Ahmed F. Bingöl
GitHub :: ahmedbing | mail :: bingol20@itu.edu.tr
July 3, 2020

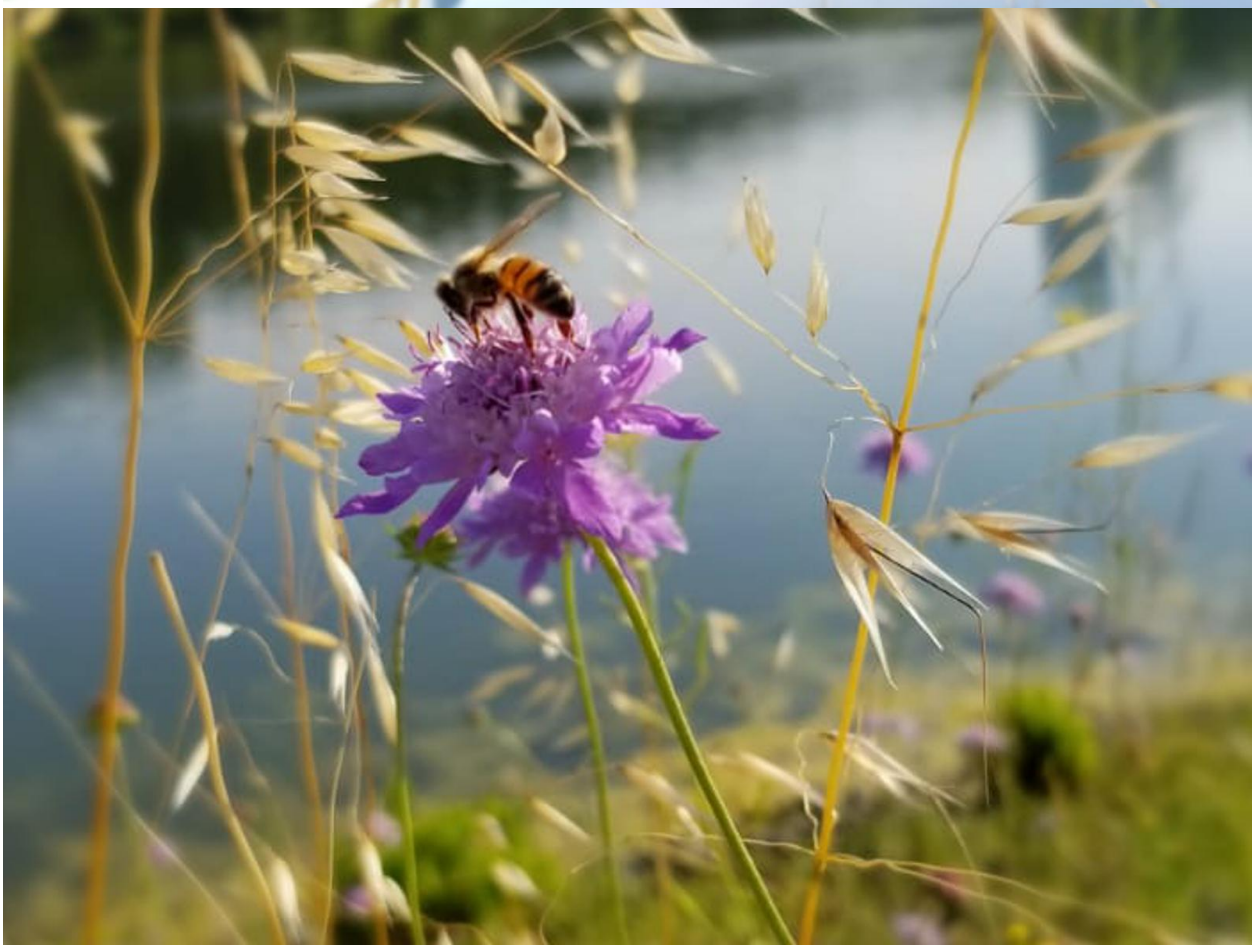
BBL 588 Final Exam

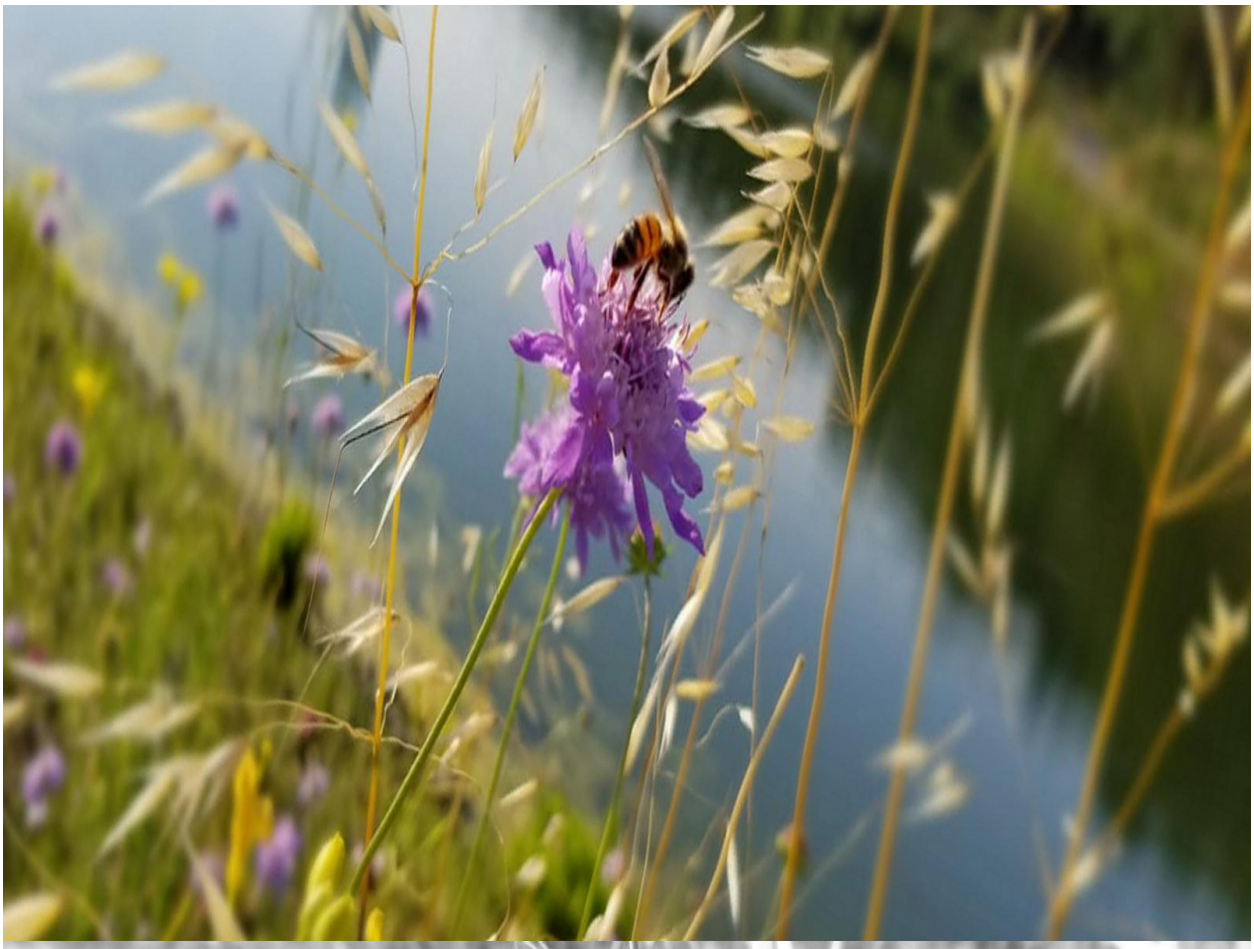
1- In the first problem, i will try to explain all steps i done for reach to result. I could not decide to upload this codes, CNN model, weights, datas because someone from class may cheat from it. Therefore i will provide portion of it, and explain rest of it here.

~ Firstly, i decided to make data augmentation for train a robust segmentation CNN for different conditioned pictures. I used "Augmentor" library which fasten to create create images from image. Code of creating new samples in "data_augmentation.py" I used different methods of Augmentor such as:

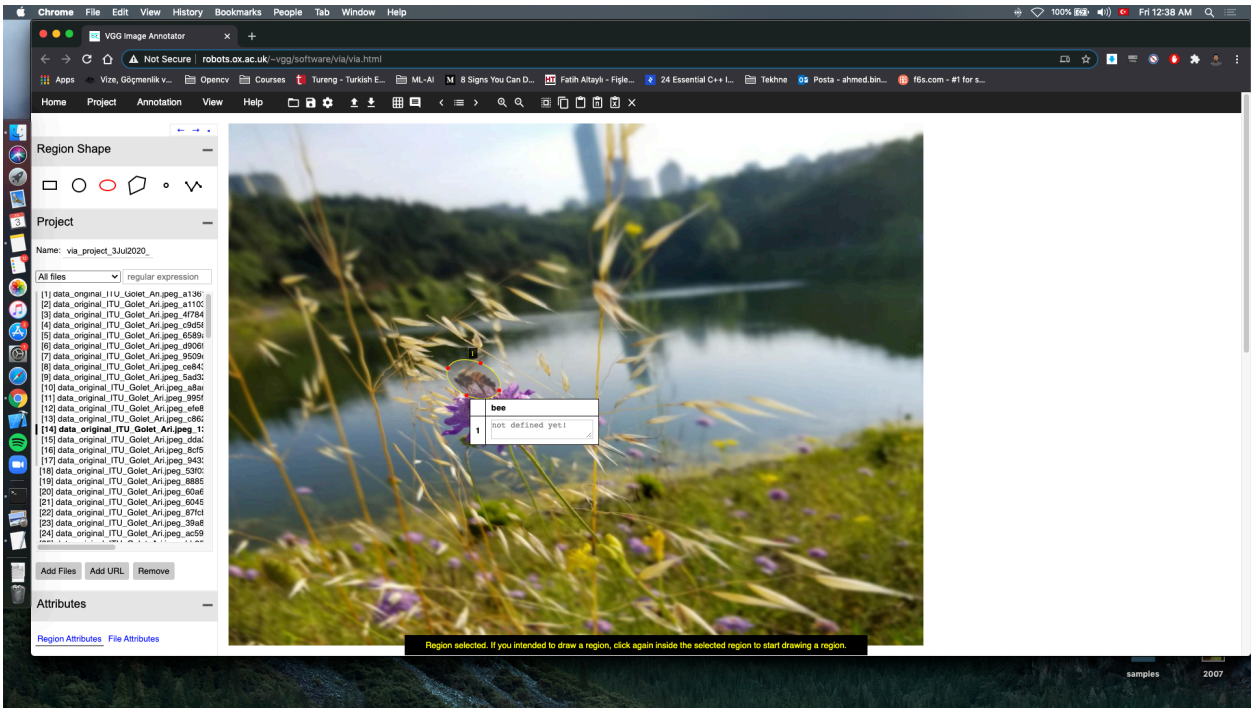
- Flip left to right with %50 prob.
- Flip top to bottom with %20 prob. This it is making picture upside down
- Change to brightness with %15 prob brighten with min factor 0.2 and max 1.8
- Some rotations between 25 angle to right or left with %30 prob.
- Rotating 90-180-270 randomly
- Zooming in and out with %20 probability.
- And lastly greyscale image with %15 probability.

With this methods, i created 125 sample images. I used 75 of this images for training 25 for testing and 25 for validation. Some of examples are below.

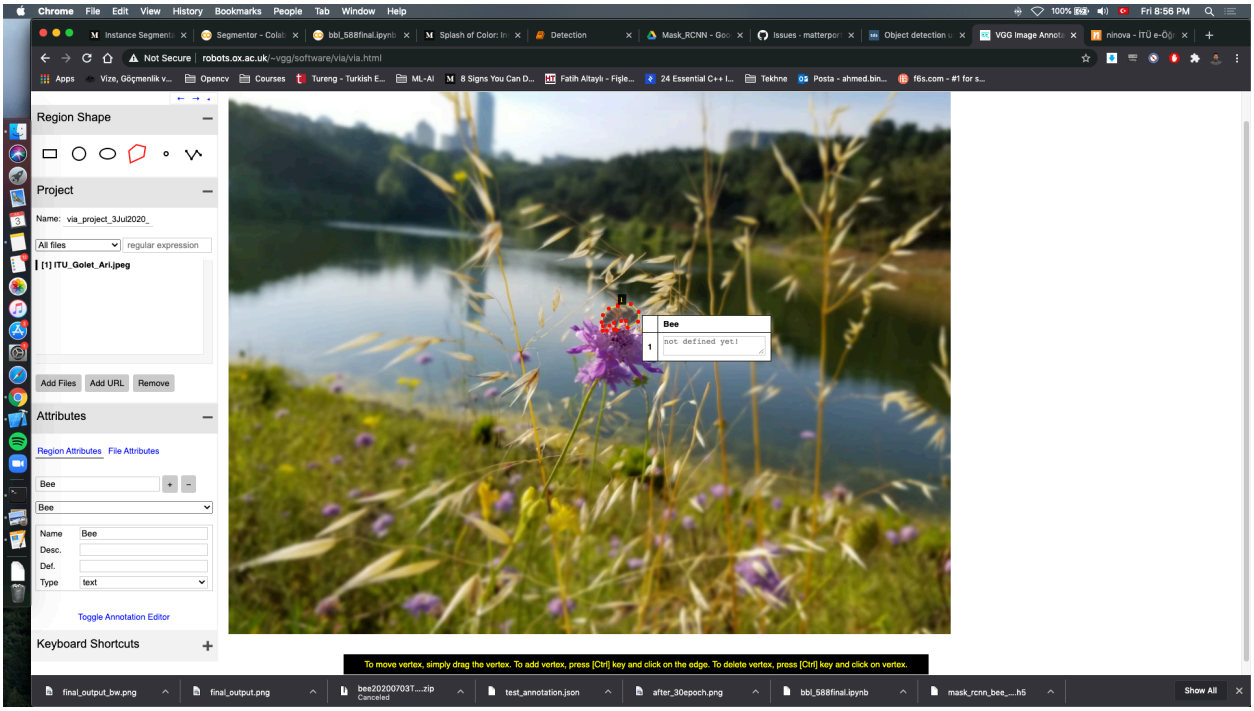




After data augmentation of 125 image, I started to label image for YOLO model. Therefore i used LabelImg and labeled all 125 images. However, then i realized that problem asks to segment bee, not just detection. Therefore , i decided to use Mask R-CNN with ResNet 101 Backbone. Since Mask R-CNN wants JSON data format with polygonal points, i label 2 more times. In below example, i mistakenly label with circle for do it fastly. Since it requires, then i label 125 images again with polygonal points. In the second part i used VGG Image Annotator for obtained required JSON format to train Mask R-CNN weights.

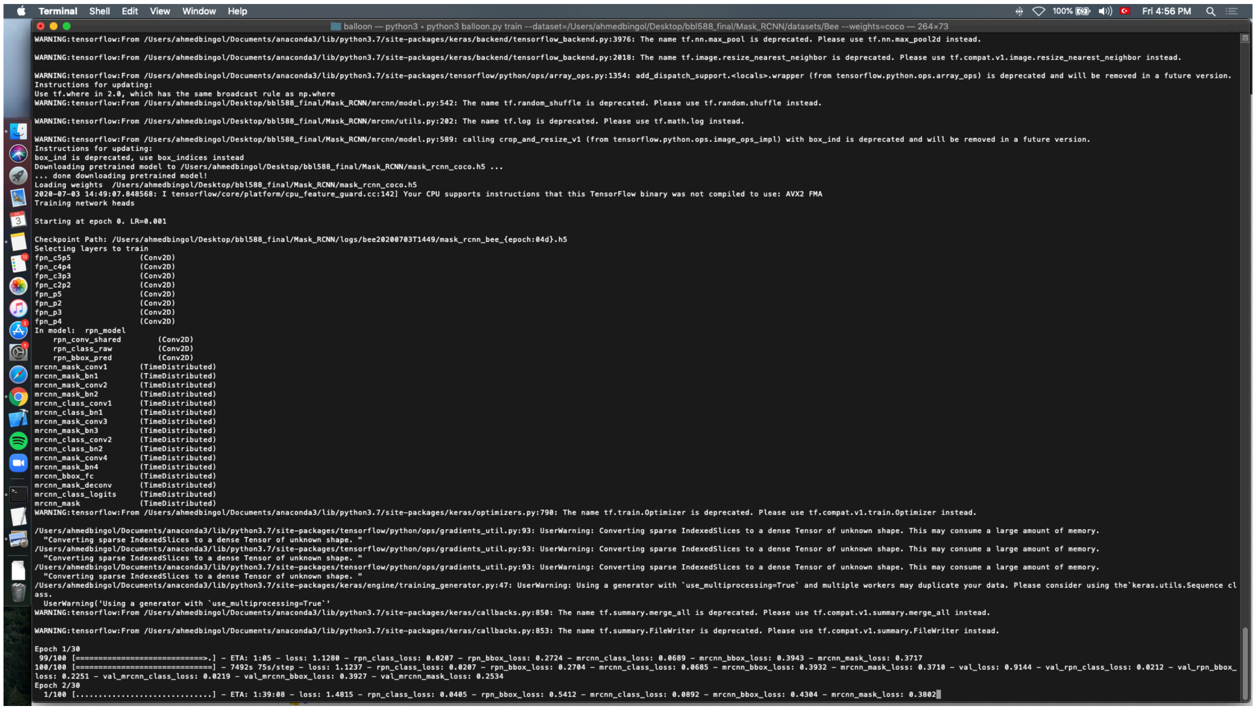


Labeled with circle as mistake

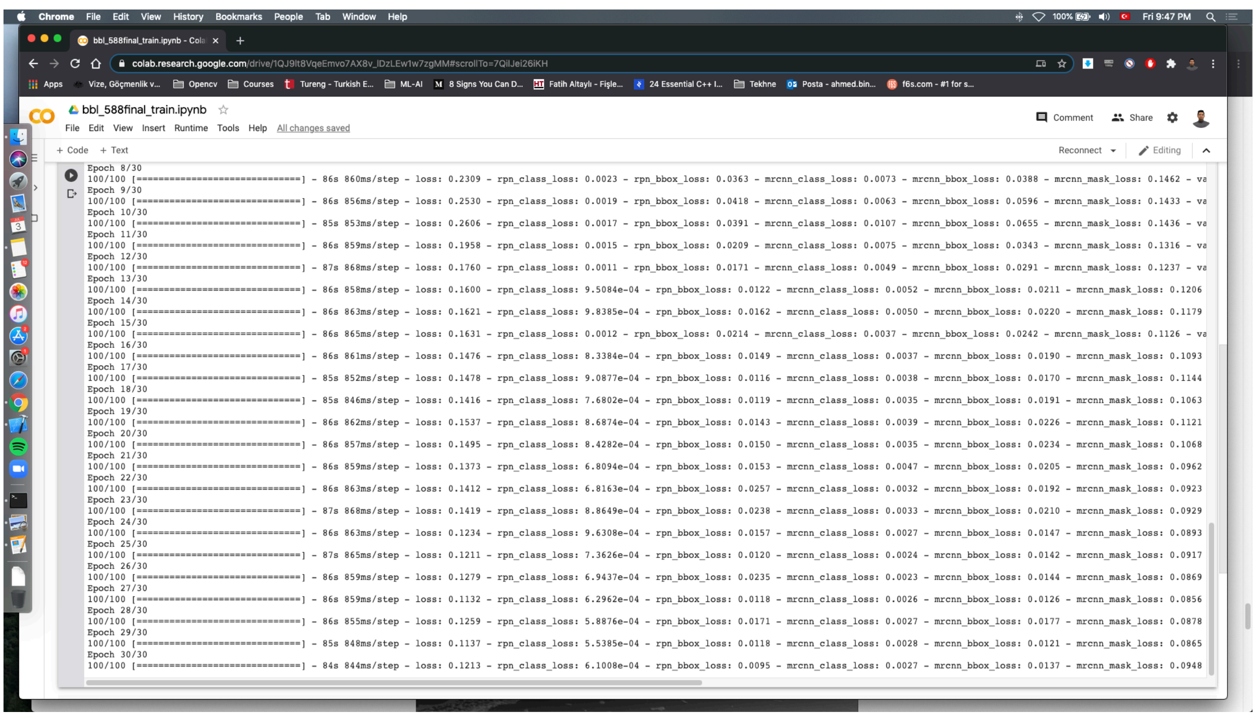


Labeled with Polygonal Points

After labeling process completed for 3 dataset, i started to train my Mask R-CNN model. I spent so much time on my MacBook to build it. However, even 1 Epoch takes around 1:30 hour to finish and i had 30 Epochs to train model. Therefore, i used Google Colab for use it is GPU's. I upload all my files to Google Drive and then worked on remote system to train my model and obtain weights. In Mask R-CNN repo, it suggest to use balloon.py example for complete all steps. Therefore i re-arrange and modify to too many codes from there to create my weights. My Colab codes inside to "bbl_588final_train.ipynb" file and it uses balloon.py for train. I modify config file for Model train configurations, model.py file for some Tensorflow Keras version adaptation and balloons.py file for create Bee class and complete dataset related works there. It suggest to give initial values of Coco dataset to train model, therefore i also downloaded to Coco dataset.

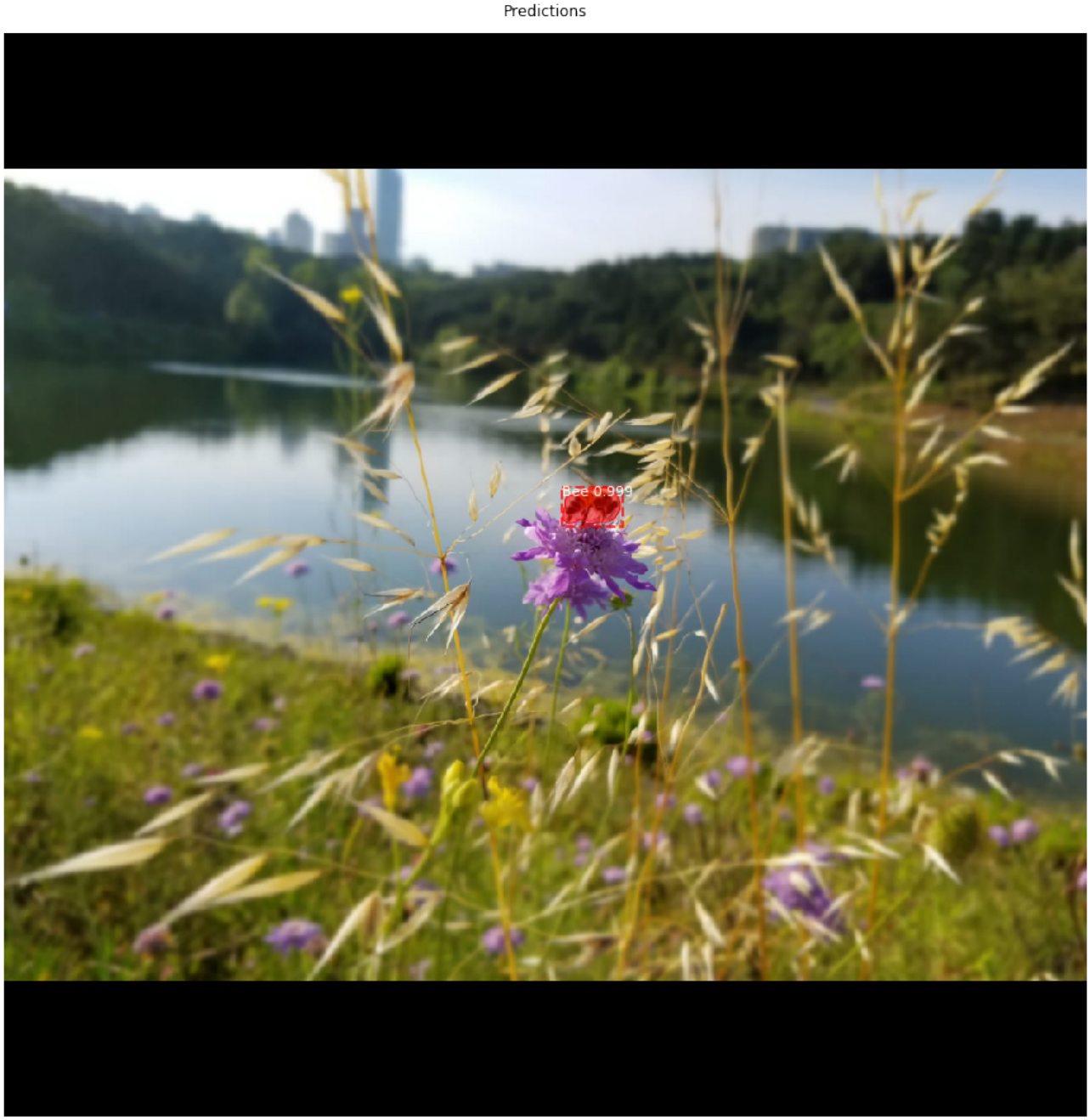


My PC terminal in training. First Epoch takes around 2 hours. Therefore i pass to Colab



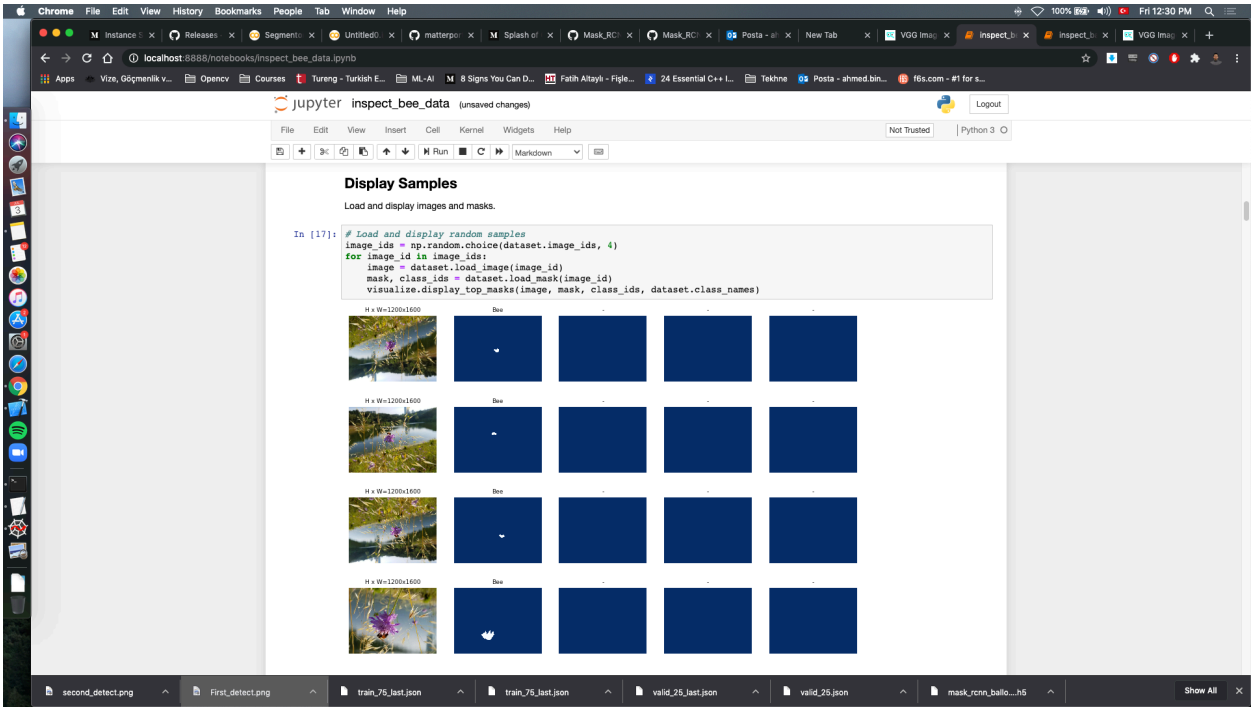
Training result in Colab. Each Epoch almost takes 86 sec. with GPU. Results saved in train.ipynb file

In the last part, i made tests with my test data in “Detection.ipynb” file. Since it is using some modules and methods from sample/balloons example some variable names stays at that. In that jupyter notebook, i shown to configuration parameters of training, load validation data (In our case which is 1), load to model with new trained weights and run to segmentation and makes a color splash version of segmented images. Outputs of the files are shown below:



Color Splash version of output, only Bee is colorful

Some final comments, since we make segmentation from only 1 image, this weights are so overfitted for this image. However, i observe that my algorithm still works very well in very bad noise greyscale and rotated images. I made most of my works on Jupyter Notebooks to show my code results, since this type of projects with too many library with different versions and Hardware dependent are very hard to run in different systems. My Loss function in the 30 Epoch was 0.1213 which was higher than 29th Epoch with 0.1137. And for the final image my ground-truth label as performance was 97% covered well to Bee.



Masks of different train images. Since there is only Bee class, other blocks empty.

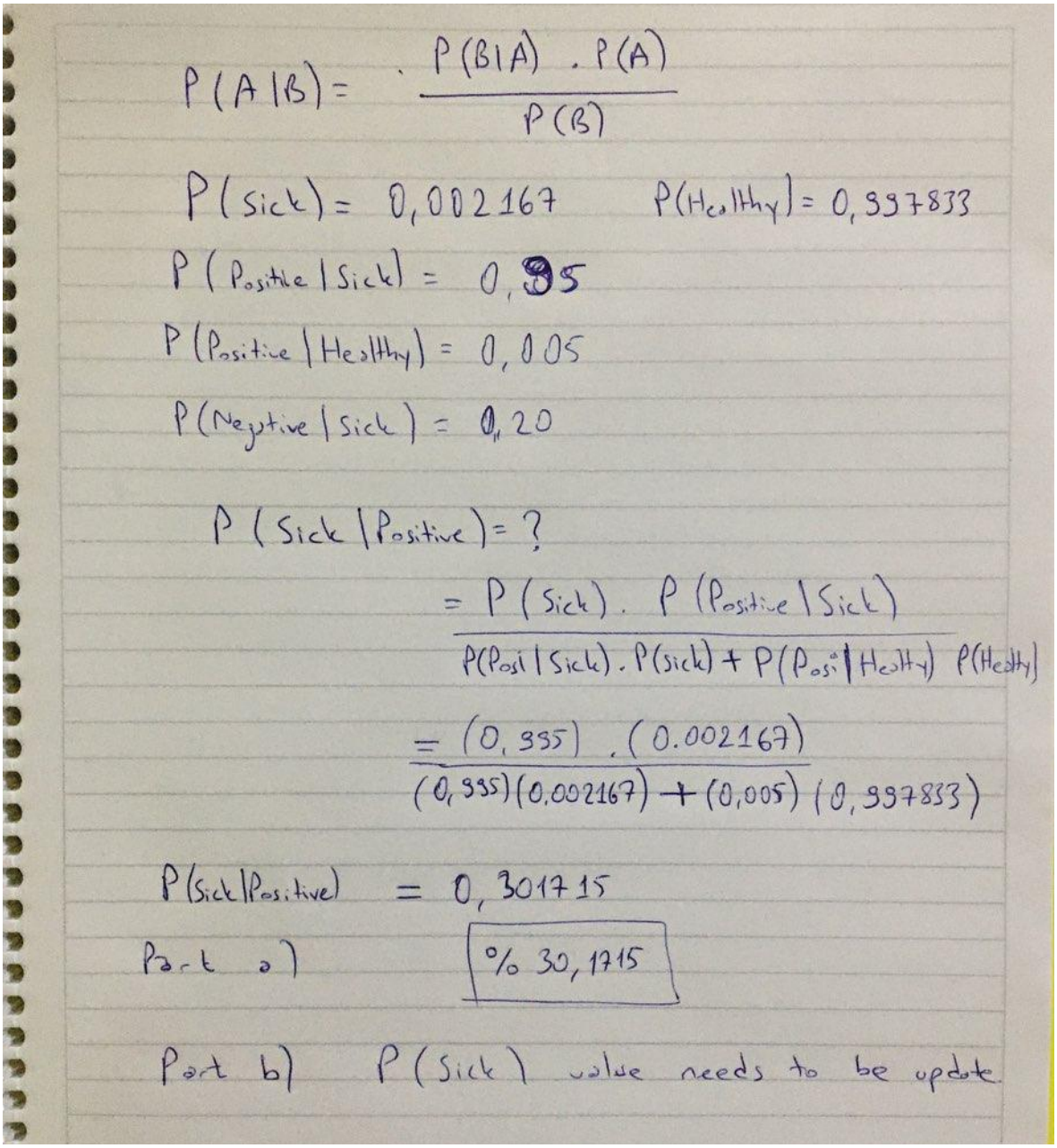
I upload files to my Github Repo under BBL588. My other works related with course also exist there.

Problem 2:

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)} \qquad \text{and} \qquad P(A \mid B) = \frac{P(A \mid B) \cdot P(A)}{P(B)}$$

Lets define P(A) and P(B) ; P(A) : Being Sick ; P(B) : Test positive

Question asks P(Sick | Positive) = ?



I used 2. formula from up and expand P(B). Result is %30,1715 as shown at below :