Microsoft Azure

Module 5: Working with Compute

Mohammed Arif

Agenda

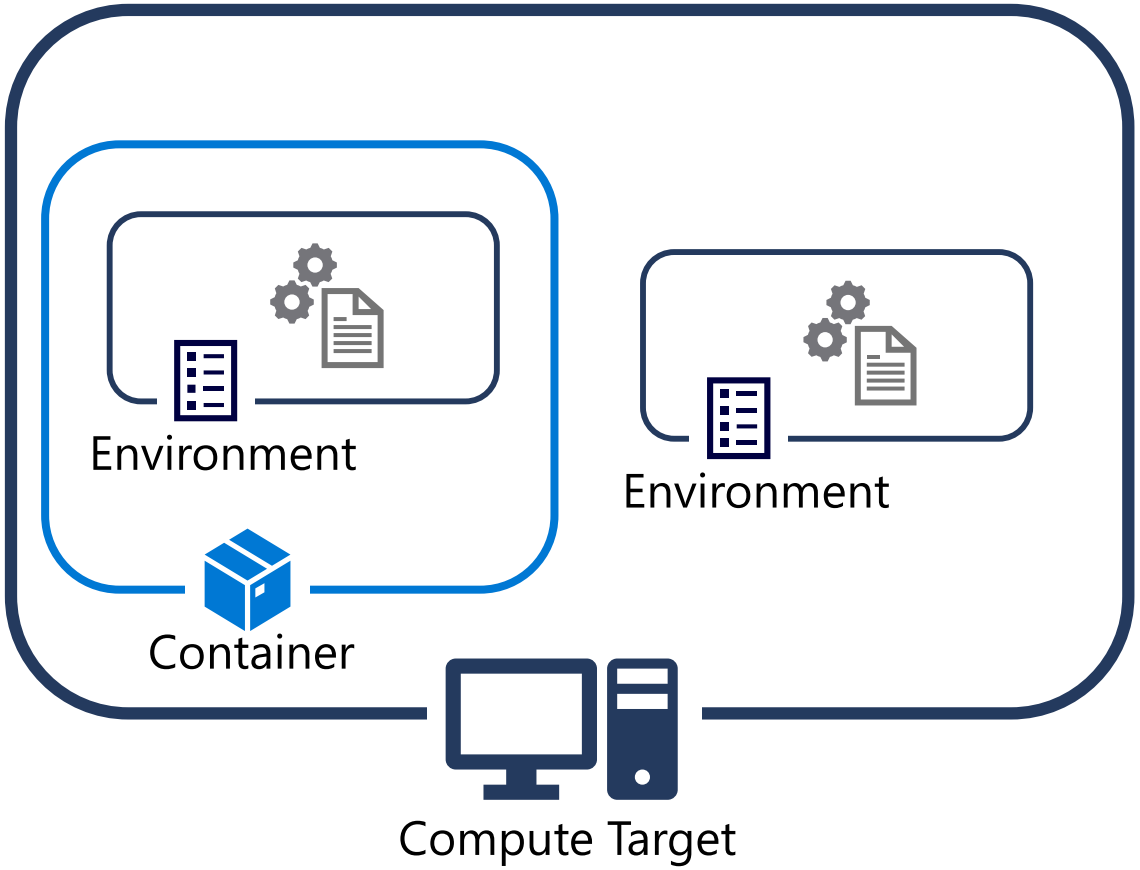
Environments

Compute Targets

Environments

Run Contexts for Experiments

- Python scripts run in a virtual *environment* that defines the Python version and installed packages
- The environment is usually (but not always) in a *container*
- The container (or environment) is hosted on a *compute target*
  - The default in most cases is the *local* compute (where the control code is run)



Explicitly Creating Environments

Create from specification file

```
env = Environment.from_conda_specification(name='training_environment',
                                           file_path='./conda.yml')
```

File in standard YAML format for Conda environments

Create from existing conda environment

```
env = Environment.from_existing_conda_environment(name='training_environment',
                                                  conda_environment_name='py36')
```

Existing conda environment on local compute

Create with specified packages

```
env = Environment('training_environment')
deps = CondaDependencies.create(conda_packages=['scikit-learn','pandas','pip'],
                                pip_packages=['azureml-defaults']
env.python.conda_dependencies = deps
```

Conda package installation is generally more efficient, so use it when possible

Most experiments require azureml-defaults

Use conda to install pip if you plan to also install pip packages

Configuring Environment Containers

Experiments run in Docker by default

```
docker_config = DockerConfiguration(use_docker=True)
script_config = ScriptRunConfig(..., docker_runtime_config=docker_config)
```

Use a DockerConfiguration class to explicitly use Docker (or not!)

Use the *docker* section of the environment to use custom container images

```
env.docker.base_image='my-base-image'
env.docker.base_image_registry='myregistry.azurecr.io/myimage'
```

Override the default base image with your own prebuilt container image…

```
env.docker.base_image = None
env.docker.base_dockerfile = './Dockerfile'
```

…or create one from a dockerfile

Override managed Python configuration

```
env.python.user_managed_dependencies=True
env.python.interpreter_path = '/opt/miniconda/bin/python'
```

If your image already includes Python and packages, manage dependencies yourself

Registering and Reusing Environments

Register an environment in the workspace

```
env.register(workspace=ws)
```

Saves a definition pf the environment in the workspace for later use

View Registered Environments

```
env_names = Environment.list(workspace=ws)
for env_name in env_names:
    print('Name:',env_name)
```

Azure Machine Learning provides a set of useful *curated* environments with names that begin "AzureML..."

Retrieve and use an environment

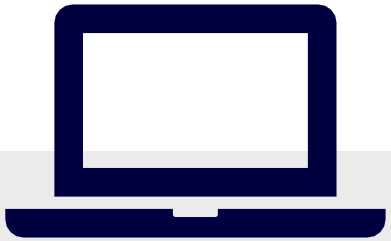
```
training_env = Environment.get(workspace=ws, name='training_environment')

script_config = ScriptRunConfig(source_directory='my_dir',
                                script='script.py',
                                environment=training_env)
```

Enables you to reuse the environment on any compute target

Environment will be created if not already on compute target

Compute Targets

Compute Options for Experiment Runs



Local Compute

- Compute where the control code for the experiment is running
- Often a development workstation or Azure Machine Learning compute instance

Compute Cluster

- Cloud-based cluster managed in an Azure Machine Learning workspace
- Starts, stops, and scales on-demand

Attached Compute

- Azure compute resource outside of a workspace
- For example:
  - Virtual Machine
  - Azure Databricks
  - Azure HDInsight

Creating a Compute Cluster

Create in Azure Machine Learning studio

or

Use the SDK

```
from azureml.core.compute import ComputeTarget, AmlCompute

compute_name = 'aml-cluster'
compute_config = AmlCompute.provisioning_configuration(vm_size='STANDARD_DS11_V2',
                                                        max_nodes=4,
                                                        vm_priority='lowpriority')
aml_compute = ComputeTarget.create(ws, compute_name, compute_config)
aml_compute.wait_for_completion(show_output=True)
```

Specify a suitable Azure VM image
(consider cores, memory, disk, GPU)

Cluster will scale up to this size as required

Low-priority or dedicated
(low-priority can be pre-empted, causing runs to restart; dedicated is more expensive)

Additional options for virtual network and managed identity for access to other Azure resources

Attaching Azure Databricks Compute

Create in Azure Machine Learning studio

or

Use the SDK

```
from azureml.core import Workspace
from azureml.core.compute import ComputeTarget, DatabricksCompute

compute_name = 'db_cluster'

db_workspace_name = 'db_workspace'
db_resource_group = 'db_resource_group'
db_access_token = '1234-abc-5678-defg-90...'
db_config = DatabricksCompute.attach_configuration(resource_group=db_resource_group,
                                                   workspace_name=db_workspace_name,
                                                   access_token=db_access_token)

databricks_compute = ComputeTarget.attach(ws, compute_name, db_config)
databricks_compute.wait_for_completion(True)
```

An existing Azure Databricks workspace in the same Azure subscription as the workspace

Generate a token in the Azure Databricks workspace and specify it here

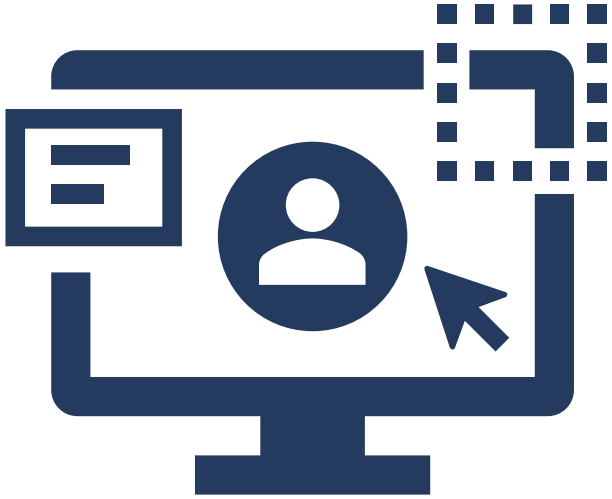
Using Compute Targets

Specify the compute target for an experiment

```
script_config = ScriptRunConfig(source_directory='my_dir',
                                script='script.py',
                                environment=env,
                                compute_target=compute_name)
```

Specify the compute target name or object

Lab: Work with Compute



1. View the lab instructions at https://aka.ms/mslearn-dp100
2. Complete the **Work with compute** exercise

Knowledge check

You need to create an environment from a Conda configuration (.yml) file. Which method of the *Environment* class should you use?

- ❑ create
- ☑ create_from_conda_specification
- ❑ create_from_existing_conda_environment

You need to run a training script on compute that scales on-demand from 0 to 3 GPU-based nodes. Which kind of compute target should you create?

- ❑ Compute Instance
- ☑ Compute Cluster
- ❑ Inference Cluster

Which ScriptRunConfig parameter causes the script to run on a compute cluster named *train-cluster*?

- ❑ arguments=['--AmlCluster', 'train-cluster']
- ❑ environment='train-cluster'
- ☑ compute_target='train-cluster'

References

Microsoft Learn: Work with Compute in Azure Machine Learning
https://docs.microsoft.com/learn/modules/use-compute-contexts-in-aml/

Azure Machine Learning environments documentation
https://docs.microsoft.com/azure/machine-learning/concept-environments

Azure Machine Learning compute targets documentation
https://docs.microsoft.com/azure/machine-learning/concept-compute-target

Microsoft Learn: Perform data science with Azure Databricks
https://docs.microsoft.com/learn/paths/perform-data-science-azure-databricks/

Microsoft Azure