How to Customize Python Environment on HAL

- Background
- Challenges and Solutions
    - How to install a specific package?
    - Users can not and should not install packages in existing python environments.
    - Lists of Existing Anaconda Environments
    - Users need to create their own python environment to install their own packages.
    - Create a New Env from Existing Environment.
    - Create Conda Environment from Scratch.
    - Users should search for all the available packages before installation.
    - Search Packages in All Default Channels.
    - Search Packages in a Specific Channel.
    - Users could install a specific python package to avoid conflict or add functionality.
    - Install a Package with a Specific Version and Build Code.
    - How to solve dependency conflict?
    - The conda has the capability to solve some dependency conflicts, but it cost time and is not guaranteed to work.
    - Users could create a python environment from scratch, then
    - Install the package which caused the conflict first.
    - Install other packages according to the conflicting package requirement.
    - How to use Customized Python Environment in HAL-OnDemand?
- Lists of Existing Anaconda Environments
- Create a New Env from Existing Environments
- Create Conda Environment from Scratch
- Search Packages in All Default Channels
- Search Packages in a Specific Channel
- Install a Package with a Specific Version and Build Code

Background

Miniconda is a free minimal installer for conda. It is a small, bootstrap version of Anaconda that includes only conda, Python, the packages they depend on, and a small number of other useful packages, including pip, zlib and a few others. Use the `conda install` command to install 720+ additional conda packages from the Anaconda repository.

Challenges and Solutions

1. **How to install a specific package?**
    a. **Users can not and should not install packages in existing python environments.**
        i. **Lists of Existing Anaconda Environments**
    b. **Users need to create their own python environment to install their own packages.**
        i. **Create a New Env from Existing Environment.**
        ii. **Create Conda Environment from Scratch.**
    c. **Users should search for all the available packages before installation.**
        i. **Search Packages in All Default Channels.**
        ii. **Search Packages in a Specific Channel.**
    d. **Users could install a specific python package to avoid conflict or add functionality.**
        i. **Install a Package with a Specific Version and Build Code.**
2. **How to solve dependency conflict?**
    a. **The conda has the capability to solve some dependency conflicts, but it cost time and is not guaranteed to work.**
    b. **Users could create a python environment from scratch, then**
        i. **Install the package which caused the conflict first.**
        ii. **Install other packages according to the conflicting package requirement.**
3. **How to use Customized Python Environment in HAL-OnDemand?**
    a. **Users need to install `conda install ipykernel` in order to select and/or switch their own conda environment in HAL-OnDemand.**

Lists of Existing Anaconda Environments

There are currently 6 Conda environments supported on the HAL system

Environment Name	Location	Description

base	/opt/miniconda3	Default conda env with basic python packages.
jupyter	/opt/miniconda3/envs /jupyter	Default conda env with jupyter server. Used for hal-ondemand service only.
deepspeed-v0.3.16	/opt/miniconda3/envs /deepspeed-v0.3.16	DeepSpeed is a deep learning optimization library that makes distributed training easy, efficient, and effective.
fastai-v0.1.18	/opt/miniconda3/envs /fastai-v0.1.18	fastai is a deep learning library that provides practitioners with high-level components that can quickly and easily provide state-of-the-art results in standard deep learning domains.
wmlce-v1.6.2	/opt/miniconda3/envs /wmlce-v1.6.2	Watson Machine Learning Community Edition is an IBM Cognitive Systems offering that is designed for the rapidly growing and quickly evolving AI category of deep learning. This is the only environment still have tensorflow-v1.15
wmlce-v1.7.0	/opt/miniconda3/envs /wmlce-v1.7.0	Watson Machine Learning Community Edition is an IBM Cognitive Systems offering that is designed for the rapidly growing and quickly evolving AI category of deep learning. This is the last version of WMLCE, the following module is opence.
opence-v1.0.0	/opt/miniconda3/envs /opence-v1.0.0	Open-CE is a community-driven software distribution for machine learning that runs on standard Linux platforms with NVIDIA GPU technologies.
opence-v1.1.2	/opt/miniconda3/envs /opence-v1.1.2	Open-CE is a community-driven software distribution for machine learning that runs on standard Linux platforms with NVIDIA GPU technologies.
opence-v1.2.2	/opt/miniconda3/envs /opence-v1.2.2	Open-CE is a community-driven software distribution for machine learning that runs on standard Linux platforms with NVIDIA GPU technologies.
opence-v1.3.1	/opt/miniconda3/envs /opence-v1.3.1	Open-CE is a community-driven software distribution for machine learning that runs on standard Linux platforms with NVIDIA GPU technologies.
opence-v1.4.1	/opt/miniconda3/envs /opence-v1.4.1	Open-CE is a community-driven software distribution for machine learning that runs on standard Linux platforms with NVIDIA GPU technologies.
opence-v1.5.1 opence-v1.5.2	/opt/miniconda3/envs /opence-v1.5.1 /opt/miniconda3/envs /opence-v1.5.2	Open-CE is a community-driven software distribution for machine learning that runs on standard Linux platforms with NVIDIA GPU technologies. This is the default python environment. Use opence-v1.5.1 as base, fixed openblas issue caused by openmp.
theano-v1.0.4	/opt/miniconda3/envs /theano-v1.0.4	Open-CE is a community-driven software distribution for machine learning that runs on standard Linux platforms with NVIDIA GPU technologies.
rapids-v0.11.0	/opt/miniconda3/envs /rapids	The RAPIDS suite of software libraries, built on CUDA-X AI, gives you the freedom to execute end-to-end data science and analytics pipelines entirely on GPUs.

Create a New Env from Existing Environments

We recommend our users to create a new environment from one of our existing **opence** environment.

Create a New Env from Existing Env

```
conda create --name=new_env --clone=opence-v1.5.1
```

The new Conda environment will be located within **$HOME/.conda/envs/new_env**, then users can search and/or install python packages via Conda

Example: Search for a New Package

```
conda search tensorflow
```

Create Conda Environment from Scratch

Users can also create a new environment from scratch

Create a New Env from Existing Env

```
conda create --name=new_env_name
```

Search Packages in All Default Channels

```
conda search openblas
```

Search Packages in a Specific Channel

```
conda search openblas -c conda-forge
```

Install a Package with a Specific Version and Build Code

Install a package from the result of the search

```
(new_env_name)[user_id@hal-login2 ~]# conda search openblas
...
openblas                      0.3.12      pthreads_hca0ad1f_0  conda-forge
openblas                      0.3.12      pthreads_hca0ad1f_1  conda-forge
openblas                      0.3.13      h6ffa863_0           pkgs/main
openblas                      0.3.13      h6ffa863_1           pkgs/main
openblas                      0.3.13      openmp_h25a920f_0    conda-forge
openblas                      0.3.13      pthreads_h92053e5_0  conda-forge
...
(new_env_name)[user_id@hal-login2 ~]# conda install openblas=0.3.13=openmp_h25a920f_0
```