Lab Guide to Train and Deploy a Convolutional Neural Network (CNN) Model

Overview

Convolutional Neural Networks (CNN) are designed to train machine learning models that can perform image classification tasks such as object detection, face detection, and emotion detection.

In this lab, we will use GPU-based virtual machines to train a CNN model that can recognize handwritten numbers. The model will be trained with popular MNIST dataset.

Objectives

- Using NVIDIA DIGITS to train a CNN model
- Testing and validating the trained model
- Running the trained model locally for inferencing

Prerequisites

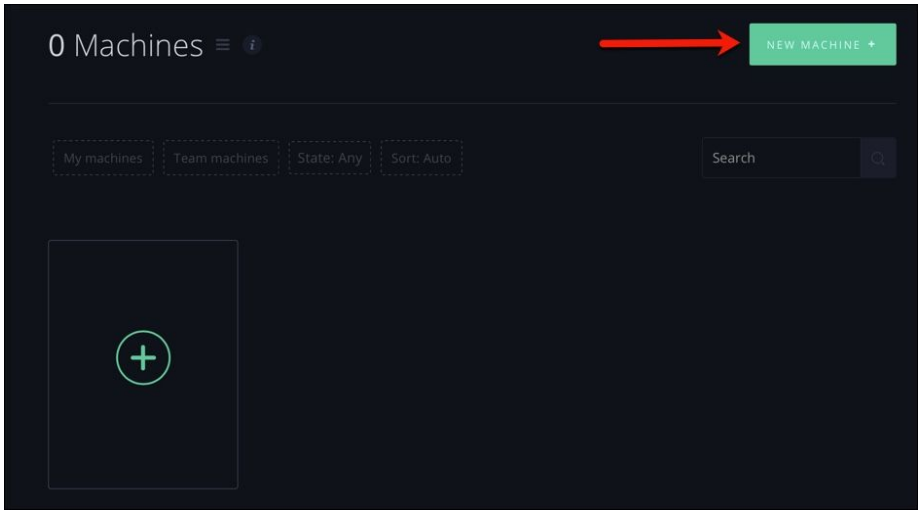
- Docker for Windows / Docker for Mac
- Git Client
- A valid account with Paperspace

Setup

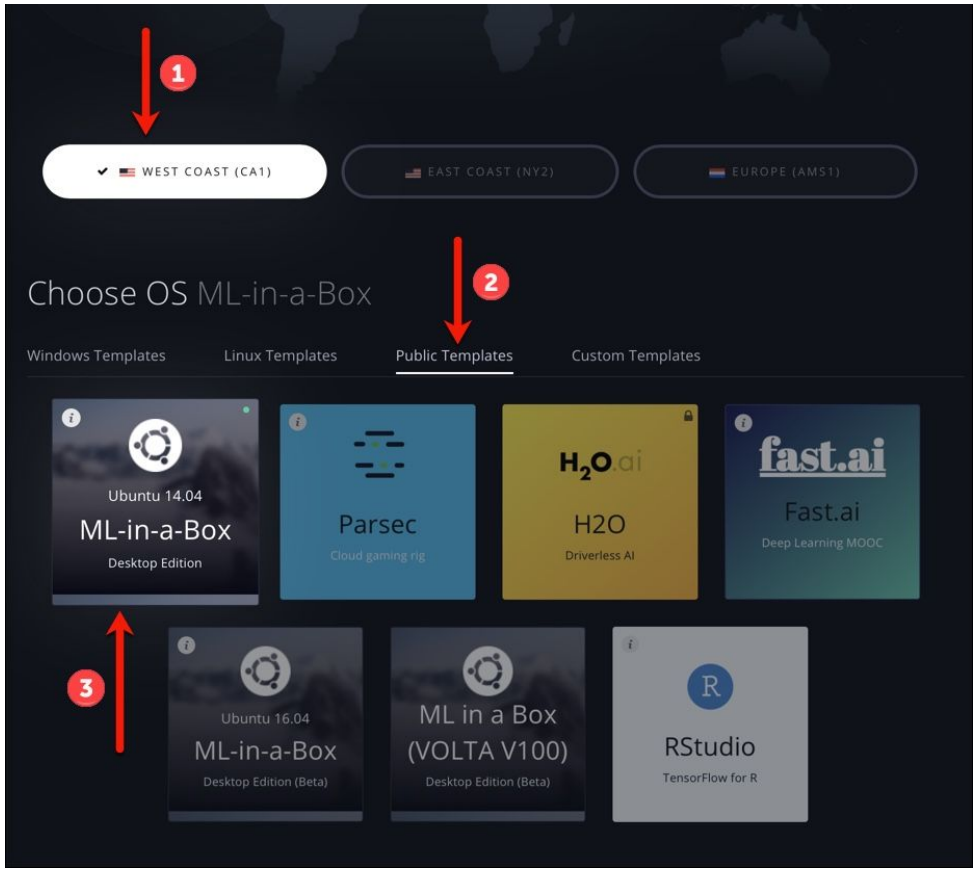
1. Clone the lab repo to access scripts and test data - *`git clone https://github.com/janakiramm/cnn-lab`*
2. Pull the Caffe CPU image for inference on local machine - *`docker pull bvlc/caffe:cpu`*

Step 1 - Launching Machine Learning in a Box VM in Paperspace

1. Sign-in to your Paperspace account
2. Click on **Machines** under Compute

1. Click on **New Machine** to create a new VM	

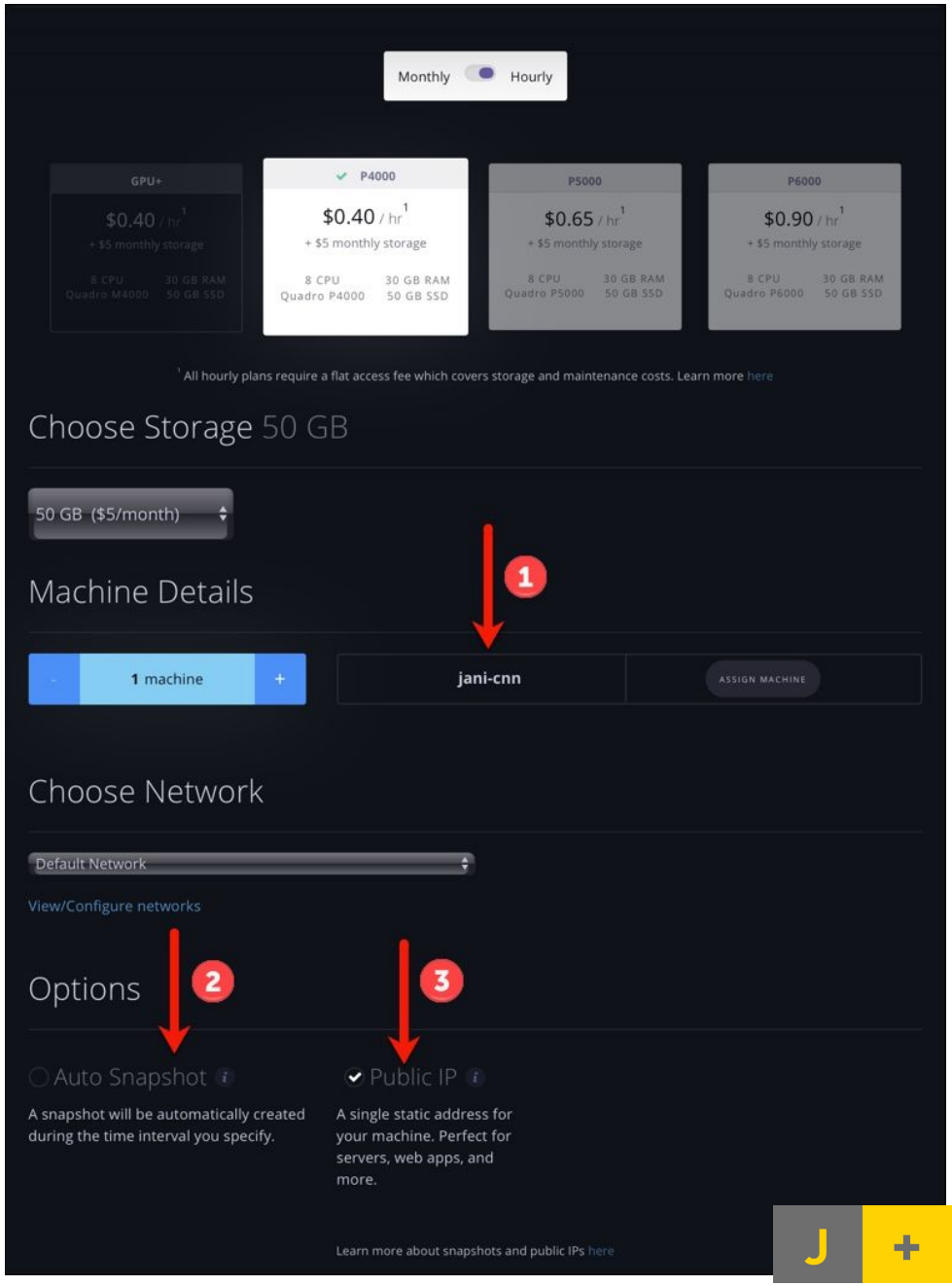
1. Choose **West Coast** as the region
2. Select **Public Templates** to access MLiaB
3. Select **ML-in-a-Box** template
4. Scroll down to access the machine configuration settings.

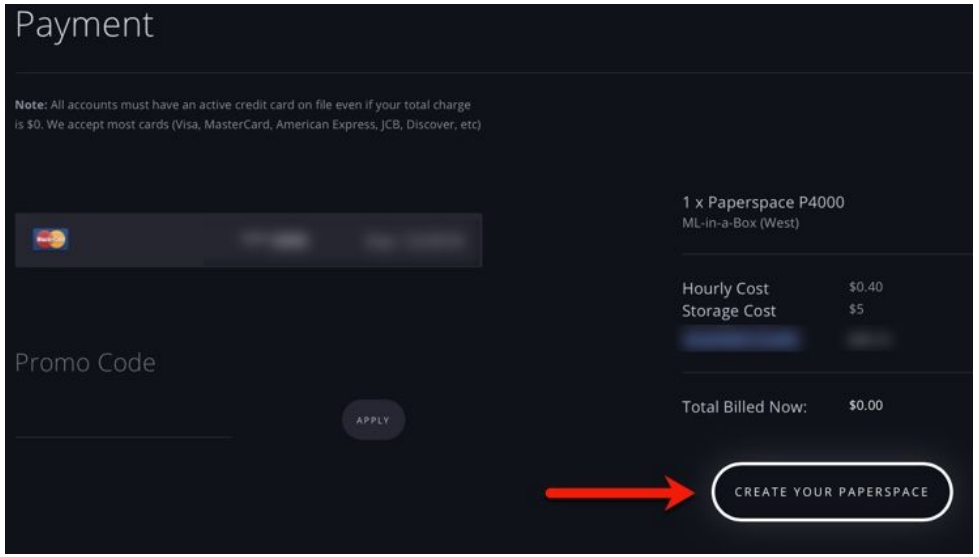


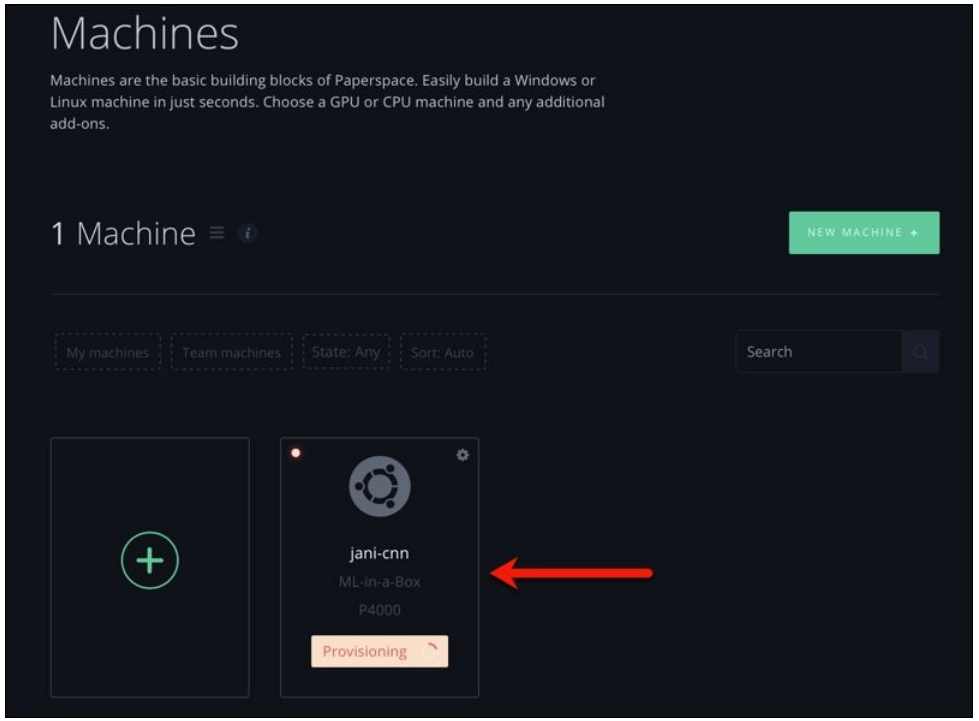
1. Change the name of the machine with **<firstname-cnn>**

2. Uncheck **Auto Snapshot**. We don't need it for the lab.

3. Check **Public IP** to assign an IP address to the VM.



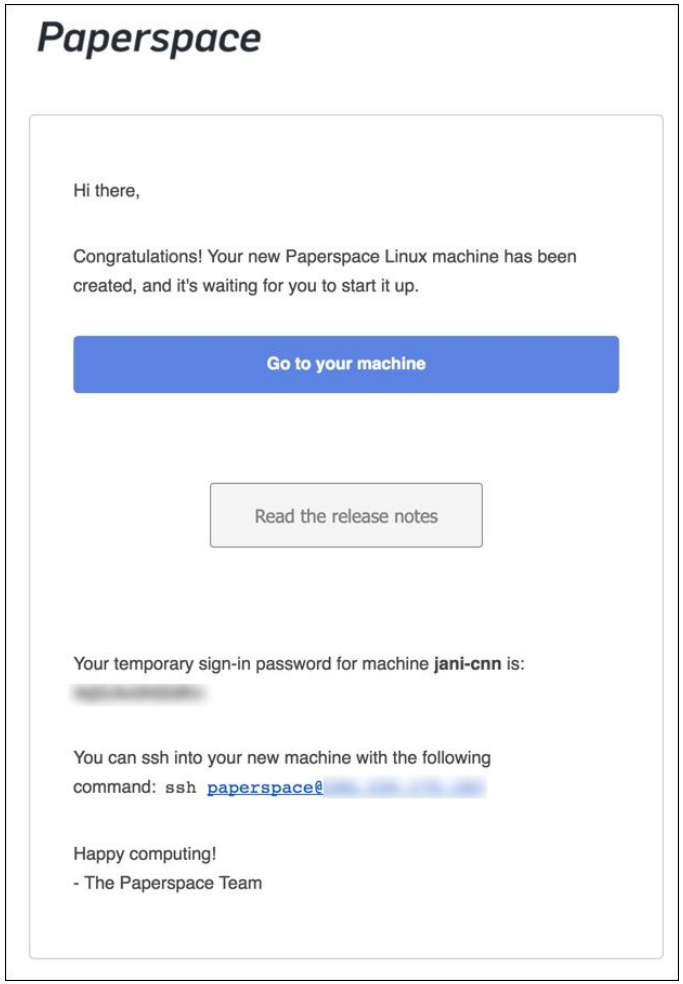
1. Click on **Create Your Paperspace** to launch the VM	

1. Wait for the machine to get provisioned.	

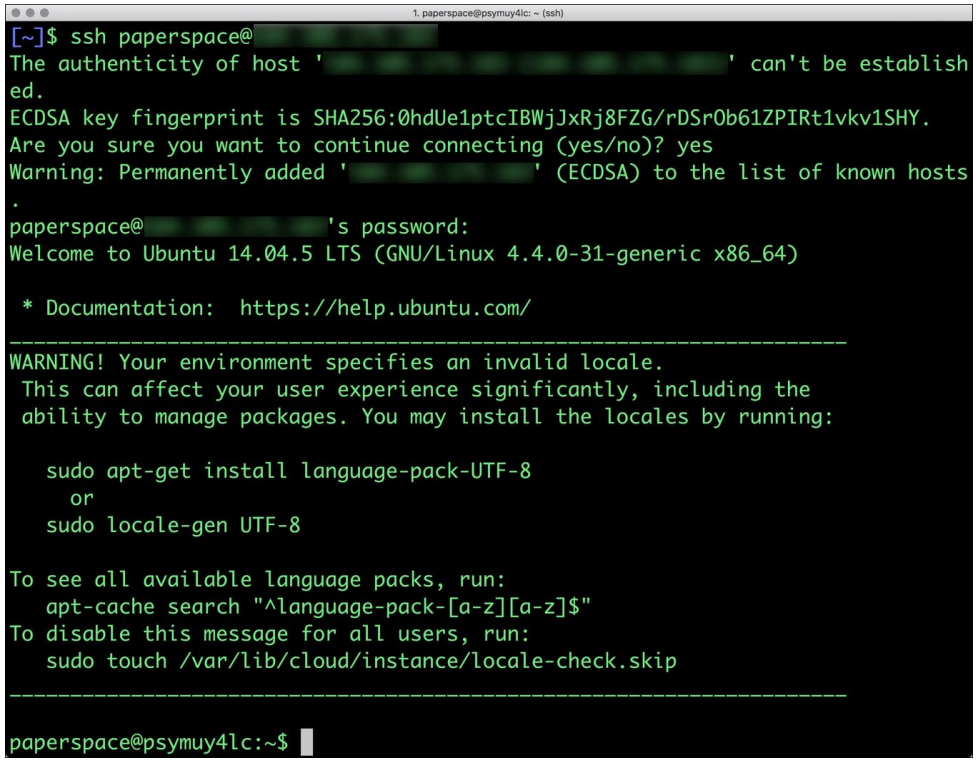
1. You should see **Ready** when the machine is provisioned	

Step 2 - Running NVIDIA DIGITS Container

1. Check your Inbox for details on accessing the VM.



1. Open terminal window and SSH into the VM with the credentials and IP address provided in the email



1. Create a directory called **data** in your home directory
2. Launch NVIDIA DIGITS container with the following command

sudo nvidia-docker run --name digits -d -p 5000:5000 -v $PWD/data:/data nvidia/digits

```
paperspace@psymuy4lc:~$ sudo nvidia-docker run --name digits -d -p 5000:5000 -v
$PWD/data:/data nvidia/digits
Using default tag: latest
latest: Pulling from nvidia/digits
99ad4e3ced4d: Pull complete
ec5a723f4e2a: Pull complete
2a175e11567c: Pull complete
8d26426e95e0: Pull complete
46e451596b7c: Pull complete
79f02f6ab059: Pull complete
59d2eaa1372d: Downloading  83.38MB/453.4MB
4421cd6e3c30: Download complete
9a47758649e5: Download complete
ce8dfeeea4e7: Download complete
0a4202988172: Download complete
505bc2c93c81: Download complete
18dccdda6b6e: Download complete
a0dbf08682d2: Download complete
2795e82bc224: Downloading  205.4MB/307.6MB
a5f44492569e: Download complete
e10245179830:
```

1. Verify that the DIGITS container is running with the following command

sudo docker ps

```
paperspace@psymuy4lc:~$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND             CREATE
D              STATUS              PORTS
   NAMES
6add4eea38c3        nvidia/digits       "python -m digits"   About
a minute ago   Up About a minute   0.0.0.0:5000->5000/tcp, 6006/tcp
   digits
paperspace@psymuy4lc:~$
```

Step 3 - Training the Model

1. Create the MNIST dataset within the container

sudo docker exec -it digits sh -c "python -m digits.download_data mnist /data"

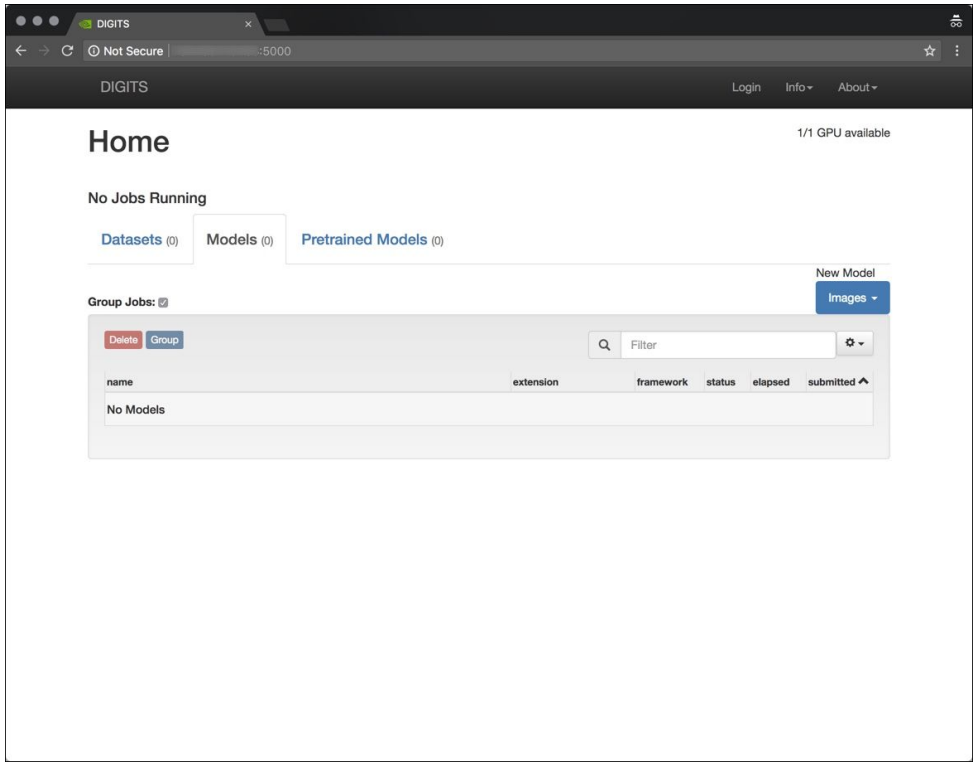
```
1. paperspace@psymuy4lc: ~ (ssh)
paperspace@psymuy4lc:~$ sudo docker exec -it digits sh -c
"python -m digits.download_data mnist /data"
Reading labels from /data/train-labels.bin ...
Reading images from /data/train-images.bin ...
Reading labels from /data/test-labels.bin ...
Reading images from /data/test-images.bin ...
Dataset directory is created successfully at '/data'
Done after 15.3298709393 seconds.
paperspace@psymuy4lc:~$
```

1. Verify the creation of the dataset by checking the local directory on the host

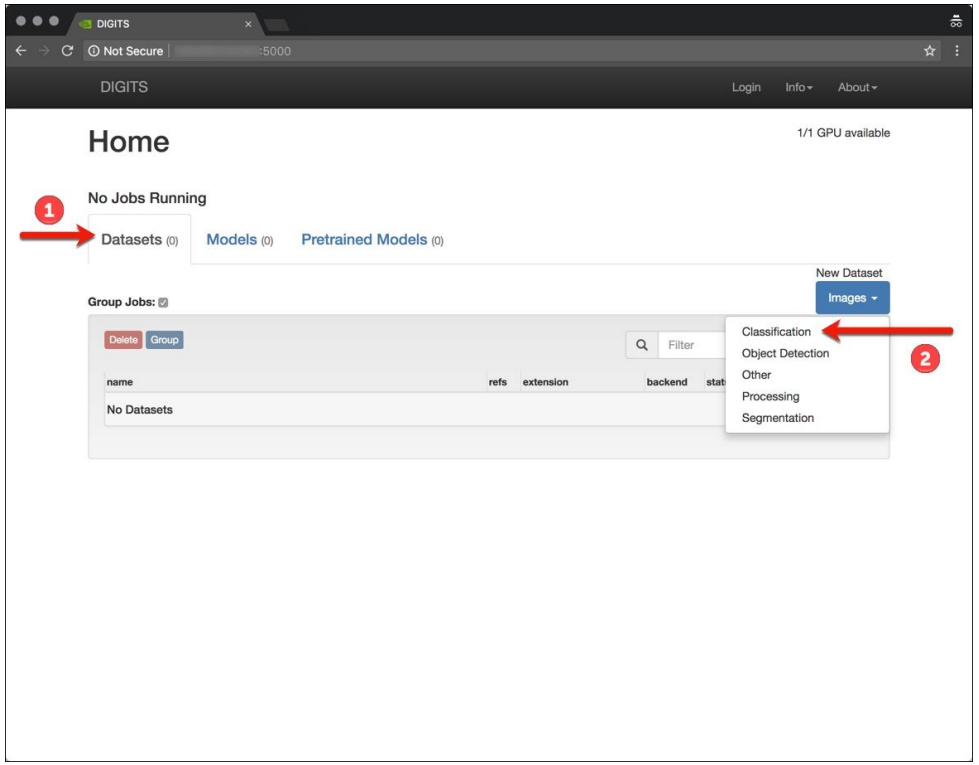
ls data

```
1. paperspace@psymuy4lc: ~ (ssh)
paperspace@psymuy4lc:~$ ls data
t10k-images-idx3-ubyte.gz  train
t10k-labels-idx1-ubyte.gz  train-images-idx3-ubyte.gz
test                       train-images.bin
test-images.bin            train-labels-idx1-ubyte.gz
test-labels.bin            train-labels.bin
paperspace@psymuy4lc:~$
```

1. Access DIGITS web application from the browser. The IP address is mentioned in the mail that you got during the creation of the VM. Don't forget to add port 5000 to the URL.



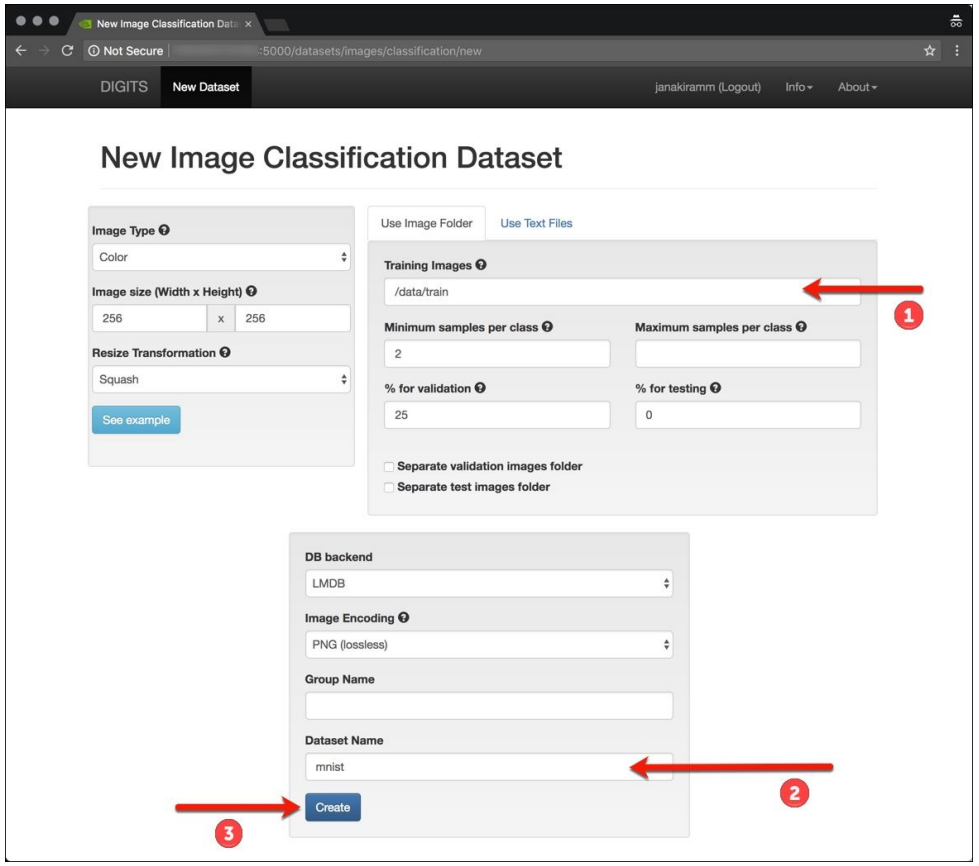
1. Click on the **Datasets** tab
2. Click **Images** and select **Classification**
3. Provide a username if asked to login



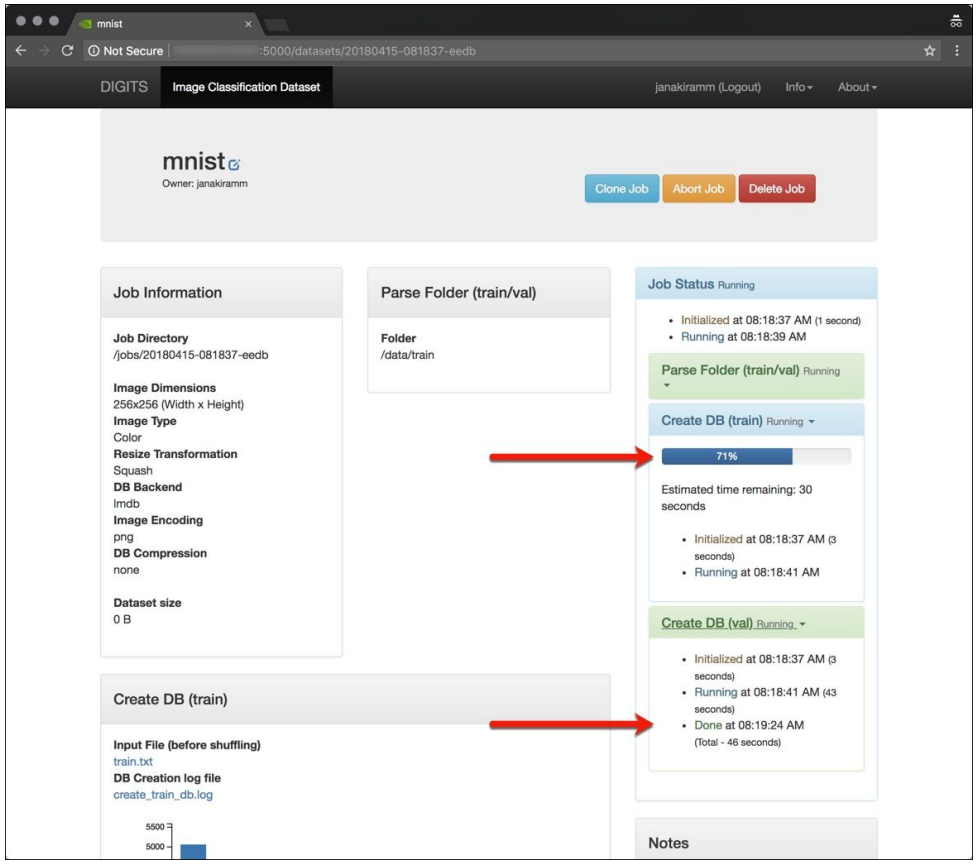
1. Under **Training Images**, type **/data/train** to point DIGITS to the MNIST dataset

2. Type **mnist** as the **Dataset Name**

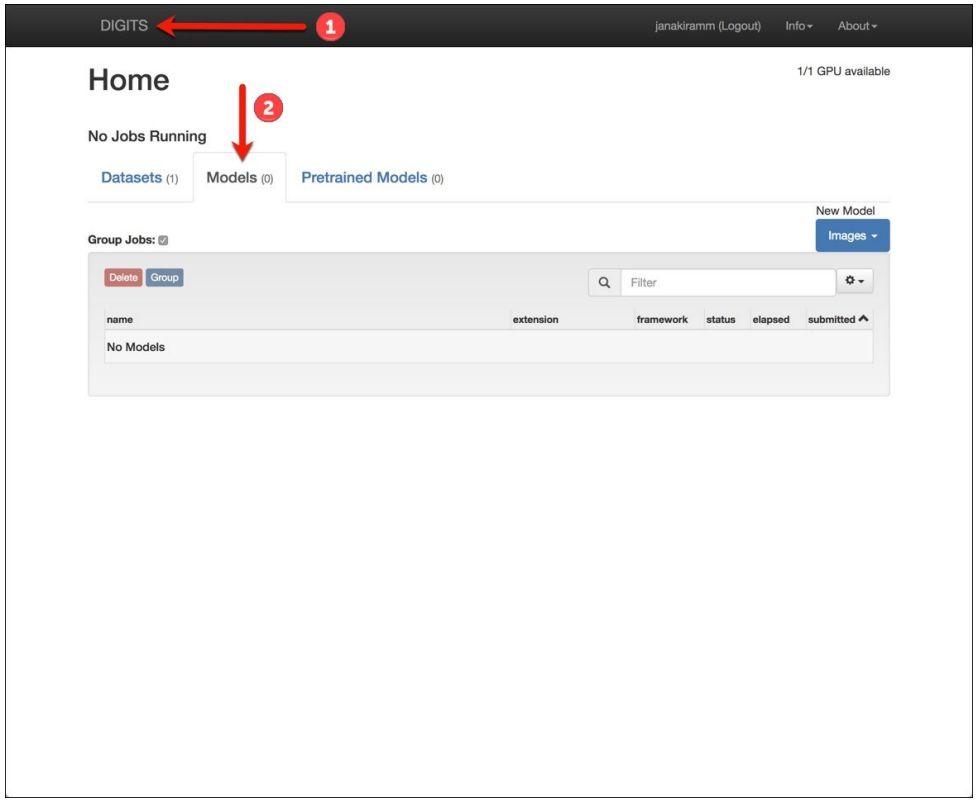
3. Click **Create** button to create the dataset



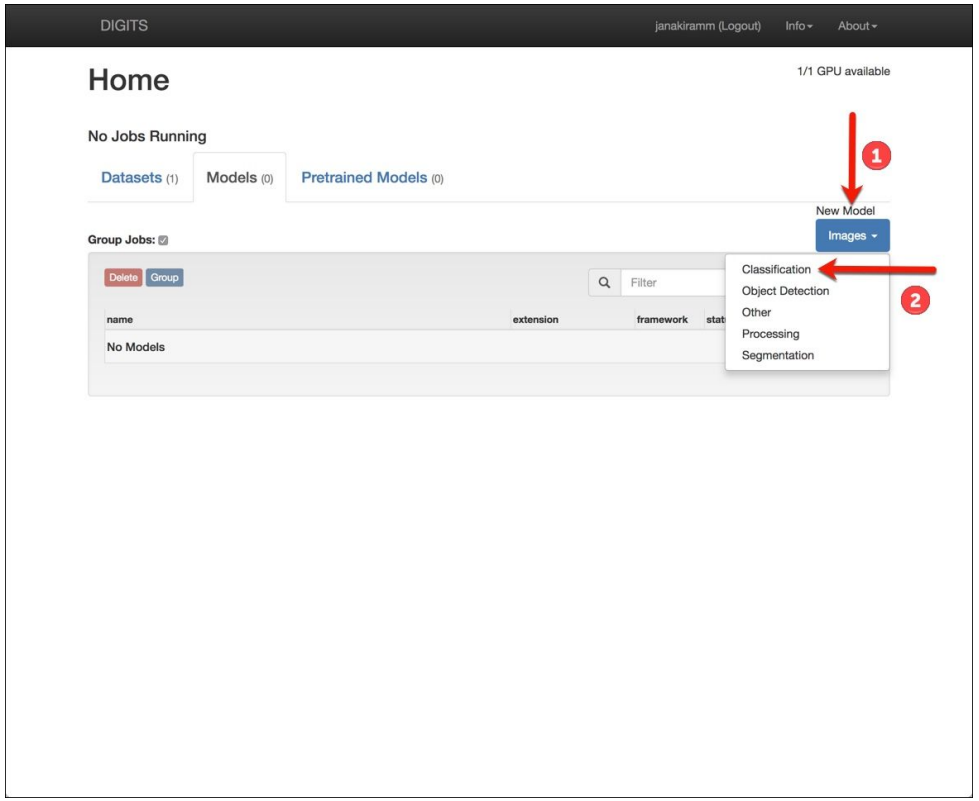
1. Wait for the train and val databases to get initialized. Both the jobs should be done before proceeding to the next step.



1. Click the **DIGITS logo** and then click on the **Models** tab.



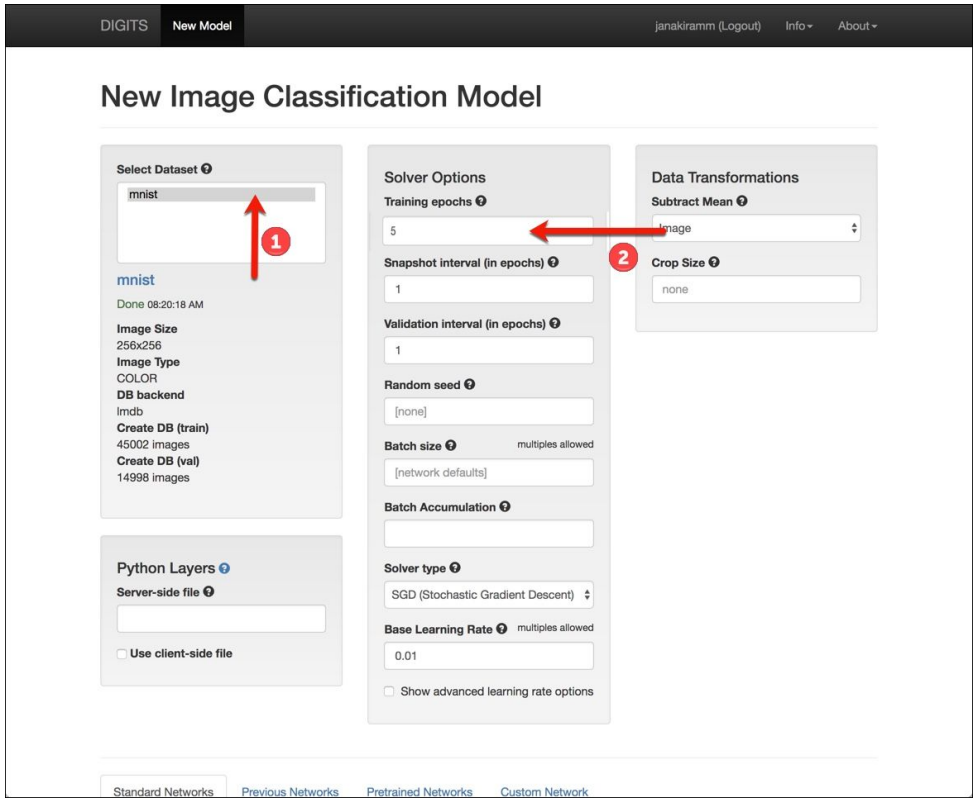
1. Click on **Images** and choose **Classification**



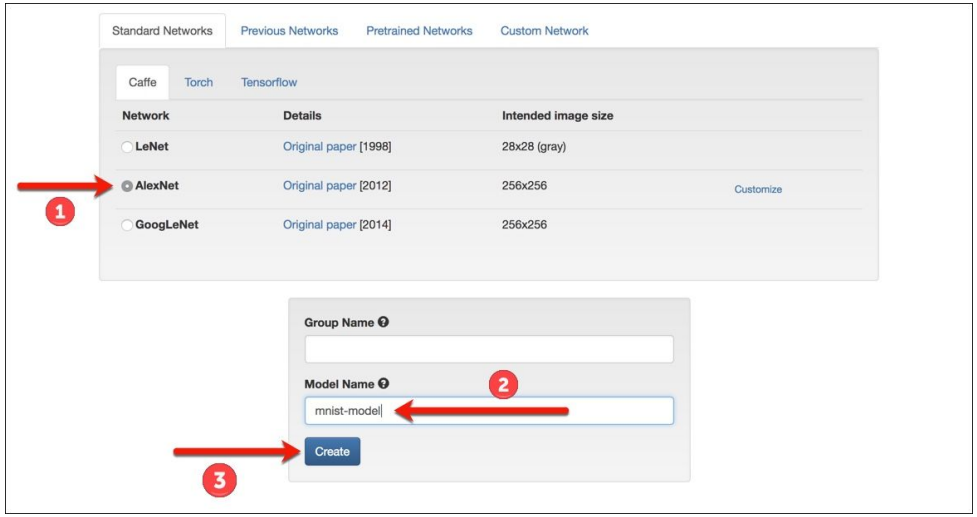
1. Choose **mnist** dataset created in the previous step.

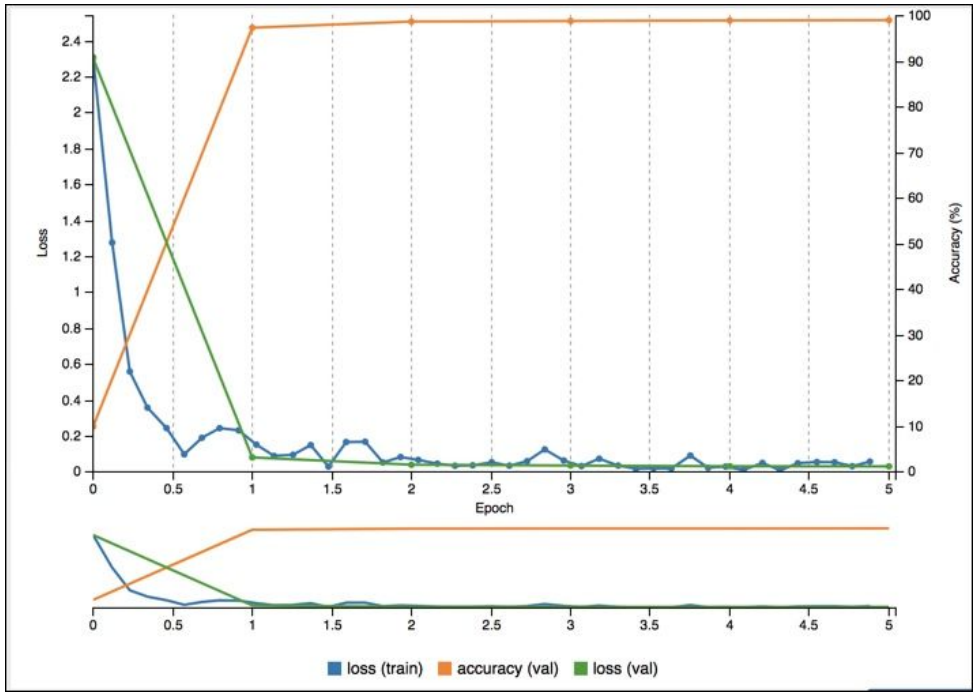
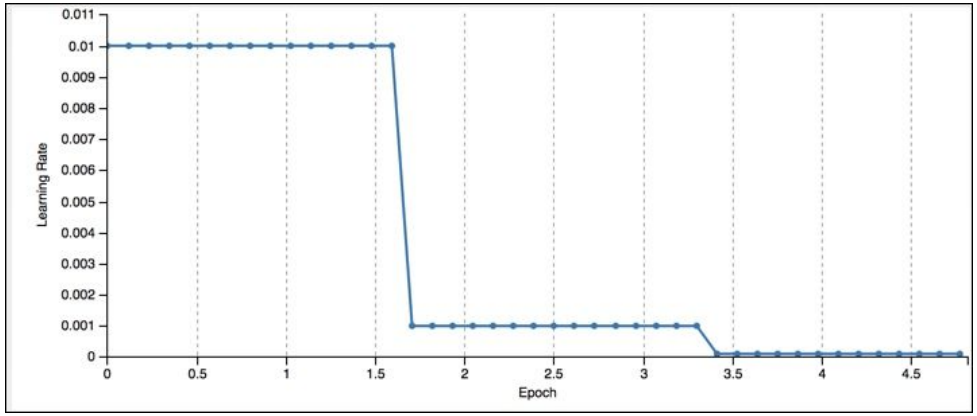
2. Change the **Training Epoche** to **5**

3. Scroll down to access the remaining options



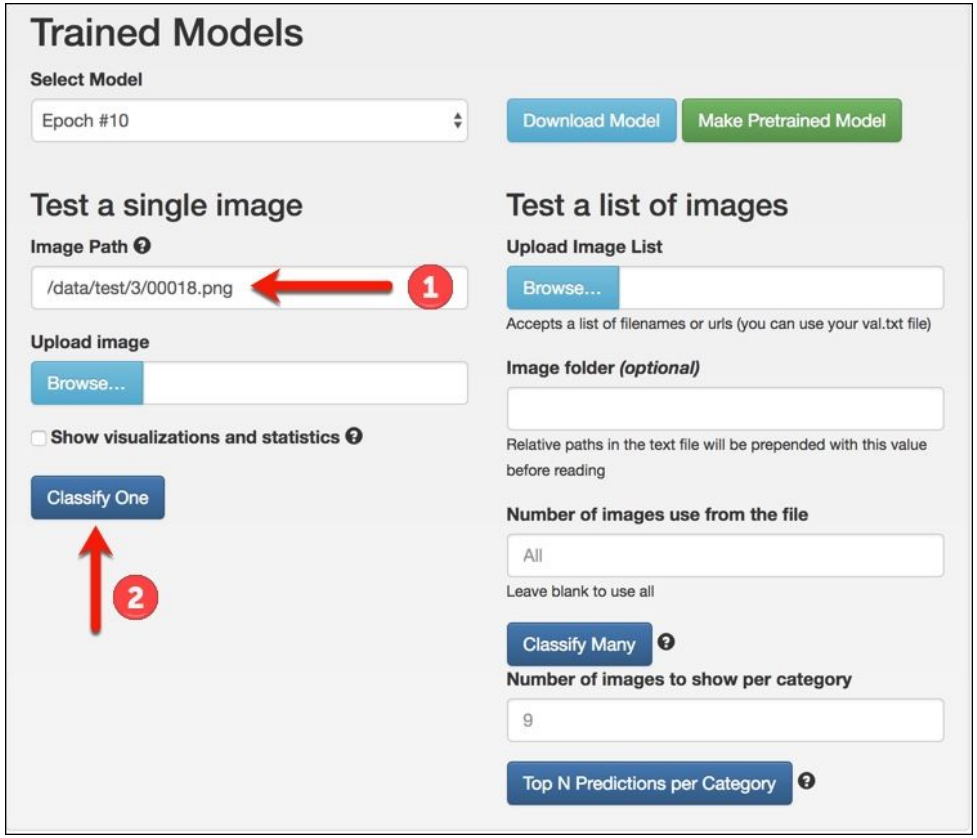
1. Under the **Standard Networks** tab, choose **AlexNet**
2. Type **mnist-model** as the **Model Name**
3. Click on **Create** button
4. Wait for the training job to start



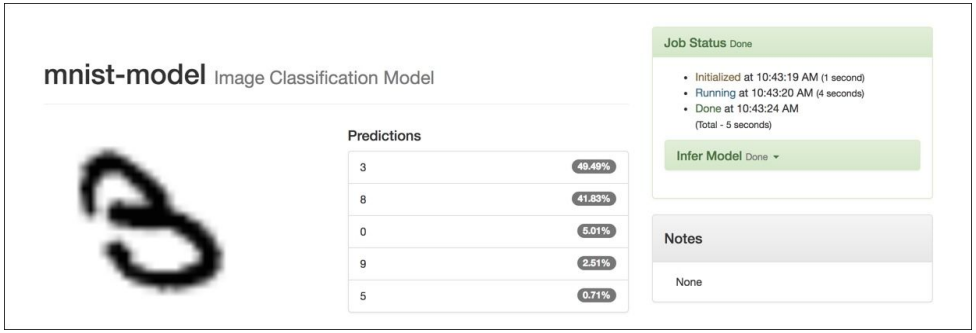
1. Notice how the accuracy improved after a few epochs	
1. Notice that the learning rate is adjusted with each passing epoch	

Step 4 - Testing and Validating the Model

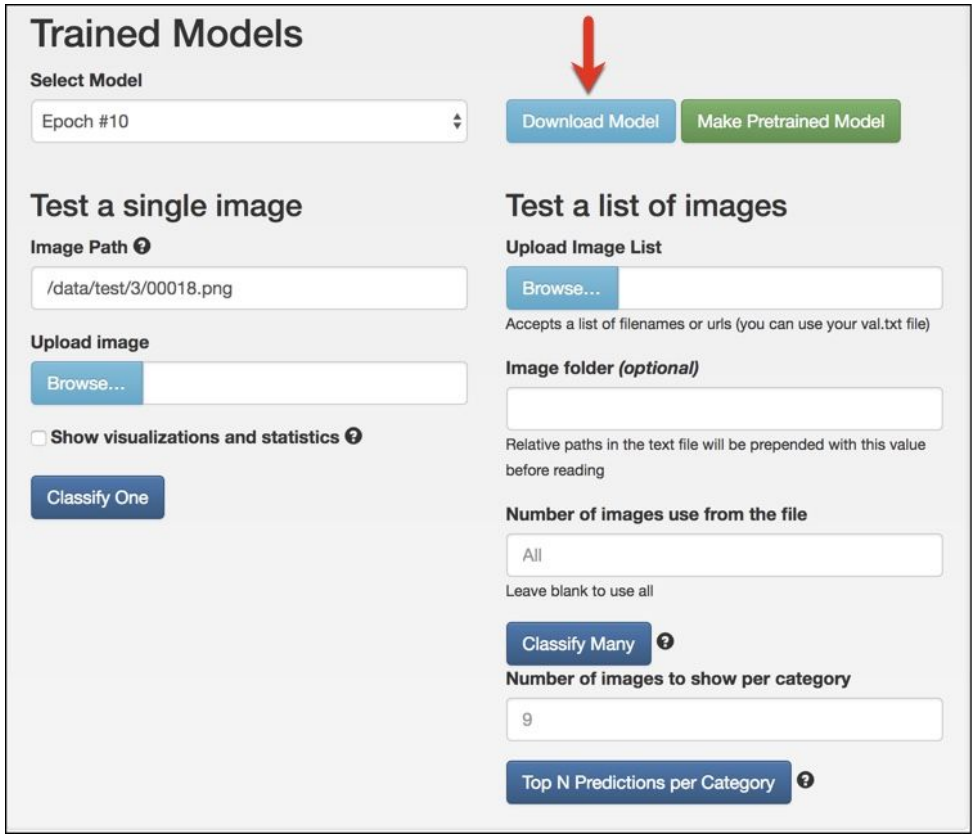
1. In the **Image Path** text box, type the following */data/test/3/00018.png*
2. Click on the **Classify One** button
3. The browser will open a new tab

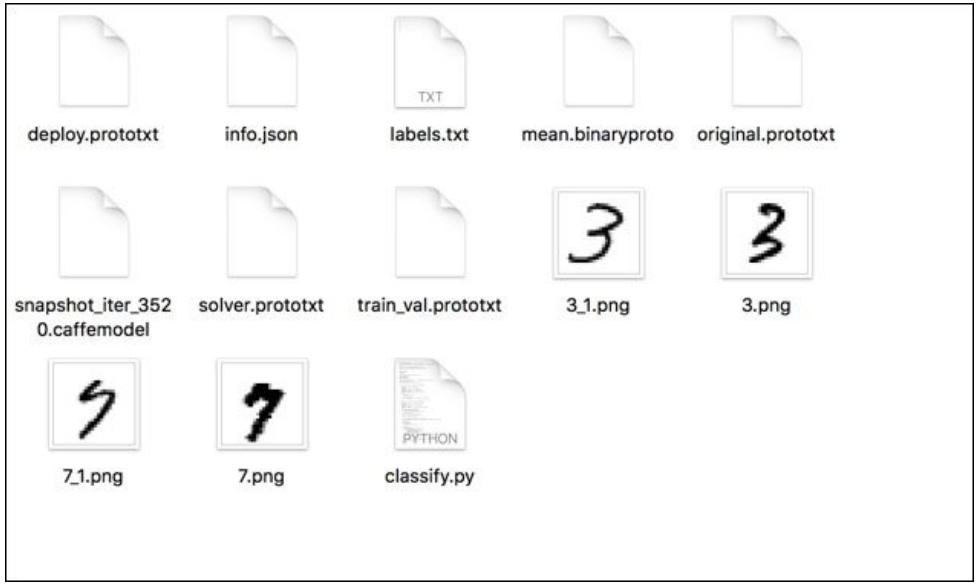


1. Notice that the model predicted the input image as digit 3 with an probability of 50%
2. Try testing the model with other images that represent different digits



Step 5 - Running the Model Locally for Inferencing

1. Click on the **Download Model** button under **Trained Models** section 2. Uncompress the downloaded file in your home directory	
1. Verify that the Caffe Intel image is available on your local machine *docker images*	
1. Clone the Github repo to access scripts and test images *git clone https://github.com/janakiramm/cnn-lab*	

1. Copy all the files from the repo to the directory containing the downloaded model from DIGITS *cd cnn-lab* *cp * ~/<Model_Directory>*	
1. Find the file with the extension **caffemodel**, and set its name as a variable *export MODEL_NAME="snapshot_iter_3520.caffemodel"*	

1. Launch the Docker container and pass the parameters

*docker run -it --rm --name caffe -v $PWD:/infer bvlc/caffe:cpu bash -c "cd /infer && python -W ignore classify.py -m mean.binaryproto -l labels.txt $MODEL_NAME deploy.prototxt **3_1.png** --nogpu"*

```
[20180415-101947-e7dc_epoch_10.0]$ docker run -it --rm --name caffe -v
$PWD:/infer bvlc/caffe:cpu  bash -c "cd /infer && python -W ignore clas
sify.py -m mean.binaryproto -l labels.txt $MODEL_NAME deploy.prototxt 3
_1.png --nogpu"
Processed 1/1 images in 0.341568 seconds ...
--------------------------- Prediction for 3_1.png ------------------
---------
 99.9999% - "3"
  0.0001% - "8"
  0.0000% - "9"
  0.0000% - "5"
  0.0000% - "7"

Script took 1.695304 seconds.
[20180415-101947-e7dc_epoch_10.0]$
```

1. Try the inferencing by sending different images

*docker run -it --rm --name caffe -v $PWD:/infer bvlc/caffe:cpu bash -c "cd /infer && python -W ignore classify.py -m mean.binaryproto -l labels.txt $MODEL_NAME deploy.prototxt **7.png** --nogpu"*

```
[20180415-101947-e7dc_epoch_10.0]$ docker run -it --rm --name caffe -v
$PWD:/infer bvlc/caffe:cpu  bash -c "cd /infer && python -W ignore clas
sify.py -m mean.binaryproto -l labels.txt $MODEL_NAME deploy.prototxt 7
.png --nogpu"
Processed 1/1 images in 0.342311 seconds ...
--------------------------- Prediction for 7.png -------------------
---------
 99.9215% - "7"
  0.0354% - "9"
  0.0119% - "1"
  0.0095% - "2"
  0.0068% - "5"

Script took 1.629630 seconds.
[20180415-101947-e7dc_epoch_10.0]$
```

Step 6 - Cleaning Up Resources

1. Access Paperspace Console by visiting https://www.paperspace.com/console/machines 2. Switch to list mode by clicking on the icon next **Machine** 3. Select the checkbox against your machine 4. Click **Deactivate** to shutdown the machine