Soccer Robot Perception

Lab Vision Systems: Cuda Vision – Learning Computer Vision on GPUs (MA-INF 4308)

Jaswanth Bandlamudi,
Venkata Santosh Sai Ramireddy Muthireddy

Hochschule Bonn-Rhein-Sieg
jaswanth.bandlamudi@smail.inf.h-brs.de, Matrikelnummer: 9034908
santosh.muthireddy@smail.inf.h-brs.de, Matrikelnummer: 9035746

Abstract. Object detection and scene segmentation are key problems in perception stack of autonomous systems including humanoid robots. In this report, we implemented NimbroNet2, a unified deep learning architecture capable of solving both detection and segmentation problems. The NimbroNet2 is trained to detect objects like soccer ball, goal post and other robots on field, as well as to segment the scene into background, field and lines. We used f1 score, accuracy, precision, recall and false detection rate to quantify detection performance (FDR), whereas segmentation performance is quantified using accuracy and Intersection over Union (IoU).

1 Introduction

According to RoboCup federation "By the middle of the 21st century, a team of fully autonomous humanoid robot soccer players shall win a soccer game, complying with the official rules of FIFA, against the winner of the most recent World Cup." [10]. Perceiving the environment of soccer field is critical for the operation of soccer playing humanoid robots. The perception stack includes object detection and scene segmentation. The task of object detection includes detecting of ball, goal posts and robots. On the other hand, segmentation task includes pixel-wise segmentation of background, field and lines. These tasks has to be carried out with high accuracy and should be robust to adverse lighting conditions.

Deep learning methods has been out-performing the classical computer vision based methods in object detection [8] and semantic segmentation tasks [12]. In RoboCup 2019, Team NimbRo implemented NimbroNet an unified encoder-decoder based deep architecture with detection and segmentation heads. In this work, we implemented NimbroNet2 and trained it on the dataset provided by Team NimbRo [11]. The dataset consists of different subsets for detection and segmentation. Sample data for detection is shown in Figure 1 and for segmentation is shown in Figure 2.

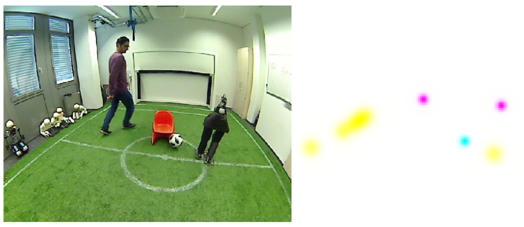


Fig. 1: Sample from detection dataset. RGB image on left and ground truth label on right.

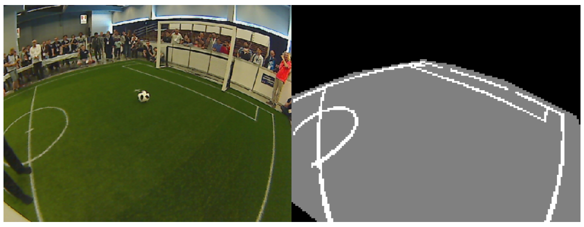


Fig. 2: Sample from segmentation dataset. RGB image on left and ground truth mask on right.

2 Related Work

Object detection and Segmentation has been one of highly studied problems by various researchers in the field of computer vision. This study is further boosted by the advent of deep learning. The state of the art methods to solve object detection problem are Faster-RCNN [9], You-Only-Look-Once [8], Single Shot Detector [6]. while semantic segmentation is solved by SegNet [1], DeepLabV3+ [3] and U-Net [12]. Consequently, these networks requires high computational resources and are mostly not suited to perform on edge devices with tight resource constraints.

In order to suit the low power applications like mobile or humanoid robot researchers proposed the usage of Tiny-YOLO [5], YOLO-lite [4] for object detection and Compact-UNet [2] for segmentation purposes. Using independent networks for different tasks is still power intensive. Hence, researchers proposed SweatyNet [13] which is a encoder-decoder based architecture with skip connections. Bilinear upscaling is used in decoder instead of transpose convolution layers as the transpose convolution layers produce checker board artifacts. This work is heavily inspired by the implementation of Sweatynet [13] and is detailed in the following section.

3 Implementation

In this section, we present NimbroNet2 architecture, datasets, loss functions and evaluation metrics used in the experiments.

3.1 Architecture

As mentioned in Section 1, the model should be robust to different conditions. Rodriguez, Farazi, Ficht et al [11] proposed a robust architecture to provide both detection and segmentation output in a single pass. Nimbronet2 consists of an encoder and a shortened decoder to achieve computational efficiency. The encoder is a pre-trained ResNet18 without Global Average Pooling (GAP) and fully connected layer. Whereas, decoder is a model made up of Transpose convolution layers cascaded with Batch Normalization and ReLU activation layers as shown in Figure 3. Location aware convolution layers proposed by [7] are used at the end of decoder to extract location dependent features. These location aware convolution layers are provided with same biases for both detection and segmentation heads to reduce the number of learning parameters.

The model has two output heads, one for detection and another for segmentation. Detection head outputs the location of the ball, goal posts and other robots as a heatmap and segmentation head gives the semantic labels of background, field and lines as a mask.

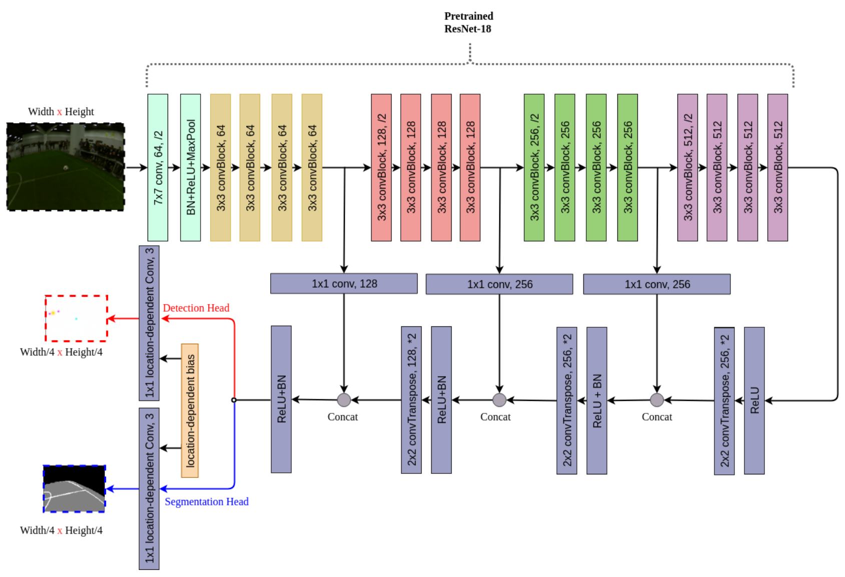


Fig. 3: NimbroNet2 architecture for simultaneous detection and Segmentation

3.2 Datasets

The dataset used in the work [11] is provided and is used in this work. The dataset consists of two subsets for detection and segmentation. The detection dataset consists of 8858 RGB images with labels in VOC format. The segmentation dataset consists of 1192 RGB images and their respective semantic masks as labels. We have split datasets into 70% train, 15% validation and 15% test sets.

The given datasets are pre-processed as explained below:

- All the input RGB-images are resized into $(480 \times 640 \times 3)$ size.
- VOC format labels provided in detection dataset are used to generate heatmaps with Gaussian blobs with mean at the center of the bounding box for respective objects.
- Gaussian blobs are generated with covariance of 5 for ball and goal post classes, where covariance of 10 is used for robot class.
- The class labels are encoded in different colors and sizes as shown in Figure 1.
- The pixels corresponding to robot and ball are mapped to field in semantic masks.
- All the semantic masks are resized into $(120 \times 160 \times 1)$ size.

3.3 Loss functions

For detection task, Mean Square Error (MSE) loss as shown in equation 1 is employed, where Y is the ground truth and $\hat{Y}$ is model prediction and n is the number of samples present.

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} \left(Y_i - \hat{Y}_i \right)^2 \tag{1}$$

For segmentation task, Negative Log Likelihood (NLL) loss as shown in equation 2 is employed, where Y is the ground truth and $\hat{Y}$ is model prediction and n is the number of samples present.

$$\text{NLL} = - \sum_{i=1}^{n} Y_i \log \hat{Y}_i \tag{2}$$

Total Variation (TV) loss is added to detection and segmentation losses except to the line channel in segmentation task. "Total variation is the sum of the absolute differences for neighbouring pixels in the output images" [tf.image.tvloss]. Total variation loss is calculated using the equation 3, where Y is the given input image and i, j gives the pixel locations.

$$\text{TV(Y)} = \sum_{i,j} |Y_{i+1,j} - Y_{i,j}| + |Y_{i,j+1} - Y_{i,j}| \tag{3}$$

3.4 Evaluation metrics

To quantify detection heads performance F1 score, accuracy, precision, recall and False Detection Rate (FDR) are used as evaluation metrics.

$$\text{Precision} \; = \frac{TP}{TP+FP} \tag{4}$$

$$\text{Recall} \; = \frac{TP}{TP+FN} \tag{5}$$

$$\text{Accuracy} \; = \frac{TP+TN}{TP+TN+FP+FN} \tag{6}$$

$$\text{F1 score} \; = \frac{2 \times Precision \times recall}{Precision + Recall} \tag{7}$$

$$\text{FDR} \; = 1 - Precision \tag{8}$$

Where TP is True Positives, FN is False Negatives, FP is False Positives and TN is True Negatives.

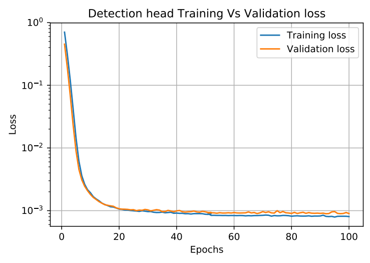
To quantify the performance of segmentation head, pixel-wise accuracy and Intersection over Union (IoU) are used.

$$\text{Accuracy} \; = \frac{Predicted\ pixel\ count \cap Ground\ truth\ pixel\ count}{Ground\ truth\ pixel\ count} \tag{9}$$

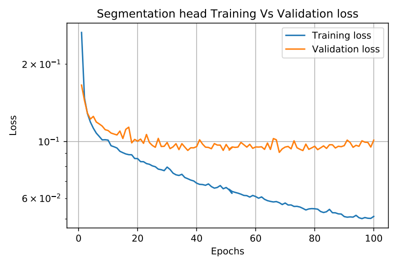
$$\text{IoU} \; = \frac{Area\ of\ Overlap}{Area\ of\ Union} \tag{10}$$

3.5 Training

We trained NimbroNet2 with batch size of 8 (segmentation data) and 40 (detection data) for 100 epochs. Pre-trained ResNet-18 weights are loaded for encoder. Adam is used as optimizer with different learning rates for parameters. We used smaller learning (1e-6) rate for parameters of pre-trained ResNet-18 model and relatively larger learning rate (1e-3) for all other parameters. During training MSE loss is employed for detection head and NLL loss is used for segmentation head. Apart from this, weighted TV loss is added for all the channels except line class channel. The combined losses are propagated as the model is considered as multiple losses and multiple outputs model. We believed that propagating the combined loss is efficient way to train in this kind of setup. The trained model is evaluated for detection and segmentation performance after every epoch. We trained the model with total variation loss and training curves for detection and segmentation heads are shown in Figure 4.



(a) Detection losses



(b) Segmentation losses

Fig. 4: Training and validation losses of detection and segmentation heads of NimbroNet2

4 Results

Results from all the experiments are summarized for detection in Table 1 and for segmentation in Table 2. Post-processing is carried out outputs from detection head. Pixel values greater than 0.95 are replaced with 1.0 to remove the noise present in the outputs. Averaging and morphological operations are applied on the individual channels of the detection head with different thresholds. No post-processing is applied on the output from segmentation head. Results for given input image are shown in Figure 5.

Table 1: Results of the object detection head. Distance threshold for the ground truth and predicted blobs is 10 pixels.

Object	F1	Accuracy	Recall	Precision	FDR
Ball	0.900	0.839	0.827	0.987	0.013
Ball (Benchmark [11])	0.998	0.996	0.996	1.000	0.000
Goal post	0.644	0.541	0.534	0.810	0.190
Goal post (Benchmark [11])	0.981	0.971	0.973	0.988	0.011
Robot	0.816	0.759	0.786	0.848	0.152
Robot (Benchmark [11])	0.979	0.973	0.963	0.995	0.004

Table 2: Results of the scene segmentation

Type	Accuracy	IOU
Field	0.994	0.983
Field (Benchmark [11])	0.986	0.975
Lines	0.947	0.878
Lines (Benchmark [11])	0.881	0.784
Background	0.985	0.982
Background (Benchmark [11])	0.993	0.981

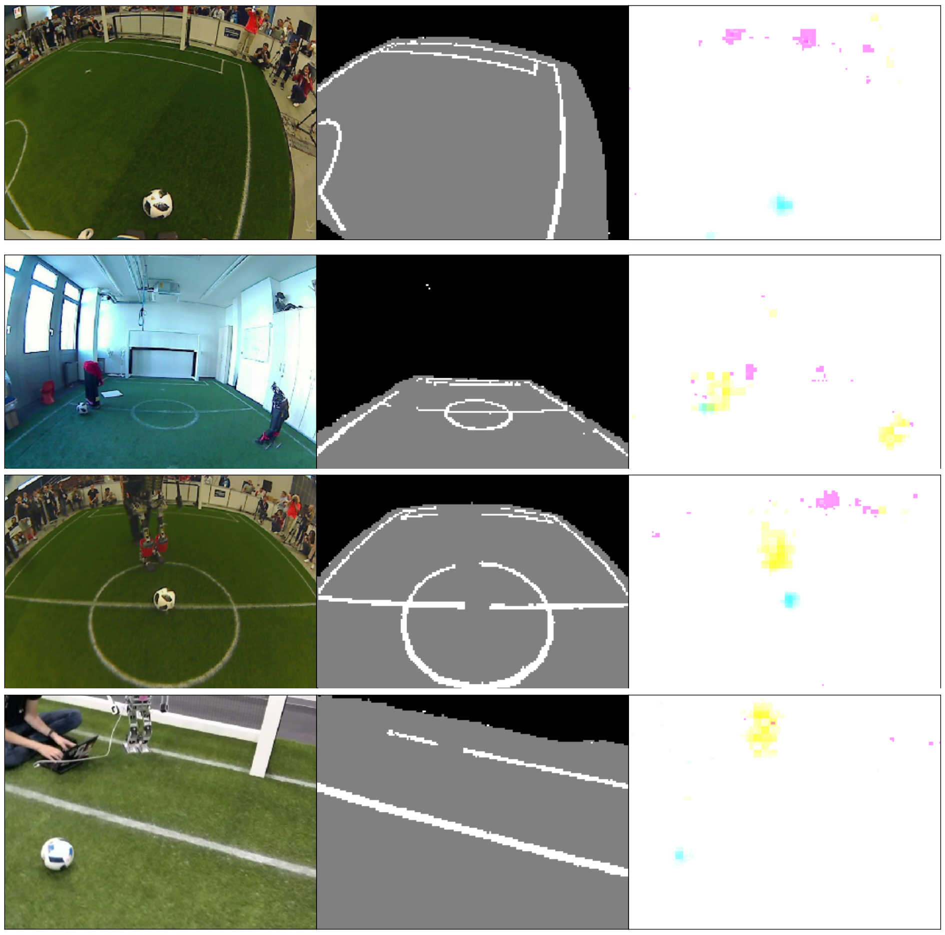


Fig. 5: Object detection results from trained model. Left column: input images. Middle column: output of the segmentation branch showing lines (white), field (gray), and background (black). Right column: output of the network indicating balls (cyan), goal posts (magenta), and robots (yellow).

5 Conclusion

In this work, an encoder-decoder based network is trained using the image data and the ground truth represented by the detection blobs and segmentation maps to perform detection and segmentation in parallel. After extensive experimentation by tuning the hyper-parameters, loss functions and post-processing we presented our results. From the results, we observed that our model performed similar to the benchmark in all cases of segmentation and detection except in case of goal post detection. We believe with different post-processing for different channels in detection output can improve this performance.

References

[1] V. Badrinarayanan et al. “SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.12 (2017), pp. 2481–2495.

[2] G. Bahl et al. “Low-Power Neural Networks for Semantic Segmentation of Satellite Images”. In: *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*. 2019, pp. 2469–2476.

[3] Liang-Chieh Chen et al. “Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation”. In: *Computer Vision – ECCV 2018*. Ed. by Vittorio Ferrari et al. Cham: Springer International Publishing, 2018, pp. 833–851.

[4] R. Huang et al. “YOLO-LITE: A Real-Time Object Detection Algorithm Optimized for Non-GPU Computers”. In: *2018 IEEE International Conference on Big Data (Big Data)*. 2018, pp. 2503–2510.

[5] I. Khokhlov et al. “Tiny-YOLO object detection supplemented with geometrical data”. In: *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*. 2020, pp. 1–5.

[6] Wei Liu et al. “SSD: Single Shot MultiBox Detector”. In: *Computer Vision – ECCV 2016*. Ed. by Bastian Leibe et al. Cham: Springer International Publishing, 2016, pp. 21–37.

[7] Niloofar Azizi, Hafez Farazi and Sven Behnke. “Location Dependency in Video Prediction”. In: *International Conference on Artificial Neural Networks (ICANN)*. Rhodes, Greece, 2018.

[8] J. Redmon et al. “You Only Look Once: Unified, Real-Time Object Detection”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 779–788.

[9] S. Ren et al. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.6 (2017), pp. 1137–1149.

[10] *RoboCup Objective*. URL: `https://www.robocup.org/objective/` (visited on 03/13/2021).

[11] Diego Rodriguez et al. “RoboCup 2019 AdultSize winner NimbRo: Deep learning perception, in-walk kick, push recovery, and team play capabilities”. In: *Robot World Cup*. Springer. 2019, pp. 631–645.

[12] Olaf Ronneberger et al. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Ed. by Nassir Navab et al. Cham: Springer International Publishing, 2015, pp. 234–241.

[13] F. Schnekenburger et al. “Detection and Localization of Features on a Soccer Field with Feedforward Fully Convolutional Neural Networks (FCNN) for the Adult-Size Humanoid Robot Sweaty”. In: 2017.