**Train Customized Model for Object Detection**

**Reference:**

* <https://blog.insightdatascience.com/how-to-train-your-own-yolov3-detector-from-scratch-224d10e55de2>
* <https://github.com/AntonMu/TrainYourOwnYOLO>

**Purpose**

To train a model which can recognize Husky robot

**Environment**

Windows 7

**Necessary Tools and Packages**

* Python 3.6
* Visual Object Tagging Tool (use for labeling images and export training files)

(<https://github.com/Microsoft/VoTT/releases>)

* Packages (install through pip install command on terminal):
  + tensorflow
  + keras
  + matplotlib
  + numpy
  + pandas
  + pillow
  + opencv-python

Use the command to install:

pip install <package name>

(Note: The packages listed above may meet the requirement for training and detecting purpose. If there is an error with the message, "No module named <package name>", it will need to install the package. A picture with the list of packages is on the last page as a reference. Please check and install if need.)

**Implementation**

**Step 1: Clone the package to your computer**

Open command prompt (or terminal) in your computer and enter:

git clone https://github.com/AntonMu/TrainYourOwnYOLO

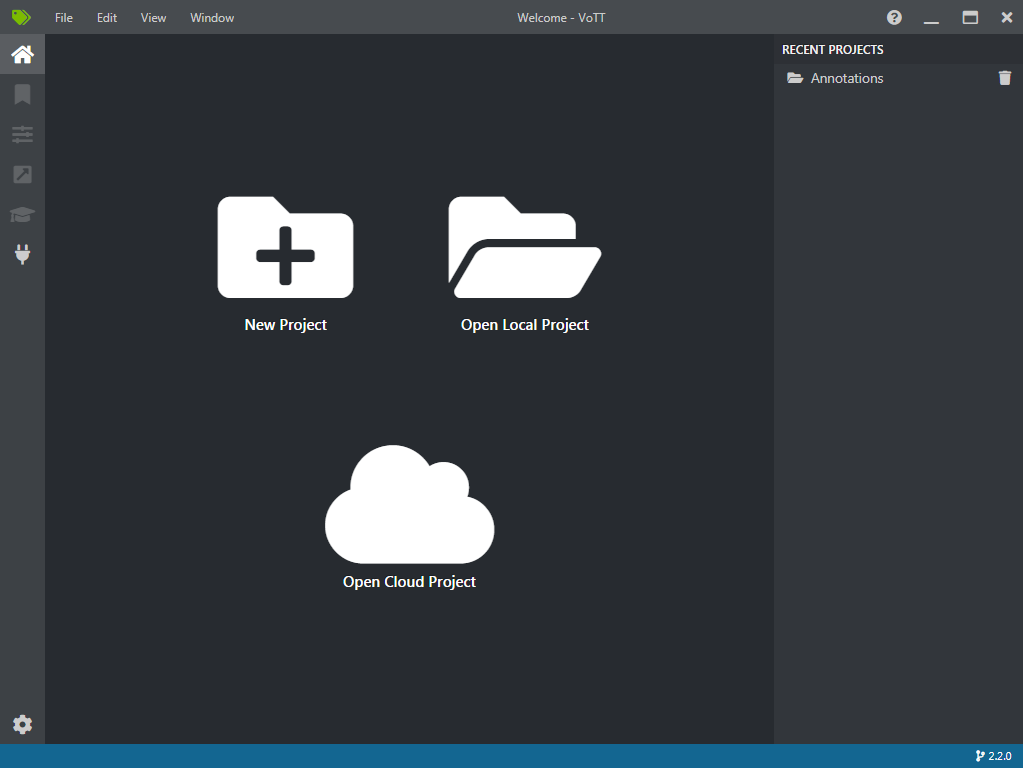
**Step 2: Prepare images for training and label them**

Go to the package you cloned and replace the pictures you want to train in the folder:

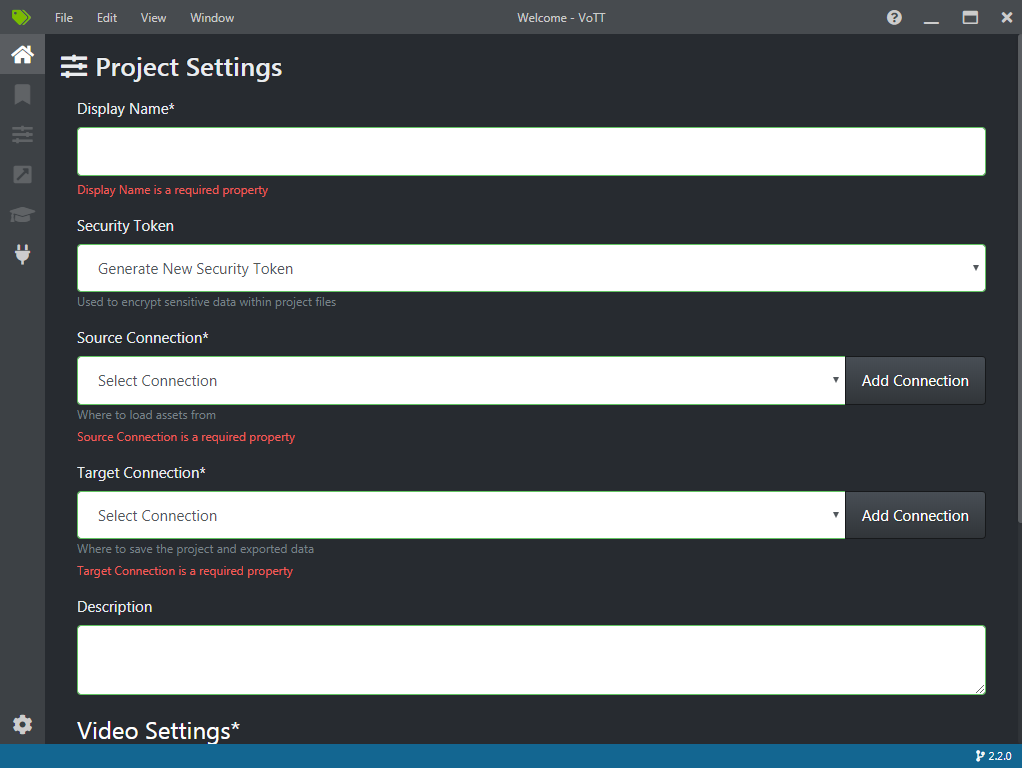
**TrainYourOwnYOLO/Data/Source\_Images/Training\_Images**

Try to prepare more than 100 images, or the training result may not be ideal.

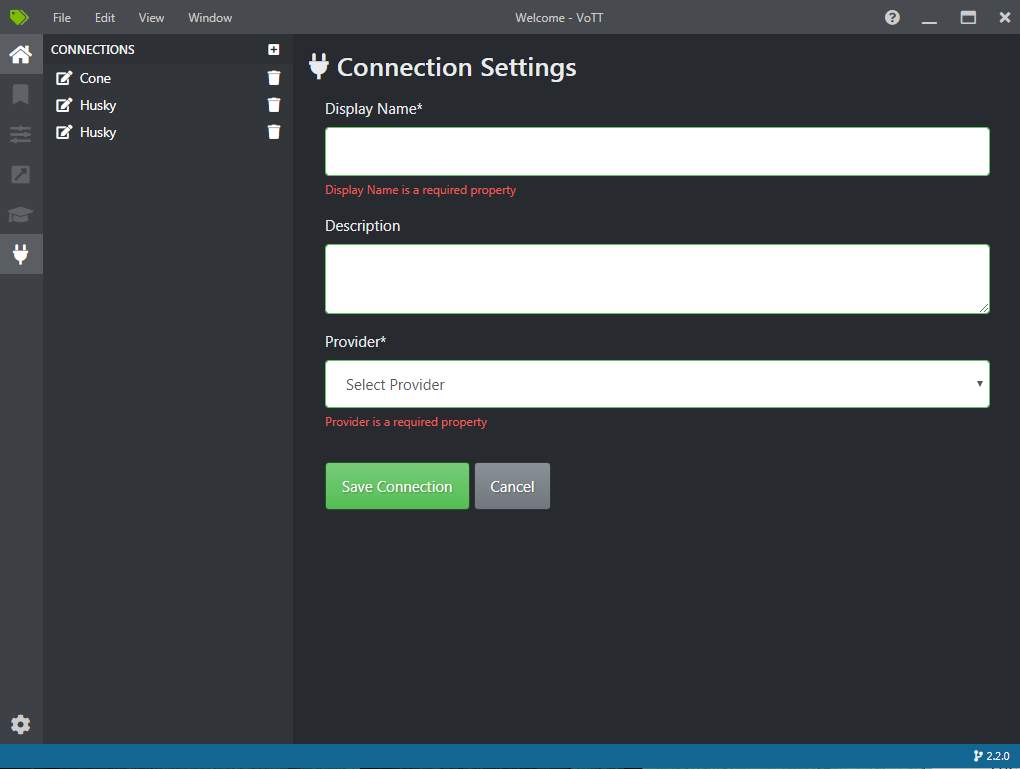
Use **Visual Object Tagging Tool** to label the images. Click on **New Project** to create the project (Image 1). Type the project name on **Display Name**, then click **Add Connection** (Image 2). In connection settings, type anything on Display Name. It could be the label name you want to use for labeling. In **Provider** box, choose **Local File System** and select the folder which the training images locate (in the case of this tutorial, images are in **TrainYourOwnYOLO/Data/Source\_Images/Training\_Images**), then click **Save Connection**. Select the created connection in **Source Connection** and **Target Connection**, then click **Save Project**. Now, the images will appear and ready to label (Image 4). After labeling, press Ctrl + E to export necessary files for training. (**Important note**: Make sure to go to the tool bar on the left side, find **Export Settings** and select **Comma Separated Values (CSV)** as **Provider**, then press **Save Export Setting**.)



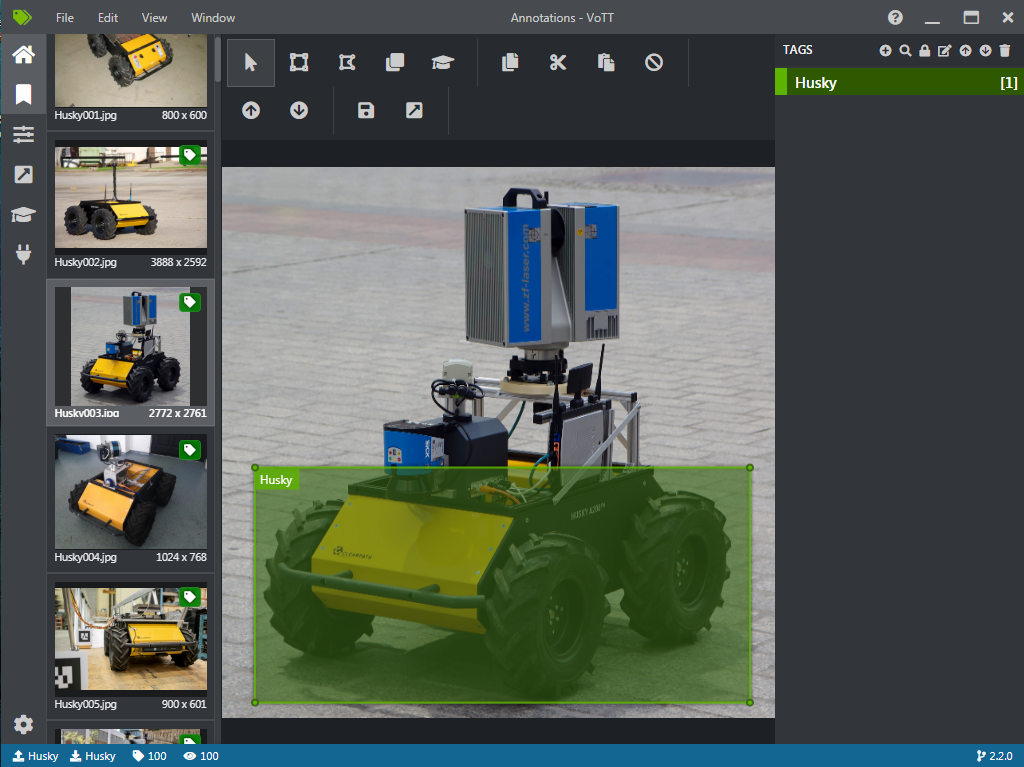
^ Image 1



^ Image 2



^ Image 3



^ Image 4

On terminal, type:

cd **TrainYourOwnYOLO/1\_Image\_Annotation**

**Then type:**

**python Convert\_to\_YOLO\_format.py**

**Now the files are prepared and ready to train.**

**Step 3:**

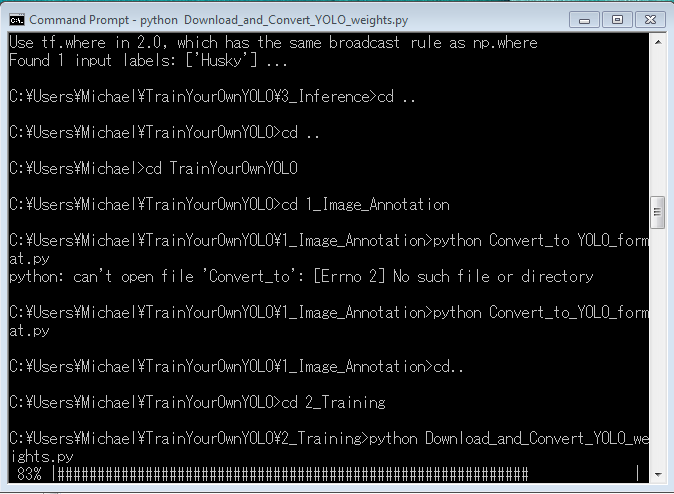
On terminal, go to another folder

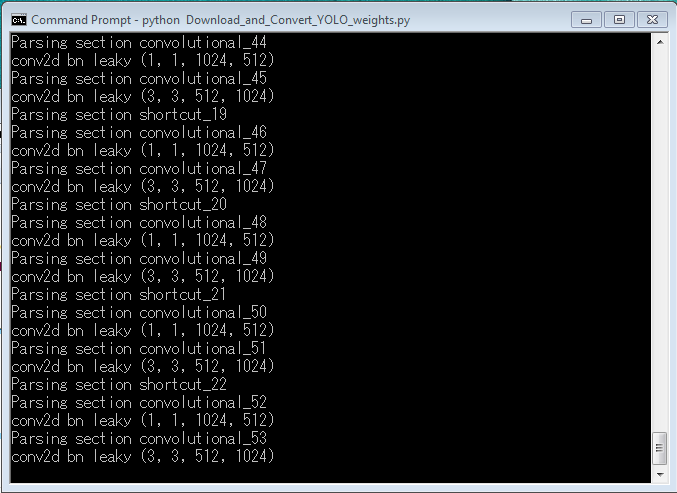
cd **TrainYourOwnYOLO/2\_Training**

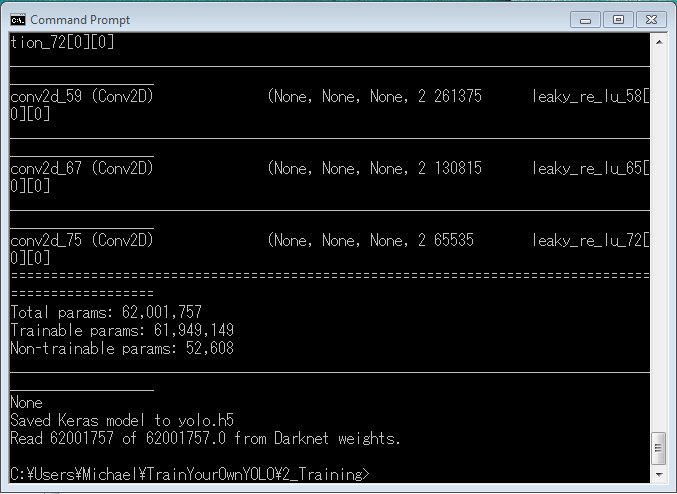
**Run the code:**

**python Download\_and\_Convert\_YOLO\_weights.py**

The process shows below:



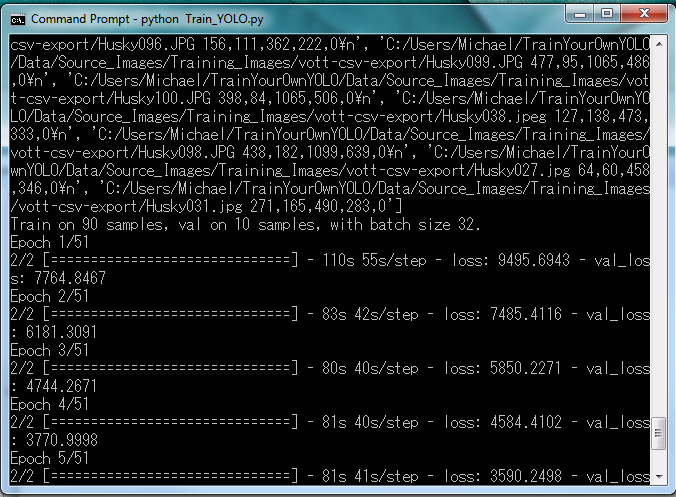


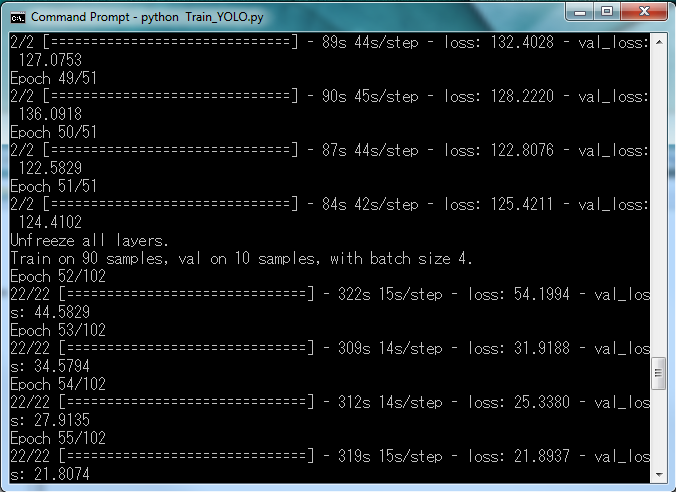


When it finishes, run another code:

python Train\_YOLO.py

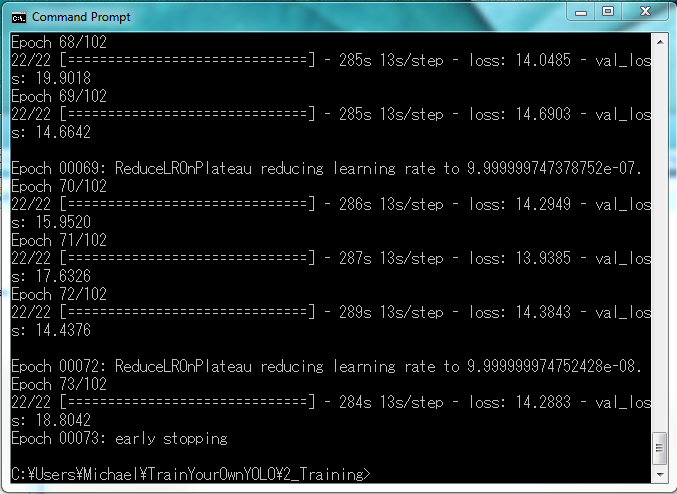
The process shows below. It can be observed that the data loss is decreased when training time increases.





Below shows the end of the process. It will cause an early stopping since the training cannot decrease data loss anymore. Don't worry about it, the model is still trained successfully. It is located in:

**TrainYourOwnYOLO/Data/Model\_Weights**



**Step 4:**

To test the trained model, put the images you want to test in the folder:

**TrainYourOwnYOLO/Data/Source\_Images/Test\_Images**

On the terminal, move to another folder by the command and run the code:

cd **TrainYourOwnYOLO/3\_Inference**

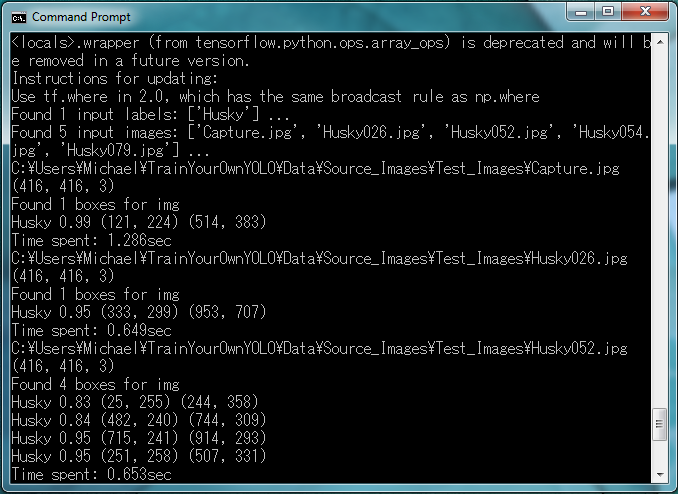
python Detector.py

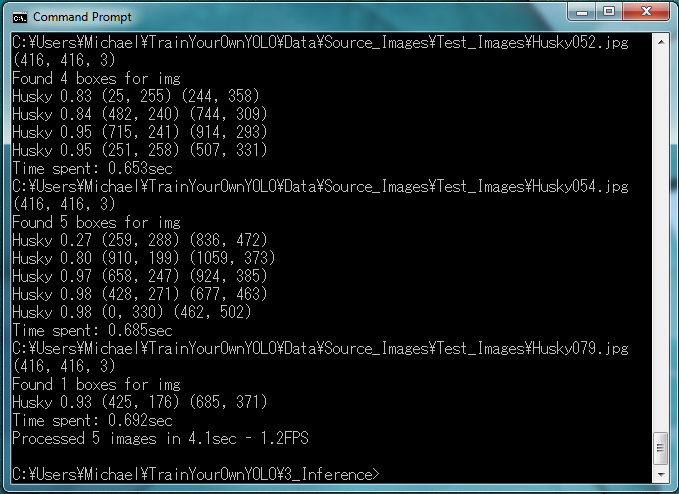
The results will show on command prompt and the folder:

**TrainYourOwnYOLO/Data/Source\_Images/Test\_Image\_Detection\_Results**

The Detector.py can also detect the video files (in mp4 format).

(Note: Test images need to be jpg file (no jpeg), or it will not be read by python script.)







^ Detect in the simulation environment



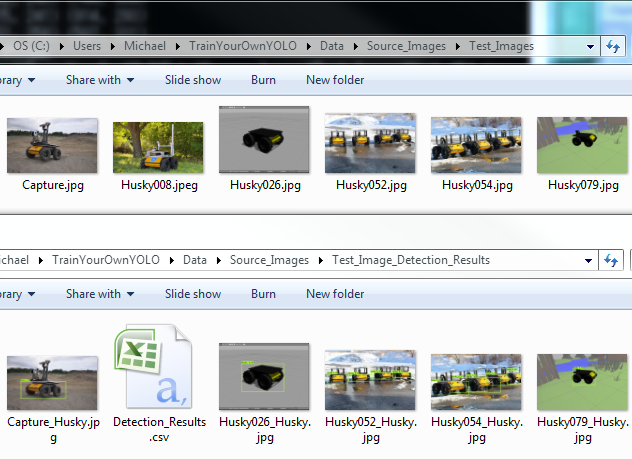
^ Detect several Husky in one picture.



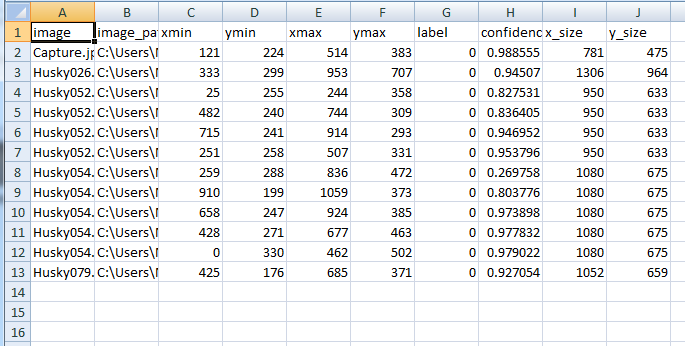
^ It could detect on the wrong position in the picture since the several same objects are too close to each other.



^ The model is trained to detect the main body of Husky robot. No matter what stuff is added on the robot, it will not influence the result.

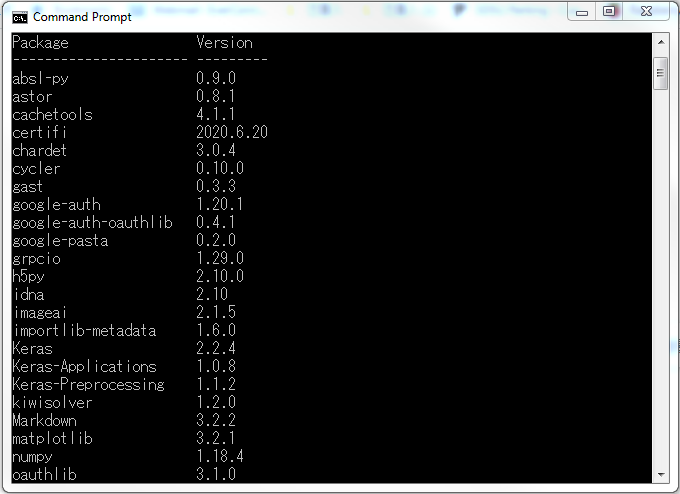


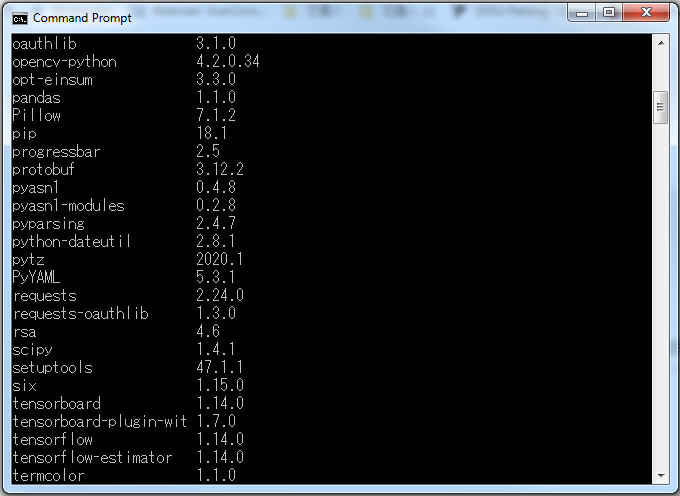
^ Husky008.jpeg is not able to detect by python script. Make sure to keep test files in jpg format.

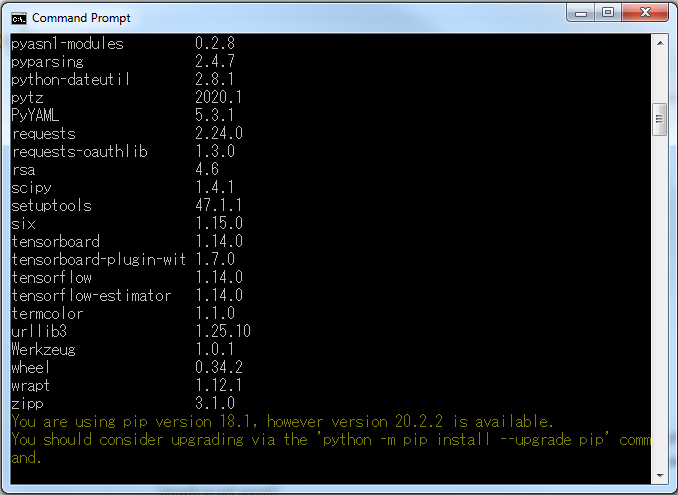


^ The data in the "Detection\_Results" file. It shows the label box positions in the pictures.

**Package list**







**- End -**