Soccer Robot Perception
LabVision

Khan, Sarah[1] and Hashem, Mahmoud[2]

[1]s6srkhan@cs.uni-bonn.de, Matrikelnummer: 3279206, Bonn University
[2]s6mahash@cs.uni-bonn.de, Matrikelnummer: 3201329, Bonn University

Universität Bonn
s6srkhan@cs.uni-bonn.de, Matrikelnummer: 3279206
s6mahash@cs.uni-bonn.de, Matrikelnummer: 3201329

Abstract. In this report we present the development details of the deep learning vision system of RoboCup Humanoid League 2019 in Sydney. These developments include a model which can detect position of balls, goalposts and robots as well as perform pixel wise segmentation of field and lines.

1 Introduction

1.1 Visual Perception System

The visual perception system in discussion[1] can recognize soccer related objects, such as a soccer ball, field boundaries, robots, line segments, and goalposts by using texture, shape, brightness, and color information.This deep learning based visual perception system is robust against brightness, viewing angles, and lens distortions. To achieve this, the model suggested is a unified deep convolutional neural network which performs object detection as well as pixel wise classification with one forward pass.

1.2 Structure of report

The report is structured in the following order -

- 1. Implementation
- 1.1 Pre-processing and preparing dataset.
- 1.2 Data Loading through custom dataloader.
- 1.3 Model
- 1.4 Training Cycle
- 2. Visualisation analytics
- 3. Metrics for detection operation
- 4. Metrics for segmentation operation
- 5. Further proposition
-

2 Implementation

2.1 Pre-processing and preparing dataset

We used the dataset provided by Hafez Farazi[2] which consisted of two parts, one for detection and one for segmentation.For detection we had an image and a corresponding xml file having coordinates of bounding box for ball, goalposts and robots in the image. We constructed gaussian blobs around the centers calculated from bounding box coordinates to obtain target image. We resized all input images to 640*480 and output images to 1/4th the size of input images as output generated from model is also 1/4th the size of input.

(a) Input image

(b) Provided information in xml

(c) Target image constructed

Fig. 1: Object detection dataset preparation

For segmentation we simply resized all input images to have same dimensions and output images to be 1/4th the size of input images.

(a) Input image

(b) Target image

Fig. 2: Segmentation dataset preparation

2.2 Data Loading through custom dataloader

We divided the complete dataset into 7:3 ratio of training and test dataset. For loading, we implemented a custom dataloader class which overloads the

__len__ and __getitem__ functions.__len__ returning the total size of dataset. and __getitem__ for returning dataset at ith index and the returned dataset consists of two images: the input image and its corresponding target image. For both segmentation as well as object detection same dataloader is used.

2.3 Model

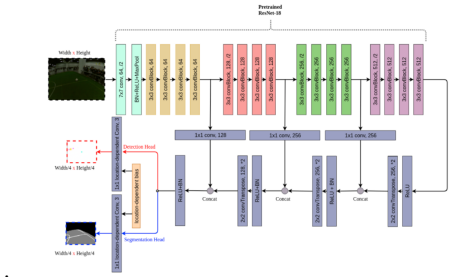

Fig. 3: Model Architecture[3]

The skip connections have been used at layers 5,6,7 and 8.

2.4 Training Cycle

Since the dataset is different for both object detection and segmentation we train in cycles in an epoch. For every cycle of training for segmentation , we train for object detection for two cycles, because we have bigger dataset for object detection as well as we consider that learning robot requires more training than other objects. For object detection, mean square loss along with total variation loss to introduce smoothness is used. For segmentation, negative log likelihood loss has been used.

3 Visualisation analytics

We plan to implement visualisation of confusion matrix, learning curves as well as feature maps/activation maps.

4 Metrics for object detection

For measuring accuracy in this task we want to use gaussian fit for detecting peak in output images as well as ground truth images. Then we want to find out the distances between the centers obtained in both images and if they are

within a threshold we want to pass it as correctly classified. For precision we will measure the percentage of correct predictions. For recall we will be using the given formula:

$$recall = \frac{TP}{TP + FN}$$

And finally F1 could be found by using precision and recall from previous step

$$F1 = 2 * \frac{precision * recall}{precision + recall}$$

5 Metrics for segmentation

For calculating accuracy we took the number of pixels correctly classified by comparing the output image with ground truth and used the following formula:

$$accuracy = correctlyClassified + totalPixels * 100$$

For calculating intersection over unit we used the previous number of correctly classified pixels to calculate the area of overlap between the predicted segmentation and the ground truth divided by the area of union between the predicted segmentation and the ground truth.

$$IoU = target \cap predicted / target \cup predicted$$

6 Flowchart of implementation

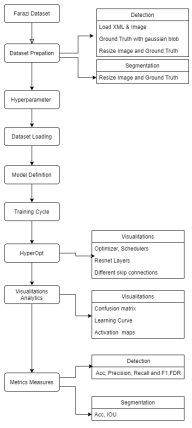

Fig. 4: Model Architecture[3]

7 Future propositions

Use hyperopt to optimize which resnet to use, skip connections layers and optimizers such as SGD,Adam.

References

1. Diego Rodriguez, Hafez Farazi, Grzegorz Ficht, Dmytro Pavlichenko, André Brandenburger, Mojtaba Hosseini, Oleg Kosenko, Michael Schreiber, Marcel Missura, and Sven Behnke. *RoboCup 2019 AdultSize Winner NimbRo: Deep Learning Perception, In-Walk Kick, Push Recovery, and Team Play Capabilities.*
2. http://bigcuda5.informatik.uni-bonn.de:8686/
3. Diego Rodriguez, Hafez Farazi, Grzegorz Ficht, Dmytro Pavlichenko, André Brandenburger, Mojtaba Hosseini, Oleg Kosenko, Michael Schreiber, Marcel Missura, and Sven Behnke. *RoboCup 2019 AdultSize Winner NimbRo: Deep Learning Perception, In-Walk Kick, Push Recovery, and Team Play Capabilities.*