INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, ALLAHABAD
VII Semester B.Tech in Information Technology

Report

HUMANOID ROBOTICS ASSIGNMENT 3 (C1)

Robot Object Grasping using Convolutional Neural Network

By :-

Chinmay Shravanbal Tayade (IIT2018138)

Table of contents

1. Introduction 2

2. Problem statement 3

3. Dataset description 4

4. Language , tools and model 4

5. Grasping method 5

6. Implementation & Results 6

7. References 12

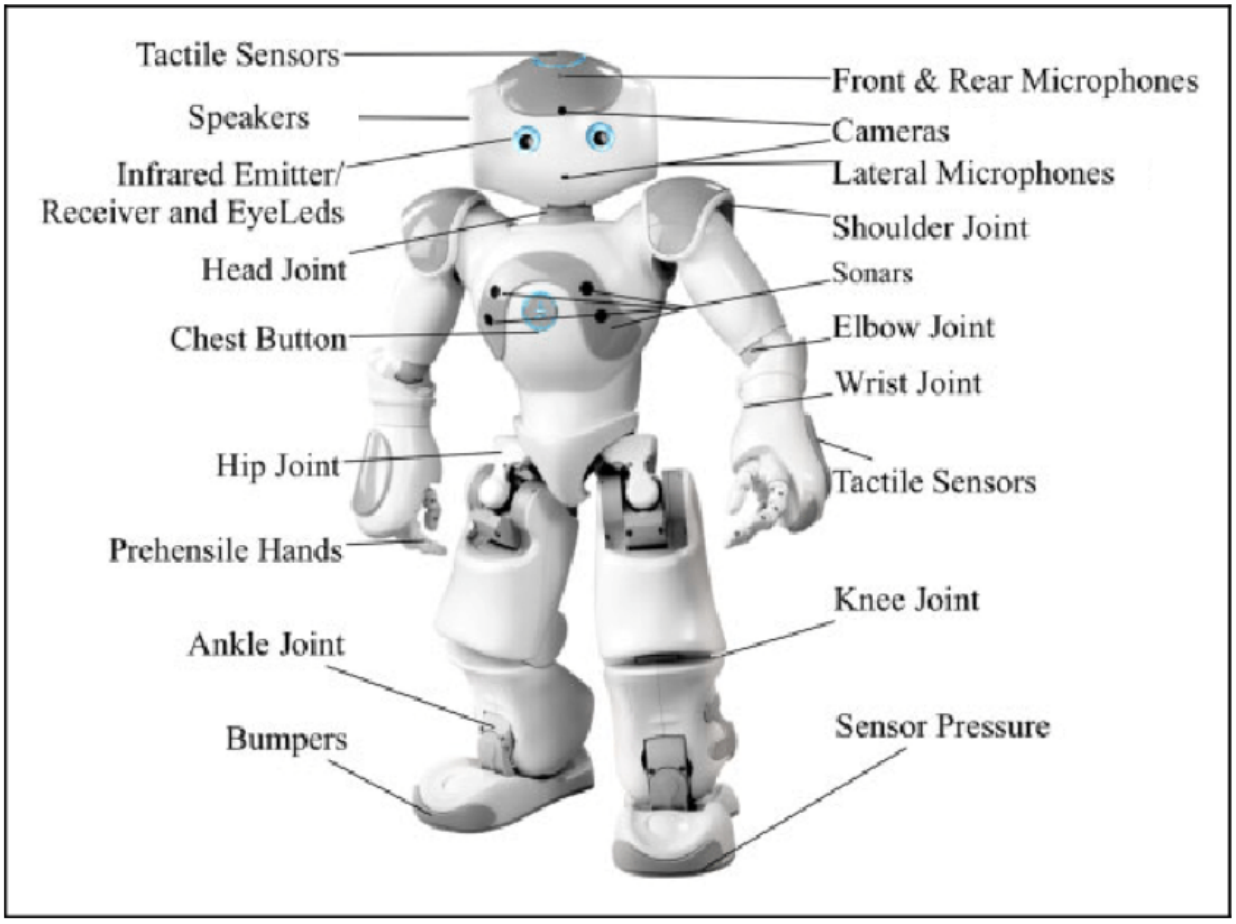
Abstract: The purpose of this paper is to determine the graspable area for a robot placed on the top table using a CNN classification model. performs a single-stage regression to graspable bounding boxes. It is possible for our network to simultaneously recognize an object and find a good grasp rectangle in one step. Using a locally constrained prediction mechanism, this model predicts multiple grasps per object. Especially on objects that are graspable in different ways.

1. Introduction

While many people associate the word "robot" with Hollywood humanoid characters, robots are mainly mechanical devices programmed to perform specific repetitive tasks. People use them routinely to perform boring, dirty, or dangerous tasks that they don't want to do. It is possible to program a robot to perform some tasks that would be too complicated for a human to complete. In general, these robots are industrial in nature and are used for a number of purposes ranging from welding parts on auto assembly lines to interacting with humans in service areas. When you use the self-checkout machines at the grocery store, purchasing tickets from computerized ticketing systems , it might not seem like you're dealing with a robot. Obviously, robots have an impact on everyday life in the service sector.

DETAILED DIAGRAM OF A HUMANIOD ROBOT

This is NAO



2. Problem statement

Solve the tabletop object grasping problems using a grasping rectangle with the use of classification model CNN.

3. Dataset description

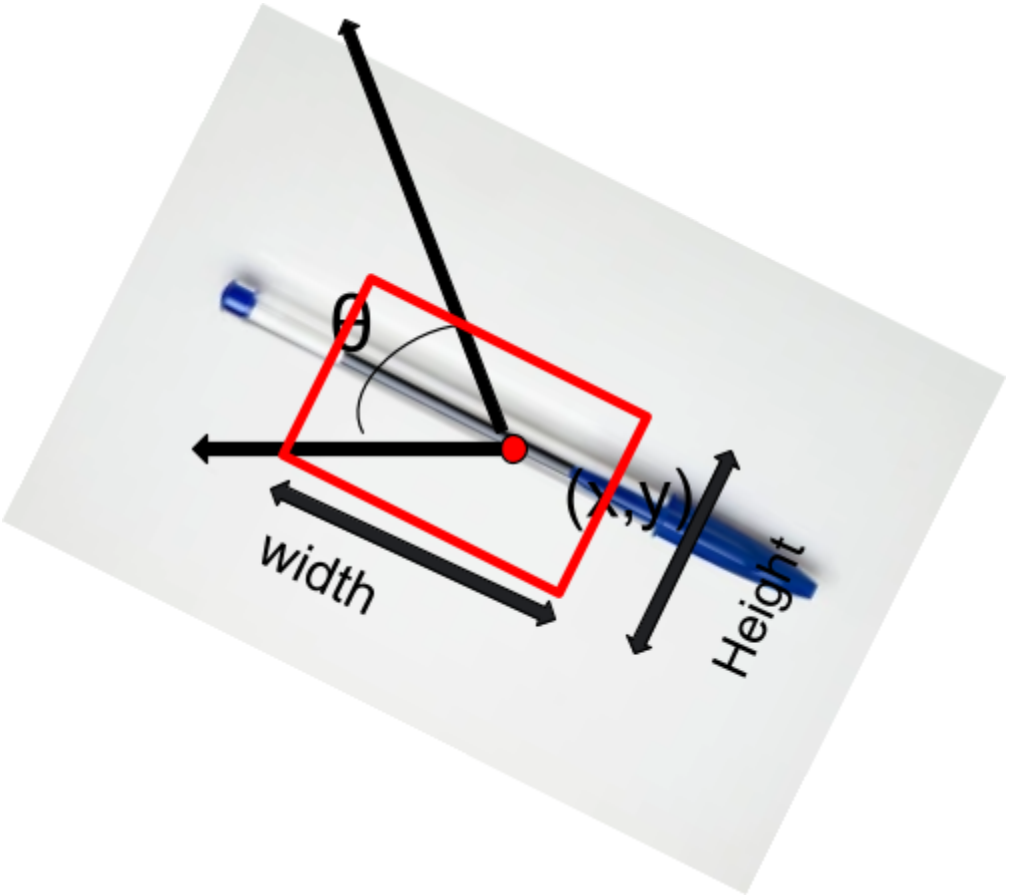
LINK :
https://drive.google.com/drive/folders/1_xPLPNAba0vAGOCZWnQVEWV1RKtRx9qV?usp=sharing

Dataset used :- Cornell grasping dataset for training my model.
Dataset Description :- consisting 855 images of 244 different kinds of objects,
Clicked from various kinds of orientations of the object view.
Each distinct image is labeled with multiple ground truth grasps corresponding to possible ways to grab the object.

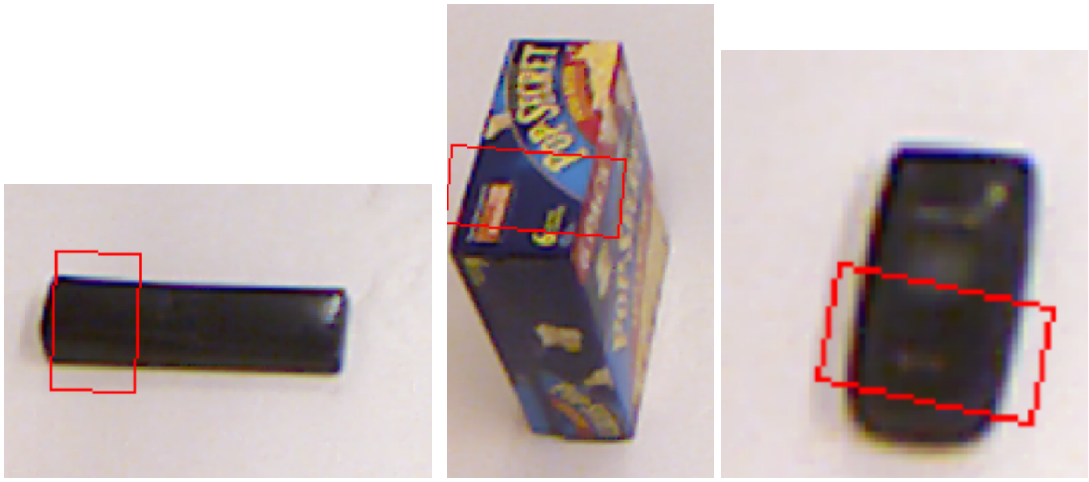
4. Language ,tools and model

- Python (3.8.3)notebook Using Conda environment system as well as Jupyter server on Google colab
- Torchvision is a helpful library for the study of computer vision. It contains datasets ,model architectures and also common image transformation for computer vision.
- matplotlib.pylab is a module which provide matlab type namespace in python
- Torch is a tensor library that allows strong GPU support for the notebook.
- Image is used to represent PIL image
- ImageDraw provides 2D graphics support for images, which is used to create and retouch preexisting images and to generate graphics.
- Glob helps to return files paths related to a specific pattern in code.
- Numpy this module helps to deal with arrays and matrices easily.
- Classification model - CNN

5. Grasping method



We are going to detect the grasp :
A grasp has X coordinate , Y coordinate , θ, height and width . Refer to the diagram above to understand in detail.
Samples of grasping rectangles on the Cornell dataset images :



- To create a dataset, create a list of examples that will be fed into a neural network
- Create a dataloader to specify how datasets are loaded into neuralnets (batch size, order, computation optimization)
- Using matrix math, create a model for the input tensor to be transformed into the output tensor
- After initializing the neural network we will move forwaionMonitoring, evaluation of matrices, logger, back and forth and finally Visualize the plots of Training and validation Accuracy and Loss.

10. Implementation & Results

- Imported the required libraries
- To create a dataset, create a list of examples that will be fed into a neural network network, use the following functions ->
- Generate_listof_grasping_rectangles

  This function generate_listof_grasping_rectangles function is used to generate/form a list consisting coordinates of the

Rectangle used to determine the grasping area of the object ,in this list each element will be either x or y coordinate

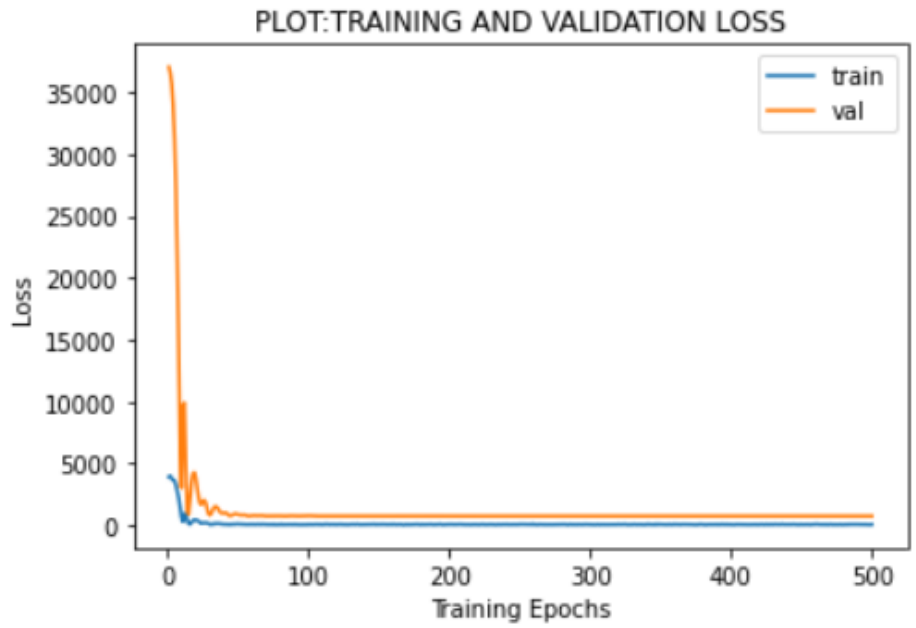
A particular vertex

- Generate_list_of_images():generate_list_images function is used to generate/form a list consisting of the images of the Cornell dataset
- Access_image_tag(name_of_file:
  This function will be used to orgnanize the labels with the images
- Display_image_tag :
  This function will be used to display the image label corresponding to the image
- Generated an organized list of all image addresses.
- Accessed PIL Image.open() is a library which provides editing access to pictures. .
- Transformation of images using torch.vision.transforamtion function
- Resized the images and sizes of tags (labels).
- Selecting the grasp randomly
- Tested the transformer
- Created a dataloader to specify how datasets are loaded into neuralnets (batch size, order, computation optimization)
- Converted list of coordinates to grasp representation -> tensor([x, y, theta, h, w])
- Convert the grasp representation to a list of coordinates
- Transform function testing
- Using matrix math, create a model for the input tensor to be transformed into the output tensor
- After initializing the neural network we moved forward on Monitoring, evaluation of matrices, logger, back and forth and finally Visualize the plots of Training and validation Accuracy and Loss.

- Trained the model on 500 epochs.

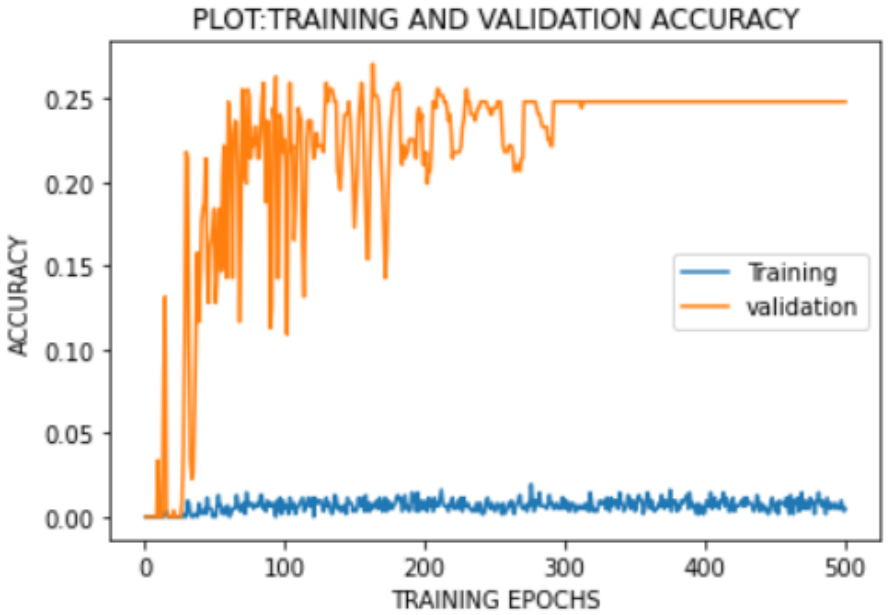
```
Epoch 0/499, current learning_rate=0.001
Copied best model weights!
train loss: 3924.512520, accuracy: 0.00
val loss: 37036.714844, accuracy: 0.00
----------
Epoch 1/499, current learning_rate=0.001
Copied best model weights!
train loss: 4010.518982, accuracy: 0.00
val loss: 36574.859375, accuracy: 0.00
----------
Epoch 2/499, current learning_rate=0.001
Copied best model weights!
train loss: 3770.485460, accuracy: 0.00
val loss: 35661.277344, accuracy: 0.00
----------
Epoch 3/499, current learning_rate=0.001
Copied best model weights!
train loss: 3664.120355, accuracy: 0.00
val loss: 34029.949219, accuracy: 0.00
----------
Epoch 4/499, current learning_rate=0.001
Copied best model weights!
train loss: 3620.387318, accuracy: 0.00
val loss: 31352.791016, accuracy: 0.00
----------

show more (open the raw output data in a text editor) ...

----------
Epoch 499/499, current learning_rate=1.52587890625e-08
train loss: 83.705157, accuracy: 0.48
val loss: 759.812134, accuracy: 24.81
```

-
- Plot of training and validation loss



-

- Plot of Traing and validation Accuracy



-

12. References

[1].Real-world Multi-object, Multi-grasp Detection by Fu-Jen Chu, Ruinian Xu, Patricio A. Vela
[2].Deep Grasp: Detection and Localization of Grasps with Deep Neural Networks | Papers With Code