Fast-Converging Depth Estimation using Transfer Learning

Rajiv Bharadwaj Reuben Gutmann Isaac Moothart
Charles Reinertson
Univeristy of Michigan, Ann Arbor, Michigan
{rajivbh, rcgutman, imoot, crein}@umich.edu

1. Introduction

Depth perception is a critical faculty for operating in the physical world. Navigating three-dimensional space requires an understanding of the surroundings, including the distance to those surroundings. Consequently, developing a system for determining the depth of the scene is essential for applications such as self-driving vehicles, image refocusing, robot-assisted surgery, and more. Many depth estimation systems utilize special sensors that emit light at some electromagnetic wavelength to build an understanding of the objects around them. However, these sensors are prohibitively costly to include in consumer grade products. Moreover, this additional hardware requirement prevents existing systems which lack such hardware from developing an understanding of the scene. Consequently, many researchers have explored the problem of depth estimation using only the simple visible light sensors with which so many consumer devices are deployed.

Traditionally, depth perception by humans and animals alike is achieved using stereo visual input, motion parallax, and/or object shading [10, 13]. This complex visual information is used to construct a three-dimensional understanding of the surrounding world. Such a reconstruction can also be performed computationally given similar input [8], however, the space requirements for stereo imagery and/or motion data are inherently larger than those for a single image. Moreover, the geometric calculations required to compute such depth maps are costly. Thus, a singular image view, known as monocular vision, presents advantages in both space and time for the problem of depth estimation.

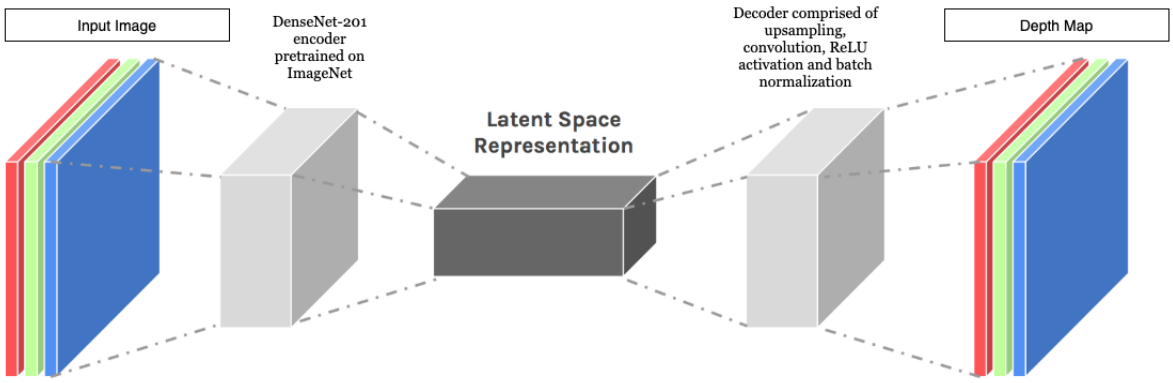
Although a more efficient prospect, monocular depth estimation presents its own challenges. Specifically, despite retaining object shading, the lack of stereo or motion references renders the depth of a single view ambiguous by its geometric definition. Nevertheless, monocular depth estimation remains an attractive area of research due to the prevalence of singular view cameras in existing systems and data. Using deep learning approaches, researchers have found systems which are able to estimate these depth maps from a single view, likely using patterns found in regular visual scenes which would not hold in geometrically irregular ones [2, 4].

The work of Fu *et al*. [4] and Alhashim and Wonka [2] provide key insights for the task of monocular depth estimation using deep learning which we apply in our approach. One problem that Fu *et al*. [4] identify in naive approaches is that the definition of the task as a regression problem defines a weight space with many local minima, making such approaches difficult to optimize. They address this problem by discritizing distance values, converting to an ordinal regression problem which converges faster and outperforms previous state-of-the-art solutions. Nevertheless, this performance comes at a cost in terms of processing complexity and thus does not fit into the objective of a consumer-grade distribution. Alhashim and Wonka [2] address this complexity problem in their approach by leveraging existing image classification models in an encoder-decoder architecture that performs competitively with state-of-the art systems such as [4].

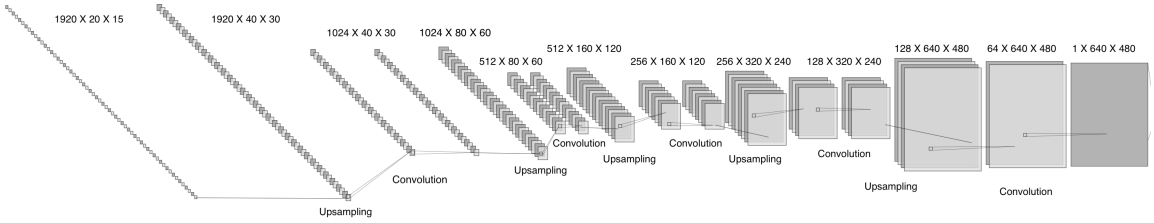
Inspired by the work of Alhashim and Wonka, we sought to develop a simple depth estimation architecture which uses transfer learning from image classification models for extracting image features but that achieves faster convergence. Like Alhashim and Wonka, the model we present uses a simple encoder-decoder architecture. However, we employ DenseNet-201 [7] pre-trained on ImageNet [3] as the encoder. The decoder we developed contains 5 dense convolutional blocks connected by upsampling to construct the estimated depth map. This model achieves competitive quantitative performance to previous work on the NYU Depth Dataset V2 [11], yet converges in less than 170 iterations.

2. Related Work

The prevalence of single-view camera systems continues to inspire much research interest in the area of monocular depth estimation. Due to the ambiguities of the task, traditional approaches must trade off between model performance and complexity [12]. Generally, state-of-the-art methods are still built on models with a large computational



(a) Encoder-decoder Architecture. Image template from [1].



(b) Decoder Architecture (Figure is an approximation and not to scale)

Figure 1: Model Architecture

footprint [4, 5]. Recent advances have begun to challenge this notion, however, developing small yet performant models [2].

One state-of-the-art method, [4], derives significant performance from converting the task to an ordinal regression one. In [4], they identify in their experiments that treating monocular depth estimation as a pure regression problem, i.e., calculating mean-squared error per pixel in the depth map, often creates models that are difficult to optimize due to the presence of many local minima and shoulders in the weight space. Instead, they convert the depth values into discrete intervals which are then optimized using ordinal regression [6]. They posit that this conversion transforms the weight space into one easier to optimize, leading to better minima in a shorter amount of time. Their state-of-the-art results on multiple benchmarks demonstrate the effectiveness of this approach, yet this system still suffers from a large computational footprint as [2] argues.

Applying recent advances in image classification research, [2] challenges the notion that deep, large networks are needed to achieve state-of-the-art performance on monocular depth estimation tasks. They implement a simple encoder-decoder architecture which uses DenseNet-169 [7] pre-trained on ImageNet [3] as their image encoder. This use of transfer learning, they argue, allows the framework they introduce to progress alongside advancements in other image learning domains such as image classification. They also employ a number of data augmentation policies, including color channel swapping, which they find to help significantly in improving and accelerating the training of their model. By reducing the task to a pre-trained encoder and simple decoder and focusing more on the data augmentation and loss functions used to train the model, [2] achieve results competitive with state-of-the-art methods while using more than 60% fewer parameters when compared to [4].

3. Approach

Our overall method of depth prediction involves training a deep neural network to accept RGB images as input and subsequently output predicted depths per pixel in the range of 0-10 meters.

Following a similar approach to that used in [2], we apply an encoder-decoder framework as the network architecture. A visualization of the overall architecture can be seen in figure 1a.

The encoder was a DenseNet-201 [7] pre-trained on ImageNet for image classification with the classification layer truncated from the encoder to obtain a high-level feature mapping of each image. While DenseNet-201 was trained to optimize its feature mapping to perform image classification, it is reasonable to expect that the feature mapping

for image classification would still provide an informative feature representation even for the new task of depth estimation. By utilizing transfer learning in the encoder, the model is able to achieve faster convergence. Specifically, transfer learning allows the training to start learning from a much higher baseline level since the model is able to immediately produce a meaningful encoding rather than needing to initially learn a meaningful encoding. Secondly, transfer learning allows the model to converge faster since the weights of the encoder portion of the network are frozen during training and do not require back propagation.

After images have been encoded via the encoder, the resulting feature map is then passed as input to the decoder portion of the network as seen in figure 1b. Our decoder is implemented via four stacked sublayers of 2x Nearest Upsampling - Convolution 3x3 - BatchNorm2d - ReLU - Convolution 3x3 - BatchNorm2d - ReLU followed lastly by one additional sequence of 2x Nearest Upsampling - Convolution 3x3 - BatchNorm2d - ReLU - Convolution 3x3. The number of convolutions used in the decoder is described by the following progression: $1920 \rightarrow 1024 \rightarrow 1024 \rightarrow 512 \rightarrow 512 \rightarrow 256 \rightarrow 256 \rightarrow 128 \rightarrow 128 \rightarrow 64 \rightarrow 1$. All 2-dimensional convolutional layer weights are initialized via sampling from a Normal distribution with mean 0 and standard deviation of $\frac{1}{5 \cdot 5 \cdot \text{num input channels}}$. All 2-dimensional convolutional layer biases are initialized to zero.

The model was trained using the Adam optimizer [9] with L1 loss and with an effective batch size of 64. The learning rate was 0.0001 and the weight decay was 0.0001.

4. Experiments

4.1. Data

We trained our model on the NYU Depth v2 [11] red-green-blue (RGB) images for different indoor scenes. The NYU Depth v2 images are comprised of 464 unique scenes from three different cities. This gives us a diversified sample of the variety of indoor scenes. These images are 640 by 480 pixels, and the depth measurements are gathered using Microsoft Kinect, ranging from 0 to 10 meters. During development, we used the labeled subset of the NYU dataset which totals 1449 RGB images. We randomly allocate 1159 of these pairs into the train set and 159 pairs into both the validate and test split. For training and evaluation purposes, we store the RGB and depth portions of the images separately which allows us to compare the depth map learned by our model to the actual depth map. We can then validate how well our model performs with the test and validate split.

4.2. Metrics

4.2.1 Training Metrics

We experimented with L1 and L2 loss because they both minimize the summed distance in color values between correct and predicted pixels. We expected L2 loss to produce better results than L1 loss since we did not anticipate significant outliers in the ground-truth depth maps, however, L1 loss ended up performing better than L2. Additionally, L1 loss caused the model to reach close to the minimal loss in a short number of training iterations, however, performance fluctuated once training was very close to the optimum.

4.2.2 Evaluation Metrics

We used Mean Squared Error (MSE) to evaluate the performance of our model including when to stop training.

4.3. Qualitative Results

In this section we refer to figure 2. The middle column represents the predicted depth map and the right column is the actual depth map. An example of our model performing well is the bottom depth map. The top depth map is a poor representation of the target depth map. An area of improvement for our system is trying to achieve crisper, more defined edges in our estimated depth maps.

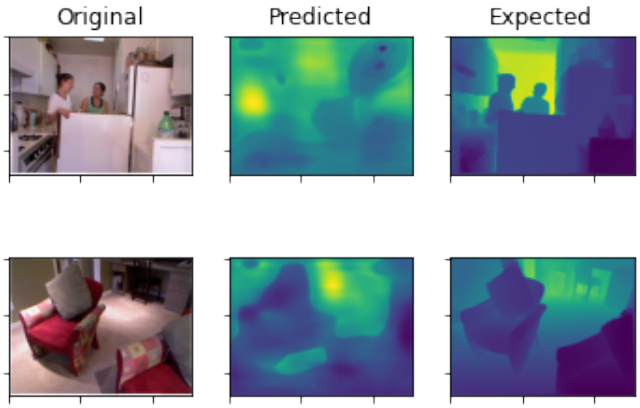


Figure 2: Qualitative results of our estimated depth maps (middle) and the ground truth (right)

4.4. Quantitative Results

Table 1 shows the quantitative results for the model. Compared to the baseline model, the final model achieved a better MSE value for the problem.

	Untrained		Epoch 7	
	L1 Loss	MSE	L1 Loss	MSE
Train	2.8063	1.2623	0.7426	0.1442
Validation	2.5040	1.0758	0.7426	0.1442
Test	-	-	0.7478	0.1446

Table 1: Quantitative Results

5. Implementation

Our project was implemented within a Google Colaboratory notebook. Our code was structured very similar to the starter code for past homeworks in EECS 442. However, all the code was written by the members of this team using PyTorch and other scientific modules that are available in Python.

Due to the computational limitations of Google Colab, we figured out an intuitive solution to increase the batch size for training and validation by having a counter that makes sure that the optimizer takes a step after n cycles of a maximum batch size of 8, thereby allowing us to get a size of $8n$. This is how we achieved a batch size of 64 for our purposes.

Our architecture is highly inspired by the paper from Alhashim and Wonka [2, 4] where they use a deep convolutional encoder-decoder neural network architecture. We used a DenseNet-201 encoder pretrained on ImageNet from the Torchvision module. The decoder architecture, along with training and evaluation functions were written by us.

6. Conclusion

We attempted to recreate with a simpler architecture the results from a paper outlining the use of a Convolutional Neural Network to perform monocular depth prediction. Our model converged within 170 iterations and produced a reasonable result as seen in figure 2.

Based on this experiment, one future project might consider using a similar model, but with additional computational resources to allow the model to train for a longer period with a smaller learning rate.

Additionally, attempting at procuring more data, whether that's through using classical depth prediction models to generate depths for unlabelled images in the NYU Dataset v2, or data augmentation methods are likely to lead to higher performance since we can potentially increase our training size to more than 100,000 images.

Furthermore, using a transfer learning model that has been trained on a more comparable task to depth estimation may allow the encoded version of each image to possess more descriptive features pertaining specifically to depth estimation.

References

[1] Explain about auto encoder? details about encoder, decoder and bottleneck?, Oct 2019. 2

[2] I. Alhashim and Peter Wonka. High quality monocular depth estimation via transfer learning. *ArXiv*, abs/1812.11941, 2018. 1, 2, 4

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 1, 2

[4] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2002–2011, 2018. 1, 2, 4

[5] Zhixiang Hao, Yu Li, Shaodi You, and Feng Lu. Detail preserving depth estimation from a single image using attention guided networks. In *2018 International Conference on 3D Vision (3DV)*, pages 304–313. IEEE, 2018. 2

[6] Frank E Harrell Jr. *Regression modeling strategies: with applications to linear models, logistic and ordinal regression, and survival analysis*. Springer, 2015. 2

[7] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2017. 1, 2

[8] H. Kim, Seung jun Yang, and Kwanghoon Sohn. 3d reconstruction of stereo images for interaction between real and virtual worlds. In *The Second IEEE and ACM International Symposium on Mixed and Augmented Reality, 2003. Proceedings.*, pages 169–176, 2003. 1

[9] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015. 3

[10] Karl Kral. Behavioural–analytical studies of the role of head movements in depth perception in insects, birds and mammals. *Behavioural Processes*, 64(1):1–12, 2003. 1

[11] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 1, 3

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

[13] Peter H. Schiller, Warren M. Slocum, Brian Jao, and Veronica S. Weiner. The integration of disparity, shading and motion parallax cues for depth perception in humans and monkeys. *Brain Research*, 1377:67–77, 2011. 1