

## Step 1 – Train using Sagemaker instance (not Studio)

1. Created Sagemaker notebook instance
2. Use 'ml.t2.medium' instance type for notebook – it is universal instance type, which will be enough for the notebook to run (and cheap)
3. (optional) when creating notebook instance it is useful to provide git repository path – you will start with a repository cloned already
4. Creation of an instance takes 5-10 minutes – if you already have an instance it is much faster to use existing one than to create a new one every time
5. Train the network using the baseline script – fix bugs in the script such as unreachable code, wrong debug prints.
6. Train using multiple instances and prepare results
7. Multi instance training ends after 2 minutes, but in fact it required more time to be initialized

Screenshot of Sagemaker notebook instance:

The screenshot shows the Amazon SageMaker console interface for a notebook instance named 'project4'. The breadcrumb navigation at the top reads 'Amazon SageMaker > Notebook instances > project4'. On the right side of the header, there are four buttons: 'Delete', 'Stop', 'Open Jupyter', and 'Open JupyterLab'. Below the header, the 'Notebook instance settings' section is displayed, featuring an 'Edit' button on the right. The settings are organized into four columns: Name, Status, Notebook instance type, and Platform identifier. The 'Name' column contains 'project4', 'ARN' (arn:aws:sagemaker:us-east-1:633572263357:notebook-instance/project4), and 'Lifecycle configuration' (-). The 'Status' column shows 'InService' with a green checkmark icon, 'Creation time' (Apr 15, 2022 10:07 UTC), and 'Last updated' (Apr 15, 2022 10:13 UTC). The 'Notebook instance type' column lists 'ml.t2.medium', 'Elastic Inference' (-), and 'Volume Size' (5GB EBS). The 'Platform identifier' column shows 'notebook-at1-v1'.

Name	Status	Notebook instance type	Platform identifier
project4	InService	ml.t2.medium	notebook-at1-v1
ARN	Creation time	Elastic Inference	
arn:aws:sagemaker:us-east-1:633572263357:notebook-instance/project4	Apr 15, 2022 10:07 UTC	-	
Lifecycle configuration	Last updated	Volume Size	
-	Apr 15, 2022 10:13 UTC	5GB EBS	

## Step 2 - run your notebooks using EC2 in your workspace

1. Create EC2 instance  
Instance creation looks completely different than the course described it, but after a couple tries I succeeded.
2. Selected instance type is 'm5.xlarge' and I used spot instance for the notebook, because it is cheaper. System type: Deep Learning AMI (Amazon Linux 2) Version 60.0. I tried to train the model on t2.micro, but it was killed due to memory requirements.
3. Spot instance snapshot – on Udacity AWS account I don't have permission to see spot instances (screenshot 1), but the instance definitely has 'spot' lifecycle (screenshot 2)

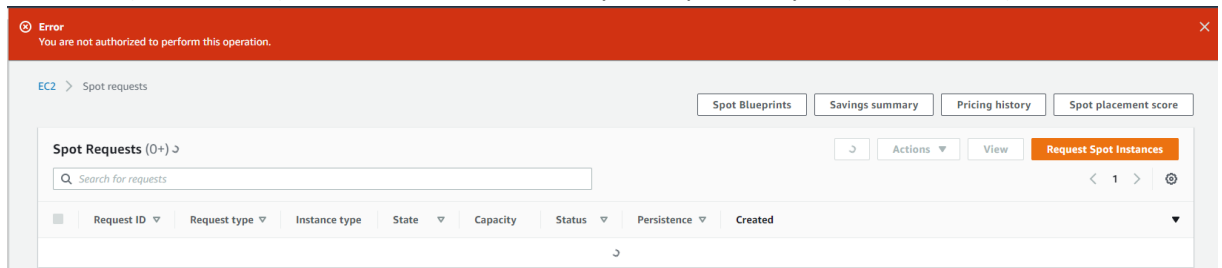


Figure 1 Error seeing spot instances on Udacity AWS account

4.

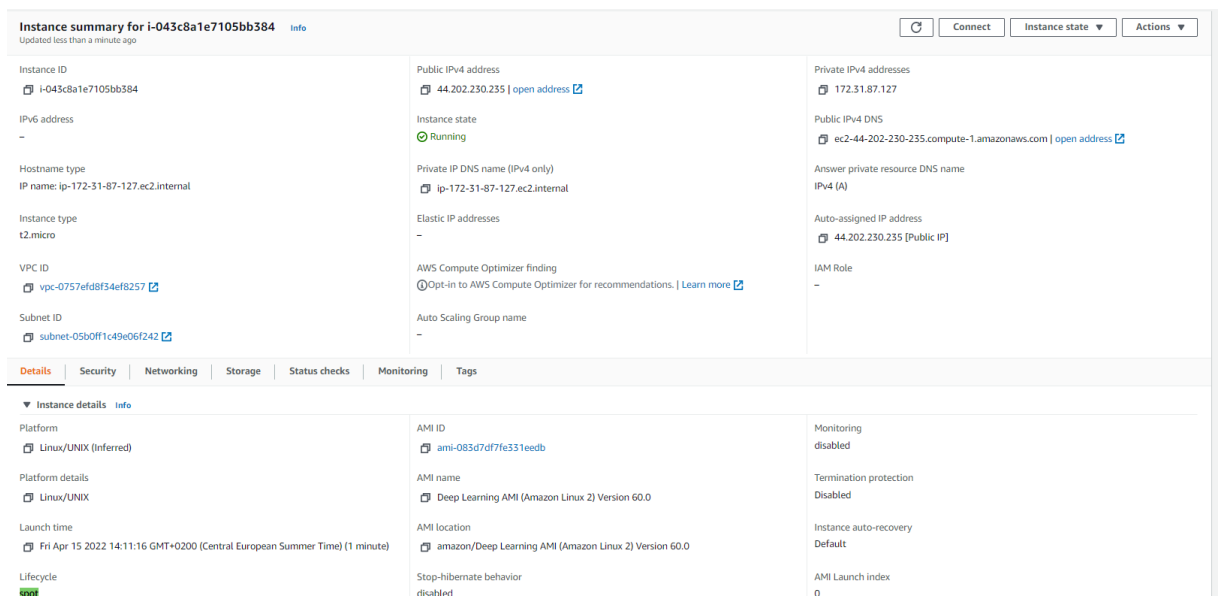


Figure 2 EC2 instance details presenting its type and lifecycle option

5. Enter the EC2 instance console and run training form inside.  
This step is especially tricky, because the console is slowly moving upwards. I don't know what causes this problem, but I have only 1-3 minutes to use for each console, later it goes out of screen.

6. Training succeeded:

```
Starting Model Training
Epoch: 0, phase train
Epoch 0, Phase train, Images [32/836 (4%)] Loss: 4.94 Accuracy: 1/32 (3.12%)
Epoch 0, Phase train, Images [64/836 (8%)] Loss: 4.87 Accuracy: 2/64 (3.12%)
Epoch 0, Phase train, Images [96/836 (11%)] Loss: 4.87 Accuracy: 3/96 (3.12%)
train loss: 4.8916, acc: 0.0312, best loss: 1000000.0000
Epoch: 0, phase valid
Epoch 0, Phase valid, Images [32/835 (4%)] Loss: 4.85 Accuracy: 1/32 (3.12%)
Epoch 0, Phase valid, Images [64/835 (8%)] Loss: 4.90 Accuracy: 1/64 (1.56%)
Epoch 0, Phase valid, Images [96/835 (11%)] Loss: 4.89 Accuracy: 1/96 (1.04%)
valid loss: 4.8780, acc: 0.0104, best loss: 4.8780
Break due to epoch limit at epoch: 0
Model saved.
(pytorch_p38) [root@ip-172-31-31-209 ~]# cd TrainedModels/
(pytorch_p38) [root@ip-172-31-31-209 TrainedModels]# ls -alh
total 92M
drwxr-xr-x 2 root root  23 Apr 15 16:51 .
dr-xr-x--- 7 root root 210 Apr 15 16:50 ..
-rw-r--r-- 1 root root 92M Apr 15 16:51 model.pth
(pytorch_p38) [root@ip-172-31-31-209 TrainedModels]#
```

Figure 3 Snapshot of the training logs and generated model.pth file

7. The code for EC2 instances is very similar to sagemaker one, but it should be self-containing python script which trains the model on any machine using local file system to access the data, while Sagemaker scripts required methods to access S3 dataset. Using EC2 instances is much more similar to usage of your own machine.

### Step 3 – Lambda functions for inference

1. Create a new lambda function using provided code.
2. Initially Lambda does not have enough permissions to run the endpoint. Additional policies has to be added to Lambda in order to run it properly.
3. Lambda can be tested using JSON file with “url” entry.



```
Response
{
  "statusCode": 200,
  "headers": {
    "Content-Type": "text/plain",
    "Access-Control-Allow-Origin": "*"
  },
  "type-result": "<class 'str'>",
  "Content-Type-In": "<__main__.LambdaContext object at 0x7f936f5a3b50>",
  "body": "[[-0.21229296922683716, -0.15309667587280273, -0.3061373829841614, 0.1
```

Figure 4 Output of successful lambda execution

4. Lambda functions were already covered in project no2, the 'invoke\_endpoint' function works like a 'predict' function in Sagemaker

## Step 4 – Actual running of Lambda function

1. In order to access Endpoint Lambda required more permissions:

lambdap4-role-cq7rhg44

Delete

Edit

Summary

Creation date

April 15, 2022, 19:29 (UTC+02:00)

Last activity

None

ARN

arn:aws:iam::633572263357:role/service-role/lambdap4-role-cq7rhg44

Maximum session duration

1 hour

Permissions

Trust relationships

Tags

Access Advisor

Revoke sessions

Permissions policies (3)

You can attach up to 10 managed policies.

Filter policies by property or policy name and press enter

< 1 >

Policy name	Type	Description
AmazonSageMaker-ExecutionPolicy-20220415T120723	Customer managed	
AWSLambdaBasicExecutionRole-9d77e790-2184-450b-bd0a-f39710dc2c1f	Customer managed	
AmazonSageMakerFullAccess	AWS managed	Provides full access to Amazon Sa

Figure 5 Permissions for the lambda role

2. Lambda output:

```
"body": "[[-0.21229296922683716, -0.15309667587280273, -0.3061373829841614, 0.11011774092912674, -0.16774681210517883, -0.008066248148679733, -0.23197175562381744, 0.15635429322719574, -0.12622188031673431, -0.12885969877243042, 0.3695796728134155, -0.002062767744064331, -0.055869728326797485, 0.14742261171340942, -0.11725631356239319, -0.3161870837211609, -0.03053201735019684, -0.3684312105178833, -0.32753172516822815, 0.15400850772857666, 0.07220902293920517, 0.09954917430877686, 0.013533521443605423, -0.06362387537956238, -0.2291933298110962, -0.07303585857152939, -0.06227679178118706, -0.39104604721069336, 0.029530785977840424, -0.18164679408073425, -0.11869032680988312, -0.34357741475105286, 0.02373727224767208, -0.042474523186683655, 0.2240193486213684, -0.005224950611591339, -0.09473496675491333, -0.036600545048713684, 0.08635075390338898, -0.303500235080719, 0.08778636157512665, -0.025020167231559753, 0.017018944025039673, 0.10680541396141052, -0.10075695067644119, -0.15924464166164398, 0.23671609163284302, 0.0304688960313797, -0.04243922978639603, -0.046819064766168594, 0.05256631597876549, -0.03278496116399765, 0.006142046302556992, -0.01153213158249855, 0.027360720559954643, 0.08312027156352997, -0.09118440002202988, -0.08264446258544922, 0.11408921331167221, -0.09587959945201874, -0.048805855214595795, 0.058021046221256256, 0.13345670700073242, -0.13694187998771667, -0.19396939873695374, -0.3879741430282593, -0.32485464215278625, -0.10602971166372299, -0.2708476781845093, -0.0530611053109169, 0.23138290643692017, -0.016494084149599075, -0.04421060532331467, -0.2205985188484192, -0.16145184636116028, 0.0401601567864418, -0.167718306183815, -0.0998743399977684, 0.028368674218654633, 0.09563405811786652, -0.3112568259239197, -0.11798204481601715, -0.14478018879890442, -0.12296410650014877, -0.3944595754146576, 0.16801868379116058, -0.008254840970039368, -0.13363496959209442, -0.3903580605983734, 0.1054837629199028, -0.233144149184227,
```

-0.12806618213653564, 0.03157461807131767, -0.06345967203378677, -  
0.163484588265419, -0.07572126388549805, -0.14463841915130615, -  
0.16430017352104187, -0.062421269714832306, -0.15379102528095245, -  
0.010581500828266144, -0.1540013700723648, 0.12033206969499588,  
0.10423266142606735, -0.1938173770904541, 0.23208999633789062, -  
0.30273863673210144, -0.11447548866271973, -0.08791511505842209, -  
0.2549930512905121, -0.19253094494342804, 0.1697363406419754, -  
0.06303775310516357, -0.38952410221099854, -0.17013460397720337, -  
0.08217258006334305, -0.09334175288677216, 0.12874142825603485, -  
0.10503050684928894, -0.10680680721998215, -0.14144812524318695, -  
0.16845321655273438, -0.09375317394733429, -0.32222914695739746, -  
0.24126026034355164, 0.049243178218603134, -0.13309355080127716, -  
0.185033917427063, 0.000845693051815033, 0.037239909172058105,  
0.07688818126916885, -0.4072481095790863, -0.04945005103945732]]"

3. For the personal application adding full access role to the lambda is OK, because I have only one endpoint and only one lambda, but in general the permissions should be monitored and probably have some due date for them.

### **Step 5 – Autoscaling**

Autoscaling for endpoint and lambda allows concurrent request and inferences to be working on the endpoint. The values chosen should be based on expected endpoint load, which for this project is 0 (it will be deleted soon), so I set up minimum accepted values of 2 concurrent runs of lambda and endpoint.