving cars, etc.

The objective of this project is to implement a deep learning system that can predict the depth of each pixel. It should map a $H \times W \times 3$ image to a $H \times W \times 1$ depth map. Starting with a self-developed convolutional autoencoder, we also reproduced more complex models such as U-Net [11] with skip connection and concatenate layers from scratch. We also tried applying transfer learning where we replace the encoder part with large pre-trained models such as Xception [4] and DenseNet [6].

[1]The source code for this project can be access at GitHub.

Quantitative evaluations will be performed on different model architectures we implemented. We compare the performance of models and we confirm the superiority of combining pre-trained models with decoder, which brings to the improvement of model performances in monocular depth prediction tasks.

2. Related Work

2.1. Monocular Depth Prediction

Depth prediction is the task of estimating the depth from a single RGB image. It requires taking an input image and predicting the distance to the camera plane at each pixel. It has been considered by many CNN methods where researches formulate the problem as a regression of the depth value from the RGB image input [5, 7, 14, 12]. While the models' performances have been improved steadily, there is still room for the further improvement in the resolution and quality of the output depth map.

2.2. Encoder-Decoder

Encoder-Decoder networks have achieved impressive results in a wild range of computer vision tasks, including image segmentation [2, 3], object detection [17], and scene text recognition [10]. Recently, researchers have also achieved better results on depth prediction problems with the encoder-decoder architectures [1, 16, 8, 15].

The encoder part of the network takes an input image and generates a high-dimensional feature vector. It aggregate features at multiple levels, while the decoder part decode features aggregated by encoder by taking a high-dimensional feature vector, performing upsampling and generating the expected output.

2.3. Transfer Learning

Transfer learning approaches have been shown very helpful in many different contexts. Pre-trained models such as Resnet [13], DenseNet [6], Xception [4] are used as the starting point on computer vision tasks given the vast compute and time resources required to develop neural network models. In recent work, Alashim et al. leveraged features extracted using high performing pre-trained networks to lead to more accurate results and achieved detailed high-resolution depth maps even for a very simple decoder. Our project is based on this idea and we will build our own networks that utilizes pre-trained models followed by a decoder architecture.

3. Method

3.1. Model Architectures

Convolutional Autoencoders (CAE) As is shown in Figure 2, Convolutional autoencoders extend the basic structure of the simple autoencoder by changing the fully connected layers to convolution layers.

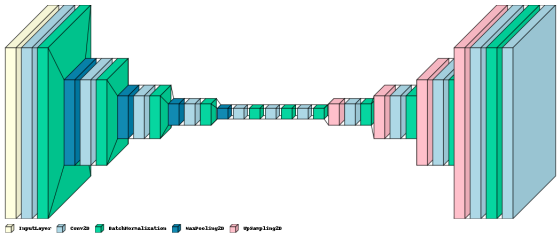

Figure 2: Convolutional autoencoder model structure with Input (yellow), Convolutional (blue), BatchNormalization (green), Max Pooling (dark blue), and Up Sampling (pink).

The encoder part of the network takes an input image and generates a high-dimensional feature vector. It aggregate features at multiple levels, while the decoder part decode features aggregated by encoder by taking a high-dimensional feature vector, performing upsampling and generating the expected output.

U-Net The U-Net [11] is a convolutional network architecture proposed by Ronneberger et al. The network consists of a contracting path and an expansive path, which gives it the u-shaped architecture. The contracting path is a typical convolutional network that consists of repeated application of convolutions, each followed by a rectified linear unit (ReLU) and a max pooling operation. During the contraction, the spatial information is reduced while feature information is increased. The expansive pathway combines the feature and spatial information through a sequence of up-convolutions and concatenations with high-resolution features from the contracting path.

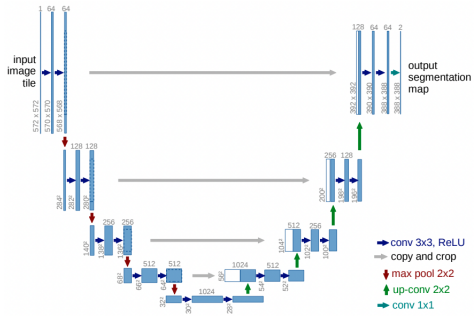

Figure 3: U-net architecture (example for 32x32 pixels in the lowest resolution).Each blue box corresponds to a multi-channel feature map. White boxes represent copied feature maps. The arrows denote the different operations.

The concatenation of feature maps from encoder layers with corresponding decoder layers is called skip connections (the white arrows in Figure 3). It provides spatial information to each decoder so that it can effectively recover fine-grained details when producing output masks.

We reproduced the U-Net architecture with skip connection from scratch, which matches with our expected input shape $480 \times 640 \times 3$ and output shape $480 \times 640 \times 1$ depending on the NYU Depth V2 [9] dataset.

Pretrained Model + Decoder The idea is to replace the traditional encoder part with a model pretrained on ImageNet. As is shown in Figure 4, for our encoder, the input RGB image is encoded into a feature vector using the pretrained model. This vector is then fed to our decoder, a successive series of up-sampling layer, in order to construct the final depth map at the same resolution as input. inspired by the U-Net architecture, we have also implemened skip connections (grey arrow) which concatenates the upsampling layers with their associated encode layers.

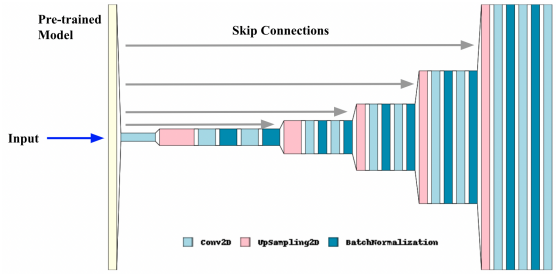

Figure 4: Pretrained model + decoder network architecture.

In this project, we chose both Xception [4] and DenseNet [6] as our pretrained models and compared their performances.

3.2. Loss Function

A standard loss function for depth regression problems considers the difference between the groundtruth depth map y and the prediction of the depth regression network $\hat{y}$. We referred to prior work [1] and defined the loss as a weighted sum of three different loss functions:

- The pointwise $L1$ loss: $L_1(y, \hat{y}) = \frac{1}{n}\sum_p^n |y_p - \hat{y}_p|$.
- The $L1$ loss defined over the image gradient g of the depth image:
  $L_{grad}(y, \hat{y}) = \frac{1}{n}\sum_p^n |g_x(y_p, \hat{y}_p)| + |g_y(y_p, \hat{y}_p)|$.
- The Structural Similarity (SSIM) loss:
  $L_{SSIM}(y, \hat{y}) = \frac{1-SSIM(y,\hat{y})}{2}$.

During the training process, we found that the models are not performing well on detecting edges and the outputs tends to have arbitrary and blurred edges, so we decided to put more weights on the edge loss L_{grad}. On the other hand, we also found that structural similarity loss L_{SSIM} and edge loss L_{grad} contributes more to the model performance than the pointwise $L1$ loss. Therefore the final formula we ended up with for training the model is:

$L(y, \hat{y}) = 1.5 * L_{grad} + 1.0 * L_{SSIM} + 0.2 * L_1$

4. Experiments

4.1. Dataset

We are going to use the NYU Depth Dataset V2 [9].

This is a classic dataset which comes with pre-processed depth data that is nicely calibrated. It is comprised of video sequences from a variety of indoor scenes as recorded by both the RGB and Depth cameras from the Microsoft Kinect. It features:

- 1449 densely labeled pairs of aligned RGB and depth images
- 464 new scenes taken from 3 cities
- 407,024 new unlabeled frames
- Each object is labeled with a class and an instance number (cup1, cup2, cup3, etc)

We made use of the 1449 densely labeled pairs of aligned RGB and depth images and we split them into training (1049), validation (200), and test sets (200). The scenes are captured at a resolution $480 \times 640 \times 3$ and our networks produce predictions at the same resolution as input $480 \times 640 \times 1$

The depth maps have a lower bound of 0.5 meters and an upper bound of 4 meters. Therefore while making predictions, we applied $tanh$ activation to the output layer of our models followed by a linear operation

$$output = 1.75 * tanh(z) + 2.25$$

In this way, we make sure our output fell in the expected depth range $[0.5, 4.0]$.

4.2. Deep Learning Framework

We used Tensorflow as our deep learning framework because it is an end-to-end open-source deep learning framework with documentation and offers multiple abstraction levels for building and training models.

4.3. Evaluation Metrics

To quantitatively evaluate and compare the performances of the convolutional autoencoer, U-Net, and pretrained + decoder models, we utilize the standard six metrics used in prior work [5]. These are defined as:

- Threshold accuracy (δ_i): $\%$ of $y_p s.t. max(\frac{y_p}{\hat{y}_p}, \frac{\hat{y}_p}{y_p}) = \delta < thr$ for $thr = 1.25, 1.25^2, 1.25^3$.
- Average relative error (rel): $\frac{1}{n}\sum_p^n \frac{|y_p - \hat{y}_p|}{y}$
- Root mean squared error (rms): $\sqrt{\frac{1}{n}\sum_p^n (y_p - \hat{y}_p)^2}$
- Average ($\log_{10}$) error: $\frac{1}{n}\sum_p^n |\log_{10}(y_p) - \log_{10}(\hat{y}_p)|$

where y_p is a pixel in ground truth depth map y, $\hat{y}_p$ is a pixel in the predicted depth map $\hat{y}$, and n is the total number of pixels for each depth map.

4.4. Results and Error Analysis

As can be seen from Table 1, the performance of pretrained + decoder model outperforms the convolutional autoencoder and U-Net, which indicates that transfer learning could bring a huge improvement in depth prediction tasks. Among the two pretrained + decoder models, the one with

Method	$\delta_1 \uparrow$	$\delta_2 \uparrow$	$\delta_3 \uparrow$	rel $\downarrow$	rms $\downarrow$	$\log_{10} \downarrow$
Convolutional Autoencoder	0.501	0.787	0.923	0.299	1.106	0.123
U-Net	0.478	0.767	0.911	0.338	1.123	0.129
Xception [4] + Decoder	0.674	0.898	0.975	0.203	0.870	0.085
DenseNet [6] + Decoder	**0.714**	**0.918**	**0.981**	**0.180**	**0.847**	**0.0769**

Table 1: Comparisons of different model's test performance on the NYU Depth V2 [9] dataset. $\uparrow$ means the higher the better and $\downarrow$ means the lower the better.

DenseNet works slightly better and gives us the best performance.

Figure 6 visualizes the performance of our best model on 4 samples taken randomly from the test set. We can observe that the model can successfully predict the depth map with small incorrectness. The edges of most objects can be recognized and the depth value on each pixel are predicted reasonably. However, the performance on small sparse objects still has space for improvement, the reason may be that our loss function not only considers point-wise differences but also optimize for edges and structural similarity, the learning process does not converge well for very sparse depth images.

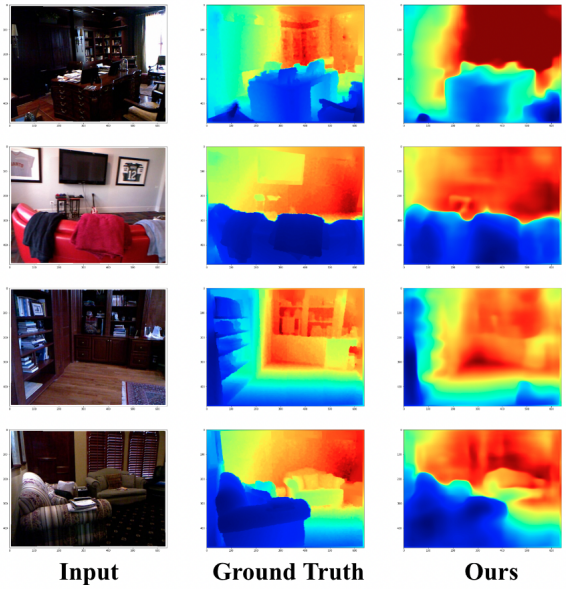

Figure 5: Performance of our best model (DenseNet + Decoder) on test set.

More visualization of our best model's performance can be found in Appendix A.

5. Conclusions

In this project, we implemented convolutional autoencoder, U-Net, and pretrained + decoder models from scratch and compared and evaluated their performances on monocular depth prediction tasks. Pretrained model + decoder architecture outperforms simple convolutional autoencder architectures, which confirms the superiority of transfer learning given the vast compute and time resources required to develop neural network models.Our model can successfully predict the depth map with small incorrectness. The edges of most objects can be recognized and the depth value on each pixel are predicted reasonably.

References

[1] Ibraheem Alhashim and Peter Wonka. High quality monocular depth estimation via transfer learning. *arXiv e-prints*, abs/1812.11941, 2018.

[2] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[3] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018.

[4] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

[5] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. *Advances in neural information processing systems*, 27, 2014.

[6] Forrest Iandola, Matt Moskewicz, Sergey Karayev, Ross Girshick, Trevor Darrell, and Kurt Keutzer. Densenet: Implementing efficient convnet descriptor pyramids. *arXiv preprint arXiv:1404.1869*, 2014.

[7] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *2016 Fourth international conference on 3D vision (3DV)*, pages 239–248. IEEE, 2016.

[8] Jin Han Lee, Myung-Kyu Han, Dong Wook Ko, and Il Hong Suh. From big to small: Multi-scale local planar guidance for monocular depth estimation. *arXiv preprint arXiv:1907.10326*, 2019.

[9] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[10] Zhi Qiao, Yu Zhou, Dongbao Yang, Yucan Zhou, and Weiping Wang. Seed: Semantics enhanced encoder-decoder framework for scene text recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13528–13537, 2020.

[11] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[12] Tomoyoshi Shimobaba, Takashi Kakue, and Tomoyoshi Ito. Convolutional neural network-based regression for depth prediction in digital holography. In *2018 IEEE 27th International Symposium on Industrial Electronics (ISIE)*, pages 1323–1326. IEEE, 2018.

[13] Sasha Targ, Diogo Almeida, and Kevin Lyman. Resnet in resnet: Generalizing residual architectures. *arXiv preprint arXiv:1603.08029*, 2016.

[14] Keisuke Tateno, Federico Tombari, Iro Laina, and Nassir Navab. Cnn-slam: Real-time dense monocular slam with learned depth prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6243–6252, 2017.

[15] Xiaohan Tu, Cheng Xu, Siping Liu, Guoqi Xie, Jing Huang, Renfa Li, and Junsong Yuan. Learning depth for scene reconstruction using an encoder-decoder model. *IEEE Access*, 8:89300–89317, 2020.

[16] Diana Wofk, Fangchang Ma, Tien-Ju Yang, Sertac Karaman, and Vivienne Sze. Fastdepth: Fast monocular depth estimation on embedded systems. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6101–6108. IEEE, 2019.

[17] Jimei Yang, Brian Price, Scott Cohen, Honglak Lee, and Ming-Hsuan Yang. Object contour detection with a fully convolutional encoder-decoder network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 193–202, 2016.

A. Appendix[2]

Visualizations of our best model (DenseNet + Decoder) performance on test set.

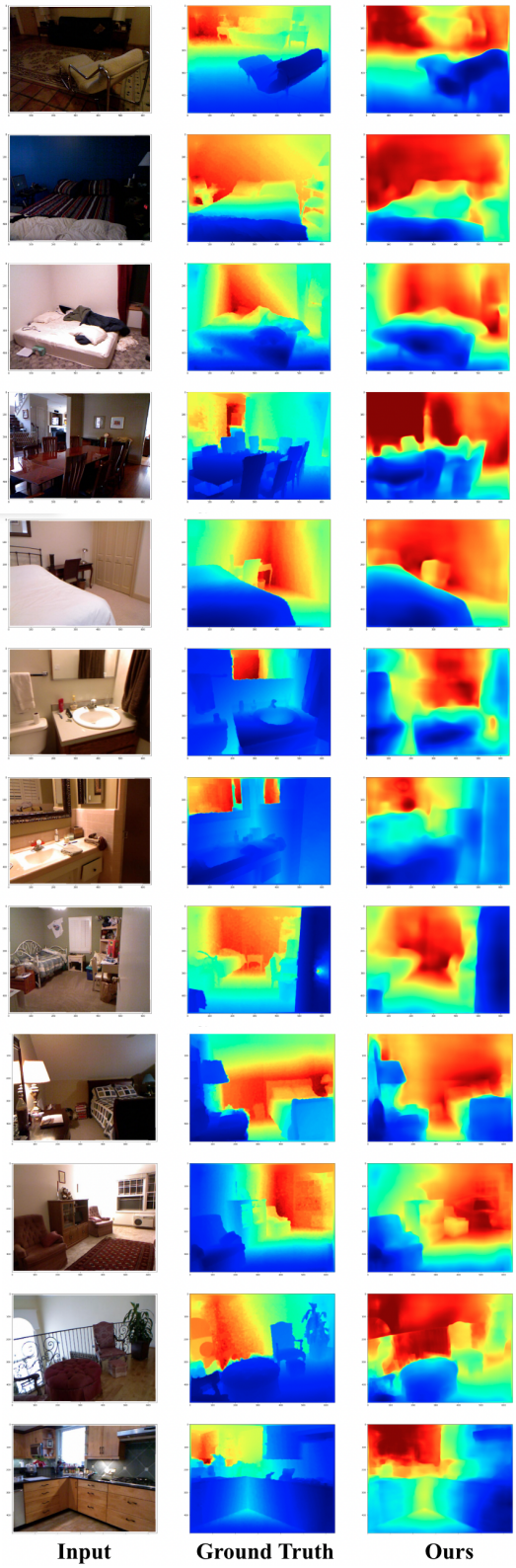

Figure 6: Performance of our best model (DenseNet + Decoder) on test set.

[2]The source code for this project can be access at GitHub.