MultitaskNet: Joint Depth Estimation and Semantic Segmentation using Deep Convolutional Network

Prateek Rajvanshi[1], Vinod Kumar[1]

Abstract

Semantic segmentation and depth estimation are two important tasks in computer vision, and many methods have been developed to tackle them. Commonly these two tasks are addressed independently, but recently the idea of merging these two problems into a sole framework has been studied under the assumption that integrating two highly correlated tasks may benefit each other to improve the estimation accuracy. In this work, we developed a hybrid convolution network for simultaneous depth estimation and semantic segmentation from a single RGB image. The proposed model is trained and evaluated on SUNRGB dataset [1] achieving results that are not too far from the state of the art methods on both the tasks.

1. Introduction

It is harder for practitioners to deploy multiple deep learning models to perform different tasks in a single system. Such a system requires huge computational power, a large memory, training and dataset overhead. For some closely related tasks like image segmentation, image classification etc., such a deployment does not present a significant obstacle as they may share a similar dataset. But tasks like semantic segmentation and depth estimation rarely share the same datasets and hence if a complex system needs to be developed which can perform both these tasks, we will have to deploy two separate networks. With this project, we aim to demonstrate that a single model can perform as well on multiple tasks as separate model for each of these tasks.

Our choice of semantic segmentation and depth estimation tasks is motivated by an observation robot navigation task. In robotics, performing tasks in interactive environments requires identification of objects as well as their distance from the camera. Likewise, autonomous navigation applications need a 3D reconstruction of the scene as well as semantic information to ensure that the agent device has enough information available to carry out the navigation in a safe and independent manner. Although RGB-D sensors are currently being used in many applications, most systems only provide RGB information.

Therefore, addressing depth estimation and semantic segmentation under a unified framework is of special interest. Semantic segmentation is a per pixel task, aims at giving a class label for each pixel on the image according to its semantic meaning. Depth estimation is another per-pixel task, the goal of which is to determine how far each pixel is from the observer.

2. Related Work

Our work touches several areas of research - multi-task learning, semantic segmentation and depth estimation. As per multi-task paradigm, if a model is forced to perform several related tasks simultaneously, it can improve the generalization of the model by imposing inductive bias on learned representations [2], [3]. In [4], the authors prove that a network for a particular task can be obtained by fine tuning a network which is trained for some other but correlated task. It describes a network which can be common for different tasks, including the estimation of depth map, surface normals, and semantic segmentation. The results obtained by [4] outperforms the ones presented in [5] and proves how the strategy of tackling multiple tasks with a common network may lead to a better performance.

Eigen & Fergus [7] performed three tasks - predict depth, surface normals and semantic segmentation by training a single architecture.In [8] a universal network is proposed which tackles 7 different vision tasks. A method to do joint semantic segmentation and object detection was proposed by Dvornik et al. [9]. Chen et al. [10] built a single network with the ResNet-50 [11] backbone performing joint semantic segmentation, depth estimation and object detection. In [12] proposed an approach to make global and local prediction using unified network where the consistency between depth and semantic segmentation is learned through a joint training process.

[1]Georgia Institute of Technology, Atlanta
[2]Code repo: https://github.com/imvinod/MultitaskNet

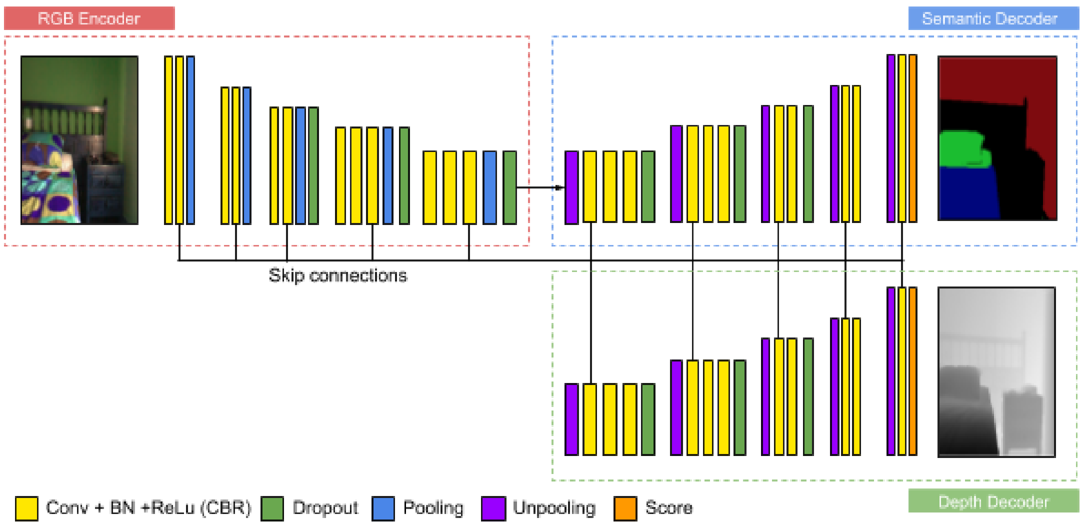

Figure 1. The proposed MultitaskNet architecture

3. Approach

3.1 Model

We propose an encoder-decoder type network architecture as shown in Figure 1. The proposed network has two major parts: 1) the encoder part extracts features and 2) the decoder part upsample the feature maps back to the original input resolution.

The encoder part of the network resembles the 16-layer VGG net [13], except for the fully connected layers *fc6, fc7* and *fc8*, since the fully connected layers reduce the resolution with a factor of 49, which increases the difficulty of the upsampling part.

We define a CBR block which is a combination of convolution, batch normalization and ReLU. We use batch normalization (BN) after convolution (Conv) and before rectified linear unit (ReLU) to reduce the internal covariate shift. The BN layer first normalizes the feature maps to have zero-mean and unit-variance, and then scales and shifts them afterwards.

The decoder part of the network has two main modules: one for semantic segmentation and other for depth estimation. where memorized unpooling is applied to upsample the feature maps. In the decoder part, we again use the CBR blocks.

We model the network as a composition of functions corresponding to L layers with parameters denoted by $W = [w^{(1)}, w^{(2)}, \ldots., w^{(L)}]$, that is

$$f(x; W) = g^{(L)}(g^{(L-1)}\left(.. g^{(2)}\left(g^{(1)}\left(x; w^{(1)}\right); w^{(2)}\right) .; w^{(L-1)}\right); w^{(L)})$$

Our dataset consists of RGB images with label set as $L = \{1, 2, .. K\}$. We assume that we are given a training set $\{(X_i; Y_i) \mid X_i\}$ of shape $H * W * 3$ and Y^i of shape $H * W$ for all $i = 1...m$ consisting of m three-channel RGB images (X_i), having the same size $H * W$, along with the ground truth label (Y^i)

3.2 Loss Function

We use the standard softmax cross entropy loss for segmentation and the inverse Huber loss for depth estimation [14]. Our total loss contains an additional scaling parameter, λ, which, for simplicity, we set to 0:5:

$$Loss(total) = \lambda * Loss_{semantic} + (1 - \lambda) * Loss_{depth}$$

3.2.1 Semantic Loss

For semantic segmentation module the network outputs a response map with the dimension of $C * H * W$ where C is the number of semantic classes and H, W are the height and width of input image. We use standard softmax cross entropy loss for segmentation which minimizes KL-divergence between the predicted and the true class distribution. All positions and labels of the output class score map are equally weighted in the overall loss function except for those unlabeled pixels which are ignored. The final output then has as many channels as there are classes.

$$L_{sem} = -\sum_{i=1}^{N} \log\left(p(C_i^* | z_i)\right)$$

where C_i^* is the ground truth label of pixel i, $p(C_i|z_i) = \frac{e^{z_i}}{\sum_c e^{z_{i,c}}}$ is the probability of estimating semantic category of C_i at pixel i and $z_{i,c}$ is the output of the response map.

3.2.2 Depth Loss

A depth map for an image is simply a 2-D matrix with the same size as the input image, that contains information relating to the distance of the surfaces of scene objects from the camera viewpoint. For depth prediction, we use a loss function comparing the predicted and ground-truth log depth maps D and D*. Letting d = D − D* be their difference, we set the loss to

$$L_{depth}(D, D^*) = \frac{1}{n}\sum_i d_i^2 - \frac{1}{2n^2}\left(\sum_i d_i\right)^2$$

$$+ \frac{1}{n}\sum_i[(\nabla_x d_i)^2 + (\nabla_y d_i)^2]$$

$D = log(Predicted\ depth\ map)$

$D* = log(Ground\ truth\ depth\ map)$

$d = D - D^*$

$d_i = D_i - D^i = difference\ between\ predicted$

$and\ ground\ truth\ depth\ for\ pixel\ i$

$n = total\ number\ of\ pixels$

where the sums are over valid pixels i (we mask out pixels where the ground truth is missing). Here, $\nabla_x d_i$ and $\nabla_y d_i$ are the horizontal and vertical image gradients of the difference.

We used the $L2$ and scale-invariant difference terms in the first line, similar to [21]. However, we also include a first-order matching term $(\nabla_x d_i)^2 + (\nabla_y d_i)^2$, which compares image gradients of the prediction with the ground truth. With this approach of calculating loss, predictions not only have close-by values, but also similar local structure.

4. Experiments and results

We followed an iterative approach for building the network architecture. The goal was to get an architecture from end to end (encoder - decoder) working and then try to improve the performance.

4.1 Training Details

Our dataset consists of RGB images which were re-sized to the resolution of 224x224 and then we applied bilinear interpolation. We applied nearest-neighbor interpolation on the depth images and the ground-truth labeling. The networks were implemented with the Pytorch frame-work and were trained with stochastic gradient descent (SGD) solver using a batch size of 4. The input data was randomly shuffled after each epoch. The learning rate was initialized to 0.001 and was multiplied by 0.9 in every 50,000 iterations. We used a momentum of 0.9 and set weight decay to 0.0005. We trained the networks until convergence, when no further decrease in the loss was observed. Encoder part of the network consists of VGG-16-layer model which was pretrained on the ImageNet database.

We discuss here some of the experiment milestones. Although we performed several more experiments, these experiments showed us clear indications of the strengths and weakness that we could rectify in the next stage of the experiment.

4.1.1 Experiment 1

We used pre trained VGG 16 without last few layers as encoder. Three stages of Unpool-CBR-Dropout, two stages of Unpool-CPR and a score layer formed the decoder. The outputs are the semantic predictions of size $224\ x\ 224\ x\ labels$. The SUN-RGBD dataset has 38 labels. For simplicity we used the same architecture for depth. The depth output is of size $224\ x\ 224\ x\ 1$. We used a subset of 1000 training dataset to overfit the model for sanity check. After 10 epochs, the semantic labels and depth predictions were visually discernible. Next, We trained the model with training dataset of size 8000 for 50 epochs and the $mIOU$ was 12.4 and the depth $RMSE$ was 188.43.

4.1.2 Experiment 2

We realised that the network was deep. We added skip connections between each CPR block in the encoder and decoder to avoid the problem of vanishing gradients. Since, one of the goals of the MultitaskNet is to have a comparatively lighter model, adding several skip connections was not ideal as it increased the number of parameters in the decoders significantly. We wanted to achieve improvement in performance and subsequently reduce the number of skip connections without hampering the performance. After 50 epochs with 8000 training size, the $mIOU$ recorded was 22.68 and $RMSE$ of 57.23.

4.1.3 Experiment 3

So far we had been using SGD for optimization. For this experiment we used Adam optimizer $(beta1: 0.9, beta2: 0.999, eps: 1e-8$ $weight_{decay}: 0.0005)$. After every 25 epochs we reduced the learning rate by a tenth. This seemed to help reduce the losses for a few subsequent epochs. We trained the model with training dataset of size 8000 for 50 epochs and the $mIOU$ was 25.1 and the depth $RMSE$ was 55.66.

4.1.4 Experiment 4

At this point, we observed that the depth loss was still reducing whereas the semantic loss was stagnant. So far the lambda $\lambda = 0.5$ which meant both semantic loss and depth loss weighed equally. We changed $\lambda = 0.3$. The weight of depth loss was now 0.7 and weight of semantic loss was 0.3. We trained for 50 epochs and $mIOU$ was 20.3 and the depth $RMSE$ was 54.35. Setting $\lambda = 0.3$ reduced both $RMSE$ but significantly hampered $mIOU$.

Batch	LR	λ	ep	Opt	RMSE	**mIOU**
4	1e-4	0.5	50	SGD	188.43	12.4
4	1e-4	0.5	50	SGD	57.23	22.68
4	1e-5	0.5	50	Adam	55.66	25.10
4	1e-5	0.3	50	Adam	54.35	20.3

Table 1: Important hyperparameters

4.1.5 Constraints

The training size was 8000 images and the validation size was 2000 images. Since the size of the dataset is comparatively small, the learning is limited and the model tends to overfit. It is hard to find a larger dataset that has both depth and semantic labels. We need both the depth and semantic labels to be of equal size to ensure that the model learns to perform equally well on both the tasks.

With limited processing power, a training session of 50 epochs takes about 15 hours. Thus, we could only run a limited number of experiments and each experiment had an overhead of making sure the model was indeed learning by running a small number of epochs before starting a long training session.

4.2 Results

In our experiment, we consider SUN-RGB dataset[1], representing indoor setting of a room with various furniture. The dataset set was split into training and testing in a 60:40 ratio. We fine tuned the network with this training data set and updated the parameters. After training is completed, we used the test dataset to measure the performance of our model.We divided our result into two categories- first one is quantitative result where we could compare our model performance directly with state of the art methods in terms of the error. The second type of result is qualitative, where we see the actual output image and compare it with the ground truth.There will be two output image for each input image- one is segmented image and the other one is depth map image.

4.2.1 Quantitative Results:

For the reported $mIOU$ the higher the better, whereas for the reported RMSE the lower the better. Quantitatively, we are able to achieve 25:2% $mIOU$ and 55.66 $RMSE(lin)$ on the validation set. Our depth estimation rmse is comparable to all the state of the art methods. In semantic segmentation, mIoU is lower compared to other approaches. This also matches with qualitative results (figure 2 and figure 3) where our segmentation results are not so good for images with complex setup but depth estimation similar to ground truth for most of the inputs.

	Semantic Segmentation		Depth Estimation	
Model	Regime	mIoU %	RMSE(lin) m	RMSE (log)
RefineNet – 101 [15]	Segm	43.6	-	-
Context [16]	Segm	40.6	-	-
Kendall and Gal [17] □	Segm, Depth	37.3	0.506	-
Ours □ MultitaskNet	**Segm, Depth**	**25.2**	**0.556**	**0.164**
Laina et al. [14]	Depth	-	0.573	0.195
Qi et al. [18]	Depth	-	0.569	-

Table 2: Results comparison, mIoU the higher the better, whereas for the reported RMSE the lower the better. (□) indicates that both tasks are performed simultaneously. using a single model

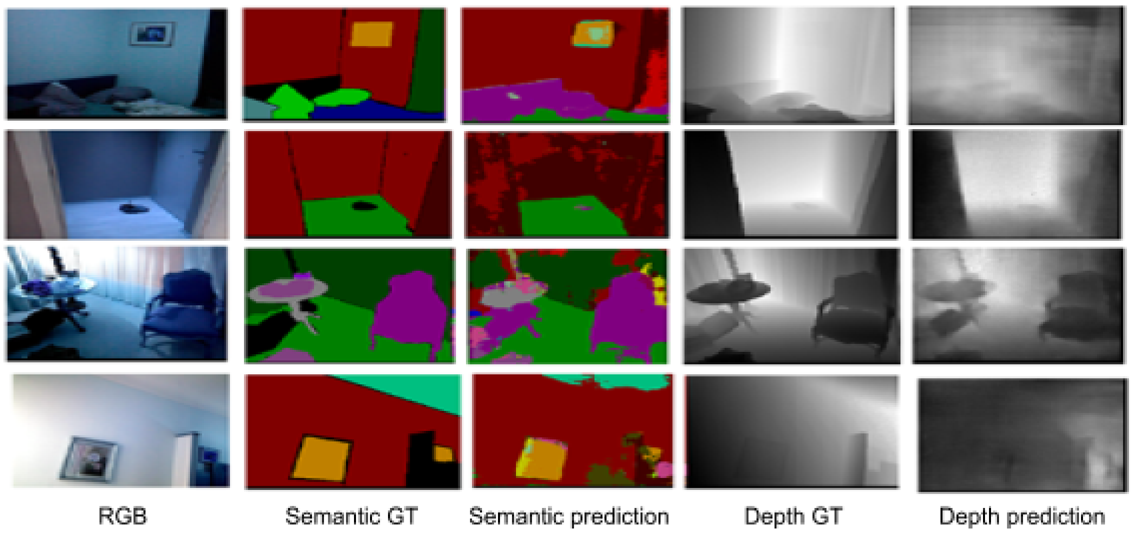

Figure 2 : Qualitative results on segmentation and depth estimation task after 50 epochs are comparable with ground truth

Conclusions

We showed an approach how we can perform semantic segmentation and depth estimation jointly in a single network which is trained in stages. We then fine tuned the network parameters using a single loss function. Our loss function is calculated as weighted sum of segmentation loss and depth loss with weight parameter lambda (λ). Our proposed network model uses VGG-16 layer network on the encoder side. On decoder side we have two branches one for semantic segmentation and depth estimation each.

Our proposed approach gives comparable quantitative results (Table 2) for segmentation task w.r.t state of the art methods. As can be seen from qualitative results (Figure 2.). Our depth estimation quantitative results (Table 2) outperformed some approaches like kendall and Gall [17].

Future Work

Designing better loss functions for a multi-task learning : Most of the state of the art methods calculate loss by linear combination of loss from all the tasks. However, these losses may have different physical meaning (cross entropy, euclidean loss) and combining such losses is hard. We need to devise some higher level evaluation metric which can combine various losses in a meaningful way. For instance, the task for prediction of the 3D oriented bounding box of objects uses both semantic segmentation and depth estimation results, and naturally combines the loss function for both tasks.

Skip connection: Currently we have added skip connections between every module of the encoder with that of the decoded. This leads to a large number of parameters and is unsuitable for real-time applications. Removing some of the skip connections while not sacrificing the performance significantly is worth exploring.

Knowledge distillation: We may also explore the practice knowledge distillation and directly incorporate a large pre-trained teacher's (expert) learnings during the pre-training stage. We may use a lighter model as student. This may not only improve the performance but also significantly reduce the number of parameters and make the model light-weight.

Acknowledgments

We created a hybrid network in which encoder was inspired by model architecture in [19]. The concept of using two branches for two separate tasks in decoder was referred from Real-Time Joint Semantic Segmentation and Depth Estimation Using Asymmetric Annotations [20].

Our code structure and utilities is referenced from the implementation of FuseNet-based CNN architecture[19]. Their official code is hosted on Github: https://github.com/tum-vision/fusenet.

References

[1] Sun RGB dataset : http://rgbd.cs.princeton.edu/

[2] R. Caruana, “Multitask learning: A knowledge-based source of inductive bias,” in ICML, 1993. 1

[3] J. Baxter, “A model of inductive bias learning,” J. Artif. Intell. Res., 2000.

[4]. Eigen, D.; Fergus, R. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 13–16 December 2015; pp. 2650–2658

[5]. Eigen, D.; Puhrsch, C.; Fergus, R. Depth map prediction from a single image using a multi-scale deep network. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–8 December 2014; pp. 2366–2374

[6]. Mousavian, A.; Pirsiavash, H.; Košecká, J. Joint Semantic Segmentation and Depth Estimation with Deep Convolutional Networks. In Proceedings of the Fourth International Conference on 3D Vision (3DV), Stanford, CA, USA, 25 -28 October 2016; pp. 611–619.

[7] D. Eigen and R. Fergus, “Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture,” in ICCV, 2015

[8] I. Kokkinos, “Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory,” in CVPR, 2017.

[9] N. Dvornik, K. Shmelkov, J. Mairal, and C. Schmid, “Blitznet: A real-time deep network for scene understanding,” in ICCV, 2017.

[10] L. Chen, Z. Yang, J. Ma, and Z. Luo, “Driving scene perception network: Real-time joint detection, depth estimation and semantic

[11] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in CVPR, 2016. segmentation,” in WACV, 2018.

[12] Wang, P.; Shen, X.; Lin, Z.; Cohen, S.; Price, B.; Yuille, A.L. Towards unified depth and semantic prediction from a single image. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 8–10 June 2015; pp. 2800–2809.

[13] Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. Proceedings of International Conference on Learning Representations (2015)

[14] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, “Deeper depth prediction with fully convolutional residual networks,” in 3DV, 2016.

[15] G. Lin, A. Milan, C. Shen, and I. D. Reid, “Refinenet: Multi-path refinement networks for high-resolution semantic segmentation,” in CVPR, 2017.

[16] G. Lin, C. Shen, I. D. Reid, and A. van den Hengel, “Efficient piecewise training of deep structured models for semantic segmentation,” CoRR, vol. abs/1504.01013, 2015.

[17] A. Kendall and Y. Gal, “What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?” in arXiv:1703.04977, 2017.

[18] X. Qi, R. Liao, Z. Liu, R. Urtasun, and J. Jia, “Geonet: Geometric neural network for joint depth and surface normal estimation,” in CVPR, 2018.

[19] FuseNet: Incorporating Depth into Semantic Segmentation via Fusion-based CNN Architecture

[20] Real-Time Joint Semantic Segmentation and Depth Estimation Using Asymmetric Annotations

[21] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. NIPS, 2014