Soccer robot perception

CudaVision - Learning Computer Vision on GPUs

Aakash Aggarwal and Suhaila Mangat Paramban

Universität Bonn
aakash@uni-bonn.de, Matrikelnummer: 3272727
s6sumang@uni-bonn.de, Matrikelnummer: 3277917

Abstract. In this paper, we present the implementation of the model in 'winner2019' for detection and localization of ball, goalposts, and robots as well as pixel-wise segmentation of field and lines. The paper 'winner2019' is about latest developments that lead team NimbRo to win the RoboCup 2019. These developments include a deep learning vision system, in-walk kicks, step-based pushrecovery, and team play strategies.

1 Introduction

We can define computer vision as the understanding of the real world through the analysis of an image or a video sequence with one or more cameras. In an abstract way, it tries to replicate human vision capabilities.These comprise from the visual perception of tangible elements to the interpretation of events, just in the way a cognitive process would do. Therefore, a vision system response usually comes in the form of visual enhancements and/or decisions from prior scene estimations. In most cases, computer vision is focused on objects, expected to appear in the scene. In this project, we focus on computer vision tasks, detection and semantic segmentation.

Detection: identification and localisation of a well-defined object class

Semantic segmentation: Labelling each pixel with the class of its enclosing object or region.

Deep-learning-based visual perception system has witnessed great success in this field.In Robocup 2019, the new unified perception convolutional neural network "NimbRoNet2" helped the team NimbRo AdultSize,to improve visual perception pipeline significantly since RoboCup 2018. Accurate recognition of soccer-related objects, like a soccer ball, robots, goalposts and field boundaries by the model plays a huge role in Nimbro's performance. Our key insight is to build fully convolutional networks that take arbitrary size input and produce correspondingly-sized output with efficient inference and learning.

In the pre-processing step, we create Gaussian blobs from the annotation files in xml format. Where the center of the annotation file in given as mean and co-variance for each class is determined based on performance.

2 Related Models

Our model uses an encoder-decoder architecture similar to pixel-wise segmentation models like SegNet, and U-Net.

2.1 U-Net

Developed for biomedical image segmentation, this network is based on the fully convolutional network. The contracting or the encoder part and the decoder or the expansive part of the network are symmetric hence the name U-net. The encoder consists of the typical convolutional network made of multiple convolutions followed by ReLU and max pooling operations. It has the addition of skip connections which allow a more refined and precise output.

2.2 SegNet

Developed for outdoor and indoor scene understanding, SegNet consists of a sequence of encoder layers and respective decoder layers. The encoder is made of convolutional layers followed by ReLU, max-pooling and sub-sampling. The decoder up-samples the image using the max-pooling indices. This helps in retaining high-frequency details in the segmented images.

3 Network Architecture

The system has two output heads; one for object detection with three output channels, and the other also with three additional output channels for pixel-wise segmentation.The detection head gives the location of the ball, robots, and goalposts. The segmentation head is for line and field detection.

A pre-trained ResNet-18 is chosen as the encoder. Since ResNet was originally designed for recognition tasks, we removed the Global Average Pooling (GAP) and the fully connected layers in the model.Transpose-convolutional layers are used for up-sampling the representations.

Similar to ResNet, each convBlock consists of two convolutional layers followed by batch-norm and ReLU activations. For simplicity, residual connections in ResNet are not depicted. Note that instead of a convolutional layer we used a location-dependent convolution in the last layer. Two different types losses were used for detection and segmentation heads. For detection head, the mean squared error is used and for segmentation head cross entropy is used. The target is constructed by Gaussian blobs around the ball center and bottom-middle points of the goalposts and robots. NimbRoNet2 uses a bigger radius for robots with the intuition that annotating a canonical center point is more difficult, thus a bigger radius would less penalize the network for not outputting the exact human labels. We added Total Variation loss to the output of all result channels except the line segmentation channel. Total Variation loss encouraged blob response thus helped to have less false positives, especially in field detection.

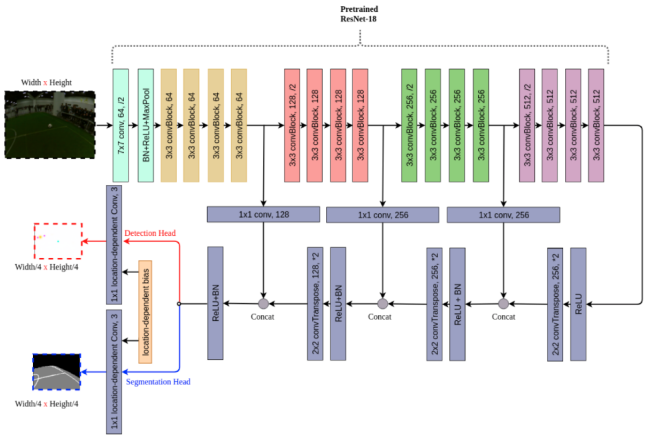

Fig. 2. NimbRoNet2 architecture. Similar to ResNet, each convBlock consists of two convolutional layers followed by batch-norm and ReLU activations. For simplicity, residual connections in ResNet are not depicted. Note that instead of a convolutional layer we used a location-dependent convolution in the last layer.

Fig. 1: NimbRoNet2 architecture.

4 Training

We used Adam optimizer with learning rate 0.001 in the training. The network converges after 80 epochs with a batch size of 16. The network is. trained by freezing the weights of the pre-trained network. The learning curve and output image examples of the training phase is shown below.

Fig. 2: Input image example

Detection

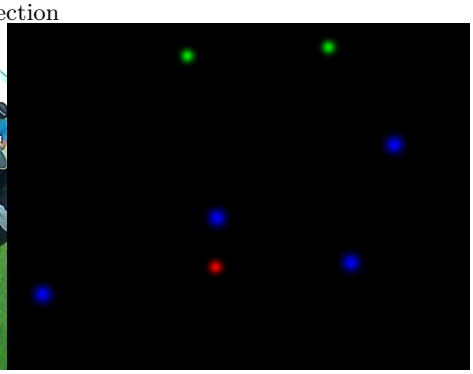

Fig. 3: Corresponding target blobs

Segmentation

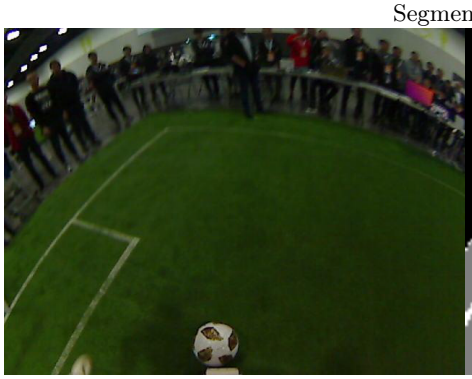

Fig. 4: Input image example

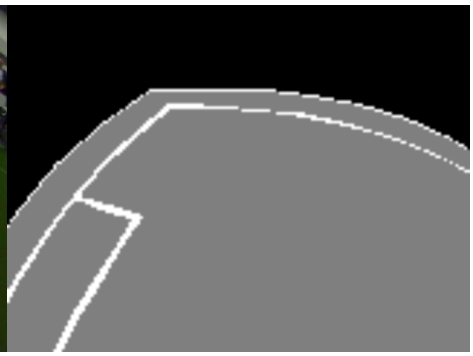

Fig. 5: segmentation target

Detection

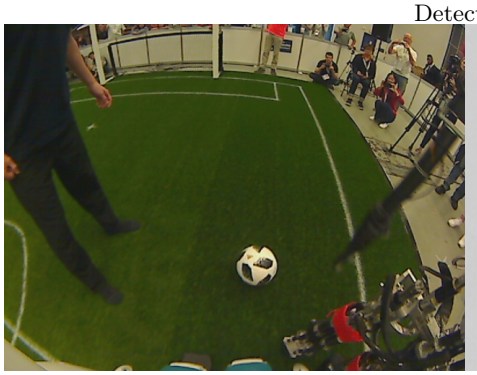

Fig. 6: Input image

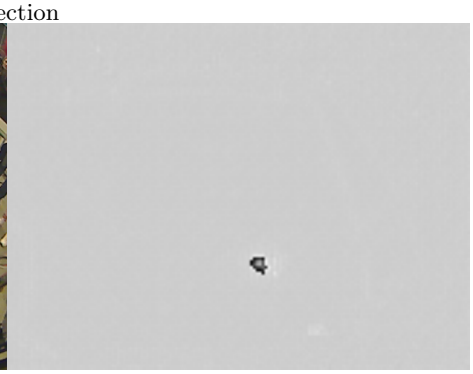

Fig. 7: Blob of detected ball

Detection

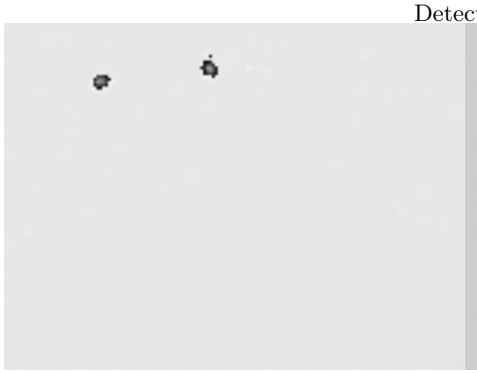

Fig. 8: Blob of detected goal post

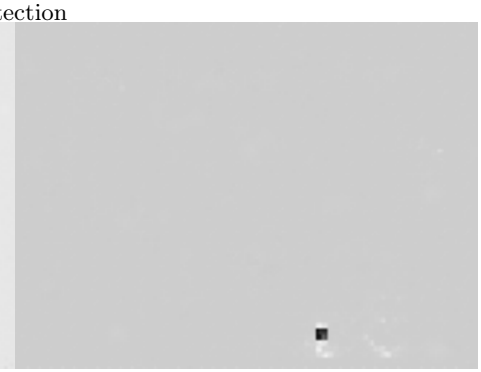

Fig. 9: Blob of detected robot

Segmentation

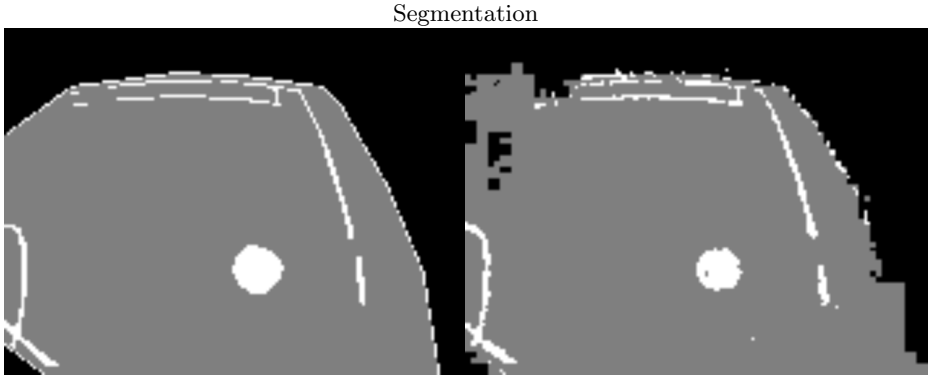

Fig. 10: Expected segmentation result

Fig. 11: Segmentation output

Loss Curves

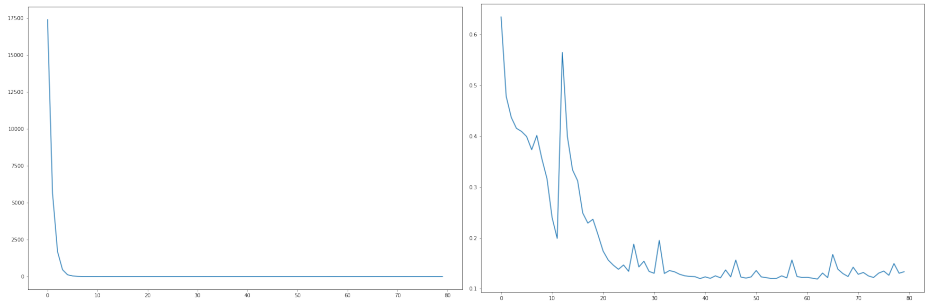

Fig. 12: MSE validation loss

Fig. 13: Negative log likelihood loss

Loss Curves

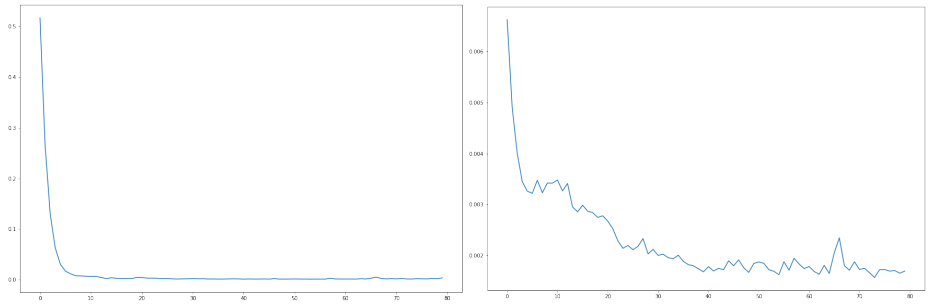

Fig. 14: Variation loss for detection

Fig. 15: Variation loss for segmentation

5 Metric Calculation

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall(RC) = \frac{TP}{TP + FN} \tag{2}$$

$$FalseDetectionRate(FDR) = 1 - Precision \tag{3}$$

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{4}$$

- TP : True Positive
- FP : False Positive
- FN : False Negative

6 Results

Detection					
Object	Accuracy	Precision	Recall	F1	FDR
Ball	0.70	0.97	0.71	0.82	0.03
Robot	0.85	0.85	0.99	0.92	0.15
Goal post	0.73	0.95	0.76	0.84	0.05

Table 1: Results of the detection branch

Segmentation		
Type	Accuracy	IOU
Field	0.976	0.949
Lines	0.707	0.591
Background	0.971	0.939

Table 2: Results of the segmentation branch

7 Conclusion

In conclusion, it is shown from the experiment that the model can learn to detect soccer related objects and robot. However there is still plenty of room for improvements.

8 Acknowledgement

We thank Hafez Farazi for his continuous guidance and helpful feedback for the successful completion of this project.

9 References

1. winner2019.pdf URL: http://www.ais.uni-bonn.de/WS1920/LabVision/Session9/winner2019.pdf

2. Badrinarayanan, Vijay et al. (2015). "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation". In: CoRR abs/1511.00561. "rXiv: 1511.00561. URL: http://arxiv.org/abs/1511.00561

3. Ronneberger, Olaf et al. (2015). "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: CoRR abs/1505.04597. arXiv: 1505.04597. URL: http://arxiv.org/abs/1505.04597
4. Goodfellow, Bengio and Courville, 2016. Deep Learning. URL: http://www.deeplearningbook.org/