Report
on

Real-Time Depth Estimation from 2D Images
26-Dec-2018

Submitted by

Love Prakash

DronaMaps Private Limited

Contents

1 Abstract:

Single image depth prediction allows depth information to be extracted from any 2-D image database or single camera sensors. We applied transfer learning to the NYU Depth Dataset V2 and the RMRC [?]challenge dataset of indoor images to form a model for single image depth prediction. Prediction on test images had a Root Mean Square Error of .833 which surpassed previous non-neural network methods.We also found that using upsampling/convolution instead of fully connected layers led to comparable results with 8 times less variables. Quantitative results show our predicted images contain course depth information of the objects in the input images. We attempt to explain the limitations of network by examining errors. Next, we attempted to run our model live on a Nvidia TK1 in real time.To test the model's ability to adapt to settings, we experimented with real time training for the model which improved performance in a localized setting.

2 Introduction:

Depth prediction from visual images is an important problem in robotics, virtual reality, and 3D modeling of scenes. With stereoscopic images, depth can be computed from local correspondences and triangularization. However, multiple images for a scene are often not available, leading to a need for [?] a system that can predict depth based on single images. Methods for inferring depth from a single image have involved supervised learning on segmented regions of images using features such as texture variations, texture gradients, interposition, and shading.Recent work using convolutional neural networks have shown promise in generating accurate depth maps from single 2-D images.

3 Exploration:

We explore the differences in training on an untrained network, and on a network pre-trained to the ImageNet Classificaion task. As a secondary goal, we will attempt to export our[?] neural network architecture and perform real-time depth perception on a Nvidia TK1 to enable real-time evaluation of scene depth. We also explore realtime learning to better understand how localized data helps to improve predictions in specific indoor settings.

4 Related Work:

4.1 In early 2010

Prior to the popularization of convolutional neural networks in early 2010s, several [?]groups worked on single image depth prediction by pairing supervised learning and feature engineering. Make3D divides each image into patches and then extracts features on texture variations, texture gradients, and haze used to represent both absolute depth and relative depth to other patches. Then the authors apply a Markov Random Fields (MRFs) to infer the depth of each path taking into account both the patch features and features relative to other patches.

4.2 In 2014

In 2014, released a hierarchical convolutional neural network model trained on the NYU Depth Dataset containing 407,024. The authors first feeds input data into a course model that predicts the course depths and then uses the output of the course model in a fine model to generate final prediction. With this model,[?] the authors are able to reach a root mean square error of .871 which improves dramatically from the RMSE error of 1.21 achieved by Make 3D. Figure 3 show results from the authors. As seen in the results, the course layers provides very general depth information and the fine layer on top provides the details. Interestingly, adding the fine layer increased the RMSE from .871 to .907. The authors of improved on their results further in using a three layer model aimed to learn depth information at three different scale. With the new model, the authors reached a RMSE of .641 on NYU Depth Dataset V2.

5 Dataset and Features:

For our dataset, we used the RMRC indoor depth challenge dataset consisting of RGB images of various indoor scenes and corresponding depth maps taken by Microsoft Kinect cameras. To increase our training data, we also used subsets of the NYU Depth Dataset V2. In total, we had 9900 RGB/Depth pairs which we split into 9160 training points, 320 validation points, and 320 test points. For each rgb/depth image,[?] we cut out 10 pixels on the border, because when the RGB and Depth images are aligned, the borders tend to have Nan values. Then we rescale each RGB image to 224x224 and each depth image to 24x24. Finally, we subtract the channels of the RGB image by (103.939, 116.779, 123.68) which are the channel means for the dataset that VGG net is trained on. We also perform data augmentation, in order to reduce overfitting. For our dataset, we randomly select images and generate rotated version of selected images with rotation angle ranging from -5 degrees to 5 degrees. Furthermore, we randomly select images and multiply each color channel by constants ranging from .95 to 1.05. Finally, for each image, there is a 50flipped image to be included in our dataset. In total, data augmentation increases our dataset size by a factor of 2.

6 Hardware:

To run our Keras models on live data, we purchased an Nvidia Tegra TK1 Development kit. The TK1 provides an NVIDIA Kepler GPU with 192 CUDA cores, and a Quad- Code ARM Cortex CPU. The board has 2GB RAM, split between GPU and CPU. We run Ubuntu 14.04 L4T, CUDA and CUDNN versions 6.5. To avoid hitting memory limitations, we added a 4GB swapfile on a high speed SD card. To stream video, we use a Microsoft Kinect v1. We use the Libfreenect library to retrieve and align both video and calibrated depth data. This allows us to display ground-truth depths along with predicted depths. This is the same camera used to collect the RMRC dataset to collect the training data we used. We built software to run and visualize the results in real time. We achieve frame rates around 1 fps. We also experimented with real-time training on captured images of scenes, by collecting data, performing preprocessing, and running training epochs in real-time.

7 Methods Used:

7.1 Make3d Method

7.2 Course Network Method

7.3 Course Fine Network Method

7.4 Hierarchical VGG Method

8 Output Screenshots

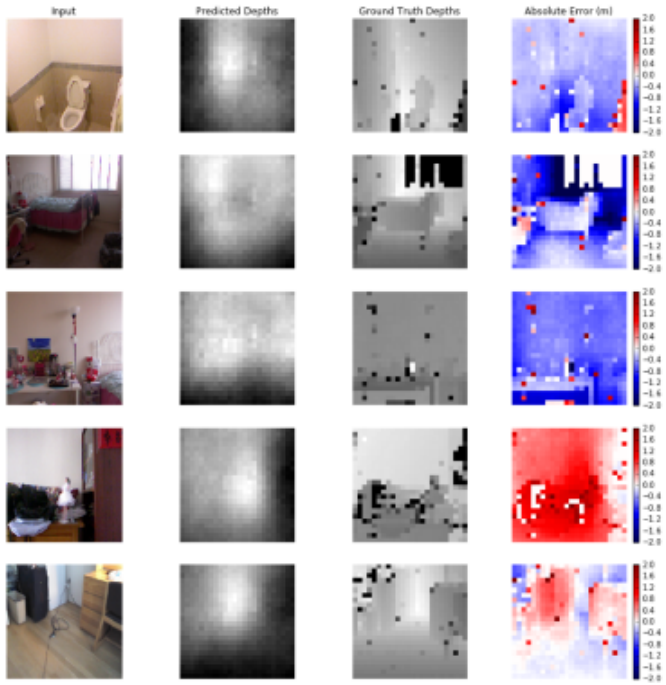

Figure 1: R
esults and errors of our network on our test dataset. From Left to Right: Input image, Our prediction, Ground-truth depth, Absolute error maps

9 Training Error Screenshot

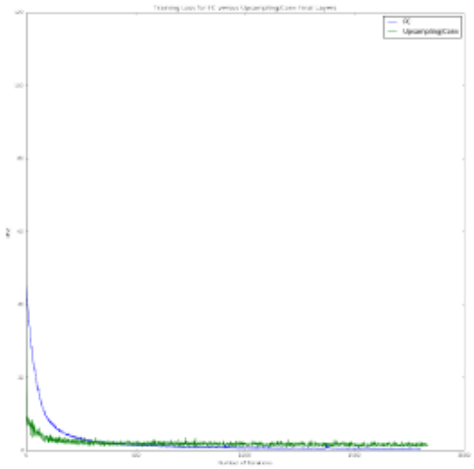

Figure 2: T
raining Error

10 Summary and Potential Improvements:

We have shown that transfer learning on convolutional neural networks can be effective in learning to estimate depth from single images. However, predicting detailed depth maps remains a hard regression problem requiring many training examples. Hierarchical models such as those in continue to out-perform our single-scale models. Adding segmentation data and surface normal data, as shown in may better separate objects whose depths may be obfuscated by very similar color or harsh lighting. We may have also benefited from varying the loss function; mentions that they could achieve outputs that better track the input data if they regularize their loss on the gradients of the log-differences. In hardware, we found difficulty engineering solutions to memory limitations in our hardware. To further decrease the memory representation of the fully connected layers, we could decrease the space and time needed to compute the feed-forward regression, and compute the SVD of the weigh matrix and drop the dimensions with low singular values instead.