Soccer Robot Perception

CudaVision: Learning Vision Systems on GPU

Padmanabhan, Deepan Chakravarthi and Mulye, Mihir

Hochschule Bonn-Rhein-Sieg
{deepan.padmanabhan, mihir.mulye}@smail.inf.h-brs.de
Matrikelnummer: 9034816. 9034909

Abstract. This project aims to develop a visual perception pipeline for a soccer playing robot. The deep learning model Rodriguez et al., 2019 implemented in this project is the winner of RoboCup Humanoid League, 2019. The model is designed to perform simultaneous detection and segmentation of soccer-related entities namely goalposts, ball, field, line demarcations along with the other robots in the scene. The model achieves an impressive F1 score of 95.8% and mean Intersection over Union of 93.3% on the segmentation task. However, the model performs poorly on the detection task and achieves 0.19% accuracy and 99.6% False Detection Rate.

1 Introduction

The task of making computers see the way humans do has been a long researched problem. With advancements of methods such as Deep Learning and increasingly available compute power, the research envelope has been pushed even further in this domain. Many novel applications of computer vision are researched and one such work is discussed in Rodriguez et al., 2019.
The network discussed in this paper is able to detect and segment objects in a given input scene. The task of classification and localization of objects in a given input image is called Object detection. Semantic segmentation is the task of performing pixel-wise labelling of different entities present in an image. This task of segmenting objects plays a pivotal role in computer vision and attributes to the success of various applications such as autonomous driving, tumor detection, remote sensing, augmented reality Forsyth and Ponce, 2002 and scene understanding Ess et al., 2009 Geiger et al., 2012 Cordts et al., 2016. CNN-based deep learning methods have been the state-of-the-art models for various segmentation tasks Bakas et al., 2018 Ning et al., 2005 Ciresan et al., 2012 Farabet et al., 2012 Hariharan et al., 2014 Gupta et al., 2014 Badrinarayanan et al., 2017 Ronneberger et al., 2015, outperforming other approaches by a substantial accuracy.

This model implemented in this work provides an approach to multi task learning in the context of a multi robot soccer competition. The network identifies objects such as balls, other robots and goalposts and is able to distinguish

between field, line demarcations and background. The ability to detect other robots in particularly crucial as it can be utilised in a variety of multi robot settings.

2 Related Work

The unified perception system of simultaneous performing detection and segmentation find applications in multi-task learning and instance segmentation tasks. The objective of this paper differs from the conventional instance segmentation approach of performing pixel-wise classification for each known object instance in the scene. In this particular work, the authors focus on detection of each ball, robot and goalpost instances and pixel-wise classification of the field and boundary lines. In the work Reisswig et al., 2019, to devise an Optical Character Recognition (OCR) system, the authors develop an architecture Chargrid-OCR to perform segmentation of different characters in a document and detect the location of word-level boxes. In the field of robotics Rodriguez et al., 2019, the paper discusses a unified perception pipeline for humanoid soccer playing robots with detection and segmentation. This project employs a multi-task learning (MTL) approach to perform detection and segmentation task. In the past few years, multi-task learning research is increasing starting from the works of Eigen and Fergus, 2014 Tian et al., 2014 Bischke et al., 2017 Teichmann et al., 2016. The paper Crawshaw, 2020 provides a detailed survey, discusses the challenges, architectures used, and applications of multi-task learning. The major challenge in multi-task learning is overcoming the *destructive inference or negative transfer*, where improving the performance of the model on one task will impact the model performance on the other task. The following sections will provide the steps taken to address negative transfer in this project and accomplish the task.

3 Soccer Robot Perception Pipeline

This section addresses the different stages and elements of the visual perception pipeline of the soccer playing robot. The soccer robot perception pipeline is implemented in PyTorch 1.7.1 and Torchvision 0.8.2. The project developed as a python package *soccer-robot-perception* is available at `https://github.com/DeepanChakravarthiPadmanabhan/Soccer_Robot_Perception.git`.

3.1 Network Architecture

The architecture, NimbRoNet2 from the paper Rodriguez et al., 2019 is implemented to accomplish the task. The architecture is capable of performing object detection and semantic segmentation. As can be seen in Figure 1, the backbone of the network is a pretrained ResNet-18 model with location-dependent convolutional layers. The location-dependent layers employ a shared learn-able bias to reduce the number of parameters. The location-dependent layers is used to incorporate the location specific features as discussed in Azizi et al., 2018.

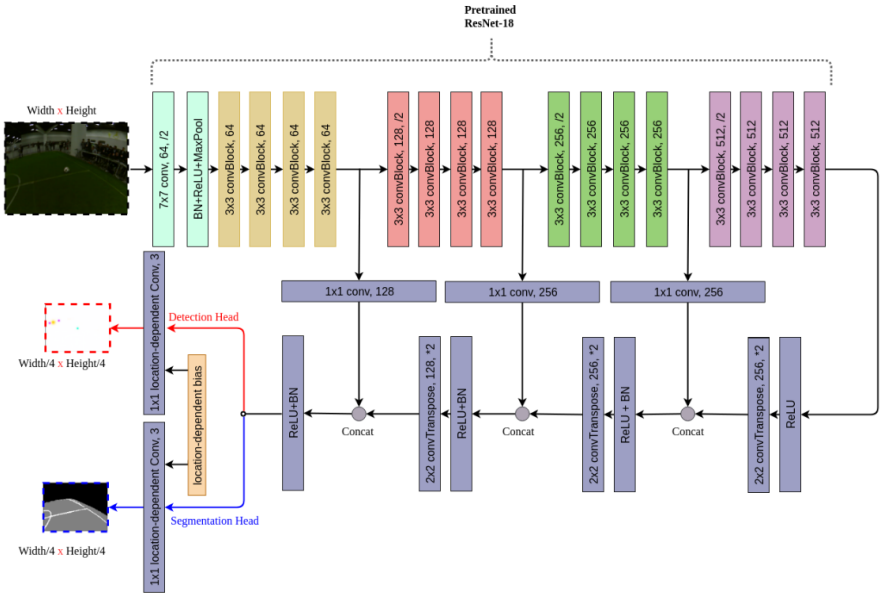

Fig. 1. NimbRoNet2 architecture (taken from Rodriguez et al., 2019)

3.2 Dataset

The dataset consists of 8869 RGB images annotated for detection task and 1192 RGB images annotated for segmentation task. A few images without labels are neglected. The remaining detection and segmentation dataset samples are 7462 and 1165, respectively. The annotations for the detection task are in the form of *xml* annotations containing the filename, object type, corresponding bounding box coordinates, along with the dimensions of the image. The object types in detection are robot, goal post and ball. The ground truth for segmentation is in the form of color coded images, where each pixel of the image belonging to the same class are mapped to the same color. The three different values available in the mask are background, field and boundary lines. The dataset is split into 0.7, 0.15 and 0.15 of the total data for train, validation and test set.

3.3 Dataset pre-processing

The data is provided in the form of RGB images, *xml* annotations for detection ground truth and color coded images for segmentation ground truth. For training the detection task, the bounding box coordinates information provided in the annotations are used to generated gaussian blobs. The gaussian blobs are treated as keypoints to be learned by the model. Each class - robot, ball and goalpost have their own heatmap with the respective gaussian blobs at the object centroids. The centroid location for balls, robots and goal posts are the

center of the bounding box coordinates, the center of the lower bounding part and the two bottom edges, respectively. The centroids are superimposed based on a channel specific mapping and the masks are generated.

For segmentation task, a few mask annotations have values denoting a fourth class, which are mapped to the field mask value. The color coded images are converted to class coded masks and then used as a ground truth for training the neural network.

3.4 Model creation

The neural network architecture implemented in this project has been replicated from Rodriguez et al., 2019. The objective of the model is to implement multi task learning. The network provides two outputs via heads: detection and segmentation. An interesting concept of location-aware convolution has been introduced in this network. The idea is to introduce a bias in the network parameters based on the prior knowledge. An example is: "field" would almost always be in the lower half of the image being processed by the pipeline at any given instance. Hence, the network parameters are initialised using a weighted function which encodes this information. This bias is added to both the detection and segmentation head before applying a 1×1 convolution to obtain the final prediction. The dimensions of the output follows the **(N, C, H, W)** convention where N = batch size, C = number of channels, H = height of the image & W = width of the image:

hence, the dimension of detection output are:

$$batch\ size \times 3 \times \frac{input\ image\ size}{4} \times \frac{input\ image\ size}{4} \tag{1}$$

& segmentation output:

$$batch\ size \times 3 \times \frac{input\ image\ size}{4} \times \frac{input\ image\ size}{4} \tag{2}$$

3.5 Data loaders

The project utilizes a detection dataloader and segmentation dataloader for detection and segmentation. In addition, a concatenated dataset batching detection and segmentation data in a shuffled manner is also available in the repository, which can be generated using the configuration file for the training. The concatenated dataset is developed using torch.utils.data.ConcatDataset(). However, only this concatenated dataset is not used in training the model.

3.6 Training

The training procedure to accomplish multi-task learning is challenging given the different learning speed for each task. In the initial experiments, it is noticed

that the detection task needs a lot more time than the segmentation task to learn. To overcome this, each epoch sampled detection samples at a higher batch size than the segmentation. Each epoch includes updating the model once after gathering detection and segmentation model output. The backbone ResNet-18 is retrained in the training loop with a lesser learning rate as the network is pretrained already.

In addition, Loss Balanced Task Weighting (LBTW) methodology Crawshaw, 2020 is used to weight the losses for each task. The weight λ_i for a task i at time step t, is given by the equation 3. LBTW measures learning speed as the ratio of the current loss to the initial loss.

$$\lambda_i(t) = (\frac{L_i(t)}{L_i(0)})^{\alpha} \tag{3}$$

α is a hyper-parameter and set to one. The value is calculated manually over many runs and set as loss factor in the trainer module by normalizing over the different tasks. The loss factor for detection and segmentation is 0.7 and 0.3, respectively. This has been carried out after noticing that detection task is learnt by the model very slowly, and the segmentation task is beginning to overfit after certain epochs. This difference in learning speed serves as the motivation for weighting the loss.

The training is carried out in Hochschule Bonn-Rhein-Sieg cluster with Tesla V100-PCIE-16GB and Linux operating system.

3.7 Hyper-parameters

The training instance was instantiated for the model with the following hyper-parameters as provided in Table 1.

Table 1. Model training hyperparameters

Hyper-parameter	Value
Number of epochs	100
Detection batch size	32
Segmentation batch size	8
Optimizer	Adam
Encoder learning rate	1e-06
Decoder learning rate	1e-03
Segmentation Head learning rate	1e-03
Detection head learning rate	1e-03
Early Stopping patience	10
Total Variation loss weight	1e-06

3.8 Loss calculation

The authors of Rodriguez et al., 2019 applied mean squared error for detection loss and cross entropy for segmentation loss computation. According to the authors, calculating total variation loss for all output channels except for the field line channel promotes blob response.

$$mean\ square\ error = \frac{\sum_{i=1}^{n}(y_i - \hat{y_i})^2}{n} \tag{4}$$

where, n = number of samples, y_i = ground truth, $\hat{y_i}$ = network prediction.

$$cross\ entropy\ loss = -\sum_{i=1}^{n} y_i \times log(p_i) \tag{5}$$

where, n = number of samples, y_i = ground truth, $\hat{p_i}$ = softmax probability of i^{th} class.

$$total\ variation\ loss = \sum_{i,j} \sqrt{|y_{i+1,j} - y_{i,j}|^2 + |y_{i,j+1} - y_{i,j}|^2} \tag{6}$$

where, $y_{i,j}$ = ground truth pixel at i, j^{th} location

3.9 Evaluation

The analysis of the results has been performed using the standard metrics given below,

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{9}$$

$$False\ Detection\ Rate(FDR) = 1 - Precision \tag{10}$$

$$F1\ score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{11}$$

where, TP : True Positive, FP : False Positive, FN : False Negative, TN : True Negative

$$IOU = \frac{Area\ of\ Intersection}{Area\ of\ Union} \tag{12}$$

3.10 Post Processing

The direct output from the detection head of the model is very noisy. In order to find the blob centers, contour detection is performed on each heatmap channel of the detection head output. The heatmap is smoothed by gaussian kernel and then contours are identified by Opencv *findContours* functions. The contour centers are found using the first moment of the contours.

4 Results and Discussion

Figures 2 and 3 depict the Soccer robot perception pipeline output. The figure on the left is the input image, the one in the middle depicts the segmentation output and the one on the right depicts the detection head output.

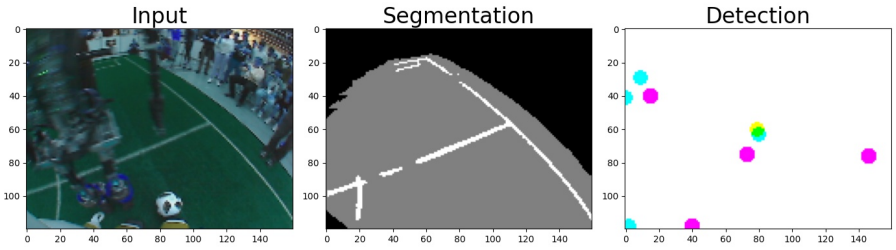

Fig. 2. Soccer robot perception pipeline output illustration 1. In detection image, yellow blobs denote goal post, magenta blobs denote robot and cyan denotes ball. In segmentation image, black pixels denote background, gray pixels denote field and white pixels denote line boundary. The segmentation head of the model is able to identify the break in the line demarcation where the robot legs obstruct the view. Also, the detection head of model produces higher False Positive while detecting the desired objects

Using the aforementioned formulae, the quantities are computed for NimbRoNet2 on the test set and the following tables provides an insight into the performance of the network:

From Tables 2 and 3, it can be seen that the model is able to perform the segmentation task. However, the performance of the model for the detection task is poor and can be improved a lot. This can be attributed to the fact that object detection and segmentation task are learnt at different speeds by the model. Even though this issue is addressed by loss weighting and training detection with relatively more samples than the segmentation the model generalization is affected for detection.

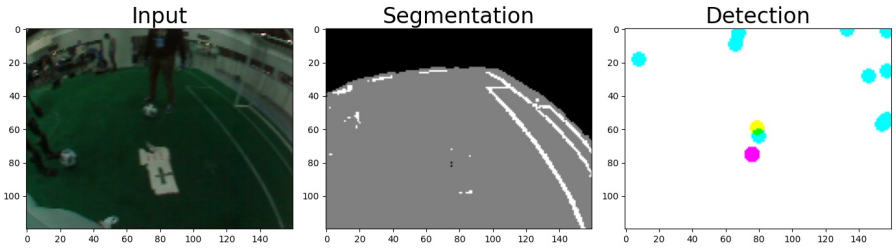

Fig. 3. Soccer robot perception pipeline output illustration 2. In detection image, yellow blobs denote goal post, magenta blobs denote robot and cyan denotes ball. In segmentation image, black pixels denote background, gray pixels denote field and white pixels denote line boundary. The segmentation head is able to distinguish between the object lying on the floor and the line demarcations. However, the detection head of the model yields higher False positives.

Table 2. Results for detection of soccer robot perception network

Type	Accuracy(%)	Precision(%)	Recall(%)	F1((%)	FDR(%)
Ball	0.23	0.24	3.3	0.4	0.997
Robot	0.24	0.41	0.72	0.4	0.995
Goalpost	0.11	0.28	0.15	0.2	0.997
Mean	0.19	0.31	1.39	0.3	0.996

5 Conclusion

In this project a visual perception pipeline system to detect and classify balls, goalposts, robot, field and boundary line in a soccer field. A multi task learning approach has been implemented as provided in Rodriguez et al., 2019. The pipeline is able to perform object detection and semantic segmentation on the given input. The performance of the pipeline on segmentation is impressive with 95.4% accuracy with 93.3% IOU on the test set. However, the pipeline performs poorly on the detection task with an accuracy of 0.19% and a precision of 0.31%

Table 3. Results for semantic segmentation of soccer robot perception network

Type	IOU(%)	Accuracy(%)	Precision(%)	Recall(%)	F1(%)
Background	98.5	98.6	98.9	98.5	98.7
Field	98.1	98.6	98.3	98.6	98.5
Lines	83.3	89	91.4	98.3	90.2
Mean	93.3	95.4	96.2	98.4	95.8

on test set. The future work will focus on training the model for larger epochs with even reduced loss factor for segmentation.

References

Azizi, Niloofar et al. (2018). "Location Dependency in Video Prediction". In: *CoRR* abs/1810.04937. arXiv: 1810.04937. URL: http://arxiv.org/abs/1810.04937.

Badrinarayanan, Vijay et al. (2017). "Segnet: A deep convolutional encoder-decoder architecture for image segmentation". In: *IEEE transactions on pattern analysis and machine intelligence* 39.12, pp. 2481–2495.

Bakas, Spyridon et al. (2018). "Identifying the Best Machine Learning Algorithms for Brain Tumor Segmentation, Progression Assessment, and Overall Survival Prediction in the BRATS Challenge". In: *CoRR* abs/1811.02629. arXiv: 1811.02629. URL: http://arxiv.org/abs/1811.02629.

Bischke, Benjamin et al. (2017). "Multi-Task Learning for Segmentation of Building Footprints with Deep Neural Networks". In: *CoRR* abs/1709.05932. arXiv: 1709.05932. URL: http://arxiv.org/abs/1709.05932.

Ciresan, Dan et al. (2012). "Deep neural networks segment neuronal membranes in electron microscopy images". In: *Advances in neural information processing systems* 25, pp. 2843–2851.

Cordts, Marius et al. (2016). "The cityscapes dataset for semantic urban scene understanding". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3213–3223.

Crawshaw, Michael (2020). "Multi-Task Learning with Deep Neural Networks: A Survey". In: *arXiv preprint arXiv:2009.09796*.

Eigen, David and Rob Fergus (2014). "Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture". In: *CoRR* abs/1411.4734. arXiv: 1411.4734. URL: http://arxiv.org/abs/1411.4734.

Ess, Andreas et al. (2009). "Segmentation-Based Urban Traffic Scene Understanding." In: *BMVC*. Vol. 1. Citeseer, p. 2.

Farabet, Clement et al. (2012). "Learning hierarchical features for scene labeling". In: *IEEE transactions on pattern analysis and machine intelligence* 35.8, pp. 1915–1929.

Forsyth, David A. and Jean Ponce (2002). *Computer Vision: A Modern Approach*. Prentice Hall Professional Technical Reference.

Geiger, Andreas et al. (2012). "Are we ready for autonomous driving? the kitti vision benchmark suite". In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 3354–3361.

Gupta, Saurabh et al. (2014). "Learning rich features from RGB-D images for object detection and segmentation". In: *European conference on computer vision*. Springer, pp. 345–360.

Hariharan, Bharath et al. (2014). "Simultaneous detection and segmentation". In: *European Conference on Computer Vision*. Springer, pp. 297–312.

Ning, Feng et al. (2005). "Toward automatic phenotyping of developing embryos from videos". In: *IEEE Transactions on Image Processing* 14.9, pp. 1360–1371.

Reisswig, Christian et al. (2019). "Chargrid-OCR: End-to-end trainable Optical Character Recognition through Semantic Segmentation and Object Detection". In: *CoRR* abs/1909.04469. arXiv: 1909.04469. URL: http://arxiv.org/abs/1909.04469.

Rodriguez, Diego et al. (2019). "RoboCup 2019 AdultSize Winner NimbRo: Deep Learning Perception, In-Walk Kick, Push Recovery, and Team Play Capabilities". In: *CoRR* abs/1912.07405. arXiv: 1912.07405. URL: http://arxiv.org/abs/1912.07405.

Ronneberger, Olaf et al. (2015). "U-net: Convolutional networks for biomedical image segmentation". In: *International Conference on Medical image computing and computer-assisted intervention*. Springer, pp. 234–241.

Teichmann, Marvin et al. (2016). "MultiNet: Real-time Joint Semantic Reasoning for Autonomous Driving". In: *CoRR* abs/1612.07695. arXiv: 1612.07695. URL: http://arxiv.org/abs/1612.07695.

Tian, Yonglong et al. (2014). "Pedestrian Detection aided by Deep Learning Semantic Tasks". In: *CoRR* abs/1412.0069. arXiv: 1412.0069. URL: http://arxiv.org/abs/1412.0069.