Object Detection within a Robotic Application

Chia-Wen Tsai[1], Felizitas Kunz[2], Christoph Caprano[1], and Oskar Haller[1]

[1]Technical University of Munich [2]Ludwig-Maximilians-University, Munich

Introduction

Our idea was to teach a Braccio Robotic Arm to play the child's game pairs. It should detect the playing cards laying on the table, their position and the motif of a card and find the second one. The robot picks one playing card up, places it on the stack and searches for the second one within the remaining cards. For object detection YOLOv2-Real-Time-Object detection is used for transfer learning with datasets containing images of our playing cards. We trained our network with Google Cloud Service.

Dataset

Datasets for the project could be divided into two parts. Darknet has pre-trained on the Pascal VOC 2007+2012, and we trained 30 images for each of the 10 classes of the playing cards. **Add sample label from our dataset**

Fig. 1: Training data of the playing cards. Ten classes of the cards are:
Boat, Girl, Boy, Horse, EuropeanBird, PacificBird, Flower, BlueCard, Tree, Pineapple

Related Work

For the object detection, we used YOLOv2 respectivly Tiny YOLO with transfer learning to detect the motifs of the playing cards in real time.
YOLOv2 trains the labels and the bounding boxes at the same time with a joint dataset consisting of one for classification and one for detection. If an image from the detection dataset is detected, it backpropagates normally. If the other case is detected, the network only backpropagates the classification specific parts.
A simplyfied network based on Goolenet architecture with additional features like batch normalization and high resolution classifiers makes YOLOv2 fast and accurate.

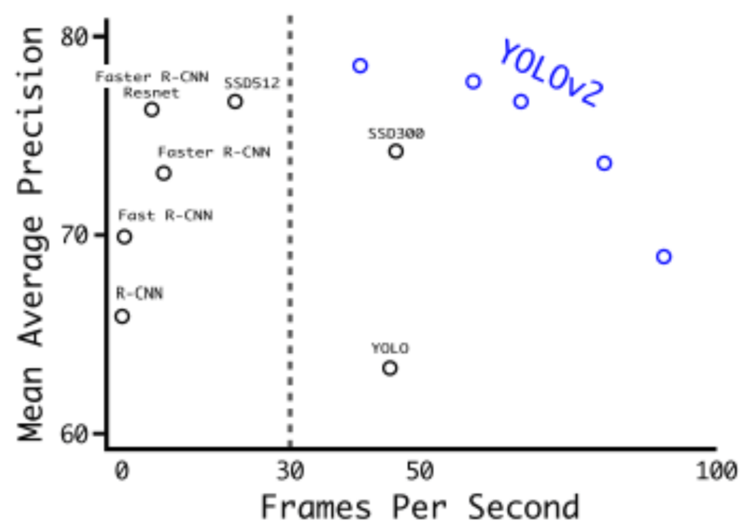

Figure 4: Accuracy and speed on VOC 2007.

Fig. 2: Network architecture.

Methodology

Network Architecture

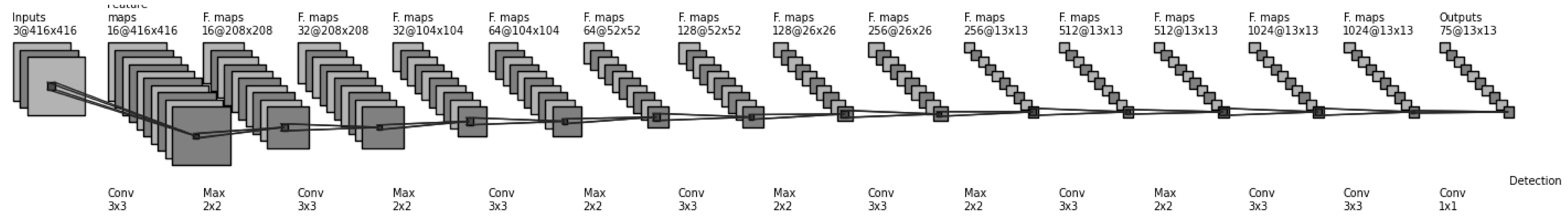

Fig. 3: Network architecture.

We used a pre-trained Tiny YOLO network (see figure 3). It has only 15 layers (Yolo has 30), which makes it much faster, but also less accurate. The architecture consits of 9 convolutional and 6 pooling layers. We adjusted some parameters **which ones again? I forgot**
We trained the last two convolutional layers with our dataset to extract the middle and high level features. The results were very accurate.
Our hardware architecture consists of a PC, an Arduino UNO, an electromagnet, a camera and the Braccio Robot Arm. They communicate with ROS.

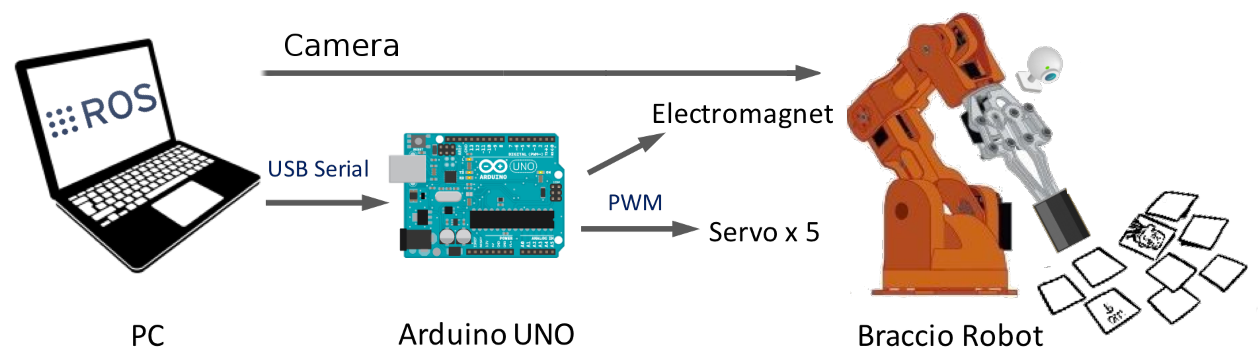

Fig. 4: Hardware setup

Outcome

Add image with weights

- Network from Tiny YOLO
- 10 cards + classes
- Architecture
- Filters from results
- Camera input with detection labels