Creating SSH keygen for enabling SSH cloning of Github repos

Git page recommends creating a 4096b RSA key but we create Ed25519 which is roughly equivalent to 3072b RSA

- ssh-keygen -t ed25519 -C "Mila"

- ```
  cat ~/.ssh/id_ed25519.pub >> ~/.ssh/authorized_keys
  chmod 600 ~/.ssh/authorized_keys
  cat ~/.ssh/authorized_keys
  ```

- Copy that string id_ed25519.pub to your Github account under settings -> SSH keys

PS: This also facilitates inter-node logins as we've added the key to authorized_keys.

Instructions for building Hanabi SAD on Mila cluster (see below for Beluga)

- First, create a conda env by following instructions in Hanabi SAD repo excluding the CPATH, LIBRARY_PATH, LD_LIBRARY_PATH

  ```
  # create new conda env
  conda create -n hanabi_SAD python=3.7
  conda activate hanabi_SAD
  ```

- Activate the hanabi_SAD environment.

  ```
  # install pytorch
  pip install torch==1.5.1+cu101 torchvision==0.6.1+cu101 -f
  https://download.pytorch.org/whl/torch_stable.html

  # install other dependencies
  pip install numpy (if installing pytorch doesn't install numpy)
  pip install psutil
  pip install wandb
  pip install pandas
  ```

- module purge
  module load anaconda/3 cuda/10.1/nccl/2.4 python/3.7/cuda/10.1/cudnn/7.6/pytorch/1.5.1
  export CUDNN_LIBRARY="/cvmfs/ai.mila.quebec/apps/x86_64/common/cudnn/10.1-v7.6"

  git clone --recursive git@github.com:chandar-lab/CMAL_Hanabi.git -b dev_akilesh
  ```

- Remove *-march=native* in CMakeLists.txt or use the CMakeLists.txt in https://github.com/akileshbadrinaaraayanan/hanabi_sad_mods
  (OR)
  UPDATE: The latest CMakeLists.txt in CMAL_Hanabi is up to date, so one can use it directly.

- salloc --mem=16G -c 2 --time=03:00:00 -w apollor15 (this was the node used for building).
- mkdir build

  cd build

  cmake .. (if I requested for a GPU in the previous step I should do

  CUDA_VISIBLE_DEVICES="" cmake ..)

  make

  mv hanalearn.cpython-37m-x86_64-linux-gnu.so ..

  mv rela/rela.cpython-37m-x86_64-linux-gnu.so ..

  mv hanabi-learning-environment/libpyhanabi.so ../hanabi-learning-environment/

- export PYTHONPATH=$PYTHONPATH:/path/to/hanabi_SAD

Once the building is done and the .so's are moved to their required places. Every time you now login to a compute node, you just need to

- module load anaconda/3
- source activate /home/mila/b/badrinaa/anaconda3/envs/hanabi_SAD
- export PYTHONPATH=/home/mila/b/badrinaa/hanabi_SAD:$PYTHONPATH
- export OMP_NUM_THREADS=1

Instructions for building Hanabi SAD on Beluga cluster

 # Module loads

module load gcc/7.3.0

module load cuda/10.1

module load cudnn/7.6.5

module load cmake/3.16.3

 # create new conda env (instructions from Hanabi SAD repo)

conda create -n hanabi_SAD python=3.7

conda activate hanabi_SAD

 # install pytorch

pip install torch==1.5.1+cu101 torchvision==0.6.1+cu101 -f

https://download.pytorch.org/whl/torch_stable.html

 # install other dependencies

pip install numpy (if it is not already installed when installing pytorch).

pip install psutil

pip install wandb

pip install pandas

- Remove *-march=native* in CMakeLists.txt or use the CMakeLists.txt in https://github.com/akileshbadrinaaraayanan/hanabi_sad_mods
  (OR)

UPDATE: The latest CMakeLists.txt in CMAL_Hanabi is up to date, so one can use it directly.

Clone the repo

git clone --recursive git@github.com:chandar-lab/CMAL_Hanabi.git -b dev_akilesh

 # Build Hanabi

- mkdir build
- cd build
- cmake .. -DPYTHON_INCLUDE_DIR=$(python -c "from distutils.sysconfig import get_python_inc; print(get_python_inc())") -DPYTHON_LIBRARY=$(python -c "import distutils.sysconfig as sysconfig; print(sysconfig.get_config_var('LIBDIR'))") -DPYTHON_EXECUTABLE:FILEPATH=`which python` -DCUDNN_INCLUDE_PATH="$EBROOTCUDNN/include" -DCUDNN_LIBRARY_PATH="$EBROOTCUDNN/lib64/libcudnn.so"

Before actually beginning the build, patch Makefiles.

- sed -i '\|.cpython-.*m-x86_64-linux-gnu.so: /usr/local/cuda/lib64/|d' CMakeFiles/hanalearn.dir/build.make rela/CMakeFiles/rela.dir/build.make

- make

- mv hanalearn.cpython-37m-x86_64-linux-gnu.so ..
- mv rela/rela.cpython-37m-x86_64-linux-gnu.so ..
- mv hanabi-learning-environment/libpyhanabi.so ../hanabi-learning-environment/

Note:

- Sample output after cmake step on Beluga is as below:

```
(hanabi_SAD) [akb@beluga1 build]$ cmake .. -DPYTHON_INCLUDE_DIR=$(python -c "from
distutils.sysconfig import get_python_inc; print(get_python_inc())")
-DPYTHON_LIBRARY=$(python -c "import distutils.sysconfig as sysconfig;
print(sysconfig.get_config_var('LIBDIR'))") -DPYTHON_EXECUTABLE:FILEPATH=`which
python` -DCUDNN_INCLUDE_PATH="$EBROOTCUDNN/include"
-DCUDNN_LIBRARY_PATH="$EBROOTCUDNN/lib64/libcudnn.so"
-- The C compiler identification is GNU 7.3.0
-- The CXX compiler identification is GNU 7.3.0
-- Check for working C compiler: /cvmfs/soft.computecanada.ca/custom/bin/cc
-- Check for working C compiler: /cvmfs/soft.computecanada.ca/custom/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler:
/cvmfs/soft.computecanada.ca/nix/var/nix/profiles/gcc-7.3.0/bin/c++
-- Check for working CXX compiler:
/cvmfs/soft.computecanada.ca/nix/var/nix/profiles/gcc-7.3.0/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Found PythonInterp: /home/akb/anaconda3/envs/hanabi_SAD/bin/python (found suitable
version "3.7.9", minimum required is "3.7")
-- Found PythonLibs: /home/akb/anaconda3/envs/hanabi_SAD/lib (found suitable version
"3.7.9", minimum required is "3.7")
-- Looking for pthread.h
```

```
-- Looking for pthread.h - found
-- Performing Test CMAKE_HAVE_LIBC_PTHREAD
-- Performing Test CMAKE_HAVE_LIBC_PTHREAD - Failed
-- Looking for pthread_create in pthreads
-- Looking for pthread_create in pthreads - not found
-- Looking for pthread_create in pthread
-- Looking for pthread_create in pthread - found
-- Found Threads: TRUE
CMake Warning (dev) at
/home/akb/anaconda3/envs/hanabi_SAD/lib/python3.7/site-packages/torch/share/cmake/Caff
e2/public/cuda.cmake:29 (find_package):
 Policy CMP0074 is not set: find_package uses <PackageName>_ROOT variables.
 Run "cmake --help-policy CMP0074" for policy details.  Use the cmake_policy
 command to set the policy and suppress this warning.
 Environment variable CUDA_ROOT is set to:
   /cvmfs/soft.computecanada.ca/easybuild/software/2017/Core/cudacore/10.1.243
 For compatibility, CMake is ignoring the variable.
Call Stack (most recent call first):

/home/akb/anaconda3/envs/hanabi_SAD/lib/python3.7/site-packages/torch/share/cmake/Caff
e2/Caffe2Config.cmake:88 (include)

/home/akb/anaconda3/envs/hanabi_SAD/lib/python3.7/site-packages/torch/share/cmake/Torc
h/TorchConfig.cmake:40 (find_package)
 rela/CMakeLists.txt:20 (find_package)
This warning is for project developers.  Use -Wno-dev to suppress it.
-- Found CUDA:
/cvmfs/soft.computecanada.ca/easybuild/software/2017/Core/cudacore/10.1.243 (found
version "10.1")
-- Caffe2: CUDA detected: 10.1
-- Caffe2: CUDA nvcc is:
/cvmfs/soft.computecanada.ca/easybuild/software/2017/Core/cudacore/10.1.243/bin/nvcc
-- Caffe2: CUDA toolkit directory:
/cvmfs/soft.computecanada.ca/easybuild/software/2017/Core/cudacore/10.1.243
-- Caffe2: Header version is: 10.1
```

```
-- Found CUDNN:
/cvmfs/soft.computecanada.ca/easybuild/software/2017/CUDA/cuda10.1/cudnn/7.6.5/lib64/libcudnn.so
-- Found cuDNN: v7.6.5  (include:
/cvmfs/soft.computecanada.ca/easybuild/software/2017/CUDA/cuda10.1/cudnn/7.6.5/include, library:
/cvmfs/soft.computecanada.ca/easybuild/software/2017/CUDA/cuda10.1/cudnn/7.6.5/lib64/libcudnn.so)
-- Automatic GPU detection failed. Building for common architectures.
-- Autodetected CUDA architecture(s): 3.5;5.0;5.2;6.0;6.1;7.0;7.0+PTX;7.5;7.5+PTX
-- Added CUDA NVCC flags for:
-gencode;arch=compute_35,code=sm_35;-gencode;arch=compute_50,code=sm_50;-gencode;arch=compute_52,code=sm_52;-gencode;arch=compute_60,code=sm_60;-gencode;arch=compute_61,code=sm_61;-gencode;arch=compute_70,code=sm_70;-gencode;arch=compute_75,code=sm_75;-gencode;arch=compute_70,code=compute_70;-gencode;arch=compute_75,code=compute_75
-- Found torch:
/home/akb/anaconda3/envs/hanabi_SAD/lib/python3.7/site-packages/torch/lib/libtorch.so
-- Found PythonInterp: /home/akb/anaconda3/envs/hanabi_SAD/bin/python (found version
"3.7.9")
-- Found PythonLibs: /home/akb/anaconda3/envs/hanabi_SAD/lib
-- pybind11 v2.3.dev1
-- Performing Test HAS_FLTO
-- Performing Test HAS_FLTO - Success
-- LTO enabled
CMake Warning (dev) at
/home/akb/anaconda3/envs/hanabi_SAD/lib/python3.7/site-packages/torch/share/cmake/Caffe2/public/cuda.cmake:29 (find_package):
 Policy CMP0074 is not set: find_package uses <PackageName>_ROOT variables.
 Run "cmake --help-policy CMP0074" for policy details.  Use the cmake_policy
 command to set the policy and suppress this warning.
 Environment variable CUDA_ROOT is set to:
   /cvmfs/soft.computecanada.ca/easybuild/software/2017/Core/cudacore/10.1.243
 For compatibility, CMake is ignoring the variable.
Call Stack (most recent call first):

/home/akb/anaconda3/envs/hanabi_SAD/lib/python3.7/site-packages/torch/share/cmake/Caffe2/Caffe2Config.cmake:88 (include)
```

```
/home/akb/anaconda3/envs/hanabi_SAD/lib/python3.7/site-packages/torch/share/cmake/Torch/TorchConfig.cmake:40 (find_package)
 CMakeLists.txt:12 (find_package)
This warning is for project developers.  Use -Wno-dev to suppress it.
-- Caffe2: CUDA detected: 10.1
-- Caffe2: CUDA nvcc is: /cvmfs/soft.computecanada.ca/easybuild/software/2017/Core/cudacore/10.1.243/bin/nvcc
-- Caffe2: CUDA toolkit directory: /cvmfs/soft.computecanada.ca/easybuild/software/2017/Core/cudacore/10.1.243
-- Caffe2: Header version is: 10.1
-- Found cuDNN: v7.6.5  (include: /cvmfs/soft.computecanada.ca/easybuild/software/2017/CUDA/cuda10.1/cudnn/7.6.5/include, library: /cvmfs/soft.computecanada.ca/easybuild/software/2017/CUDA/cuda10.1/cudnn/7.6.5/lib64/libcudnn.so)
-- Automatic GPU detection failed. Building for common architectures.
-- Autodetected CUDA architecture(s): 3.5;5.0;5.2;6.0;6.1;7.0;7.0+PTX;7.5;7.5+PTX
-- Added CUDA NVCC flags for: -gencode;arch=compute_35,code=sm_35;-gencode;arch=compute_50,code=sm_50;-gencode;arch=compute_52,code=sm_52;-gencode;arch=compute_60,code=sm_60;-gencode;arch=compute_61,code=sm_61;-gencode;arch=compute_70,code=sm_70;-gencode;arch=compute_75,code=sm_75;-gencode;arch=compute_70,code=compute_70;-gencode;arch=compute_75,code=compute_75
-- Configuring done
-- Generating done
-- Build files have been written to: /home/akb/hanabi_SAD/build
```

- Sample output after make step on Beluga is as below:

```
(hanabi_SAD) [akb@beluga1 build]$ make
Scanning dependencies of target hanabi
[  5%] Building CXX object hanabi-learning-environment/hanabi_lib/CMakeFiles/hanabi.dir/hanabi_card.cc.o
[ 10%] Building CXX object hanabi-learning-environment/hanabi_lib/CMakeFiles/hanabi.dir/hanabi_game.cc.o
[ 15%] Building CXX object hanabi-learning-environment/hanabi_lib/CMakeFiles/hanabi.dir/hanabi_hand.cc.o
```

```
[ 20%] Building CXX object
hanabi-learning-environment/hanabi_lib/CMakeFiles/hanabi.dir/hanabi_history_item.cc.o
[ 25%] Building CXX object
hanabi-learning-environment/hanabi_lib/CMakeFiles/hanabi.dir/hanabi_move.cc.o
[ 30%] Building CXX object
hanabi-learning-environment/hanabi_lib/CMakeFiles/hanabi.dir/hanabi_observation.cc.o
[ 35%] Building CXX object
hanabi-learning-environment/hanabi_lib/CMakeFiles/hanabi.dir/hanabi_state.cc.o
[ 40%] Building CXX object
hanabi-learning-environment/hanabi_lib/CMakeFiles/hanabi.dir/util.cc.o
[ 45%] Building CXX object
hanabi-learning-environment/hanabi_lib/CMakeFiles/hanabi.dir/canonical_encoders.cc.o
[ 50%] Linking CXX static library libhanabi.a
[ 50%] Built target hanabi
Scanning dependencies of target rela
[ 55%] Building CXX object rela/CMakeFiles/rela.dir/transition.cc.o
[ 60%] Building CXX object rela/CMakeFiles/rela.dir/pybind.cc.o
[ 65%] Linking CXX shared library rela.cpython-37m-x86_64-linux-gnu.so
[ 65%] Built target rela
Scanning dependencies of target hanalearn
[ 70%] Building CXX object CMakeFiles/hanalearn.dir/cpp/hanabi_env.cc.o
[ 75%] Building CXX object CMakeFiles/hanalearn.dir/cpp/pybind.cc.o
[ 80%] Linking CXX shared module hanalearn.cpython-37m-x86_64-linux-gnu.so
[ 80%] Built target hanalearn
Scanning dependencies of target pyhanabi
[ 85%] Building CXX object
hanabi-learning-environment/CMakeFiles/pyhanabi.dir/pyhanabi.cc.o
[ 90%] Linking CXX shared library libpyhanabi.so
[ 90%] Built target pyhanabi
Scanning dependencies of target game_example
[ 95%] Building CXX object
hanabi-learning-environment/CMakeFiles/game_example.dir/game_example.cc.o
[100%] Linking CXX executable game_example
[100%] Built target game_example
```

Instructions for building Hanabi SAD on Cedar cluster

 # Module loads

```
module load gcc/7.3.0
module load cuda/10.1
module load cudnn/7.6.5
module load cmake/3.16.3
module load scipy-stack/2019b
```

 # Create virtualenv and activate

```
conda create -n hanabi_SAD python=3.7

conda activate hanabi_SAD
```

 # Install dependencies

```
pip install torch==1.5.1+cu101 torchvision==0.6.1+cu101 -f
https://download.pytorch.org/whl/torch_stable.html

pip install --no-index psutil

pip install wandb

pip install pandas
```

 # Clone repo recursively

git clone --recursive git@github.com:chandar-lab/CMAL_Hanabi.git -b dev_akilesh

- Remove *-march=native* in CMakeLists.txt or use the CMakeLists.txt in https://github.com/akileshbadrinaaraayanan/hanabi_sad_mods
  (OR)
  UPDATE: The latest CMakeLists.txt in CMAL_Hanabi is up to date, so one can use it directly.

#
 # Build with CMake
#

- cd hanabi_SAD
- mkdir build
- cd build
- cmake .. -DPYTHON_INCLUDE_DIR=$(python -c "from distutils.sysconfig import get_python_inc; print(get_python_inc())")
  -DPYTHON_LIBRARY=$(python -c "import distutils.sysconfig as sysconfig; print(sysconfig.get_config_var('LIBDIR'))")
  -DPYTHON_EXECUTABLE:FILEPATH=`which python`
  -DCUDNN_INCLUDE_PATH="$EBROOTCUDNN/include"
  -DCUDNN_LIBRARY_PATH="$EBROOTCUDNN/lib64/libcudnn.so"

Before actually beginning the build, patch Makefiles.

- `sed -i '\|.cpython-.*m-x86_64-linux-gnu.so: /usr/local/cuda/lib64/|d' CMakeFiles/hanalearn.dir/build.make rela/CMakeFiles/rela.dir/build.make`
- `make`
- `mv hanalearn.cpython-37m-x86_64-linux-gnu.so ..`
- `mv rela/rela.cpython-37m-x86_64-linux-gnu.so ..`
- `mv hanabi-learning-environment/libpyhanabi.so ../hanabi-learning-environment/`
- `cd ..`
- `export PYTHONPATH=$PYTHONPATH:`pwd``

```
# Run Python
# python
# Python 3.7.9 (default, Jul 18 2019, 19:34:02)
# [GCC 5.4.0] on linux
# Type "help", "copyright", "credits" or "license" for more information.
# >>> import torch
# >>> import rela
# >>> import hanalearn
# >>> quit()
```