Project Proposal: Object detection in the RoboCup Standard Platform League

Jakob Hartmann

Motivation

One of the most important problem areas for the NAO robots in the RoboCup Standard Platform League (SPL) is vision: the robots must be able to detect the ball, the border of the playing field, the goals, and other robots. This recognition is the basis for localizing the robot, planning its actions, and ultimately scoring points.

The RoboCup initiative has set itself an ambitious long-term goal: "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." [1] To achieve this goal, "RoboCup´s rules change in order to promote advances in the science and technology of robots and to make the league challenges closer to real world, rather than to impose artificial setups to improve league specific performance." [1] Examples of adjustments that made the conditions more realistic were changing the ball color from orange to black and white and playing under natural lighting conditions. [2] This requires better object recognition algorithms, as the ball is no longer as noticeable by its color and can be mistaken for other objects more easily.

Problem definition

In my final project, I would like to use current advances in computer vision and deep learning to detect the most important objects in the RoboCup Standard Platform League: balls, goals, and robots. Given an image as input, a neural network should detect all objects in the image, mark them with a bounding box and output the corresponding class. This could help future projects to develop algorithms that localize the robot in the playing field or plan its next actions such as moving to the ball.

State of the art

In recent years, RoboCup SPL teams have increasingly focused on machine learning for vision tasks. Switching to a black and white ball has driven this development, since the simple recognition by the orange color is no longer possible. The basic procedure used for object detection tasks implemented e.g., by DAInamite [3], Nao-Team HTWK [4], HULKs [5], B-Human [6] and which been studied in many theses [7-12] is as follows: First, the images taken by the NAO robot are analyzed and different areas, which could possibly contain objects such as balls, goals, robots, or penalty spots are identified as regions of interest (sometimes also called hypotheses or candidates). This preselection is necessary because the NAO robot has only limited computing resources and the images must be analyzed in real time. The identified candidates then serve as input to a Convolutional Neural Network (CNN), which classifies the objects in the image and outputs a vector with the corresponding class probabilities.

Convolutional Neural Networks were introduced by Yann LeCun in the 1990s [13] and had their breakthrough in 2012 when "AlexNet" [14] achieved impressive results in image classification tasks. Since then, they have been the foundation for many computer vision applications. The main component of CNNs are convolutional layers that slide a kernel over the input and calculate the dot product between the kernel and an image section. The aim is to recognize certain features such as edges in the image. A convolutional layer is usually followed by a pooling layer, which calculates the maximum or average of neighboring pixels and thereby discards unnecessary information, and downsamples the image. These convolutional and pooling layers alternate several times before fully connected layers are ultimately used to generate the output vector. To further improve the performance of the CNN, batch normalization layers and regularization methods such as dropout can be implemented. [7-12, 15] For many machine learning applications, including computer vision, developers can use powerful libraries and frameworks such as TensorFlow, PyTorch or OpenCV to accelerate development.

Approach

I will use YOLO (You Only Look Once) [16], a neural network framework for object detection, for development. YOLO allows to train a neural network that receives an image as input and outputs both the bounding boxes and the corresponding classes. This eliminates the need for prior candidate generation, resulting in significantly faster computations and real-time usability. [17]

Since there is no publicly available data for my task, the first thing I will have to do is create images for training and testing purposes. These will then be preprocessed to have the correct format and size and afterwards be labeled with the bounding boxes and the appropriate classes, this can for example be done by [18], [19] or [20]. I can´t yet estimate how many images I will need for training; I suspect between 250 and 300 for each object. The training requires powerful hardware, but private I only have a NVIDIA GeForce 770 with 2GB VRAM to work with. If this is not enough and the training takes to long, I will use services from Google Cloud Platform or Amazon Web Services. After training, the performance will be evaluated on the test data. If the object detection works well on single images, I plan to test it with videos for real-time usability.

Simulation

Since a real NAO robot is not available, the images for the train and test data are created in the SimSpark simulator. Real-time testing could also be performed by using video recordings in SimSpark. The results in the simulator are somewhat less realistic, since, for example, there are no natural lighting conditions, and the limiting hardware of the NAO robot is also not a problem. But the results can provide valuable information about whether the chosen approach is in principle suitable for object recognition in the RoboCup Standard Platform League.

References

[1] RoboCup Federation: *Objective*. RoboCup.org. https://www.robocup.org/objective (accessed Jan. 29, 2021)

[2] Wikipedia contributors: *RoboCup Standard Platform League*. Wikipedia.org. https://en.wikipedia.org/wiki/RoboCup_Standard_Platform_League (accessed Jan. 29, 2021)

[3] Xu, Y., Berger, M., Qian, Q., Tuguldur, E.O.: *DAInamite Team Description 2016*. RoboCup 2016: Robot Soccer World Cup XX Proceedings (2016)

[4] Nao-Team HTWK: *Team Research Report*. (2019)

[5] Basler, J. et al.: *HULKs Team Research Report 2019*.

[6] Röfer, T. et al.: *B-Human Team Report and Code Release 2018*. (2018)

[7] Kahlefendt, C.: *A Comparison and Evaluation of Neural Network-based Classification Approaches for the Purpose of a Robot Detection on the Nao Robotic System*. Project thesis, Technische Universität Hamburg-Hamburg (2017)

[8] Felbinger, G.C.: *Optimal CNN Hyperparameters for Object Detection on NAO Robots*. Master thesis, Technische Universität Hamburg-Hamburg (2018)

[9] Felbinger, G.C.: *A genetic approach to design convolutional neural networks for the purpose of a ball detection on the NAO robotic system*. Project thesis, Technische Universität Hamburg-Hamburg (2017)

[10] Mewes, F.: *Objekterkennung und Zuordnung im Roboterfußball (SPL) – Roboter*. Master thesis, Hochschule für Technik, Wirtschaft und Kultur Leipzig (2017)

[11] Büchel, F.: *Objekterkennung und Zuordnung im Roboterfußball (SPL) – Tore*. Master thesis, Hochschule für Technik, Wirtschaft und Kultur Leipzig (2017)

[12] Poppinga, B.: *Nao You See Me: Echtzeitfähige Objektdetektion für mobile Roboter mittels Deep Learning*. Master thesis, Universität Bremen (2018)

[13] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: *Gradient-Based Learning Applied to Document Recognition*. Proceedings of the IEEE 86 (11), 436-444 (1998)

[14] Krizhevsky, A., Sutskever, I., Hinton, G.E.: *ImageNet Classification with Deep Convolutional Neural Networks*. Advances in Neural Information Processing Systems 25, 1097-1105 (2012)

[15] Wikipedia contributors: *Convolutional Neural Network*. Wikipedia.org. https://de.wikipedia.org/wiki/Convolutional_Neural_Network (accessed Jan. 29, 2021)

[16] Bochkovskiy, A.: *darknet: YOLOv4v / Scaled-YOLOv4 - Neural Networks for Object Detection (Windows and Linux version of Darknet).* Github.com. https://github.com/AlexeyAB/darknet (accessed Jan. 29, 2021)

[17] Bochokovskiy, A., Wang C.Y., Liao, H.Y.M.: *YOLOv4: Optimal Speed and Accuracy of Object Detection*. (2020)

[18] Qiu, S.: *BBox-Label-Tool: A simple tool for labeling object bounding boxes in images*. Github.com. https://github.com/puzzledqs/BBox-Label-Tool (accessed Jan. 29, 2021)

[19] Lin, T.T.: *labelImg: LabelImg is a graphical image annotation tool and label object bounding boxes in images*. Github.com. https://github.com/tzutalin/labelImg (accessed Jan. 29, 2021)

[20] Grinover, R.: *darknet: Convolutional Neural Networks*. Github.com. https://github.com/RiccardoGrin/darknet (accessed Jan. 29, 2021)