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

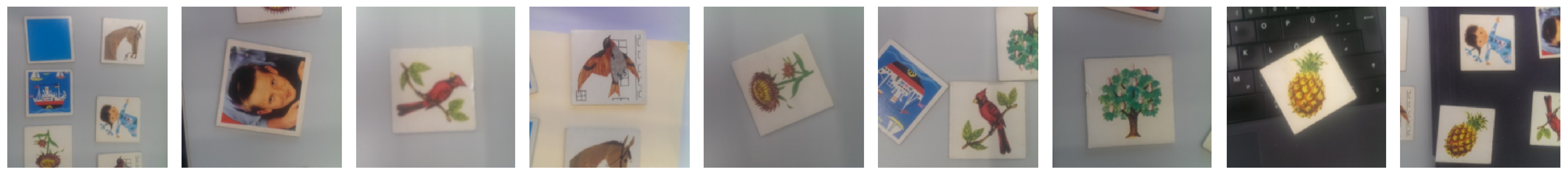


Fig. 1: Training data of the playing cards. Ten classes of the cards are:
Boat, Girl, Boy, Horse, EuropeanBird, PacificBird, Flower, BlueCard, Tree, Pineapple

Related Work

For the object detection, we used Yolo9000 with transfer learning to detect the motifs of the playing cards in real time. *Add table? Add link?*
The communication between the Braccio Robotic Arm and the computer is enabled with ROS. *Also with the Arduino?*
How does Yolo train the bounding boxes and labels at same time? Yolo uses anchor boxes to predict bounding boxes
Labels trained with ImageNet at the full 448 x 448 resolution for 10 epochs
What makes it fast and accurate? Simplify network, make representations easier to learn
Don't use VGG-16 network, but custom network based on Googlenet architecture (less accurate)

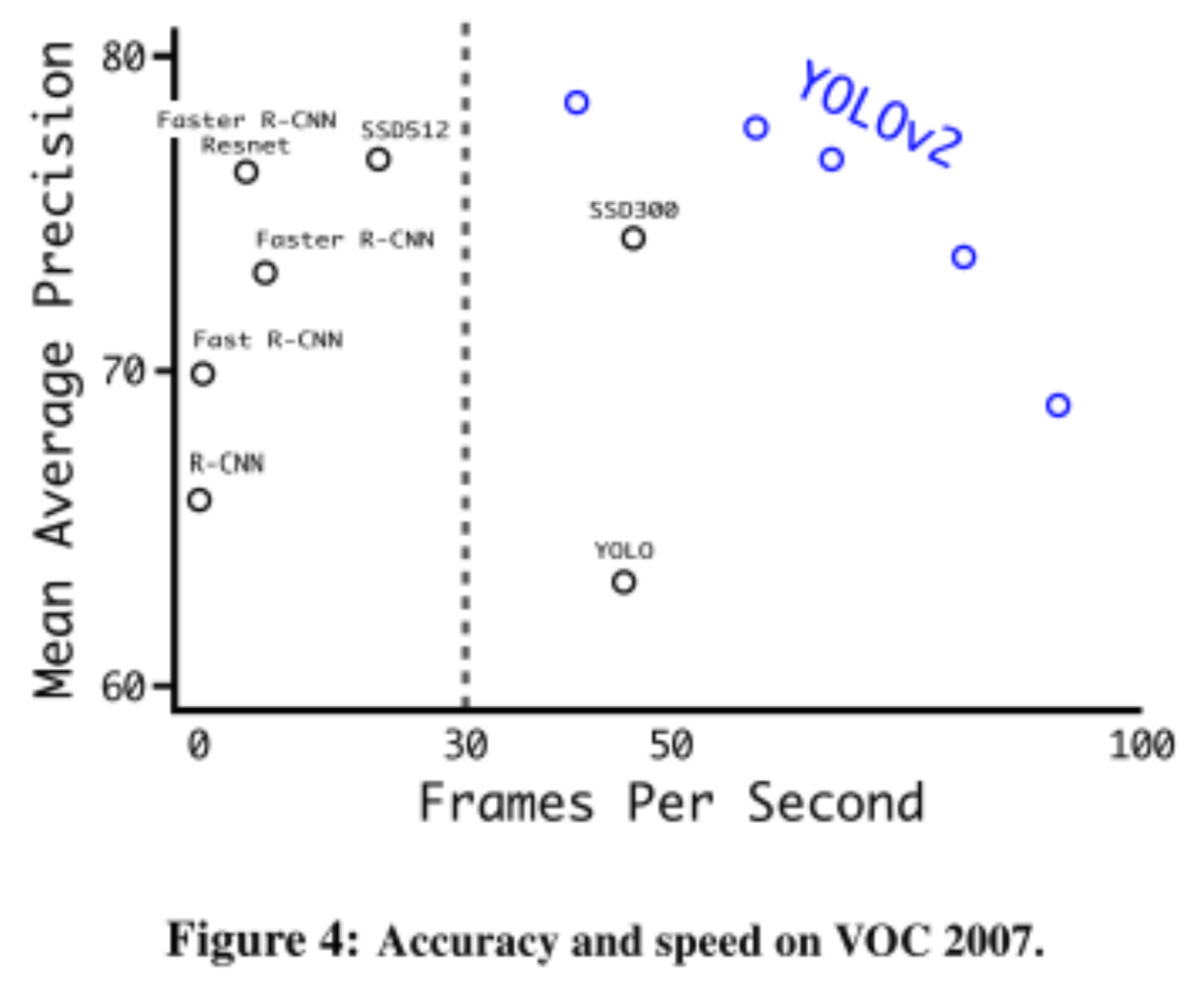


Fig. 2: Network architecture.

Methodology

Network Architecture

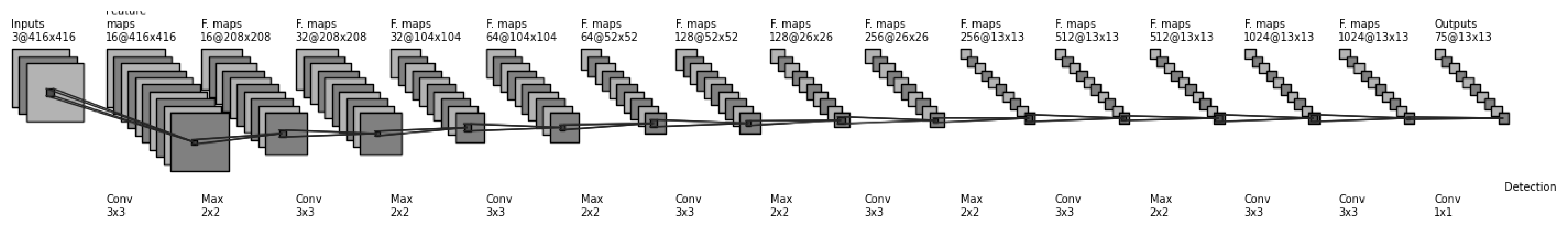


Fig. 3: Network architecture.

Text about Transfer learning and network architecture, training - validation set

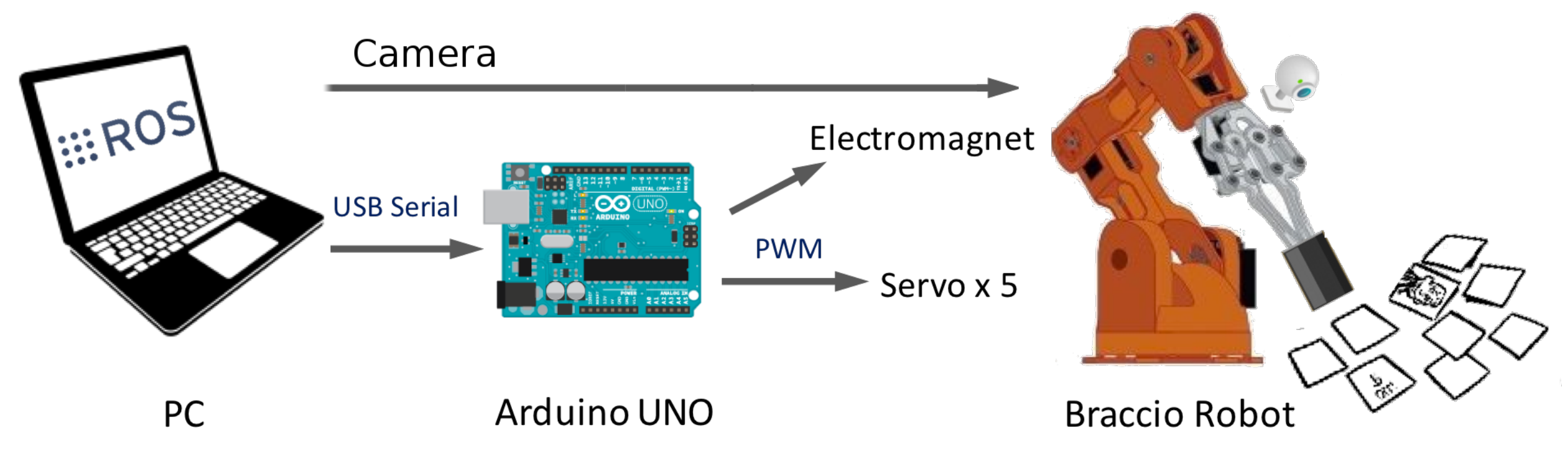
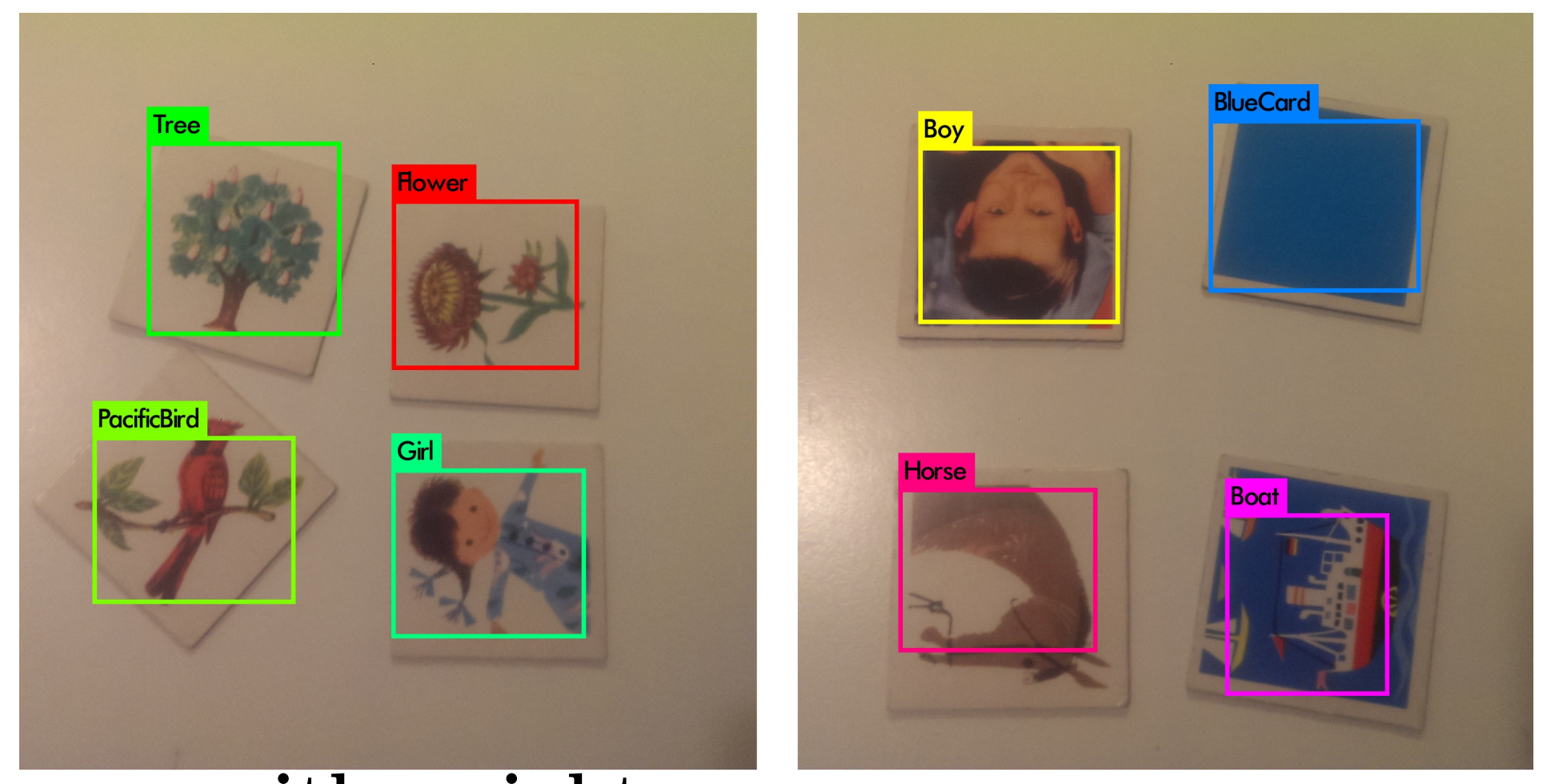


Fig. 4: Hardware setup

Hardware Architecture (shorter)
We used a pre-trained TinyYolo net (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.

Outcome



Add image with weights

- network from tiny yolo
- 10 cards + classes
- architecture
- Filters from results
- Camera input with detection labels