

# CITS5551 & CITS5552 Software Engineering Design Project

## Weed Detection - Maintenance Manual

*Jiaqi Han 22292229*

### I. Introduction

This document describes maintenance process of the project and is intended as a guide for future developers to reproduce and alter the processes by which the final product is made live. It mainly focuses on how to deploy the Mask\_RCNN on Google Cloud Platform. If you have any other questions, you can send email to Google Support Team.

### II. Deployment - Google Cloud Platform

The Google cloud platform can provide GPU and huge storage without PC limitations.

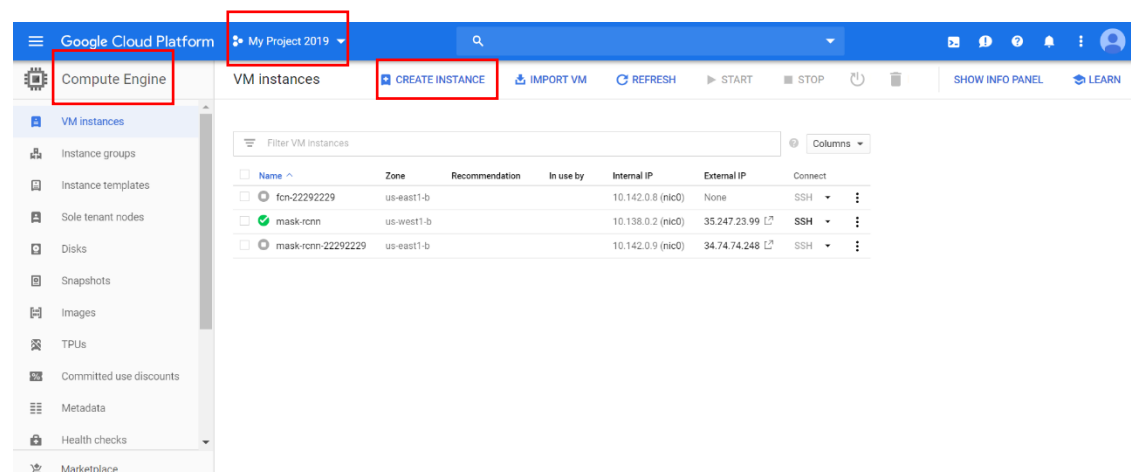


Figure 1: 'mask-rcnn' Weed Detection VM instances of Google Cloud platform

Steps in detail:

**Step 1: Apply GPU quotas:** The algorithm will run faster and better with GPU support, so the first step is to apply GPU quotas for the VM instance of Weed Detection project. For the project "My Project 2019" I have applied 4 quotas for Global and 4 quotas for GPU "NVIDIA P100 GPUs"

Choose menu: IAM & admin/Quotas → Setting parameters: Services as "Compute Engine API" / Metric as "NVIDIA P100 GPUs" (the kind of GPU you want)/ Region as "us-east1" & "Global" (Not all regions have GPUs support) → Edit Quotas: give your personal details and reasons, the email will be sent to Google Support Team automatically.

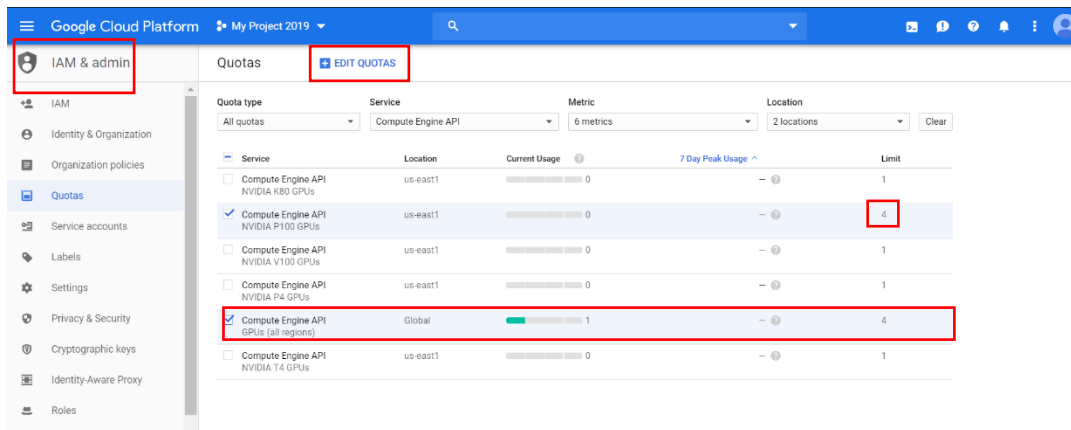


Figure 2: GPU quotas checking and application for the project

The application is successful once received such emails. (Sometimes they may forget to increase the Global Quota, best to apply both quotas at the same time.)

Hello,

Your **quota** request for project '598733786702' has been approved and your **quota** has been adjusted accordingly.

Changed **Quota**:

+-----+-----+

| GLOBAL Attribute | GPUS\_ALL\_REGIONS |

+-----+-----+

| Changes | 0 -> 4 |

+-----+-----+

Figure 3: Global GPU quotas confirmation email

Hello,

Your **quota** request for project '598733786702' has been approved and your **quota** has been adjusted accordingly.

Changed **Quota**:

+-----+-----+

| Region: us-east1 | NVIDIA\_P100\_GPUS |

+-----+-----+

| Changes | 1 -> 4 |

+-----+-----+

Figure 4: “us-east1” Region “NVIDIA\_P100” GPU quotas confirmation email

## Step 2: Setup a Google Cloud VM Instance:

Create an instance with instance name → Choose “us-east1b” as region (Not all regions have GPUs support, the GPU quota has been introduced in step1, Choose the region you applied GPUs) → Choose the number of CPU in the machine type and customize by selecting number and kind of GPUs. → Choose “Ubuntu 16.04 LTS” and increase the disk size in Boot Disk option. → Allow HTTP and HTTPS traffic for your instance.

Figure 5: Create VM instance with GPU in Google Cloud Platform

Then a VM instance is created successfully and remember to stop it when you leave the system. The cost can be checked at Bill.

#### Credits

Promotion ID	Expires ^	Promotion value	Amount remaining
Free Trial	Apr 14, 2020	\$418.25	\$95.47

Figure 6: Bill of the project

**Step 3: Connect to instance by SSH Server and deployment Weed detection Project:**  
Once the instance is set up and started, click the SSH button to connect with SSH server.

<input type="checkbox"/>	<input checked="" type="checkbox"/> mask-rcnn	us-west1-b	10.138.0.2 (nic0)	35.247.82.57 ↗	SSH	⋮
--------------------------	---	------------	-------------------	----------------	-----	---

Install Anaconda3 → Install NVIDIA, Cuda9 and cuDNN9-v7.1 library. (an easy bash file to check and install Cuda as following code1) → Install pycocotools with PythonAPI → Install python packages as following code2 and Tensorflow-GPU.  
(\$pip list to check the version of all packages)

```
#!/bin/bash
echo "Checking for CUDA and installing."
# Check for CUDA and try to install.
if ! dpkg-query -W cuda; then
    # The 16.04 installer works with 16.10.
    wget http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64/cuda-repo-ubuntu1604_9.0.176-1_amd64.deb
    sudo dpkg -i cuda-repo-ubuntu1604_9.0.176-1_amd64.deb
    sudo apt-key adv --fetch-keys
    http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64/7fa2af80.pub
    sudo apt-get update
    sudo apt-get install cuda-9.0
fi
echo 'export CUDA_HOME=/usr/local/cuda-9.0' >> ~/.bashrc
echo 'export PATH=$PATH:$CUDA_HOME/bin' >> ~/.bashrc
echo 'export LD_LIBRARY_PATH=$CUDA_HOME/lib64' >> ~/.bashrc
source ~/.bashrc
```

code 1: Bash file to check and install Cuda

```
jiaqihan1108@mask-rcnn-22292229:~$ nvcc --version
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2017 NVIDIA Corporation
Built on Fri_Sep_1_21:08:03_CDT_2017
Cuda compilation tools, release 9.0, V9.0.176
jiaqihan1108@mask-rcnn-22292229:~$ nvidia-smi
Wed May 15 08:47:33 2019
```

NVIDIA-SMI 418.40.04 Driver Version: 418.40.04 CUDA Version: 10.1									
GPU	Name	Persistence-M	Bus-Id	Disp.A	Volatile	Uncorr.	ECC		
Fan	Temp	Perf	Pwr:Usage/Cap	Memory-Usage	GPU-Util	Compute	M.		
0	Tesla P100-PCIE...	Off	00000000:00:04.0	Off				0	
N/A	38C	P0	28W / 250W	100MiB / 16280MiB	0%	Default			
1	Tesla P100-PCIE...	Off	00000000:00:05.0	Off				0	
N/A	40C	P0	28W / 250W	0MiB / 16280MiB	0%	Default			
2	Tesla P100-PCIE...	Off	00000000:00:06.0	Off				0	
N/A	38C	P0	27W / 250W	0MiB / 16280MiB	0%	Default			
3	Tesla P100-PCIE...	Off	00000000:00:07.0	Off				0	
N/A	40C	P0	29W / 250W	0MiB / 16280MiB	0%	Default			

Processes:					GPU Memory
GPU	PID	Type	Process name		Usage
0	2156	G	/usr/lib/xorg/Xorg		100MiB

Figure 7: Nvidia details

```
pip install numpy
pip install scipy
pip install cython
pip install h5py
pip install Pillow
pip install scikit-image
pip install keras
pip install theano
pip install jupyter
pip install six
pip install opencv-python
pip install imgaug
```

Code 2: the install code for some important packages

#### Step 4: Connect the system with Jupyter Notebook:

The Jupyter Notebook of the instance is running at [http://mask\\_RCNN:8888/](http://mask_RCNN:8888/), because the External IP is 35.247.82.57 (it always changes up to the instance) so the link is <http://35.247.82.57:8888/>. “Ctrl + c” to shut down the server. The password of Jupyter Notebook is “hanjiaqi”



Figure 8: External IP of the instance

```
jiaqihan1108@mask-rcnn: ~ - Google Chrome
https://ssh.cloud.google.com/projects/my-project-2019-237613/zones/us-west1-b/instances/mask-rcnn?authuser...
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage

Get cloud support with Ubuntu Advantage Cloud Guest:
http://www.ubuntu.com/business/services/cloud


30 packages can be updated.
0 updates are security updates.

New release '18.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Fri May 10 20:00:55 2019 from 74.125.41.103
jiaqihan1108@mask-rcnn:~$ jupyter notebook
[I 20:07:31.278 NotebookApp] Writing notebook server cookie secret to /run/user/1001/jupyter/notebook_cookie_secret
[W 20:07:31.467 NotebookApp] WARNING: The notebook server is listening on all IP addresses and not using encryption. This is not recommended.
[I 20:07:31.499 NotebookApp] JupyterLab beta preview extension loaded from /home/jiaqihan1108/anaconda3/lib/python3.6/site-packages/jupyterlab
[I 20:07:31.499 NotebookApp] JupyterLab application directory is /home/jiaqihan1108/anaconda3/share/jupyterlab
[I 20:07:31.504 NotebookApp] Serving notebooks from local directory: /home/jiaqihan1108
[I 20:07:31.504 NotebookApp] 0 active kernels
[I 20:07:31.504 NotebookApp] The Jupyter Notebook is running at:
[I 20:07:31.504 NotebookApp] http://mask-rcnn:8888/
[I 20:07:31.504 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).

302 GET / (125.253.56.175) 0.73ms
[W 20:09:28.471 NotebookApp] Clearing invalid/expired login cookie username-35-247-23-99-8888
[W 20:09:28.472 NotebookApp] Clearing invalid/expired login cookie username-35-247-23-99-8888
[I 20:09:28.472 NotebookApp] 302 GET /tree? (125.253.56.175) 2.12ms
[I 20:09:36.752 NotebookApp] 302 POST /login?next=%2Ftree%3F (125.253.56.175) 1.22ms
^C[I 20:11:44.038 NotebookApp] interrupted
Serving notebooks from local directory: /home/jiaqihan1108
0 active kernels
The Jupyter Notebook is running at:
http://mask-rcnn:8888/
Shutdown this notebook server (y/[n])? y
[C 20:11:45.728 NotebookApp] Shutdown confirmed
[I 20:11:45.729 NotebookApp] Shutting down 0 kernels
jiaqihan1108@mask-rcnn:~$
```

Figure 9: Running Jupyter Notebook of the instance and shut down

 jupyter

Password:

Log in

Figure 10: The Password of Jupyter Notebook is “hanjiaqi”

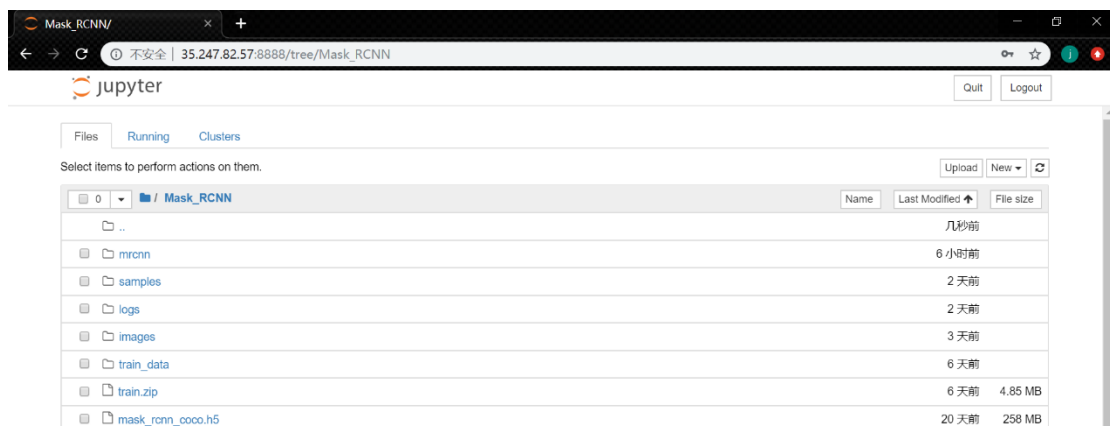


Figure 11: The Jupyter Notebook of the instance and the project interface

## Step 5: Create Static external IP for the Instance:

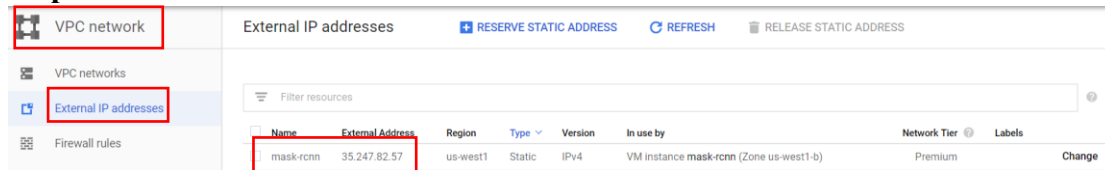


Figure 12: 'mask-rcnn' Weed Detection VM instances of Google Cloud platform with a Static External IP. Then this instance has a Static External IP (35.247.82.57), the final link is always be [http:// 35.247.82.57:8888/](http://35.247.82.57:8888/).

## III. Mask\_RCNN System Directory Tree.

/Mask\_RCNN folder

- \_\_ mrcnn folder: contained the main code for Mask\_RCNN algorithm and .h5 model for further retrain.
- \_\_ samples folder: contained Weed\_retrain.py and Weed\_detection.py
- \_\_ image folder: stored the farmer images
- \_\_ logs folder: saved the retrained model and logs
- \_\_ train\_data folder: stored the training data for the model retrain.
- \_\_ ...

## IV. System Maintenance Problems and Limitations

### 1. Deployment Problems

1> Compile errors

This kind of errors happened because of the versions of system's Python packages do not match with the versions of anaconda Python. And when compile the Makefile, it will use the system's Python as default.

The solution is to replace the system's Python with Anaconda Python. For example, use Cython-0.28.1 (/home/jiaqihan1108/anaconda3/bin/python setup.py build\_ext install)

```
jiaqihan1108@mask-rcnn-22292229:~/Mask_RCNN/coco/PythonAPI$ which python
/home/jiaqihan1108/anaconda3/bin/python
```

Figure 13: Change the path of python for compiling

2> Make sure all packages' version matched.

### 2. Quota Allocation (Resource Limitation)

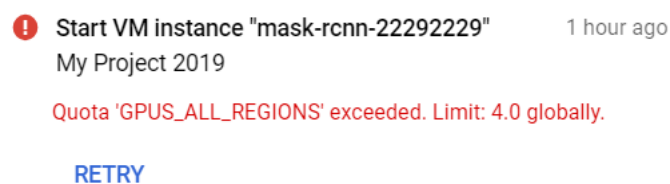


Figure 14: GPU limit of all regions

This error happened because of the unreasonable resource allocation. There is a limitation of GPUs of all regions. Sometimes we have two or more instances running together, the total number of these instances' GPU cannot over 4 globally.

The recommend solution is to deploy and balance the workload across multiple zones or regions to reduce the likelihood of an outage. So never allocate all GPU resources in one instance since sometimes we need to run more than one instance.

### 3. VM Instances Dead and Storage (Platform Limitation)

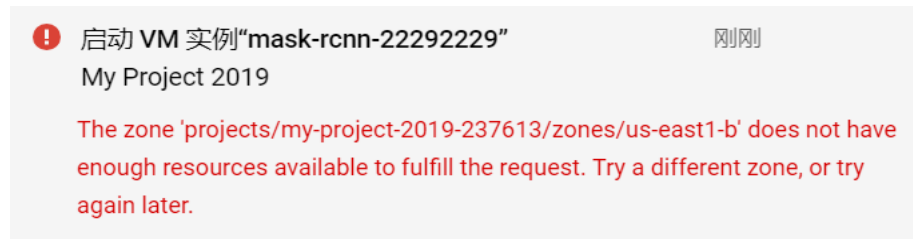


Figure 15: Instance dead

The reason of this kind of error is because the Google Cloud Platform is not as stable as we want.

The solution is to use snapshot to recreate instances. If something wrong happened with the instance, we can recreate the instance through its snapshot. A snapshot should be created once the instance is upgraded to a new version or implemented new functions.

Step details:

#### Step 1: Create a snapshot of the instance mask-rcnn

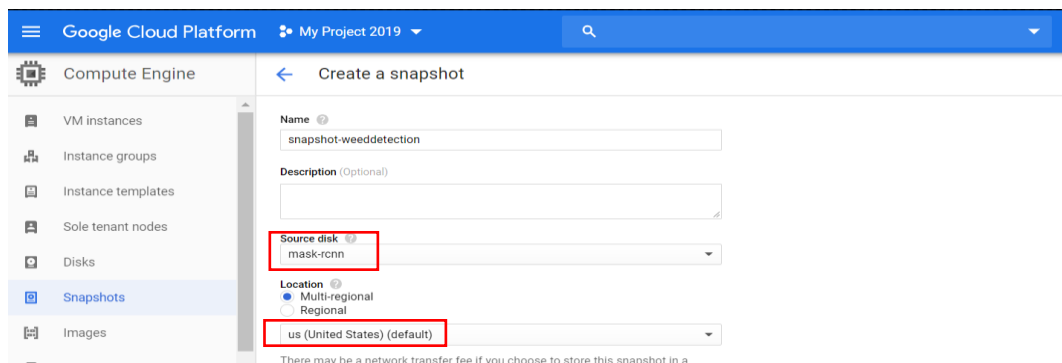


Figure 16: Create a Snapshot of mask-rcnn instance

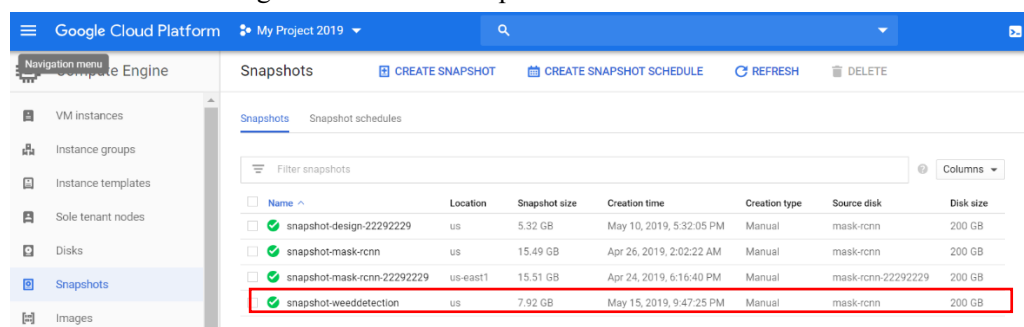


Figure 17: Snapshots of all milestones during the development

## Step 2: Create a new instance based on the snapshot

To create a VM instance, select one of the options:

- New VM instance**  
Create a single VM instance from scratch
- New VM instance from template**  
Create a single VM instance from an existing template
- Marketplace**  
Deploy a ready-to-go solution onto a VM instance

**Name**  
detectiontest

**Region**  
us-east1 (South Carolina)

**Zone**  
us-east1-b

**Machine type**  
Customize to select cores, memory and GPUs.

**Cores**  
4 vCPU | 1 - 96

**Memory**  
15 GB | 3.6 - 26

**CPU platform**  
Automatic

**GPUs**  
The number of GPU dies is linked to the number of CPU cores and memory selected for this instance. For the current configuration, you can select no fewer than 1 GPU

**Number of GPUs**  
1

**GPU type**  
NVIDIA Tesla P100

**Display device**  
Turn on a display device if you want to use screen capturing and recording tools.  
☐ Turn on display device

**Container**  
☐ Deploy a container image to this VM instance. [Learn more](#)

**Boot disk**  
New 200 GB standard persistent disk  
Snapshot: snapshot-mask-rcnn

**Identity and API access**  
Service account: Compute Engine default service account

You have \$94.880075 free trial credits remaining  
\$851.15 monthly estimate  
That's about \$1.166 hourly  
Pay for what you use: No upfront costs and per second billing  
[Details](#)

Figure 18: Create a instance based on the snapshot

Remember to allow HTTP and HTTPS traffic for your instance

<input type="checkbox"/> Name ^	Zone	Recommendation	In use by	Internal IP	External IP	Connect
<input type="checkbox"/> <input checked="" type="checkbox"/> detectiontest	us-east1-b			10.142.0.13 (nic0)	34.73.82.207 <a href="#">↗</a>	SSH <a href="#">▼</a> <a href="#">⋮</a>
<input type="checkbox"/> <input checked="" type="checkbox"/> fcn-22292229	us-east1-b			10.142.0.8 (nic0)	None	SSH <a href="#">▼</a> <a href="#">⋮</a>
<input type="checkbox"/> <input checked="" type="checkbox"/> mask-rcnn	us-west1-b			10.138.0.2 (nic0)	35.247.82.57 <a href="#">↗</a>	SSH <a href="#">▼</a> <a href="#">⋮</a>
<input type="checkbox"/> <input checked="" type="checkbox"/> mask-rcnn-22292229	us-east1-b			10.142.0.9 (nic0)	34.74.74.248 <a href="#">↗</a>	SSH <a href="#">▼</a> <a href="#">⋮</a>

Figure 19: A new instance “detectiontest” same with instance “mask-rcnn”

## 4. Model Retrain (Platform Limitation)

1> The python file cannot run.

That because the mode of the python file should be changed.

Use “chmod a+x Weed\_retrain.py” and directly run the python file “./Weed\_retrain.py”.

2> OS Error: can't allocate memory

That because the system cannot support Multi-threading perfectly.

The solution is to make sure the system running with single multithread and let “worker = 1” in model.py. (in the mrcnn folder) As below:



```

2364         self.keras_model.fit_generator(
2365             train_generator,
2366             initial_epoch=self.epoch,
2367             epochs=epochs,
2368             steps_per_epoch=self.config.STEPS_PER_EPOCH,
2369             callbacks=callbacks,
2370             validation_data=val_generator,
2371             validation_steps=self.config.VALIDATION_STEPS,
2372             max_queue_size=100,
2373             #workers=workers,
2374             workers=1,|
2375             use_multiprocessing=False,
2376             # use_multiprocessing=True,
2377         )
2378         self.epoch = max(self.epoch, epochs)
2379

```

Code 3: Use single threading

## 5. Detection result (Development Limitation)

The system only can automatically show the detection image on the Jupyter Notebook page but cannot be saved in local folder automatically. The only way to save the results is right-click and save the result image one by one.

## V. Further Development – FCN System

### 1. Deployment Requirement

Python3.5 (Anaconda3) + tensorflow-gpu1.4.0 + Cuda 8.0 + cudnn-8.0-2.windows10-x64-v6.0

### 2. Training dataset

224\*224 images and their 8bit mask in 2 separate folders (Image and Mask).

### 3. FCN System Directory Tree

/FCN folder

- \_\_ Model folder: the Vgg19 model for further retrain
- \_\_ samples folder: contained Weed\_retrain.py and Weed\_detection.py
- \_\_ images folder: stored the farmer images
- \_\_ logs folder: saved the retrained model and logs
- \_\_ Data\_zoo folder: stored the training data for the model retrain.
- \_\_ FCN.py: main function to retrain the model and achieve detection
- \_\_ ...

### 4. System Maintenance Problems and Limitations

1. training loss is not good (Training Dataset Limitation)

The FCN algorithm needs a large training dataset for training own model, but we don't have too much weed images with its masks.