G100 - Scientific Python user environment and tools for AI: the CINECA Artificial Intelligence project

The bulk of the cineca-ai package, provided by the deeplrn profile, is based on the Open Cognitive Environment (Open-CE) tool, which includes (for example) Tensorflow, Pytorch, XGBoost, and other related packages and dependencies.

This cognitive environment has been personalised by CINECA AI experts and will be published in a public channel. You can find several versions of the cineca-ai module in profile/deeplrn, differing in the versions of their main components (pytorch, tensorflow etc.). The module help reports the versioning for these components. For a complete list load the module and launch the "pip list" command.

The CINECA AI project can be used in several ways, depending on the method more suited to your needs and on the availability of conda/pip packages.

1. **Loading cineca-ai module**

The first way to use the cineca-ai environment goes through the loading of the module:

```
module load profile/deeplrn
module load autoload cineca-ai/<version>
# see all available packages
pip list
# enjoy the environment
```

If you prefer working in a conda environment, you can activate the cineca-ai env via "conda activate $CINECA_AI_ENV", and explore the environment with "conda list".

If you need to use a package not included in the list of those provided by the cineca-ai modules, you can always rely on the cineca-ai environment for the dependencies and install what you need within a personal virtual environment and/or a conda environment.

1.1 install additional packages within a virtual environment

If your package is available to pip:

```
module load profile/deeplrn
module load autoload cineca-ai/<version>
python -m venv <myvenv> --system-site-packages
source <myvenv>/bin/activate
pip install PACKAGE
```

NOTES:

- <myvenv>: choose an arbitrary, up-to-you name for your personal virtual env
- the *--system-site-packages* flag gives the virtual environment access to the system site-packages directory (otherwise you cannot access the cineca-ai environment)
- it is advised to create your personal envs in your $WORK area, since the $HOME disk quota is limited to 50 GB
- to test the installation launch: python -c "import PACKAGE"
- to use the installed PACKAGE, just *source* your env (source <myvenv>/bin/activate): you will access your packages AND those of the cineca-ai environment, no need to load the cineca-ai module
- if PACKAGE is not available to pip, see Section 2.2 and 2.3

Example: torch-scatter

It requires torch, which will be taken from the cineca-ai env (i.e., no need to install it):

```
$ module load profile/deeplrn
$ module load autoload cineca-ai/2.1.0
$ python -m venv torch_venv  --system-site-packages
$ source torch_venv/bin/activate
$ pip install torch-scatter
```

2. **create a personal conda environment with selected packages from CINECA channel**

```
module load autoload cineca-ai/<version>
conda create --prefix <mycondaenv> -y
conda activate <mycondaenv>
# see all packages in the channel
conda search -c $CINECA_AI_CHANNEL --override-channels
# install a package from the channel
conda install -c $CINECA_AI_CHANNEL PACKAGE
```

2.1 install additional packages within a virtual environment

```
# after activating <mycondaenv> and installed the needed prerequisites of PACKAGE from CINECA channel
python -m venv <myvenv> --system-site-packages
source <myvenv>/bin/activate
pip install PACKAGE
```

2.2 install additional packages within a conda environment

```
# after activating <mycondaenv> and installed the needed prerequisites of PACKAGE from CINECA channel, install
the additional packages
conda install PACKAGE1
conda install -c conda-forge PACKAGE2
```

NOTES:

- for some packages you need to explicit the external channel to use, e.g. conda-forge

2.3 install additional packages within a conda and a virtual environment

```
# after activating <mycondaenv> and installed the needed prerequisites of PACKAGE from CINECA channel, install
the additional conda packages
conda install PACKAGE
# create a virtual env and install the additional pip packages
python -m venv <myvenv> --system-site-packages
source <myvenv>/bin/activate
pip install PACKAGE
```

NOTES:

- <mycondaenv>: choose an arbitrary, up-to-you name for your personal conda env
- install the selected packages from the $CINECA_AI_CHANNEL
- install the additional packages (specifying specific external channels if needed, e.g. conda-forge)
- <myvenv>: choose an arbitrary, up-to-you name for your personale conda env
- the *--system-site-packages* flag gives the virtual environment access to the system site-packages directory (otherwise you cannot access the cineca-ai environment)
- it is advised to create both personal envs in your $WORK area, since the $HOME disk quota is limited to 50 GB
- to test the installation launch: python -c "import PACKAGE"
- to use the installed PACKAGE, just *source* your env (source <myvenv>/bin/activate): you will access your packages AND those installed in your conda environment, no need to load the cineca-ai module and to activate <mycondaenv>

Example: netket

It requires some of the conda packages provided by the cineca-ai environment, and netket is only available via pip.

```
$ module load autoload cineca-ai/2.1.0
$ conda create --prefix <mycondaenv> -y
$ conda activate <mycondaenv>
$ conda install -c $CINECA_AI_CHANNEL mpi4py jaxlib jax dm-tree mpi4jax
$ conda install cmake h5py -y
$ conda install -c conda-forge llvmlite=0.38.0 orjson -y
$ python -m venv <myvenv> --system-site-packages
$ source <myvenv>/bin/activate
$ pip install 'git+https://github.com/netket/netket.git#egg=netket[all]'
```

3. **clone the CINECA channel in your personal channel**

```
conda create --prefix <mycondaenv> --clone /cineca/prod/opt/libraries/cineca-ai/<version>/none/cineca-ai-conda-
env-py3.8-cuda-openmpi-11.0 -y
conda activate mycondaenv
# see all packages in the channel
conda search -c <mycondaenv> --override-channels
# install a package from your personal channel
conda install -c <mycondaenv> PACKAGE
```