

Depth Prediction with CNN on Sequential RGB Inputs

Alex Fu, Yannick Meier, Elena Chen, Nithin Poduval

1. Introduction

Depth prediction has been a problem as old as computer vision in computer science. The goal is to create a depth map based on single or multiple RGB image inputs. More specifically, given only the input of a standard RGB camera on mobile devices, i.e. mobile phones, tablets, etc, without extra inputs like depth sensor or stereo cameras, we would like to explore how to predict an accurate enough dense point cloud that represents the 3D geometry of real world topology. This topic is of great interests for real-world applications in augmented reality, automatic driving, robotics, 3D scanning, etc.

In this paper, we would like to demonstrate further performance improvement by using serial pictures taken for the same scene. As compared to past literature that also uses multiple images for prediction, we will experiment with modifying existing CNNs built specifically for monocular depth prediction.

2. Related Work

The initial solution to depth prediction was spatial triangulation based on pairs of consecutive pictures and camera poses (Szeliski, 2011). Current state of art literature predicts with single RGB image input, namely, monocular depth prediction. With the development of Convolutional Neural Networks (CNN), we have seen deep networks such as AlexNet and ResNet used to solve the depth prediction problem (Laina, Rupprecht, Belagiannis, & Tombari, 2016).

3. Dataset

ScanNet is a large RGB-D video dataset consisting of 2.5M images with semantic segmentation (Dai, Chang, Savva, Halber, Funkhouser, & Nießner, 2017). This dataset consists of 1,513 scenes of 707 unique indoor environments with estimated camera parameters, surface reconstructions, textured meshes and semantic segmentations provided, captured using the Microsoft Kinect v1. Each video scene has a large number of images focussed on the same room captured through varying camera angles. Figure 1 shows images that belong to the same scene. These images can be potentially combined in some form to provide the network more information about an object that can assist with its depth prediction. The diversity of spaces in the dataset and the instance-level semantic category of labels available made this dataset a natural choice for our problem.

3.1 Preprocessing

The dataset is shuffled and divided into batches before feeding into the network. In this analysis, we extracted 100 scenes from ScanNet and divide them into a training set of 400 images and a validation set of 100 images.

Figure 1: Examples of input images from the same sequence



4. Methods

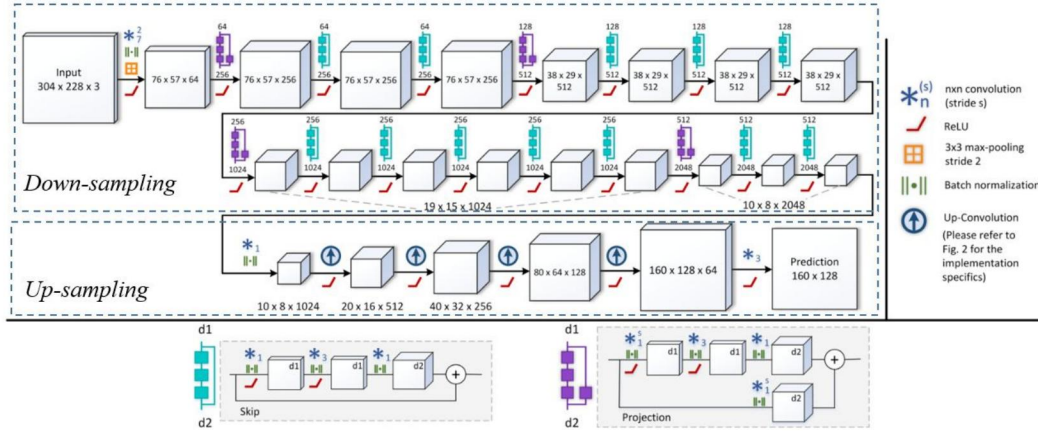
Convolutional neural networks consist of a contractive part that progressively reduces the dimension of the input image through successive convolutional and Maxpooling layers.

4.1. Architecture 1 - Base Line

In [1], ResNet-50[4] is utilized to down-sample RGB and reduce its the dimension from $[304 \times 228 \times 3]$ to a feature map of $10 \times 8 \times 2048$. ResNet-50 utilizes skip connections that help mitigate the problem of vanishing gradients that help neural networks become deeper. At this point, the FCNR (Fully convolutional residual network) utilizes upsampling and up-projection layers (upsampling layers with skip connections) to increase the dimension of the vectors to $160 \times 128 \times 1$, producing a depth map. The FCNR model was trained on the NYU dataset, which consist of indoor images similar to the ScanNet dataset. Therefore, we use the pre-trained weights the authors provide to make our baseline predictions on the ScanNet dataset.

We experiment with two architectures that make use of multiple input images to increase the receptive field and capture more global information. Figure 2 sets out details of the baseline model.

Figure 2: Network Architecture proposed by Laina et. al. (2016)



4.2. Architecture 2 - Stacking/Concatenation

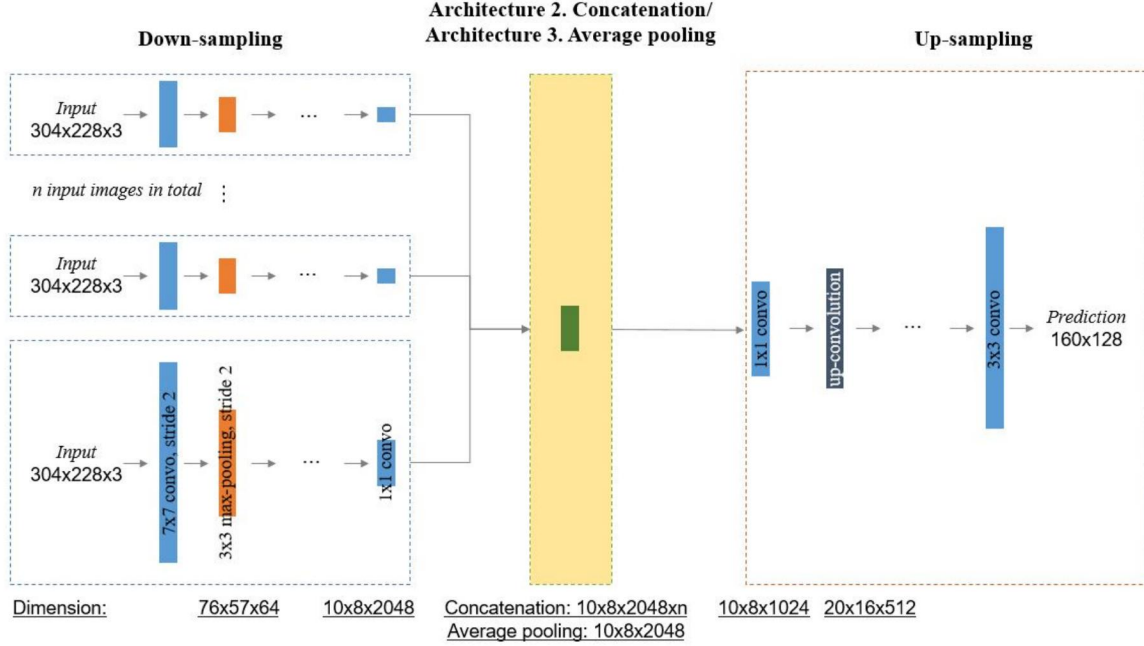
The idea of utilizing concatenation to increase the receptive field is discussed in literature. Rather than concatenating along the height or width of the image, we concatenate along the channels. The input images that are to be concatenated are first passed through the pretrained ResNet-50 layers, utilizing transfer learning and the vectors are extracted prior to the upsampling network. These images are concatenated along the channels axis. The dimension of the image at this layer grows from $10 \times 8 \times 2048$ to $10 \times 8 \times (2048 \times n)$, where n is the number of images we concatenate with and is tuned as a hyperparameter. The upsampling and up-projection half of the network architecture is kept the same as in our baseline architecture with the number of 1×1 convolution layers tuned as another hyperparameter. This network will be re-trained as there are $(2048 \times n)$ additional parameters to be learnt.

4.3. Architecture 3 - Average Pooling

In addition to concatenation, we have also explored another way of predicting depth from multiple images by averaging the value of corresponding pixel from each input image. The concatenated image will have the same

dimension as each of the input image. The structure of the up-sampling layers are the same as our baseline architecture with parameters re-trained.

Figure 3: Illustration of Architecture 2 and Architecture 3



5. Loss function/Optimization/Batch Normalization

In our analysis, we used the standard loss function, L_2 -loss, and minimized the sum of squared distances between the predicted depth map and the ground truth for each pixel. In terms of optimization algorithms, we started off with Gradient Descent and expanded to Momentum and Adam optimizers. Based on performance on our data set, Adam optimizer turned out to be the best among all three options. When training Architecture 2, 3 and 4, we have also applied the batch normalization technique.

6. Metric for evaluation

The metric we use for evaluation of different models is the mean square error validation error. This is obtained by computing the mean square error between the ground truth label and the prediction for every pixel and averaging it.

7. Results and Discussion

7.1 Model evaluation

Based on our results on the current dataset, Architecture 2, which utilizes concatenation of 5 images performs better than the baseline model and marginally better than the average pool architecture. We believe that concatenation of images over the channels axis leads to improved feature selection as opposed to averaging of pixels. We use this architecture to conduct further hyperparameter tuning experiments shown in Figure 6.

Figure 4: Depth map prediction on an example scene

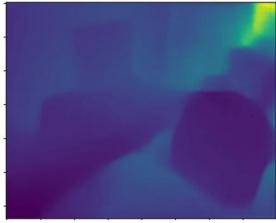
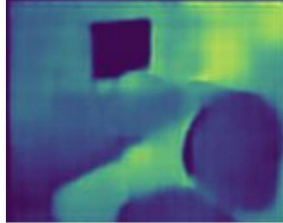
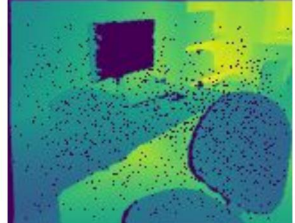
	Architecture 1. Baseline model	Architecture 2. Concatenation	Architecture 3. Average pooling
L_2 -loss (training)	N/A	0.12	0.28
L_2 -loss (validation)	3.3	1.8	2.4
Prediction (Depth map)			

Figure 5: L_2 -loss on training set

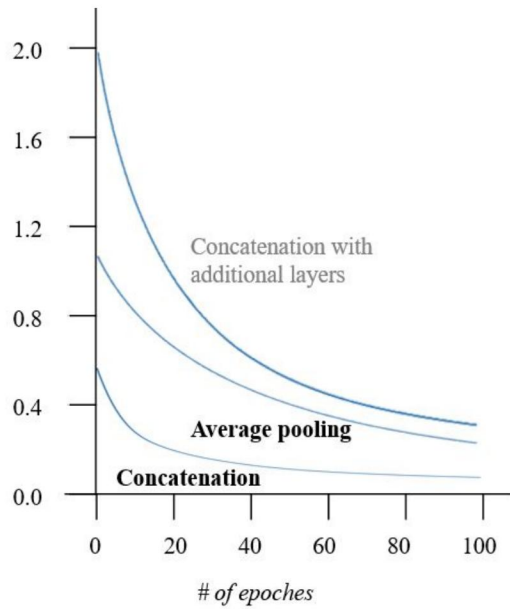
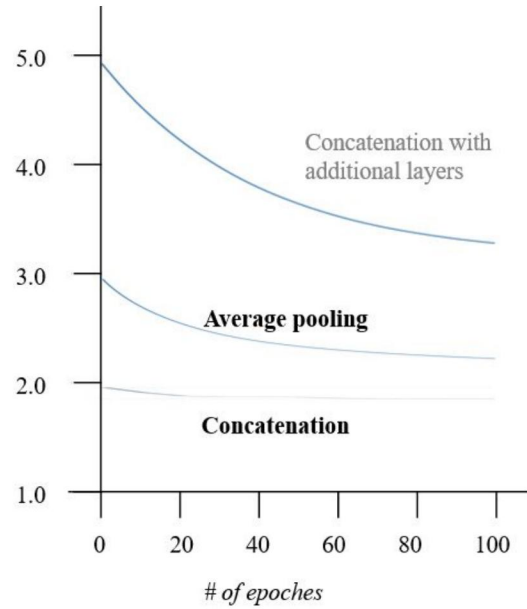


Figure 6: L_2 -loss on validation set



7.2 Hyper-parameter tuning

Figure 7: Hyper parameter tuning performance

Models	L_2 -loss (validation)
Architecture 2	1.8
3 images concatenated with no additional layers	2.3
5 images concatenated with two additional 1×1 convolution layers	1.9

7.2.1. Number of layers in the upsampling network

After stacking 5 images, channels dimension increases by 5 times. This implies up-sampling architecture from Laina et. al. (2016), there is a 90% reduction of dimension from 10,240 ($=2048*5$) to 1,024 after one 1x1 convolutional layer. Our hypothesis is that prediction accuracy may be compromised by such sharp reduction in dimension, or inflexibility of neural network. Therefore, we have experimented with adding one or two extra layers right after image stacking so that channel dimensions are reduced more gradually, namely, from 10,240 to 4,096 and then from 4,096 to 1,024.

It is observed that increasing the number of layers only leads to a marginal improvement in the mean square loss over simple concatenation. We believe this model needs more training.

7.2.2. Number of images that are concatenated or averaged over

Increasing the number of images that are concatenated can prove beneficial depending on the number of images in the dataset from the same sequence. Combining dissimilar images on the other hand will lead to the network losing information about the frame and should cause a drop in performance. The Scannet dataset has more than 10 images that look very similar to each other taken from different camera angles. Based on intuition, increasing the number of images that are concatenated from 3 to 5 should help improve performance and this is observed experimentally.

8. Conclusion/Future Work

Due to the small training set size, overfitting was observed. This was mitigated through the early stopping technique by stopping training when the validation error starts increasing. We would like to train the model on a much larger dataset to avoid the problem of overfitting. In our future research, we will also consider experimenting with other loss functions, e.g. the reverse Hber (berHu) loss, which was used by [1]. As next steps, we will use a larger training set and implement regularization methods like inverted dropout for further training of the up-sampling and 1x1 convolution layers.

We would also like to explore different ways to combine images from the training set as we have access to many images from a single scene. These images can be combined in different ways such as concatenating every alternate image or every 5 images as a means to introduce more variation in our series of iand capture more information.

Contribution

Alex Fu: She proposed project area and narrowed down the scope of this project to a specific manageable problem after much research on state of art literature. She implemented the training and testing pipeline for baseline model in Tensorflow according to the reference paper. She proposed the intuition behind using multiple images as input to the depth prediction model, and further proposed several possible architecture to experiment with. She contributed to the hyper-parameter tuning and model evaluation.

Yannick Meier: He wrote the code to test the baseline model, which predicts the depth map of a single image. He also helped modify the baseline model to make the prediction based on multiple images instead of just one and helped writing the training code to fit the weights of this modified model. He also contributed to the pre-processing of the dataset.

Elena Chen: She brainstormed design ideas for different architectures of neural networks and drafted one of the architecture choice in the milestone deliverable. She also led the drafting of final report. She created, designed and printed the poster.

Nithin Poduval: He wrote the code to train the baseline model and helped modify it to read in a large dataset.. He also modified the code to average pool over multiple images which is the third architecture discussed in the paper. He helped brainstorm and translate the vision of using multiple input images to improve performance into two specific architecture ideas. He contributed towards dataset downloading, pre-processing and preparation of the validation and test set. He also worked on hyperparameter tuning and model evaluation.

Github link: <https://github.com/futuyao/CS230Project>

Reference

R. Szeliski (2011), Structure from motion. In *Computer Vision* (pp.303–334). Springer

[1]Laina, I., Rupprecht, C., Belagiannis, V., Tombari, F., & Navab, N. (2016, October). Deeper depth prediction with fully convolutional residual networks. In *2016 Fourth international conference on 3D vision (3DV)* (pp. 239-248). IEEE.

[2]Dai, A., Chang, A. X., Savva, M., Halber, M., Funkhouser, T., & Nießner, M. (2017). Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 5828-5839).

[3]Zwald, L., & Lambert-Lacroix, S. (2012). The berhu penalty and the grouped effect. *arXiv preprint arXiv:1207.6868*.

[4]K. He et al (2015),Deep Residual Learning for Image Recognition,