

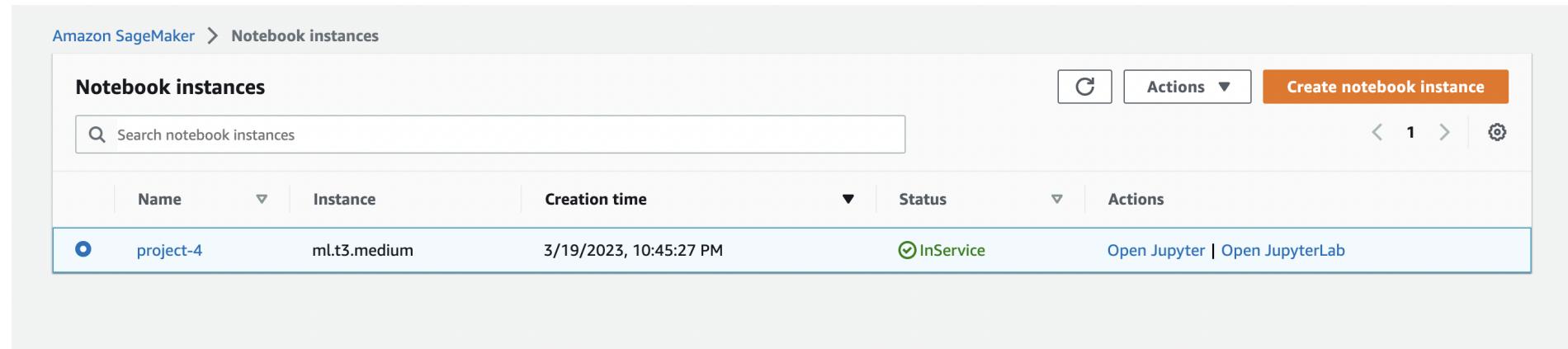
README.md

Udacity_project4

Step 1:

Used an ml.t3.medium notebook as from my previous experiences with similar projects this instance seemed sufficient with a low cost

Notebook set up:



The screenshot shows the 'Notebook instances' page in the Amazon SageMaker console. The page title is 'Amazon SageMaker > Notebook instances'. At the top right, there are buttons for 'Actions' (with a dropdown arrow) and 'Create notebook instance'. Below the title, there is a search bar labeled 'Search notebook instances'. The main table has columns: 'Name', 'Instance', 'Creation time', 'Status', and 'Actions'. There is one row visible, representing a notebook instance named 'project-4' with the 'ml.t3.medium' instance type, created on '3/19/2023, 10:45:27 PM', and currently 'InService'. The 'Actions' column for this row contains links 'Open Jupyter' and 'Open JupyterLab'.

Name	Instance	Creation time	Status	Actions
project-4	ml.t3.medium	3/19/2023, 10:45:27 PM	InService	Open Jupyter Open JupyterLab

S3 Bucket used:

Amazon S3 > Buckets > sagemaker-studio-bd01rhiuv4h

sagemaker-studio-bd01rhiuv4h [Info](#)

Objects (3)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Actions

- C
- Copy S3 URI
- Copy URL
- Download
- Open
- Delete
- Actions
- Create folder
- Upload

Find objects by prefix

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	test/	Folder	-	-	-
<input type="checkbox"/>	train/	Folder	-	-	-
<input type="checkbox"/>	valid/	Folder	-	-	-

Deployed endpoint:

Amazon SageMaker > Endpoints

Endpoints

Search endpoints

Actions

- C
- Update endpoint
- Actions
- Create endpoint

Name	ARN	Creation time	Status	Last updated
pytorch-inference-2023-03-19-18-54-03-855	arn:aws:sagemaker:us-east-1:775241169951:endpoint/pytorch-inference-2023-03-19-18-54-03-855	3/20/2023, 2:54:04 AM	InService	3/20/2023, 2:56:24 AM

Step 2:

I chose the g5.2xlarge as it was on the list of approved instances that work with the Deep Learning AMI, I needed the pytorch env to be able to run the `solution.py` script. It was one of the cheaper options providing enough cpu and memory (32 GiB) to train the model. I played it safe and chose a model with more specs than needed since it only cost \$1.212/hr and i would only be using it for 10-20 mins.

Recommended GPU Instances

[PDF](#) | [RSS](#)

We recommend a GPU instance for most deep learning purposes. Training new models is faster on a GPU instance than a CPU instance. You can scale sub-linearly when you have multi-GPU instances or if you use distributed training across many instances with GPUs. To set up distributed training, see [Distributed Training](#).

The following instance types support the DLAMI. For information about GPU instance type options and their uses, see [EC2 Instance Types](#) and select **Accelerated Computing**.

 **Note**

The size of your model should be a factor in selecting an instance. If your model exceeds an instance's available RAM, select a different instance type with enough memory for your application.

- [Amazon EC2 P3 Instances](#) have up to 8 NVIDIA Tesla V100 GPUs.
- [Amazon EC2 P4 Instances](#) have up to 8 NVIDIA Tesla A100 GPUs.
- [Amazon EC2 G3 Instances](#) have up to 4 NVIDIA Tesla M60 GPUs.
- [Amazon EC2 G4 Instances](#) have up to 4 NVIDIA T4 GPUs.
- [Amazon EC2 G5 Instances](#) have up to 8 NVIDIA A10G GPUs.
- [Amazon EC2 G5g Instances](#) have Arm-based [AWS Graviton2 processors](#).

DLAMI instances provide tooling to monitor and optimize your GPU processes. For more information about monitoring your GPU processes, see [GPU Monitoring and Optimization](#).

EC2 instance with model.pth saved to TrainedModels directory

```
BUILD_FROM_SOURCE_PACKAGES_LICENCES  PYTHON_PACKAGES_LICENSES      dogImages           solution.py
LINUX_PACKAGES_LICENSES              THIRD_PARTY_SOURCE_CODE_URLS  dogImages.zip
LINUX_PACKAGES_LIST                 TrainedModels                  nvidia-acknowledgements
[pytorch) python solution.py
/opt/conda/envs/pytorch/lib/python3.9/site-packages/torchvision/models/_utils.py:208: UserWarning: The parameter 'pretrained' is deprecated since 0.13 and may be removed in the future, please use 'weights' instead.
    warnings.warn(
/opt/conda/envs/pytorch/lib/python3.9/site-packages/torchvision/models/_utils.py:223: UserWarning: Arguments other than a weight enum or `None` for 'weights' are deprecated since 0.13 and may be removed in the future. The current behavior is equivalent to passing `weights=ResNet50_Weights.IMGNET1K_V1`. You can also use `weights=ResNet50_Weights.DEFAULT` to get the most up-to-date weights.
    warnings.warn(msg)
Downloading: "https://download.pytorch.org/models/resnet50-0676ba61.pth" to /home/ec2-user/.cache/torch/hub/checkpoints
/resnet50-0676ba61.pth
100%|██████████| 97.8M/97.8M [00:00<00:00, 384MB/s]
Starting Model Training
saved
[pytorch) ls
BUILD_FROM_SOURCE_PACKAGES_LICENCES  PYTHON_PACKAGES_LICENSES      dogImages           solution.py
LINUX_PACKAGES_LICENSES              THIRD_PARTY_SOURCE_CODE_URLS  dogImages.zip
LINUX_PACKAGES_LIST                 TrainedModels                  nvidia-acknowledgements
[pytorch) cd TrainedModels/
[pytorch) ls
model.pth
[pytorch) ]
```

EC2 Code vs Sagemaker Code:

- Not a lot of hyperparameter tuning in the EC2 code compared to Sagemaker code
- Deployment method is different, one generates a model.pth file and one uses an endpoint that can be accessed publicly

Step 3:

I created a lambda function using the `lambdafunction.py` script where I replaced the `endpoint_Name` variable with my previously created endpoint `pytorch-inference-2023-03-19-18-54-03-855`. The lambda function simply calls this endpoint and returns the response. I created a test event to see if my function works correctly:

```
{ "url": "https://s3.amazonaws.com/cdn-origin-etr.akc.org/wp-content/uploads/2017/11/20113314/Carolina-Dog-s
```

Result of lambda function test:

Execution result: succeeded ([Logs](#)) X

▼ Details

The area below shows the last 4 KB of the execution log.

```
access-control-allow-origin: ,  
"type-result": "<class 'str'>",  
"Content-Type-In": "LambdaContext([aws_request_id=940c8c52-b83c-4797-bc5d-a8cc79ee1988,log_group_name=/aws/lambda/project4,log_stream_name=2023/03/20[$LATEST]d6abca5676624089a21830d01fd426a1,function_name=project4,memory_limit_in_mb=128,function_version=$LATEST,invoked_function_arn=arn:aws:lambda:us-east-1:775241169951:function:project4,client_context=None,identity=CognitoIdentity([cognito_identity_id=None,cognito_identity_pool_id=None]))]",  
"body": "[[-14.143568992614746, -5.684410095214844, -1.8749933242797852, 1.0599637031555176, 1.5907402038574219, -9.410658836364746, 3.0227255821228027, 1.385068416595459,  
-13.199945449829102, 0.2824665307998657, 1.7210664749145508, 0.4888707399368286, -9.935746192932129, 2.262202024459839, 2.4222753047943115, 3.347254991531372, -9.389200210571289,  
-0.5058841109275818, -0.33502423763275146, 1.7883074283599854, -7.682830333709717, -4.413837432861328, 1.0785170793533325, -6.095279693603516, -0.4689871072769165, -10.435538291931152,  
-2.356555938720703, 0.19481587409973145, -2.488048553466797, -4.760436534881592, -3.755894184112549, -4.093752861022949, -9.263851165771484, -1.754446268081665, -13.825780868530273,  
-8.840593338012695, -1.4469965696334839, -9.024213790893555, 1.1076340675354004, -2.542921781539917, -3.6020495891571045, -9.03349781036377, 2.61519718170166, -4.319530963897705, ]]
```

Summary

Code SHA-256	Request ID
+pGuT/DmkEXSqnYSnUZJatMh+ncMjjMNqOELDchdZUw=	940c8c52-b83c-4797-bc5d-a8cc79ee1988
Duration	Billed duration
899.61 ms	900 ms
Resources configured	Max memory used
128 MB	76 MB

Log output

The section below shows the logging calls in your code. [Click here](#) to view the corresponding CloudWatch log group.

```
START RequestId: 940c8c52-b83c-4797-bc5d-a8cc79ee1988 Version: $LATEST  
Context::: LambdaContext([aws_request_id=940c8c52-b83c-4797-bc5d-a8cc79ee1988,log_group_name=/aws/lambda/project4,log_stream_name=2023/03/20[$LATEST]d6abca5676624089a21830d01fd426a1,function_name=project4,memory_limit_in_mb=128,function_version=$LATEST,invoked_function_arn=arn:aws:lambda:us-east-1:775241169951:function:project4,client_context=None,identity=CognitoIdentity([cognito_identity_id=None,cognito_identity_pool_id=None]))]  
Event Type::: <class 'dict'>
```

Test Event Name

test

Response

```
{  
  "statusCode": 200,  
  "headers": {
```

```
        "Content-Type": "text/plain",
        "Access-Control-Allow-Origin": "*"
    },
    "type-result": "<class 'str'>",
    "Content-Type-In": "LambdaContext([aws_request_id=09360260-4de3-4ea2-af2e-96d067e143fa, log_group_name=/aws
    "body": "[[-14.143568992614746, -5.684410095214844, -1.8749933242797852, 1.0599637031555176, 1.59074020385
}
```

Function Logs

```
START RequestId: 09360260-4de3-4ea2-af2e-96d067e143fa Version: $LATEST
Context::: LambdaContext([aws_request_id=09360260-4de3-4ea2-af2e-96d067e143fa, log_group_name=/aws/lambda/pro
EventType::: <class 'dict'>
END RequestId: 09360260-4de3-4ea2-af2e-96d067e143fa
REPORT RequestId: 09360260-4de3-4ea2-af2e-96d067e143fa Duration: 936.98 ms      Billed Duration: 937 ms Memo
```

Request ID

09360260-4de3-4ea2-af2e-96d067e143fa

Step 4:

Lambda function setup:

project4

[Throttle](#)[Copy ARN](#)[Actions ▾](#)

Function overview [Info](#)



Layers

(0)

[+ Add trigger](#)[+ Add destination](#)

Description

-

Last modified

13 minutes ago

Function ARN

[arn:aws:lambda:us-east-1:775241169951:function:project4](#)Function URL [Info](#)

-

[Code](#) | [Test](#) | [Monitor](#) | [Configuration](#) | [Aliases](#) | [Versions](#)**Execution result: succeeded** ([logs](#))[▼ Details](#)

The area below shows the last 4 KB of the execution log.

```
Access-Control-Allow-Origin: *  
},  
"type-result": "<class 'str'>",  
"Content-Type-In": "LambdaContext([aws_request_id=940c8c52-b83c-4797-bc5d-
```

Security policy:

My policy has full SageMaker access, which is a lot more permissive than it needs to be. Also roles that are old or inactive can be dangerous as people who no longer work on the project can have access to the resources and can jeopardize the project with malicious intent. Likewise, roles with policies for functions that the project is no longer using can be dangerous because users can use resources to modify other unrelated resources that might affect the success and security of the project. I can make it secure by reducing the scope to my specific endpoint by naming the resource and specifying only read actions on the endpoint. I believe my AWS Workspace is relatively secure but constant monitoring of security policies across EC2, SageMaker, IAM roles, Lambda functions is required to make it as restrictive as possible.

✓ Policy was successfully attached to role.

IAM > Roles > project4-role-tg9xvn7v

project4-role-tg9xvn7v

Delete Edit

Summary

Creation date	ARN
March 20, 2023, 16:04 (UTC+08:00)	arn:aws:iam::775241169951:role/service-role/project4-role-tg9xvn7v
Last activity	Maximum session duration
5 hours ago	1 hour

Permissions Trust relationships Tags Access Advisor Revoke sessions

Permissions policies (2) [Info](#)

You can attach up to 10 managed policies.

<input type="checkbox"/>	Policy name	Type	Description
<input type="checkbox"/>	AWSLambdaBasicExecutionRole-38939ae2-da64-417a-af37-f387d2736ef2	Customer managed	
<input type="checkbox"/>	AmazonSageMakerFullAccess	AWS managed	Provides full access to Amazon SageMa...

Filter policies by property or policy name and press enter.

Simulate Remove Add permissions ▾

< 1 > ⚙

The screenshot shows the AWS IAM 'Roles' section for the role 'project4-role-tg9xvn7v'. A green banner at the top indicates that a policy was successfully attached. The 'Summary' tab is selected, displaying creation date (March 20, 2023), last activity (5 hours ago), ARN, and maximum session duration (1 hour). Below the summary, the 'Permissions' tab is active, showing two managed policies attached: 'AWSLambdaBasicExecutionRole' and 'AmazonSageMakerFullAccess'. The table lists the policy names, types (Customer managed or AWS managed), and descriptions. A search bar and navigation controls are also present.

Step 5:

Concurrency:

I set up a provisioned concurrency of 3 so that my Lambda function can handle that many more calls concurrently.

Code | Test | Monitor | Configuration | Aliases | Versions

General configuration

Triggers

Permissions

Destinations

Function URL

Environment variables

Tags

VPC

Monitoring and operations tools

Concurrency

Asynchronous invocation

Code signing

Concurrency

Function concurrency
Use unreserved account concurrency

Unreserved account concurrency
997

Provisioned concurrency configurations (1)

To enable your function to scale without fluctuations in latency, use provisioned concurrency. You can use Application Auto Scaling to automatically adjust provisioned concurrency to maintain a configured target utilization. Provisioned concurrency runs continually and has separate pricing for concurrency and execution duration. [Learn more](#)

C Edit Remove Add

Find configuration

Qualifier	Type	Provisioned concurrency	Status	Details
1	version	3	Ready	-

Auto-scaling

I set min instance to 1 and max instance to 5 so that it can scale accordingly. I also set the scale-in/out cool down to 60 seconds so that instances can scale in/out faster and it doesn't affect the availability of the endpoint.

Variant automatic scaling [Learn more](#)

Variant name

AllTraffic

Instance type

ml.m5.large

Current instance count

1

Elastic Inference

-

Current weight

1

Minimum instance count

1

Maximum instance count

5

IAM role

Amazon SageMaker uses the following service-linked role for automatic scaling. [Learn more](#)

AWSServiceRoleForApplicationAutoScaling_SageMakerEndpoint

Built-in scaling policy [Learn more](#)

Policy name

SageMakerEndpointInvocationScalingPolicy

Target metric

[SageMakerVariantInvocationsPerInstance](#)

Target value

20

Scale in cool down (seconds) - optional

Scale out cool down (seconds) - optional

Disable scale in