Deep Learning for Computer Vision: Final Project Proposal (Learning depth maps from single RGB image)

Neha Das
neha.das@tum.de

Saadhana Venkataraman
saadhana.venkataraman@tum.de

SanYu Huang
ga59hoc@tum.de

Sumit Dugar
sumit.dugar@tum.de

Proposal I

1. Introduction

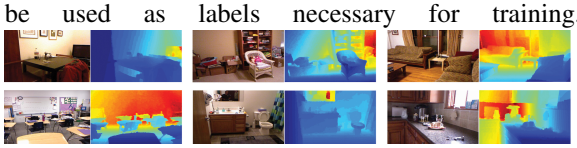
We are aiming to generate Depth Images from Single RGB images following the approach outlined by the paper "Learning Fine-Scaled Depth Maps from Single RGB Images"

1.1. Related Works

In recent years, deep learning methods have shown their outstanding performance in vision-based tasks. In fact, predicting depth information for 3D reconstruction from monocular camera represents one of very important topics. D. Eigen et. al.[1] proposed a single multiscale convolutional network architecture dealing with depth prediction, surface normals and semantic labels. In this work, each scale possesses different sizing of the image and specific kernel types. As to depth prediction, NYUDepth v2 was selected for the training part. Following that, Jun Li et. al.[2] inherited the same idea for estimating depth from a single RGB image. Beyond creating finer architecture literally, significant speedup for convergence by adding skip layers has been claimed. A new loss function, jointly over images, was also proposed for improving accuracy. Shortly, multi-scale convolutional network has great potemtial in general.

2. Dataset

- We will be working with the existing datatset from RMRC indoor depth challenge and a subset of NYU Depth Dataset V2.
- Dataset consists of 4105 RGB and Depth image pairs of indoor scenes.
- Depth images from the dataset would be used as labels necessary for training.



- RMRC dataset is not large enough for training and NYU dataset is too large(400GB). We are thinking of using RMRC plus some training samples from NYU dataset. We may also apply some data augmentation techniques to increase our dataset size. NYU dataset is in the raw format so we will need to do pre-processing as well.
- Input is a pair of rgb/depth image and output is a corresponding depth image.

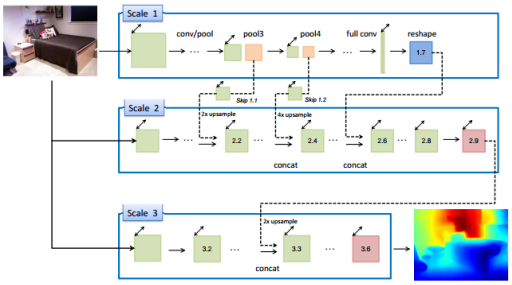
3. Methodology

- We intend to predict a pixel-wise depth map using a learning based approach. For this, we use a three level (corresponding to three scales) coarse to fine CNN with four layer fusions. Each stage corresponds to a different scale.

  a. The first level has five Convolution layers and two fully connected layers and provides a global view of the scene. We would apply transfer learning on the five Convolution layers where the weights are set from a pre-trained VGG net.

  b. The second level has nine Convolution layers with three layer fusions, two from skip layers and one from the final output of level-a. The output of this level would be a coarse depth map.

  c. The third level had six Convolution layers with one layer fusion from the final output of level-b. The expected output here is a one-channel depth image.



- All layers except the Convolution layers in level-a are trained from the scratch. In the original paper, the network was trained on a subset of NYU Depth v2 dataset of about 225K images using a TITAN X GPU. The training took around 50 hours. We will be working on a much smaller dataset of about 12K images.

4. Outcome

We are aiming to generate accurate depth image given a single RGB image.

References

[1] Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture:
http://www.cs.nyu.edu/ deigen/dnl/

[2] Learning Fine-Scaled Depth Maps from Single RGB Images:
https://arxiv.org/pdf/1607.00730.pdf

[3] Depth Map Prediction from a Single Image using a Multi-Scale Deep Network:
https://arxiv.org/pdf/1406.2283.pdf

[4] RMRC indoor depth challenge dataset:
http://cs.nyu.edu/ silberman/rmrc2014/indoor.php

[5] NYU depth dataset v2:
http://cs.nyu.edu/ silberman/datasets/nyu_depth_v2.html

Proposal II

1. Introduction

Our goal for the backup proposal will be to implement the object detection and instance segmentation approach outlined in the paper "Masked R-CNN"

1.1. Related Works

Since this work outputs both object bounding boxes and pixel-wise labels, it is closely related to instance segmentation. It yields significantly better results over its predecessors (R-CNN, Fast-R-CNN and Faster-R CNN), owing to the replacement of the ROI Pool layer by the ROI align layer used in this approach. It also yields better result than one shot object detection techniques such as SSD and YOLO, albeit longer inference times.

2. Dataset

- We will be working with the existing data set from the COCO dataset. This dataset contains 80K Training Samples, 40K Validation Samples and 80K Testing Images (from Challenge 2015). We will be using a only a subset of the testing samples

- COCO currently has three annotation types: object instances, object keypoints, and image captions. The annotations are stored using the JSON file format.

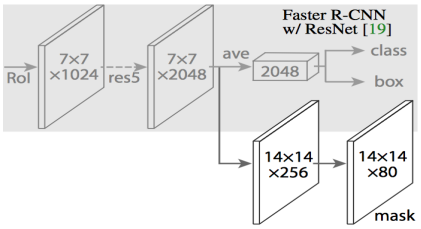
- Challenges: The dataset is very large. The training, validation and the test dataset add up to 32 GBs

- The inputs shall be RGB images. As an intermediary output we predict the captioned bounding boxes on the detected objects. For the final result, we should predict pixel wise labels for the input image.

3. Methodology

The architecture is largely similar to that of "Faster R-CNN", but adds another output branch that outputs the object mask.

- Network architecture: The architecture for masked R-CNN can be divided into two parts: a convolutional backbone (ResNet-50 upto the 4th conv layer), 3 network heads (1 each for classification, bounding boxes and masks). We feed the ROI output to the the backbone.



- Transfer learning: For the convolutional backbone we use a pretrained ResNet-50 uptothe 4th conv layer.

- Resource management: The original implementation This model runs at 195ms per image on an Nvidia Tesla M40 GPU (plus 15ms CPU time resizing the outputs to the original resolution). It also uses 8 GPUs

4. Outcome

Benchmark results for segmentation on the COCO test set or better

References

[1] Mask R-CNN :
https://arxiv.org/pdf/1703.06870.pdf

[2] COCO Dataset :
http://mscoco.org/dataset/download