DTI Platform Resource Management Guide

Here, we bring together information regarding how to control your computing resources on the C3 AI Suite.

Jupyter

Access Through Static Console

The Jupyter service is controlled through the 'Jupyter' type. To start the Jupyter service with the default container profile, execute Jupyter.inst().start() within the static console. To stop the Jupyter service execute Jupyter.inst().stop().

To launch Jupyter service with a specific config, use Jupyter.startWithConfig(). This accepts an argument of type JupyterServiceConfig. The JupyterServiceConfig can represent multiple types of services one of which is the 'resourceProfile'. This can represent pre-designated resource profiles of type ResourceProfile. To list the available resource profiles do c3Grid(ResourceProfile.fetch())

By default, two ResourceProfiles exist, 'Basic', and 'BasicGpu'. For example, to launch the Jupyter service using the 'BasicGpu' profile, you would execute

Jupyter.startWithConfig({'resourceProfile': 'BasicGpu'})

Creating new Resource Profile

As a default, tags come with "Basic" and "BasicGpu" profiles. In case these do not fit your needs, it is possible to create a new profile from your static console using:

```javascript
var rp = ResourceProfile.upsert({'id': 'YourProfileName', 'name': 'YourProfileName', 'cpuCount': <cpus>, 'memoryMb': <cpu_memory>, 'diskGb': <disk_memory>, 'gpuCount': <gpus>, 'gpuMemoryMb': <gpu_memory>});
```

In addition to that, effectively use the total memory requested in your resource profile, you will need to change the memory limits that are set at the user level of your Jupyter type. This can be done from the static console with:

Jupyter.setMemoryOverride(<cpu_memory>, ConfigOverride.USER)

where <cpu_memory> matches the field 'memoryMb' of your RP instance.

If you want to make this your default profile whenever launching a Jupyter service, you can do so with:

Jupyter.setDefaultResourceProfile('YourProfileName')

Keep in mind that, currently, the maximum number of GPUs that can be used on the training cluster is one. If you request more than one, your static console is going to be unresponsive for some time, and you won't be able to start your Jupyter service. The same thing will happen if you request resources that surpass the limits of the current underlying hardware infrastructure. These are summarized in the table below:

<table>
<thead>
<tr>
<th>Resource</th>
<th>Max Count</th>
<th>Max Memory (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>vCPUs, no GPU</td>
<td>7</td>
<td>12'1024</td>
</tr>
<tr>
<td>vCPUs, with GPU</td>
<td>5</td>
<td>12'1024</td>
</tr>
<tr>
<td>GPU</td>
<td>1</td>
<td>12'1024</td>
</tr>
</tbody>
</table>

Access Through IDS

Through IDS, clear controls regarding Jupyter are exposed under the 'ML Studio' section for Application. When you've selected a project, you need to start the Jupyter service to access notebooks on that project. In the creation process, you're able to select the container profile to use for your Jupyter service. By default, the 'BasicGpu' profile offers a single K80 GPU.

You can create new container profiles. Select the 'App Settings' Menu, and there's a tab called 'Container Profiles'. You can create new profiles which have different resources like more GPUs, and more vCPUs, and more RAM. Create a new container profile, then when creating the Jupyter service select the new profile to gain access to those resources.