



Litter Detection Algorithm: Programmer Manual

Table of Contents

1. Introduction

1.1 Tools	2
2. Creating an AWS ec2 Server Instance	2
2.1 Spot instance server	2
2.2 On Demand Server	7
3. Initial ec2 Server Installation & Configuration	11
3.1 Installing TensorFlow (GPU) on Ubuntu Server.	11
3.2 Installing TensorFlow Object Detection API	14
4. Training a Custom Dataset	Error! Bookmark not defined.
5. Accessing Jupyter Notebook	16
5.1 Windows	16
5.2 Mac	18
5.3 Initial configuration	18
5.4 Using Jupyter Notebook	19
6. Future Use	21
6.1 How to gather images for Data Set	21
6.2 How to label images for Data Set	23
6.3 How to convert the dataset into TFrecord.	25

1. Introduction

FixIT worked with Keep America Beautiful to create a Litter Detection Algorithm that will identify litter, visually, when given a particular image. This manual was written as an instructional guide for programmers and/or users with prior experience in computer science. We will outline our step-by-step process on how to access our server, use our algorithm, as well as how to manipulate the algorithm for future use.

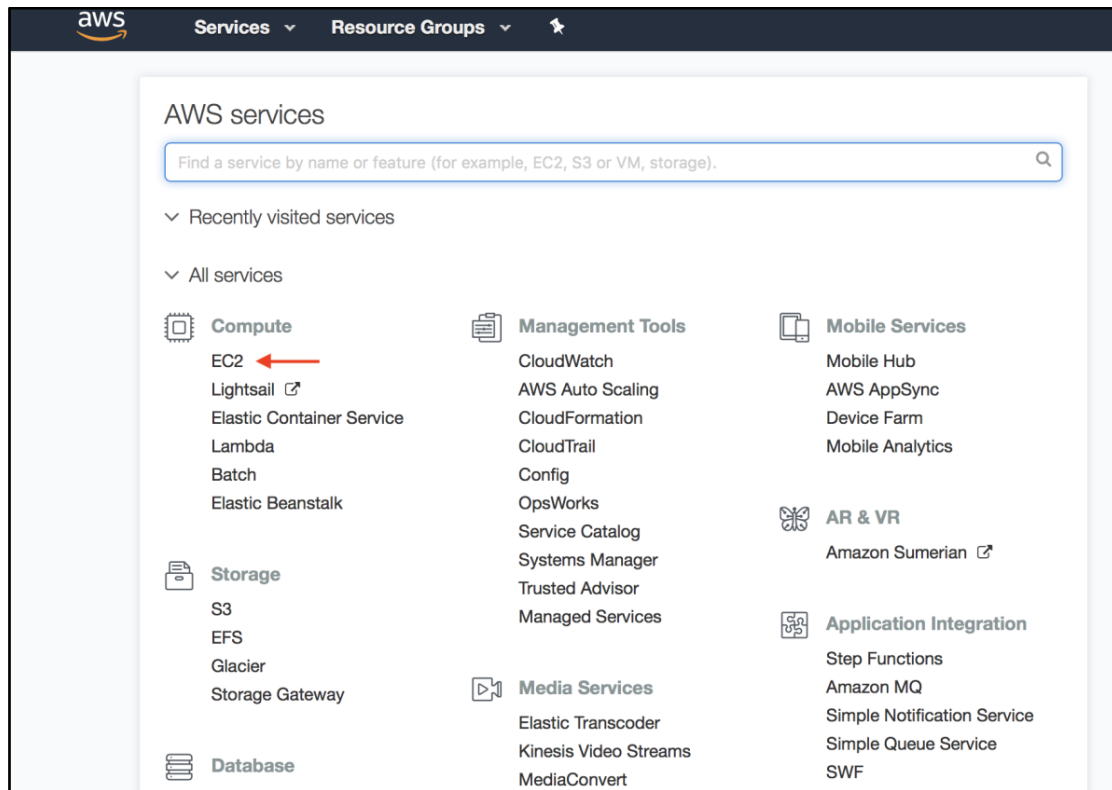
1.1 Tools

- **TensorFlow**
- **Amazon Web Services**
 - P2 Large Ubuntu Server
- **Jupyter Notebook**
- **Windows:**
 - PuTTY
- **Mac:**
 - Terminal

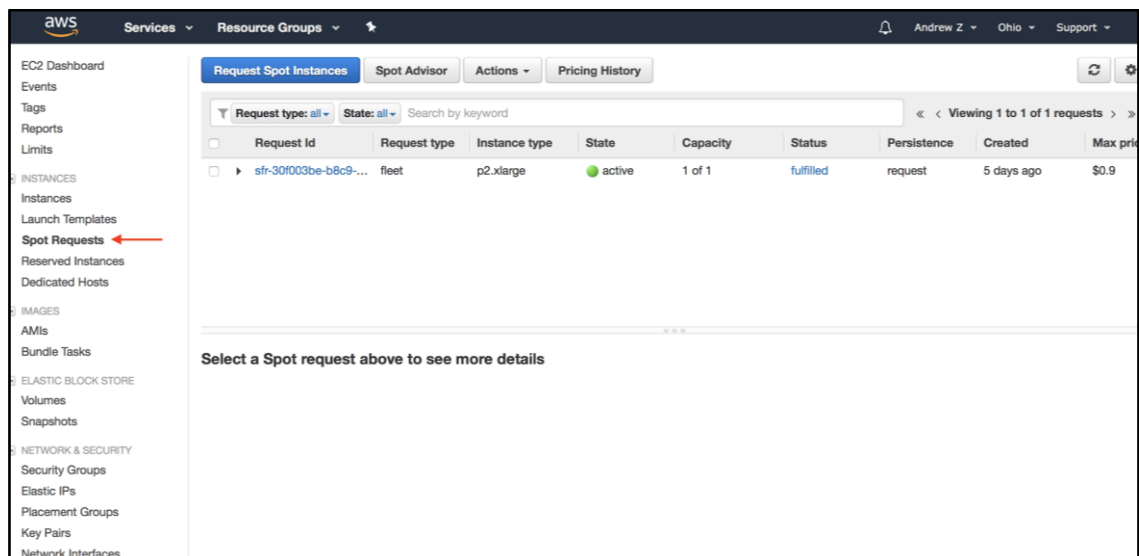
2. Creating an AWS ec2 Server Instance

2.1 Spot instance server

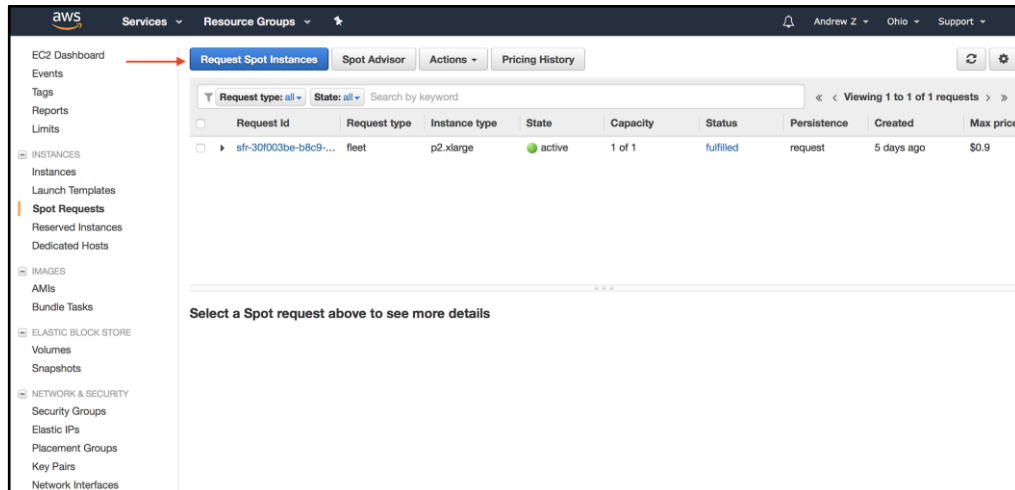
1. Create aws account:
<https://portal.aws.amazon.com/billing/signup?nc2=h ct&redirect url=https%3A%2F%2Faws.amazon.com%2Fregistration-confirmation#/start>
2. Go to aws service dashboard.
<https://us-east-2.console.aws.amazon.com/console/home?region=us-east-2>
3. Look under Computer category in all services and select EC2



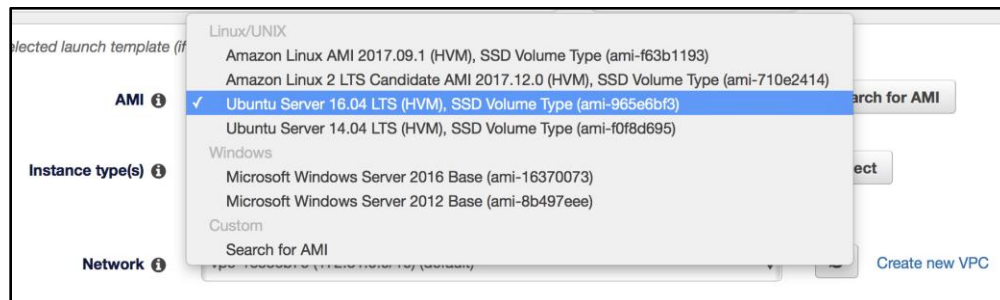
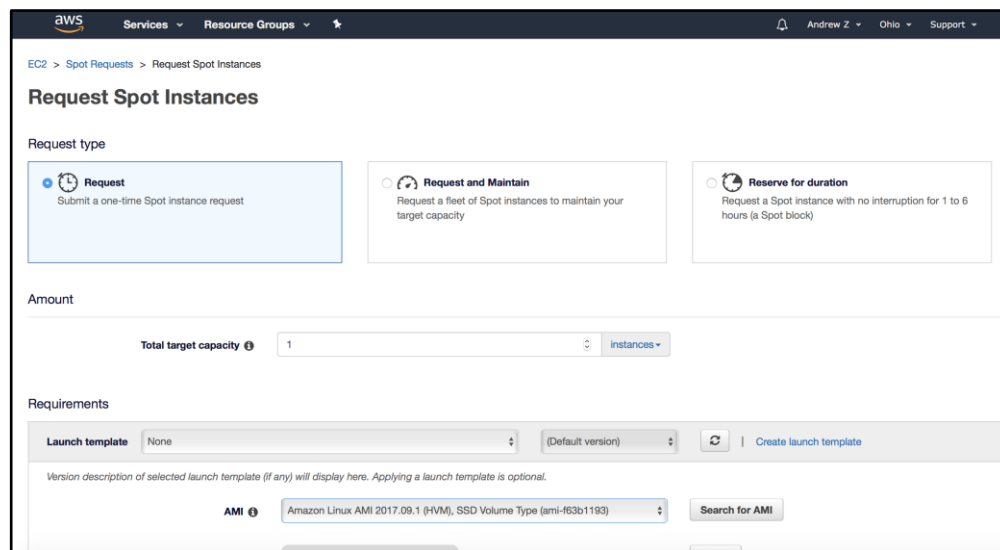
- Once you get to the EC2 dashboard page, look at the left hand navigation bar and select Spot Requests under instances.



- Click on Request spot instances.




6. You'll end up at the page for editing the settings of your spot instances. For most you will just use the default settings



7. First go to AMI option, scroll through the options listed or click on search for AMI. For ours we used an Ubuntu Server 16.04.
8. Next go to Instance types, and hit select.

Instance type(s) ⓘ

p2.xlarge (4 vCPU, 61 GiB, EBS only) ✕



Select

Select multiple instance types to find the lowest priced instances available

9. It'll bring you to a list of instance types.

Select instance types

☐ Supports dedicated tenancy

Pricing History

<input type="checkbox"/> m5.2xlarge	8	32	EBS only	Up to 10 Gigabit	\$0.0744	81%
<input type="checkbox"/> m5.4xlarge	16	64	EBS only	Up to 10 Gigabit	\$0.1498	80%
<input type="checkbox"/> m5.12xlarge	48	192	EBS only	10 Gigabit	\$0.3917	83%
<input type="checkbox"/> m5.24xlarge	96	384	EBS only	25 Gigabit	\$0.7834	83%
<input checked="" type="checkbox"/> p2.xlarge	4	61	EBS only	High	\$0.2993	67%
<input type="checkbox"/> p2.8xlarge	32	488	EBS only	10 Gigabit	\$2.16	70%
<input type="checkbox"/> p2.16xlarge	64	732	EBS only	25 Gigabit	\$4.32	70%
<input type="checkbox"/> p3.2xlarge	8	61	EBS only	High	\$1.1009	64%
<input type="checkbox"/> p3.8xlarge	32	244	EBS only	High	\$4.488	63%
<input type="checkbox"/> p3.16xlarge	64	488	EBS only	High	\$9.5337	61%
<input type="checkbox"/> r3.large	2	15	1 x 32 SSD	Moderate	\$0.0177	89%
<input type="checkbox"/> r3.xlarge	4	30.5	1 x 80 SSD	Moderate	\$0.0354	89%
<input type="checkbox"/> r3.2xlarge	8	61	1 x 160 SSD	High	\$0.0709	89%

Cancel

Select

10. The one we use is a p2xlarge. The reason we use a fairly expensive one is because it is recommended that you use a server with at least one gpu. GPU's improve the efficiency and time between each step when training significantly. P2xlarge to us is considered the minimum required to run the model training.

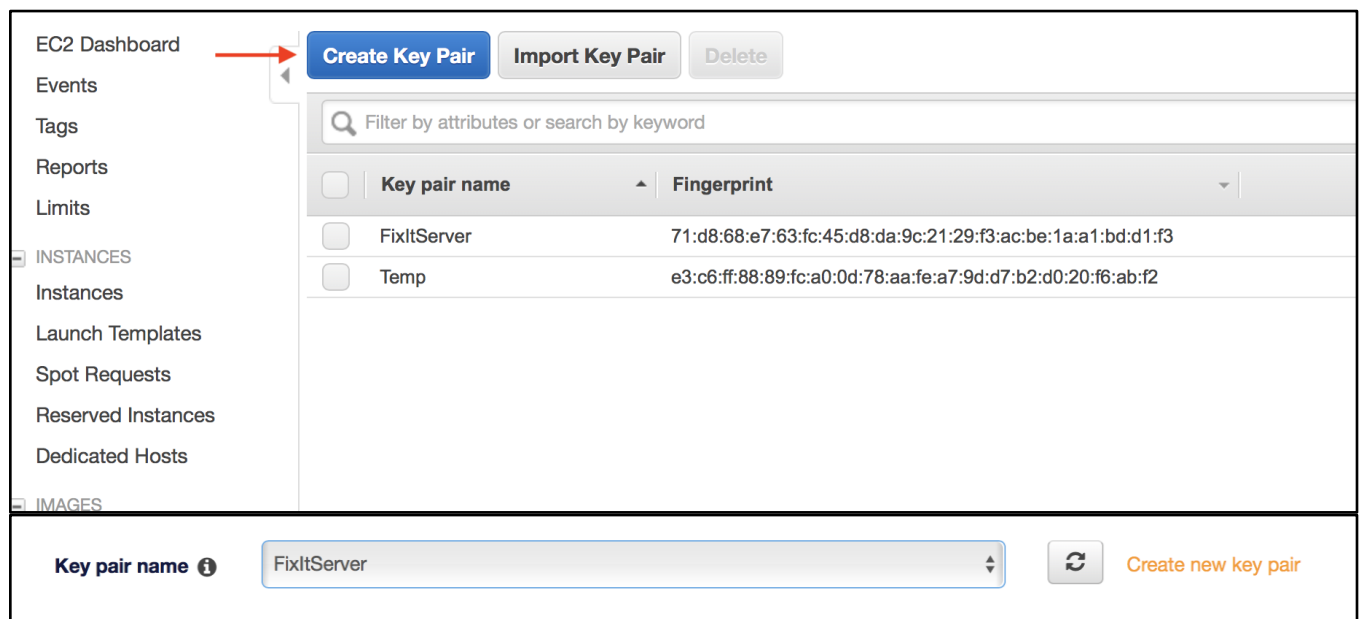
11. Next go to the EBS volume section and where it says Size(GiB) increase the amount from 8 to 75. It is important to have at least 75 gigs because loading the dataset and supporting libraries take up a large amount of space, not to mention training file checkpoints.

EBS volumes ⓘ	Device ⓘ	Snapshot	Size (GiB)	Volume Type	IOPS	Delete ⓘ	Encrypt
	Root: /dev/sda1	snap-089f597d798b3e945	75	General Purpose (SSD)		<input checked="" type="checkbox"/>	<input type="checkbox"/>
No additional EBS volumes configured							
+ Add new volume							

12. For security groups just choose default option.

EBS-optimized ⓘ	<input type="checkbox"/> Launch EBS-optimized instances
Instance store ⓘ	<input type="checkbox"/> Attach at launch
Monitoring ⓘ	<input type="checkbox"/> Enable CloudWatch detailed monitoring
Tenancy ⓘ	Default - run a shared hardware instance
Security groups ⓘ	<div><input checked="" type="checkbox"/> default <input type="checkbox"/> launch-wizard-1 <input type="checkbox"/> launch-wizard-2</div> <div>Create new security group</div>
Auto-assign IPv4 Public IP ⓘ	Use subnet setting
Key pair name ⓘ	FixItServer <div>Create new key pair</div>
IAM instance profile ⓘ	(optional) <div>Create new IAM profile</div>
User data ⓘ	<div><input checked="" type="radio"/> As text <input type="radio"/> As file <input type="checkbox"/> Input is already base64 encoded</div> <div></div>

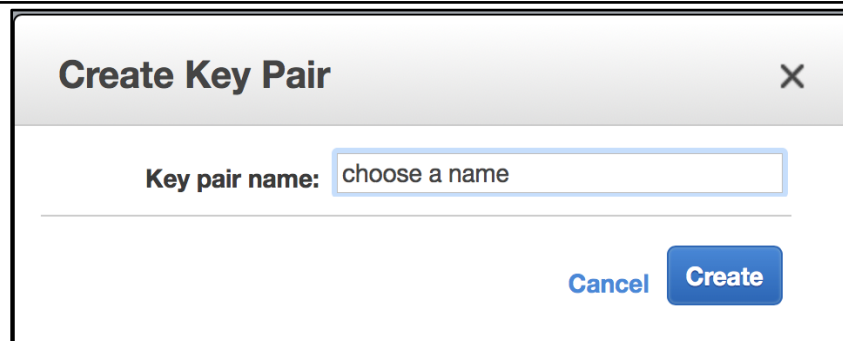
13. In order to access the server you have to give it a key pair. If you don't have one yet click on Create new key pair and follow the instructions.



The screenshot shows the AWS Management Console interface. On the left sidebar, the 'EC2 Dashboard' is selected, and a red arrow points to the 'Create Key Pair' button in the top navigation bar. Below the navigation bar, there is a table of existing key pairs:

Key pair name	Fingerprint
FixItServer	71:d8:68:e7:63:fc:45:d8:da:9c:21:29:f3:ac:be:1a:a1:bd:d1:f3
Temp	e3:c6:ff:88:89:fc:a0:0d:78:aa:fe:a7:9d:d7:b2:d0:20:f6:ab:f2

Below the table, there is a form to create a new key pair. The 'Key pair name' field is set to 'FixItServer'. To the right of the field is a 'Create new key pair' button.

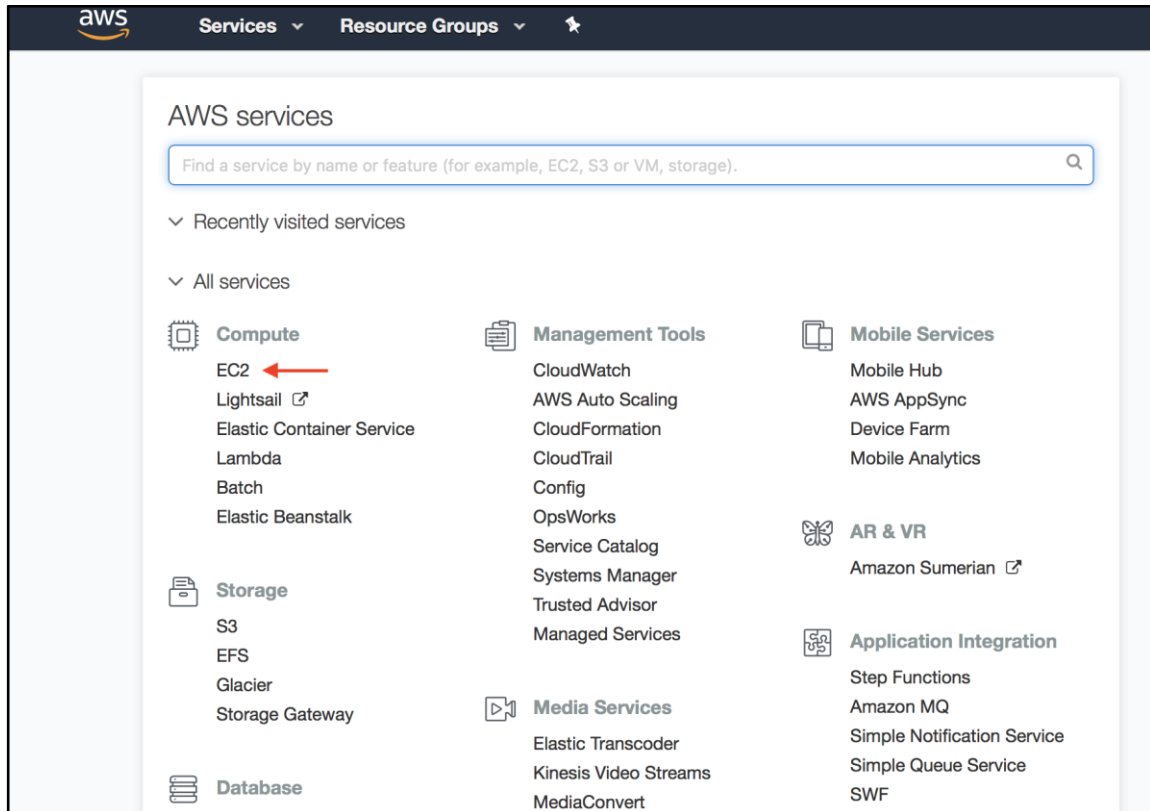


The 'Create Key Pair' dialog box is shown. It has a title bar with a close button (X). Inside the dialog, there is a label 'Key pair name:' followed by a text input field containing the placeholder text 'choose a name'. At the bottom right of the dialog, there are two buttons: 'Cancel' and 'Create'.

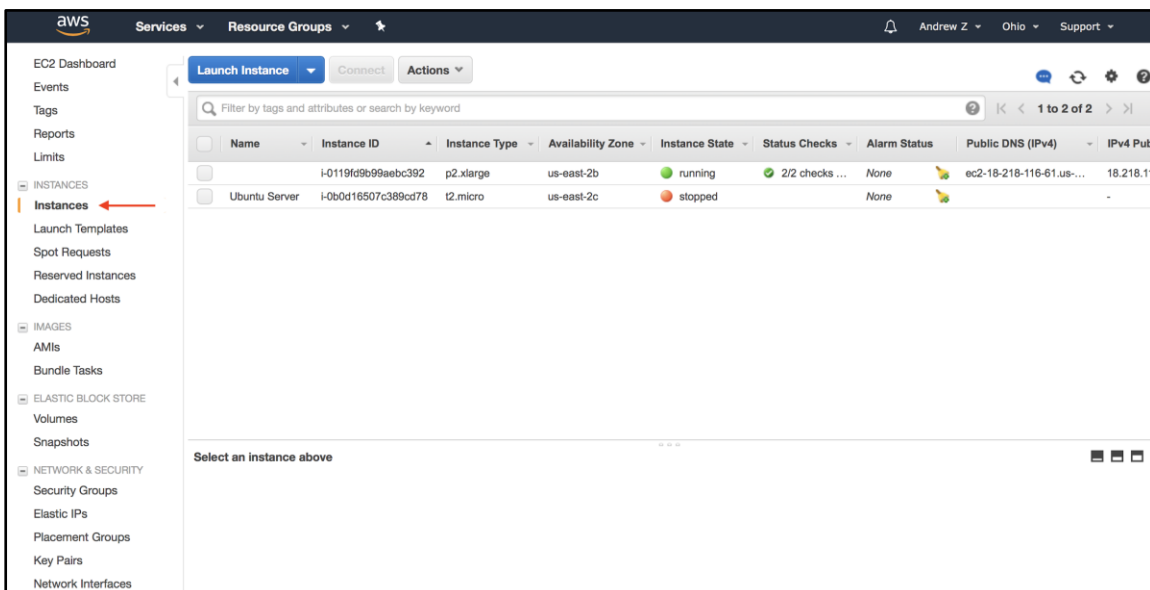
14. Once you make it, it'll automatically save the key to your local desktop. Some different ssh clients like putty need the key in different formats, but there are many instructions online on how to convert it to the specified formats.
15. Once all that is complete go to the very bottom and hit launch in the right corner. It will take a couple of minutes for the request to be fulfilled but after it is you will see the server show up under your EC2 instances.

2.2 On Demand Server

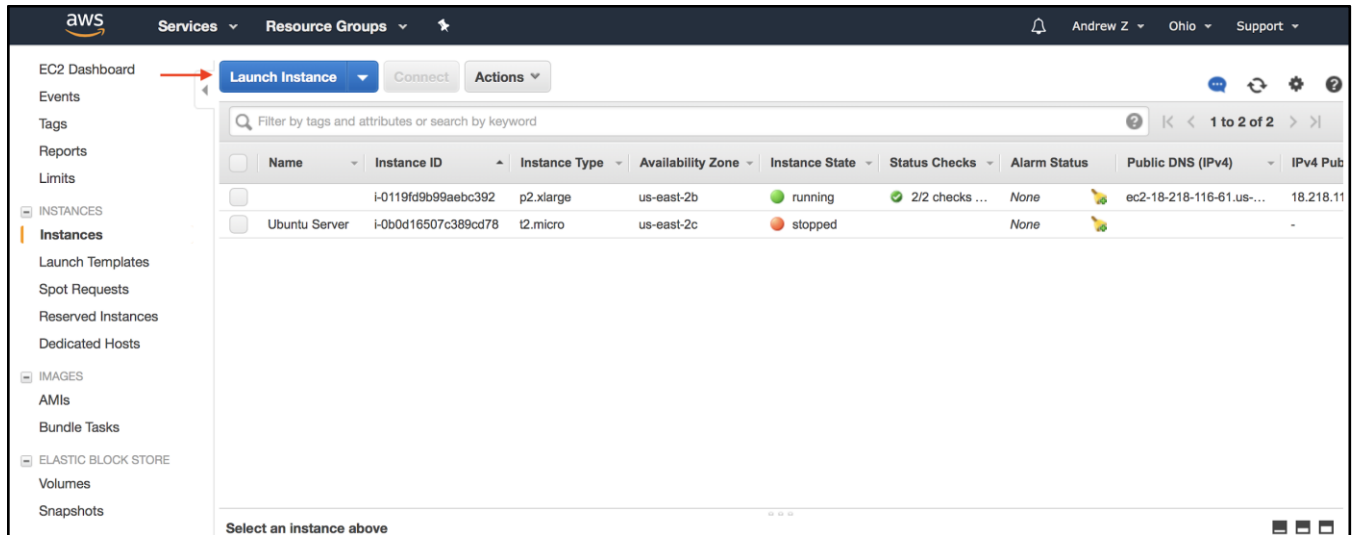
1. Create aws account:
https://portal.aws.amazon.com/billing/signup?nc2=h_ct&redirect_url=https%3A%2F%2Faws.amazon.com%2Fregistration-confirmation#/start
2. Go to aws service dashboard.
<https://us-east-2.console.aws.amazon.com/console/home?region=us-east-2>
3. Look under Computer category in all services and select EC2



- Once you get to the EC2 dashboard page, look at the left hand navigation bar and select instances.



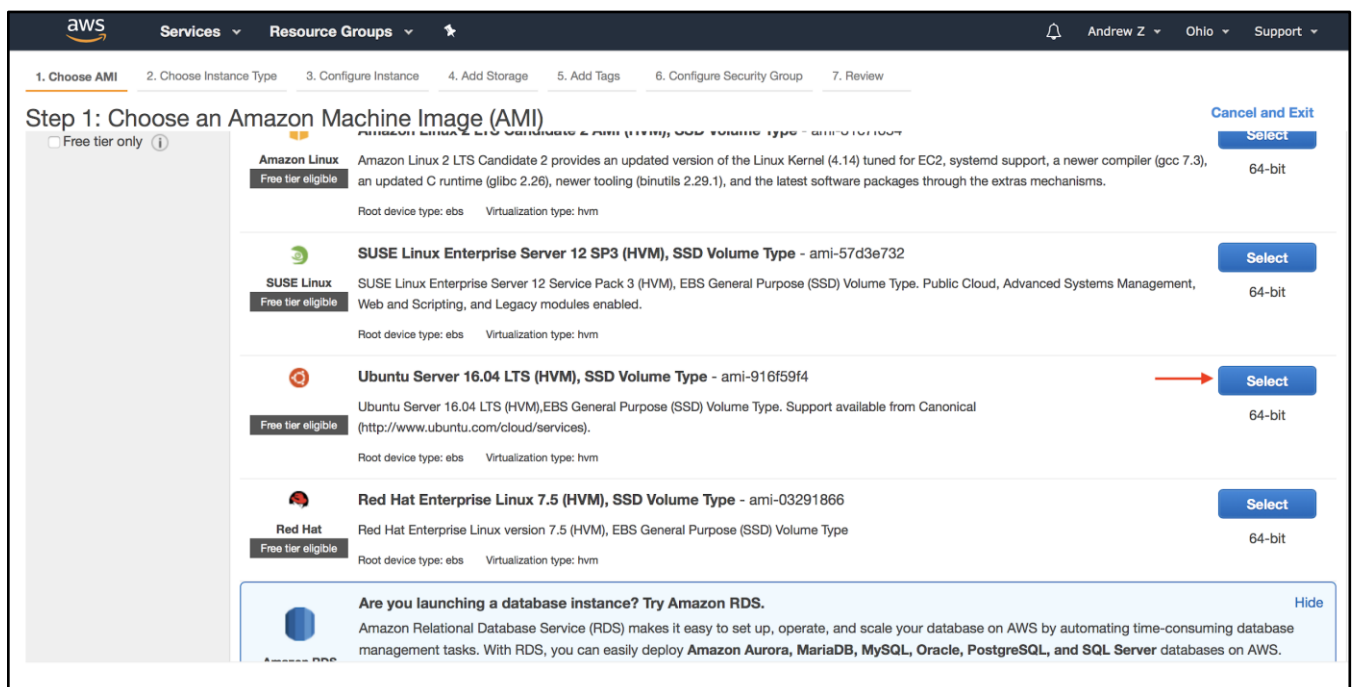
- Click on launch instance.



The screenshot shows the AWS Management Console interface. On the left sidebar, the 'EC2 Dashboard' is selected. In the top navigation bar, the 'Launch Instance' button is highlighted with a red arrow. The main content area displays a table of existing EC2 instances. The table has columns for Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, Public DNS (IPv4), and IPv4 Public IP. Two instances are listed: 'Ubuntu Server' (ID: i-0b0d16507c389cd78, Type: t2.micro, State: stopped) and another instance (ID: i-0119fd9b99aebc392, Type: p2.xlarge, State: running).

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public IP
Ubuntu Server	i-0b0d16507c389cd78	t2.micro	us-east-2c	stopped	2/2 checks ...	None	ec2-18-218-116-61.us-...	18.218.11...
	i-0119fd9b99aebc392	p2.xlarge	us-east-2b	running	2/2 checks ...	None	ec2-18-218-116-61.us-...	18.218.11...

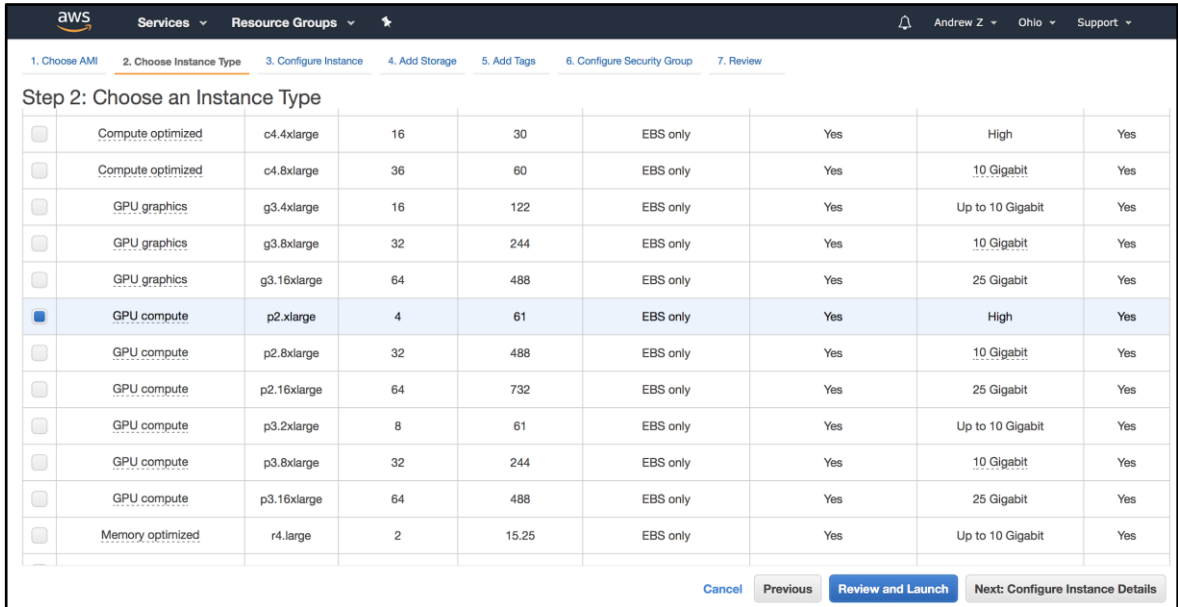
6. At this point select an AMI from the list, in our case Ubuntu 16.04



The screenshot shows the 'Step 1: Choose an Amazon Machine Image (AMI)' screen in the AWS Management Console. The left sidebar shows the 'Choose AMI' step is active. The main content area lists several AMIs. The 'Ubuntu Server 16.04 LTS (HVM), SSD Volume Type' AMI is selected, indicated by a red arrow pointing to its 'Select' button. The AMI list includes Amazon Linux, SUSE Linux Enterprise Server, Ubuntu Server, and Red Hat Enterprise Linux. Each entry shows the AMI name, description, root device type, and virtualization type.

AMI Name	Description	Root Device Type	Virtualization Type	Architecture
Amazon Linux 2 LTS Candidate 2	Amazon Linux 2 LTS Candidate 2 provides an updated version of the Linux Kernel (4.14) tuned for EC2, systemd support, a newer compiler (gcc 7.3), an updated C runtime (glibc 2.26), newer tooling (binutils 2.29.1), and the latest software packages through the extras mechanisms.	ebs	hvm	64-bit
SUSE Linux Enterprise Server 12 SP3 (HVM), SSD Volume Type	SUSE Linux Enterprise Server 12 Service Pack 3 (HVM), EBS General Purpose (SSD) Volume Type. Public Cloud, Advanced Systems Management, Web and Scripting, and Legacy modules enabled.	ebs	hvm	64-bit
Ubuntu Server 16.04 LTS (HVM), SSD Volume Type	Ubuntu Server 16.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).	ebs	hvm	64-bit
Red Hat Enterprise Linux 7.5 (HVM), SSD Volume Type	Red Hat Enterprise Linux version 7.5 (HVM), EBS General Purpose (SSD) Volume Type	ebs	hvm	64-bit

7. Select instance type. Preferably one with a GPU and in our case a p2.xlarge.

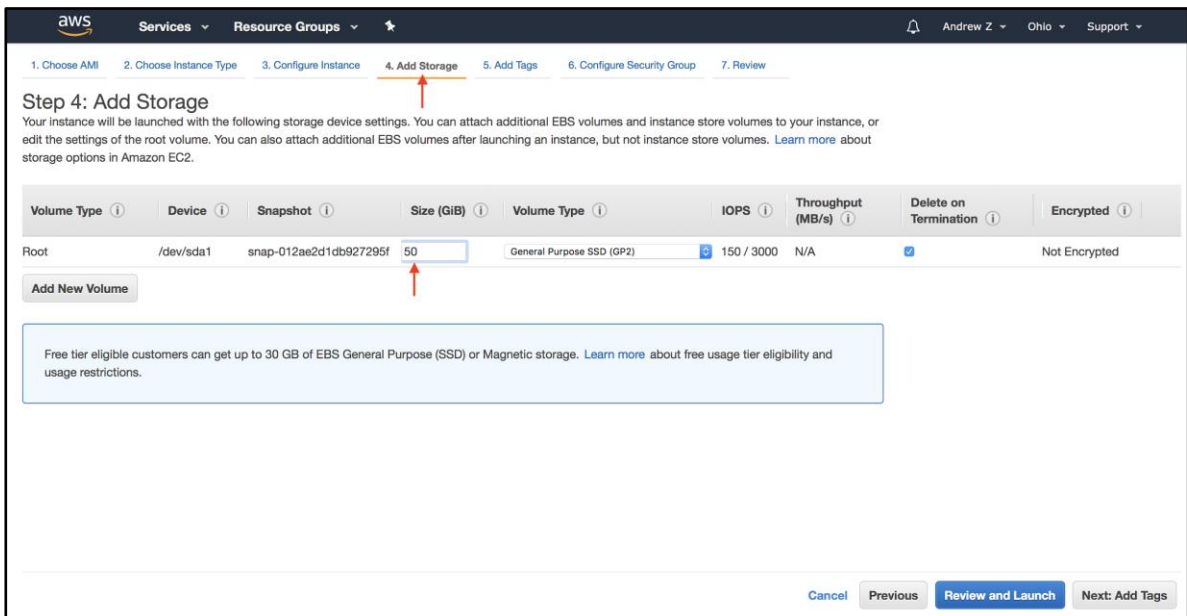


Step 2: Choose an Instance Type

	Instance Type	VCpus	Memory (GiB)	Storage	Network	Accelerated Networking	High Memory
<input type="checkbox"/>	Compute optimized	c4.xlarge	16	30	EBS only	Yes	High
<input type="checkbox"/>	Compute optimized	c4.8xlarge	36	60	EBS only	Yes	10 Gigabit
<input type="checkbox"/>	GPU graphics	g3.4xlarge	16	122	EBS only	Yes	Up to 10 Gigabit
<input type="checkbox"/>	GPU graphics	g3.8xlarge	32	244	EBS only	Yes	10 Gigabit
<input type="checkbox"/>	GPU graphics	g3.16xlarge	64	488	EBS only	Yes	25 Gigabit
<input checked="" type="checkbox"/>	GPU compute	p2.xlarge	4	61	EBS only	Yes	High
<input type="checkbox"/>	GPU compute	p2.8xlarge	32	488	EBS only	Yes	10 Gigabit
<input type="checkbox"/>	GPU compute	p2.16xlarge	64	732	EBS only	Yes	25 Gigabit
<input type="checkbox"/>	GPU compute	p3.2xlarge	8	61	EBS only	Yes	Up to 10 Gigabit
<input type="checkbox"/>	GPU compute	p3.8xlarge	32	244	EBS only	Yes	10 Gigabit
<input type="checkbox"/>	GPU compute	p3.16xlarge	64	488	EBS only	Yes	25 Gigabit
<input type="checkbox"/>	Memory optimized	r4.large	2	15.25	EBS only	Yes	Up to 10 Gigabit

Cancel Previous **Review and Launch** Next: Configure Instance Details

8. Next click on add storage and change EBS size to above 50 because server needs to hold a large data set and many files.



Step 4: Add Storage

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. [Learn more](#) about storage options in Amazon EC2.

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Throughput (MB/s)	Delete on Termination	Encrypted
Root	/dev/sda1	snap-012ae2d1db927295f	50	General Purpose SSD (GP2)	150 / 3000	N/A	<input checked="" type="checkbox"/>	Not Encrypted

Add New Volume

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. [Learn more](#) about free usage tier eligibility and usage restrictions.

Cancel Previous **Review and Launch** Next: Add Tags

9. Everything else should be left to default.
10. Hit launch and review and then again on launch in the bottom right corner.

3. Initial ec2 Server Installation & Configuration

These are the following steps to configure the server in order to run the litter recognition algorithm.

3.1 Installing TensorFlow (GPU) on Ubuntu Server.

Reference: https://www.tensorflow.org/install/install_linux

0) Download installation files

Download and extract the required installation files from Google Drive.

<https://drive.google.com/file/d/1Wcko1t2QYRX7DW-Tu2lY4l0-CQgAuqA-/view?usp=sharing>

Then, using an FTP client like a [FileZilla](#), move the files into the home directory of the server.

1) Installing GCC

The `gcc` compiler is required for development using the CUDA Toolkit.

```
sudo apt-get update && \  
sudo apt-get install build-essential software-properties-common -y && \  
sudo add-apt-repository ppa:ubuntu-toolchain-r/test -y && \  
sudo apt-get update && \  
sudo apt-get install gcc-snapshot -y && \  
sudo apt-get update && \  
sudo apt-get install gcc-6 g++-6 -y && \  
sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-6 60 --  
slave /usr/bin/g++ g++ /usr/bin/g++-6 && \  
sudo apt-get install gcc-4.8 g++-4.8 -y && \  
sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-4.8 60 --  
slave /usr/bin/g++ g++ /usr/bin/g++-4.8;  
gcc -v
```

2) Installing CUDA Toolkit 9.0 [[Reference](#)]

This toolkit provides the development environment for running high performance GPU-accelerated applications. It includes the NVIDIA drivers needed to interface with the GPU.

```
sudo apt-get install linux-headers-$(uname -r)

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_64/7fa2af8
0.pub

sudo apt-get update
sudo apt-get install cuda-9-0

export PATH=/usr/local/cuda-9.0/bin${PATH:+:${PATH}}
```

2.1) Validating CUDA Toolkit 9.0 installation

First reboot the server

```
sudo reboot
```

Then, enter the following command. This will check the status of the GPU.

```
nvidia-smi
```

If installed correctly, you will see the following screen below:

```
ubuntu@ip-172-31-19-154:~$ nvidia-smi
Tue May 1 21:02:14 2018
+-----+
| NVIDIA-SMI 390.30                 Driver Version: 390.30          |
+-----+-----+
| 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 K80        Off      | 00000000:00:1E:0 Off |                    0 |
| N/A   52C    P0      59W / 149W | 11216MiB / 11441MiB |      0%      Default |
+-----+-----+
```

If you don't see this screen, remove the cuda-9.0 toolkit using the following command, then reinstall it.

```
sudo apt-get purge $(dpkg -l | awk '$2~/nvidia/ {print $2}')
```

3) Installing cuDNN SDK v7 [[Reference](#)]

This is a GPU-accelerated library of primitives for deep neural networks.

```
sudo dpkg -i libcudnn7_7.0.4.31-1+cuda9.0_amd64.deb
sudo dpkg -i libcudnn7-dev_7.0.4.31-1+cuda9.0_amd64.deb
```

4) Installing cudna command-line-tools

```
sudo apt-get install cuda-command-line-tools-9-0
export
LD_LIBRARY_PATH=${LD_LIBRARY_PATH:+${LD_LIBRARY_PATH}:}/usr/local/cuda/extras/CUP
TI/lib64
```

5) Installing NVIDIA TensorRT 3.0

This is used to optimize inference performance.

```
tar xzvf TensorRT-3.0.4.Ubuntu-14.04.5.x86_64.cuda-9.0.cudnn7.0.tar.gz
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:~/TensorRT-3.0.4/lib
```

6) Setting up the TensorFlow virtual environment

This will create the tensorflow virtual environment under the directory ~/tensorflow

```
sudo apt-get install python3-pip python3-dev python-virtualenv
virtualenv --system-site-packages -p python3 ~/tensorflow
source ~/tensorflow/bin/activate
```

7) Installing TensorFlow (GPU)

First make sure you are in the tensorflow virtual environment, then run the following

```
easy_install -U pip
pip3 install --upgrade tensorflow-gpu
sudo pip3 install six
```

7.1) Validating the TensorFlow (GPU) installation

First reboot the server.

```
sudo reboot
```

Then, run the following python code:

```
python
import tensorflow as tf
hello = tf.constant('Hello, TensorFlow!')
sess = tf.Session()
print(sess.run(hello))
```

If the system outputs "Hello, TensorFlow!", then you have installed it correctly. Congratulations!

3.2 Installing TensorFlow Object Detection API

Reference:

https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/installation.md

1) Activate your TensorFlow environment.

```
source ~/tensorflow/bin/activate
```

1.1) You should now see something similar to the following snippet:

```
(tensorflow) ubuntu@<ip-address-of-server>: ~$
```

2) Install the necessary Python dependencies.

```
pip3 install pillow
pip3 install lxml
pip3 install jupyter
pip3 install matplotlib
```

3) Install the updated protobuf zip.

3.1) Download the Python protobuf zip folder from the Google protobuf releases webpage. All protobuf releases are located at the following web address: <https://github.com/google/protobuf/releases>.

```
cd
wget https://github.com/google/protobuf/releases/download/v3.5.0/protobuf-
```

```
python-3.5.0.zip
```

3.2) Unzip the downloaded folder, creating the “protobuf-3.5.0” directory in your current directory.

```
sudo apt install unzip # install unzip
unzip protobuf-python-3.5.0.zip
```

3.3) Install protobuf.

```
cd ~/protobuf-3.5.0
sudo ./configure
sudo make check
sudo make install

# export to the correct library path
export LD_LIBRARY_PATH=/usr/local/lib
```

3.3.1) Verify protobuf 3.5.0 is currently installed.

```
protoc --version

# expected output
libprotoc 3.5.0
```

4) Download our models directory from using the following link. Do not unzip the file:

<https://drive.google.com/file/d/1z-PaHvQ3UcjQJRv-3HrVgQudpgDnBtok/view?usp=sharing>

4.1) Transfer zip file into the server’s home directory using the FTP client you used transferring the dependency files in step 0 of section 3.1.

5.) Use the following command to unzip the file.

```
unzip models.zip
```

This results in a newly created directory named “models” in your home directory.

4. Training a Custom Dataset

Refer to this tutorial:

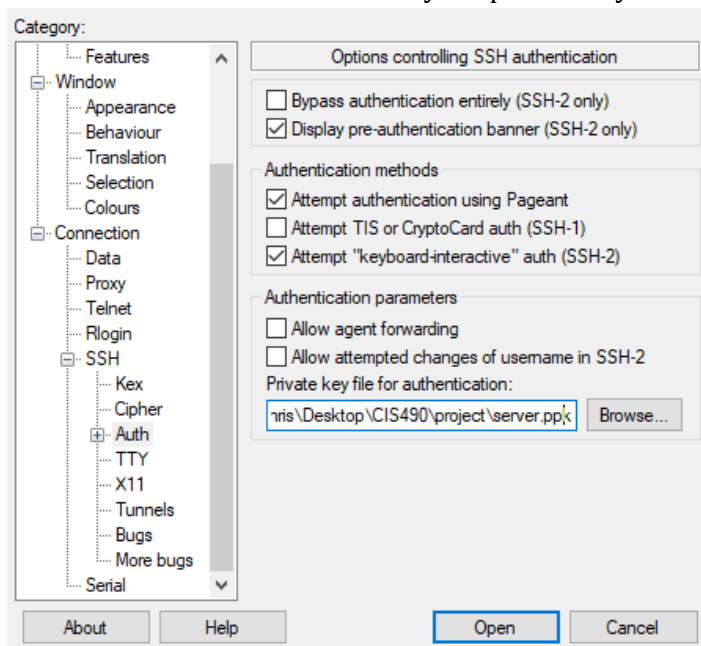
<https://pythonprogramming.net/introduction-use-tensorflow-object-detection-api-tutorial/>

5. Accessing Jupyter Notebook

5.1 Windows

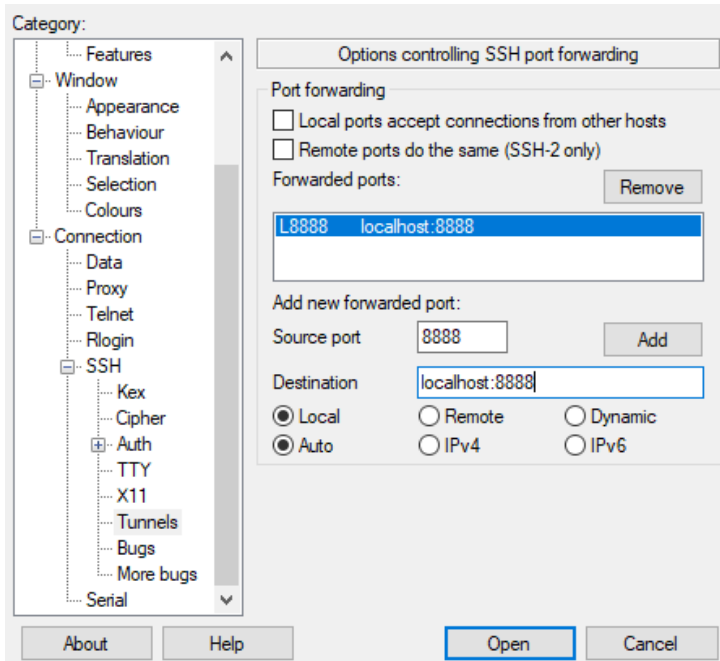
For the purposes of this project, Putty was used on Windows computers to interface the ec2 server.

1. Download and install Putty from <https://www.putty.org/>
2. Load in the private key under [Connection > SSH > Auth]
 - a. Click “browse” and select your private key file



3. Setup SSH tunneling under [Connection > SSH > Tunnels]. Enter these values:

Source Port:	8888
Destination:	localhost:8888



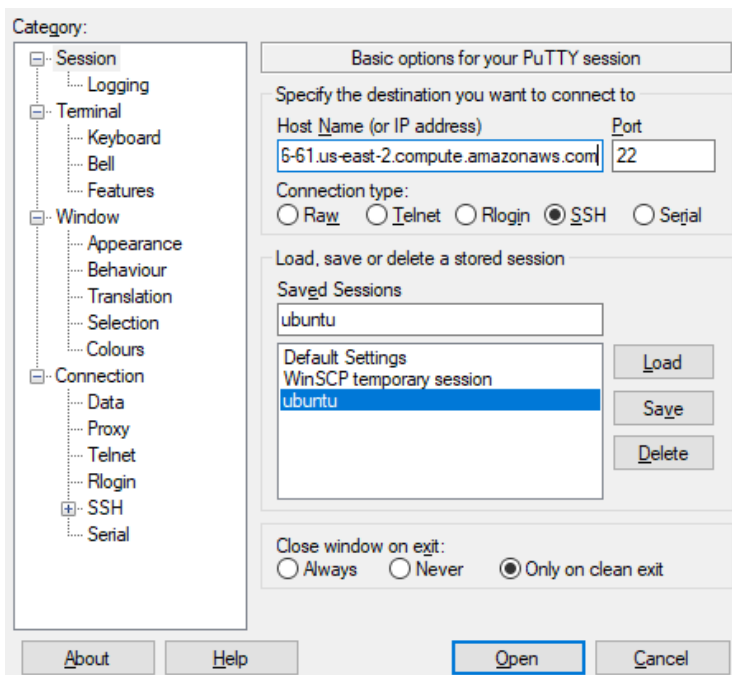
This is needed for jupyter notebook to work correctly.

4. Enter the server connection details under [Session]

IP: ec2-18-218-116-61.us-east-2.compute.amazonaws.com

Port: 22

Connection Type: SSH



5. Save the session and click open to login to the server.

6. Login as **Ubuntu**.

5.2 Mac

For the purposes of this project, Terminal, a native Mac application, was used on Mac computers to interface the ec2 server.

1. In the Terminal application, change directories to the location of the private key file that you created when you launched the instance. This private key file will have the format: *filename.pem*. Execute the following command into the Terminal window:

```
cd path/to/file.pem
```

2. Use the **chmod** command to make sure that your private key file isn't publicly viewable. For example, if the name of your private key file is *my-key-pair.pem*, use the following command:

```
chmod 400 /path/my-key-pair.pem
```

3. Use the **ssh** command to connect to the instance. You specify the private key (.pem) file and *user_name@public_dns_name*. For example, if you used an Amazon Linux AMI, *user_name* is *ec2-user*. If you used Ubuntu, *user_name* is *ubuntu*. Replace *<ec2-domain>* with your actual domain address.

```
ssh -i "FixItServer.pem" -L 8157:127.0.0.1:8888 user_name@<public_dns_name>
```

The first time you login, you should see the following response:

```
The authenticity of host 'ec2-198-51-100-1.compute-1.amazonaws.com
(10.254.142.33)'
can't be established.
RSA key fingerprint is
1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f.
Are you sure you want to continue connecting (yes/no)?
```

Type **yes** to connect. After your initial login, after you execute the **ssh** command, you will be automatically logged into the server.

5.3 Initial configuration

1. Activate Tensorflow environment by using the following command:

```
source ~/tensorflow/bin/activate
```

2. Change directories to the models/research directory.

```
cd ~/models/research
```

3. Compile the protoc library. Execute the following command:

```
protoc object_detection/protos/*.proto --python_out=.
```

4. Export the appropriate Python path. Execute the following command:

```
export PYTHONPATH=$PYTHONPATH:`pwd`:`pwd`/slim
```

5.4 Using Jupyter Notebook

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. This application will be used to execute the algorithm on chosen test images. These test images are located in the `~/models/research/object_detection/test_images` folder.

1. Change directories models/research/object_detection by using the following command:

```
cd ~/models/research/object_detection
```

2. Run Jupyter Notebook using the following command:

```
jupyter-notebook
```

You should see the following results:

```
Serving notebooks from local directory:
/home/ubuntu/models/research/object_detection
[I 22:00:04.373 NotebookApp] 0 active kernels
[I 22:00:04.373 NotebookApp] The Jupyter Notebook is running at:
[I 22:00:04.373 NotebookApp]
http://localhost:8888/?token=dec0e2a8d30add83c31c735213336503524e16e9f952b5b5
[I 22:00:04.373 NotebookApp] Use Control-C to stop this server and shut down all
kernels (twice to skip confirmation).
[W 22:00:04.373 NotebookApp] No web browser found: could not locate runnable
browser.
```

[C 22:00:04.373 NotebookApp]

Copy/paste this **URL into** your browser **when you connect for the first time,**
to login with a token:

http://localhost:8888/?token=dec0e2a8d30add83c31c735213336503524e16e9f952b5b5

3. In any web browser type **localhost:8157** to access the Jupyter Notebook interface. Copy the token string (i.e. the red bolded line of text bolded in the above snippet) into the password textbox to login.

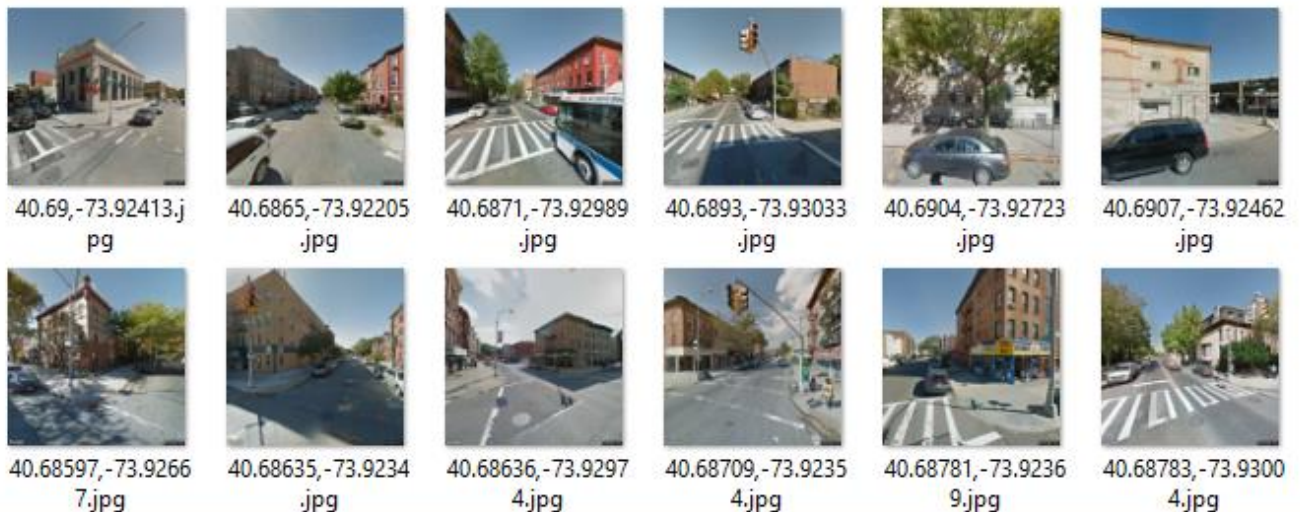
6. Future Use

6.1 How to gather images for Data Set

To gather images of litter for the data set, we have created a python script tool that pulls google street images along a given path specified by a .gpx file. The resolution of these images pulled are 640x640.

Download link:

<https://github.com/isaychris/gpx-streetview-imager>

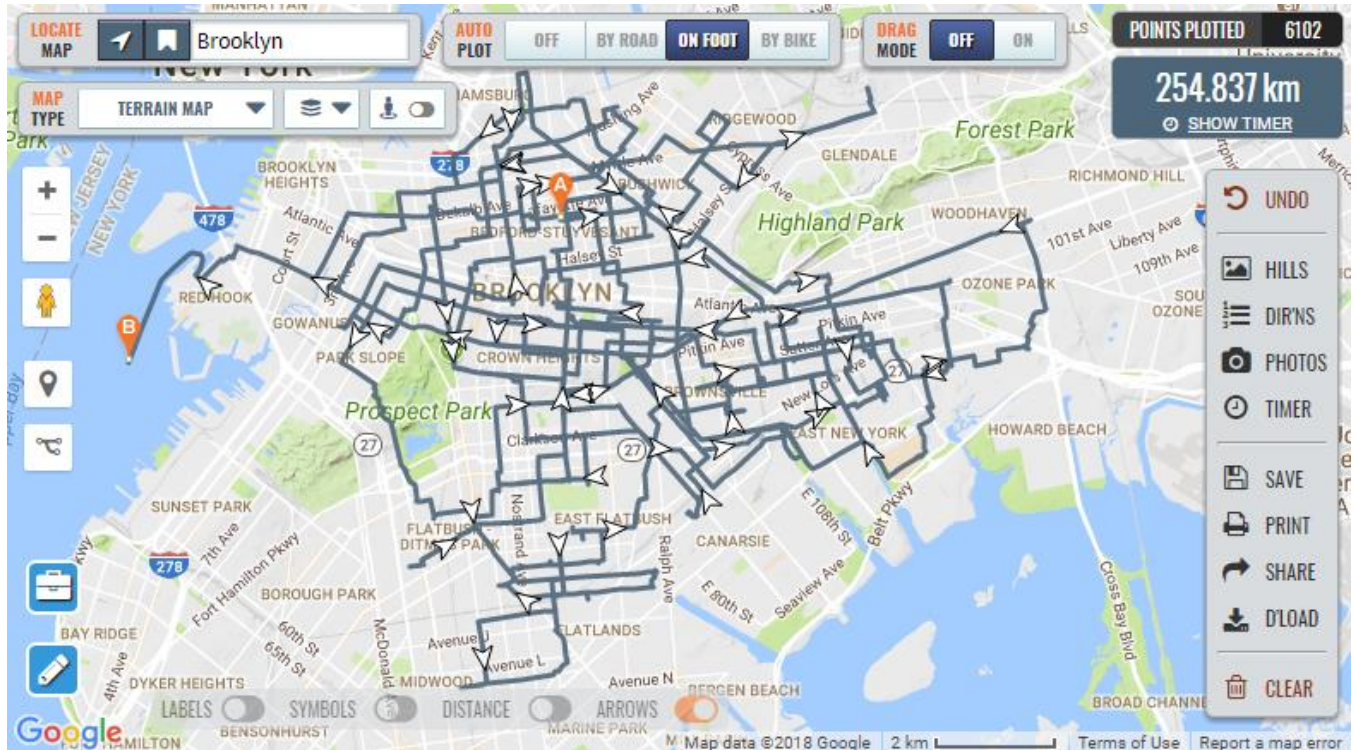


```
Running script ...  
[13 / 5110] Location: 40.69073,-73.92454
```

Prerequisites:

- Python 2
- .gpx file (instructions below)
- Google API key from <https://developers.google.com/maps/documentation/streetview/>.

Creating a Path:



1. Go to <https://www.plotaroute.com/routeplanner>
2. Enter the location you want to take street view images from.
3. Change Auto Pilot to 'On Foot'.
4. Enable street view Overlay.
5. Create your desired route along the blue street view paths. The number of points plotted displayed on the upper right corner is how many images the script will pull.
6. Download and save as 'File type: GPX, File Format: GPX, GPX TYPE: Track'.

Running the script:

1. Put your .gpx file in the same directory as the script.
2. Open the script and change the API_KEY, SAVE_PATH, and GPX_FILE variables.
3. Run the script by typing:

```
python street.py
```

After running the script:

<https://github.com/isaychris/gpx-streetview-imager/tree/master/other>

In the link provided above, there are two python scripts you can run after gathering the images.

You may have noticed there are several blank grey images in your folder. This is because the script couldn't find the google street data for that particular coordinate. To remove these blank grey images, use the cleaner.py script.

```
python cleaner.py
```

Another thing you may want to do is rename these images. To simply rename the files in bulk given a particular naming format like image001.jpg, images002.jpg, ect ..., use the renamer.py script.

```
python renamer.py
```

Before any of these scripts can be ran, edit in your image directory, then simply run.

6.2 How to label images for Data Set

In order to label the images in your data set, a image labeling software like **LabelImg** is required. The software is compatible with windows, linux, and *mac.

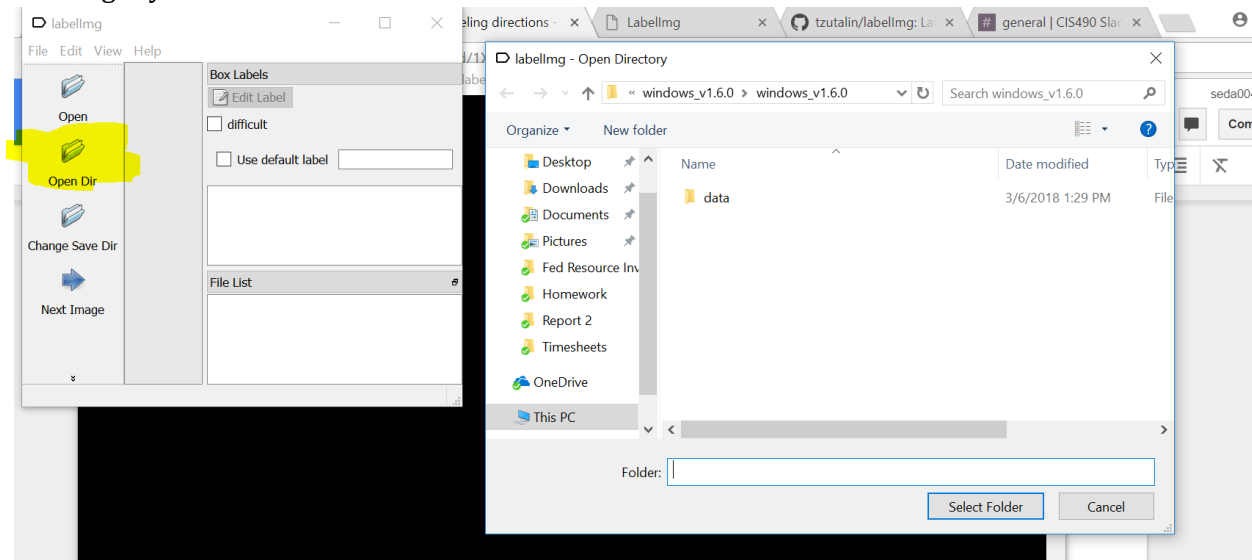
Download Link:

- <https://github.com/tzutalin/labelImg>

**If you wish to run the software on a mac, you will have to compile the software from source. Because of this, we recommend installing the windows version.*

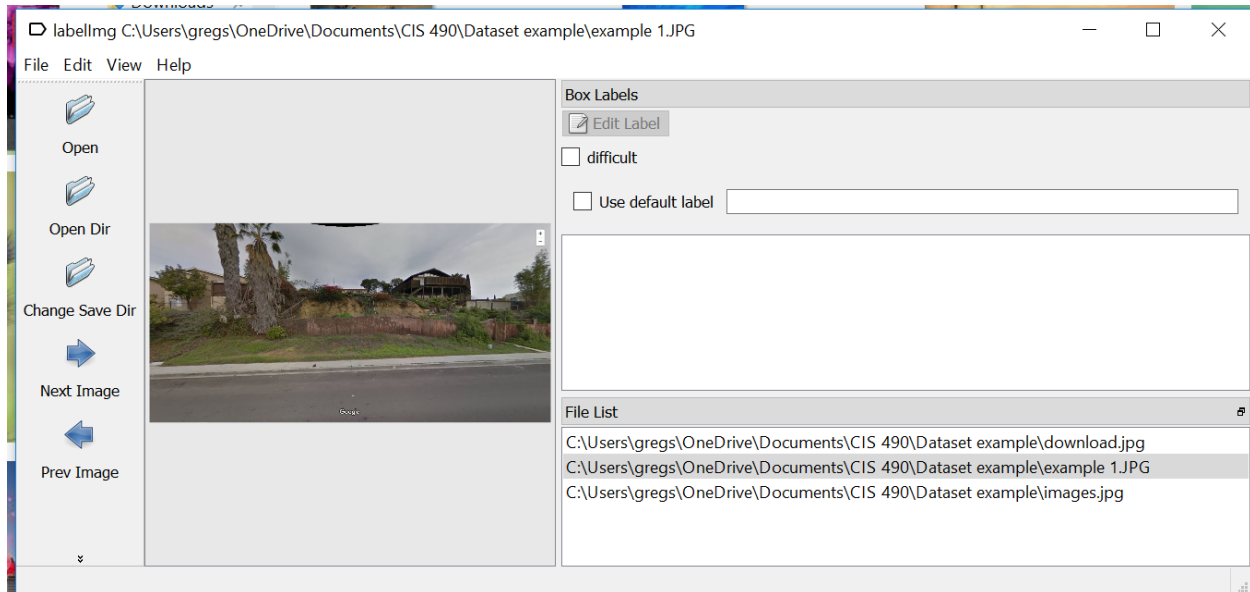
How to use the software:

1. Launch the application
2. Open the directory to your image dataset by clicking on "Open Dir". Then select the folder with the images you want to label.

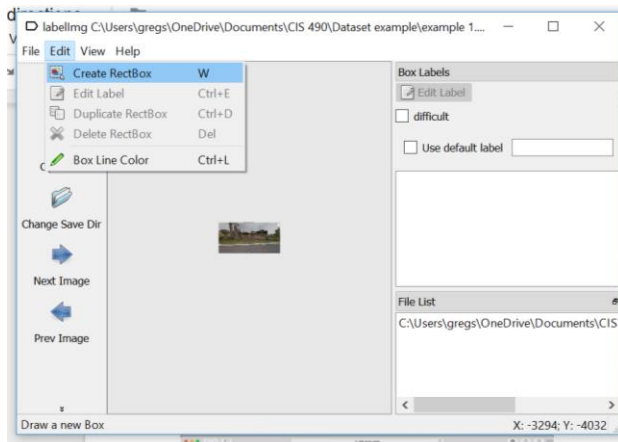


Your list of images will now show on the right side of the application.

3. To navigate between the different images in your dataset, double click on each file name under 'File List' on the bottom right corner.

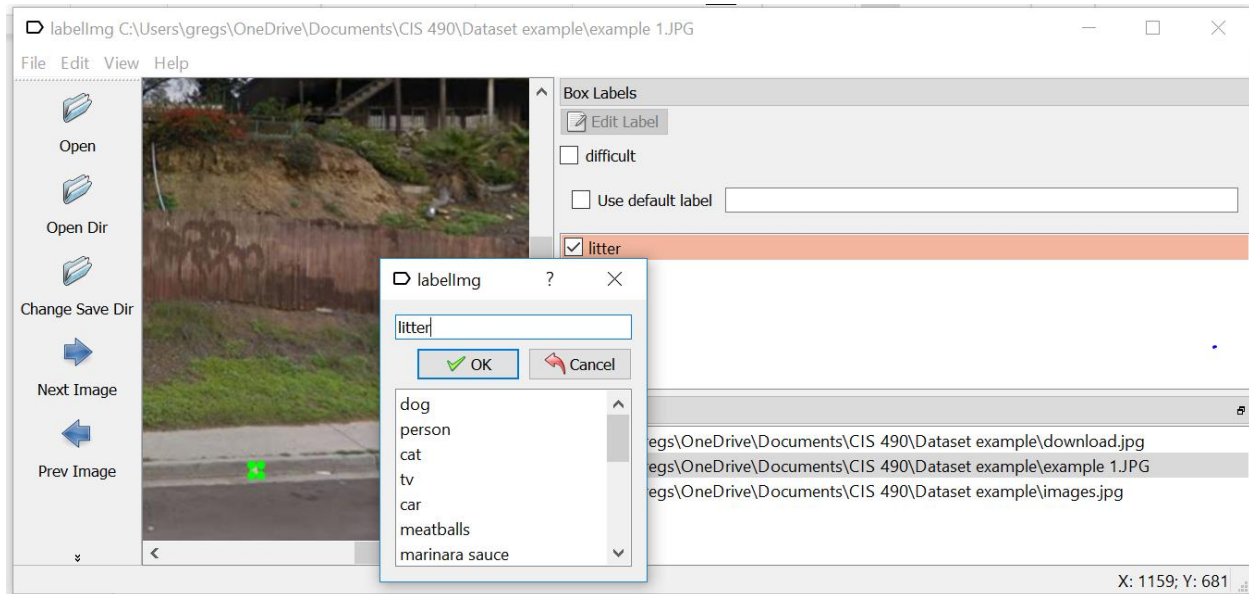


4. Select an image. To annotate the image, click on Edit -> 'Create RectBox'.(shortcut 'W')



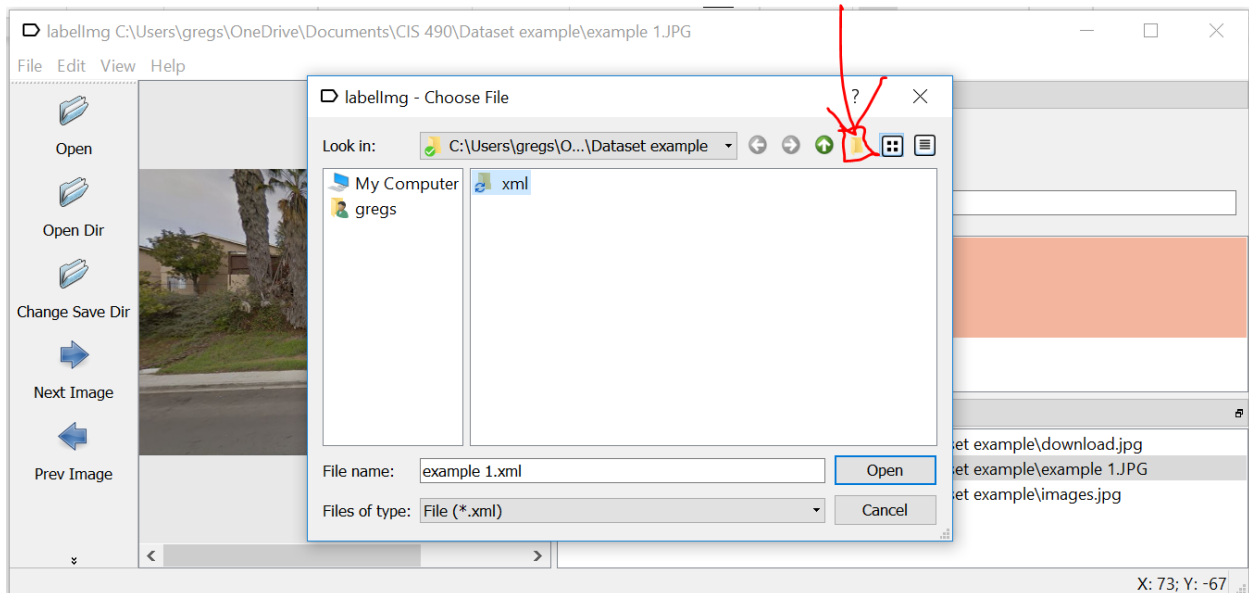
5. Locate the litter objects in the image, then click and release the left mouse button to draw a rectangle over the object.

Try to be as precise as possible with the boxing even though the litter might be really small.



6. Next type in 'litter' as the label name. (all lowercase)

7. Save the image. If done correctly, it should produce an .xml file with the same name as the image. Go to Edit -> 'save as' and create a new folder (click the marked icon) in your dataset folder called 'xml'. Save all the labeled images in there.



8. Repeat steps for every image in your dataset folder

6.3 How to convert the dataset into TFrecord.

Once the images in the dataset have been labeled, it must be converted to a format tensorflow can read called a TFrecord.

This is done by running two scripts that first converts the .xml files into a .csv file, then converting the .csv file into a TFrecord file.

Before that can be done, the dataset must first be split into a training and test sets. For our project, we split 80% into training, and 20% into testing.

Instructions:

1) Login to the server, then make a directory called "Object-Detection" and structure it as the following:

Object-Detection

- training/
- data/
- images/
- test/
- train/

2. Download these two files and place in into the root of the directory.

- xml_to_csv.py = [\[Download Link\]](#)
- generate_tfrecord.py = [\[Download Link\]](#)

2) Split the dataset into training and testing sets.

- Move the images into images/, then move 80% of .xml files into train/ and 20% into test/.

3) Create the .csv files for training and testing.

- The created csv files will be placed in data/

```
cd object-detection
python3 xml_to_csv.py
```

4) Convert the .csv files for training and testing into TFRecords.

- The created TFrecords will be placed in data/

```
source ~/tensorflow/bin/activate
python3 generate_tfrecord.py --csv_input=data/train_labels.csv --
output_path=data/train.record
python3 generate_tfrecord.py --csv_input=data/test_labels.csv --
output_path=data/test.record
```