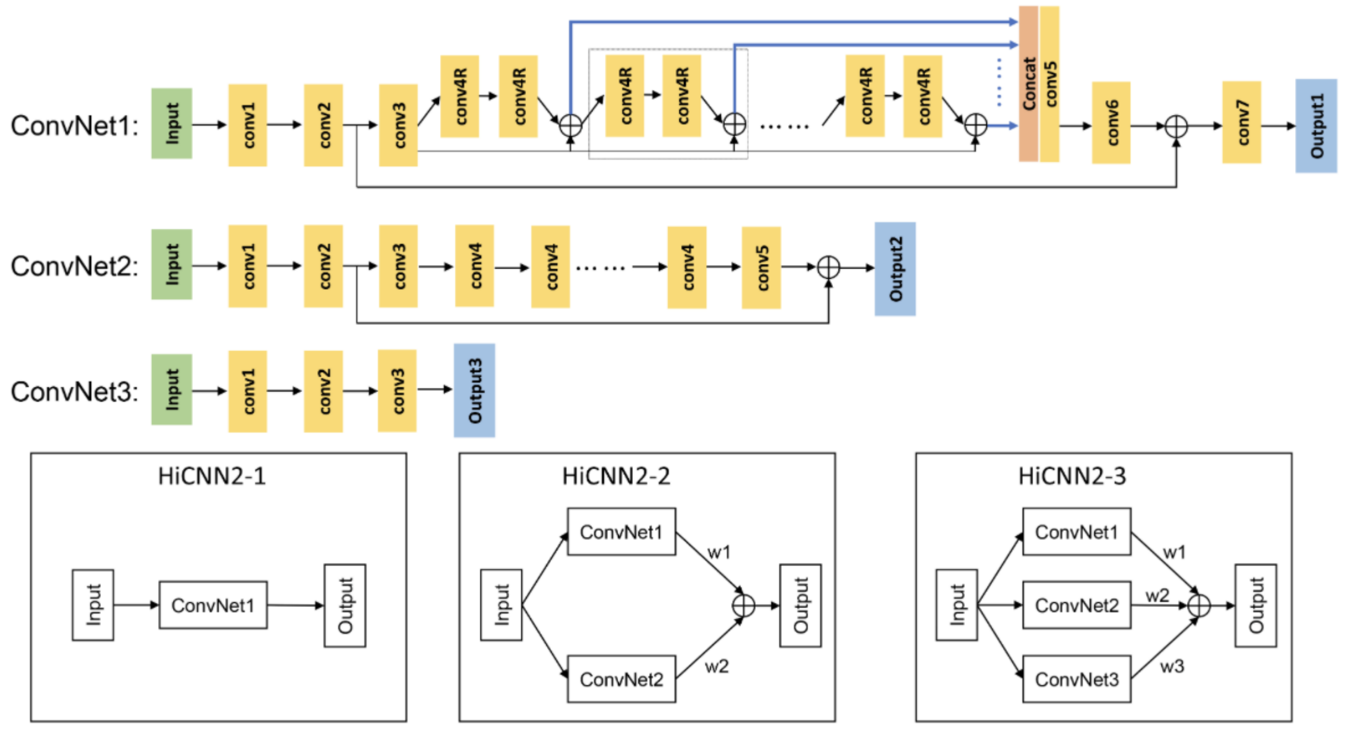
Implementation

Although I've learned a lot about HiCNN, they came out with an updated and improved HiCNN2



HiCNN2 is an improved version of our previously developed tool HiCNN for enhancing resolution of Hi-C data and uses three architectures to learn the mapping between low-resolution and high-resolution Hi-C contact matrices. HiCNN2-1 uses one convolutional neural network (ConvNet1); HiCNN2-2 consists of an ensemble of two different ConvNets (ConvNet1 and ConvNet2); HiCNN2-3 uses an ensemble of three different ConvNets (ConvNet1, ConvNet2, and ConvNet3)

Set Up

The requirements for this project are:
PyCharm
PyTorch
Numpy

Anaconda and CUDA are optional but can help with organization and GPU speed, but is not supported by MacOS.

Materials and Methods

Their `readme` section outlines how to use this this on the HIC071 Hi-C sample.

To download the "HIC071" that was used, in terminal:

```
curl -O
ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM1551nnn/GSM1551620/suppl/GSM155162
0_HIC071_merged_nodups.txt.gz
```

Extract the file and end up with a file named: `GSM1551620_HIC071_merged_nodups.txt`

Generate a Hi-C read-pair file for one chromosome. Here I'll be using chromosome 15:

```
python get_chr_reads.py GSM1551620_HIC071_merged_nodups.txt 15 chr15.reads
```

- "15", the second argument, is the chromosome ID of interest.
- The Hi-C read-pair file for chromosome 15 `chr15.reads` can be found in the "data" folder.

Generate the input of `HiCNN2_predict` using a python script:

```
python get_HiCNN2_input.py chr15.reads 102531392 10000 chr15.subMats
chr15.index
```

- "chr15.reads" is the output file of step (3);
- "102531392" is the length of chromosome 15;
- "10000" is the resolution of interest;
- "chr15.subMats" is the output submatrix file with shape (n140*40);
- "chr15.index" is the output index file with shape (n*2) for us to rebuild the whole Hi-C matrix after running HiCNN2_predict.

Make sure to install torch:

```
pip install torch
```

Run `HiCNN2_predict`:

```
python HiCNN2_predict.py -f1 data/chr15.subMats.npy -f2
data/chr15.subMats_HiCNN23_16 -mid 3 -m checkpoint/model_HiCNN23_16.pt -r 16
```

```
python HiCNN2_predict.py -f1 data/chr15.subMats.npy -f2
data/chr15.subMats_HiCNN21_16 -mid 1 -m checkpoint/model_HiCNN21_16.pt -r 16
```

- "-f1" is followed by the input file generated in step (4).
- "-f2" is followed by the output file.
- "-mid 3" means that we are using HiCNN2-3.
- "-m" indicates the best model we want to use. We provide 6 checkpoint files in the "checkpoint" folder. The checkpoint files are named with the format "model*HiCNN2**#.pt", where "*" may be 1/2/3 representing the three architectures and "#" may be 8/16/25 representing the three down sampling ratios (1/8, 1/16, and 1/25).

Because I'm using my MacOS system, I had to modify `HiCNN2_predict.py`:

Replace this line:
`Net.load_state_dict(torch.load(args.file_best_model))`

With this:
`Net.load_state_dict(torch.load(args.file_best_model, map_location='cpu'))`

Now combine predicted sub-matrices to get a big predicted high-resolution Hi-C matrix for one chromosome.

```
python combine_subMats.py data/chr15.subMats_HiCNN23_16.npy
data/chr15.index.npy 102531392 10000 data/chr15.predictedMat
```

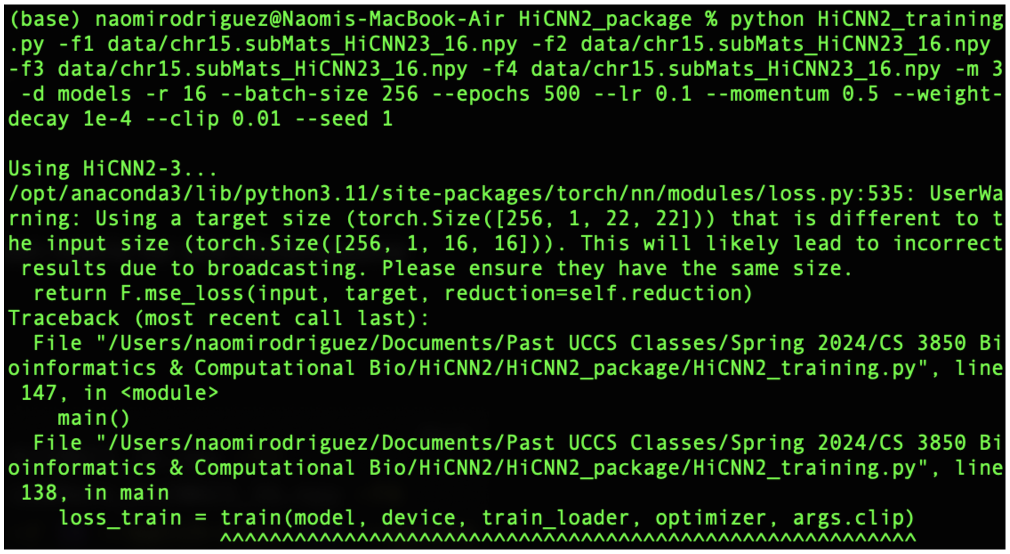
- "data/chr15.subMats_HiCNN23_16.npy" is from step (5).
- "data/chr15.index.npy" is from step(4).
- "102531392" is the chromosome length.
- "10000" is the resolution.
- "data/chr15.predictedMat" is the predicted high-resolution matrix for one chromosome (chr15).

Training

Now to train we can do the following:

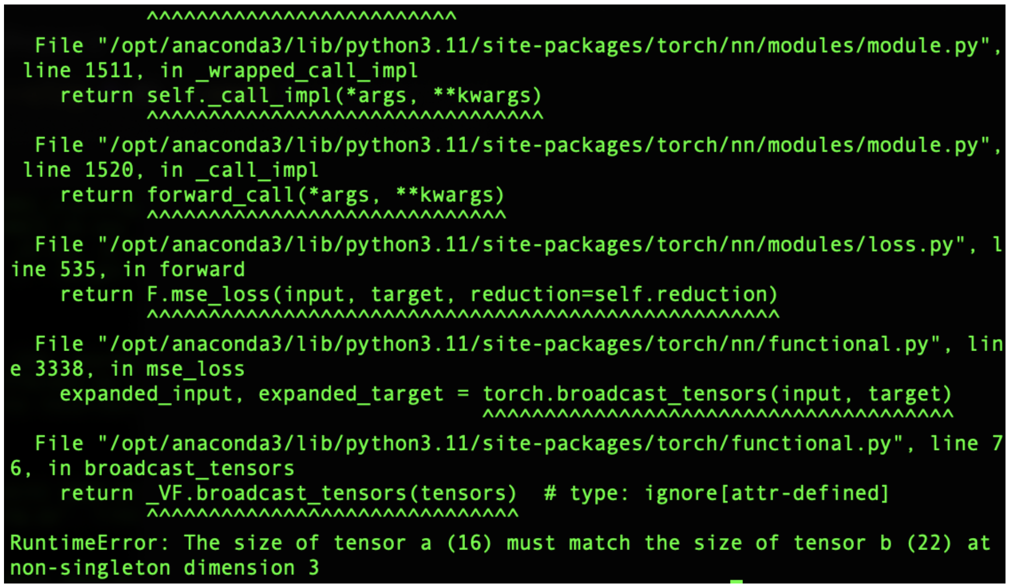
```
python HiCNN2_training.py -f1 data/chr15.subMats_HiCNN23_16.npy -f2
data/chr15.subMats_HiCNN23_16.npy -f3 data/chr15.subMats_HiCNN23_16.npy -f4
```

```
data/chr15.subMats_HiCNN23_16.npy -m 3 -d models -r 16 --batch-size 256 --
epochs 500 --lr 0.1 --momentum 0.5 --weight-decay 1e-4 --clip 0.01 --seed 1
```



```
(base) naomirodriguez@Naomis-MacBook-Air HiCNN2_package % python HiCNN2_training
.py -f1 data/chr15.subMats_HiCNN23_16.npy -f2 data/chr15.subMats_HiCNN23_16.npy
-f3 data/chr15.subMats_HiCNN23_16.npy -f4 data/chr15.subMats_HiCNN23_16.npy -m 3
 -d models -r 16 --batch-size 256 --epochs 500 --lr 0.1 --momentum 0.5 --weight-
decay 1e-4 --clip 0.01 --seed 1

Using HiCNN2-3...
/opt/anaconda3/lib/python3.11/site-packages/torch/nn/modules/loss.py:535: UserWa
rning: Using a target size (torch.Size([256, 1, 22, 22])) that is different to t
he input size (torch.Size([256, 1, 16, 16])). This will likely lead to incorrect
 results due to broadcasting. Please ensure they have the same size.
  return F.mse_loss(input, target, reduction=self.reduction)
Traceback (most recent call last):
  File "/Users/naomirodriguez/Documents/Past UCCS Classes/Spring 2024/CS 3850 Bi
oinformatics & Computational Bio/HiCNN2/HiCNN2_package/HiCNN2_training.py", line
 147, in <module>
    main()
  File "/Users/naomirodriguez/Documents/Past UCCS Classes/Spring 2024/CS 3850 Bi
oinformatics & Computational Bio/HiCNN2/HiCNN2_package/HiCNN2_training.py", line
 138, in main
    loss_train = train(model, device, train_loader, optimizer, args.clip)
                 ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

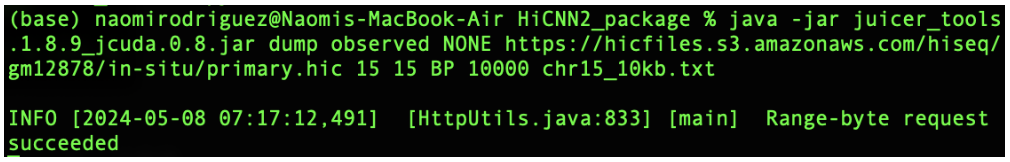


```
           ^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/opt/anaconda3/lib/python3.11/site-packages/torch/nn/modules/module.py",
 line 1511, in _wrapped_call_impl
    return self._call_impl(*args, **kwargs)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/opt/anaconda3/lib/python3.11/site-packages/torch/nn/modules/module.py",
 line 1520, in _call_impl
    return forward_call(*args, **kwargs)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/opt/anaconda3/lib/python3.11/site-packages/torch/nn/modules/loss.py", l
ine 535, in forward
    return F.mse_loss(input, target, reduction=self.reduction)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/opt/anaconda3/lib/python3.11/site-packages/torch/nn/functional.py", lin
e 3338, in mse_loss
    expanded_input, expanded_target = torch.broadcast_tensors(input, target)
                                      ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/opt/anaconda3/lib/python3.11/site-packages/torch/functional.py", line 7
6, in broadcast_tensors
    return _VF.broadcast_tensors(tensors)  # type: ignore[attr-defined]
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
RuntimeError: The size of tensor a (16) must match the size of tensor b (22) at
non-singleton dimension 3
```

Something that is recommended for this process is juicer version 1.8.9.

Since we've been using the chromosome 15 data, I used the following command:

```
java -jar juicer_tools.1.8.9_jcuda.0.8.jar dump observed NONE
https://hicfiles.s3.amazonaws.com/hiseq/gm12878/in-situ/primary.hic 15 15 BP
10000 chr15_10kb.txt
```



```
(base) naomirodriguez@Naomis-MacBook-Air HiCNN2_package % java -jar juicer_tools
.1.8.9_jcuda.0.8.jar dump observed NONE https://hicfiles.s3.amazonaws.com/hiseq/
gm12878/in-situ/primary.hic 15 15 BP 10000 chr15_10kb.txt

INFO [2024-05-08 07:17:12,491]  [HttpUtils.java:833] [main]  Range-byte request
succeeded
```

Create a `getInput.py` file:

```
resolution = 10000

# Step 1: Read the chr15_10kb.txt file
with open('chr15_10kb.txt', 'r') as f:
    lines = f.readlines()

# Step 2: Parse each line to extract indices and values
contact_matrix = {}
for line in lines:
    parts = line.strip().split()
    i, j, value = int(parts[0]), int(parts[1]), float(parts[2])
    contact_matrix[(i, j)] = value

# Step 3: Convert indices to match the expected format
converted_matrix = {}
for (i, j), value in contact_matrix.items():
    new_i, new_j = i // resolution, j // resolution
    if new_i == new_j:
        continue  # Ignore diagonal entries
    converted_matrix[(new_i, new_j)] = value

# Step 4: Write the converted matrix to a new file
with open('chr15_10kb_converted.txt', 'w') as f:
    for (i, j), value in converted_matrix.items():
        f.write(f'{i}\t{j}\t{value}\n')

# Step 5: Run get_HiCNN2_input_fromMat.py with the new file
import subprocess
subprocess.run(['python', 'get_HiCNN2_input_fromMat.py',
'chr15_10kb_converted.txt', '102531392', '10000', 'chr15.subMats',
'chr15.index'])
```

HiCNN2 trains from low to high resolution. We'll use 25kb to 10kb.

```
java -jar juicer_tools.1.8.9_jcuda.0.8.jar dump observed NONE
https://hicfiles.s3.amazonaws.com/hiseq/gm12878/in-situ/primary.hic 15 15 BP
25000 chr15_25kb.txt
```

Use the same `getInput.py` file as above but change the values to 25kb.

Now we can run the command:

```
python HiCNN2_training.py -f1 data/chr15.subMats.npy -f2
data/chr15.subMats.npy -f3 data/chr15.subMats.npy -f4 data/chr15.subMats.npy
```

```
-m 3 -d models -r 16 --batch-size 256 --epochs 5 --lr 0.1 --momentum 0.5 --
weight-decay 1e-4 --clip 0.01 --seed 1
```

Deliverables

.pdb file: Create a `pdbOutput.py`

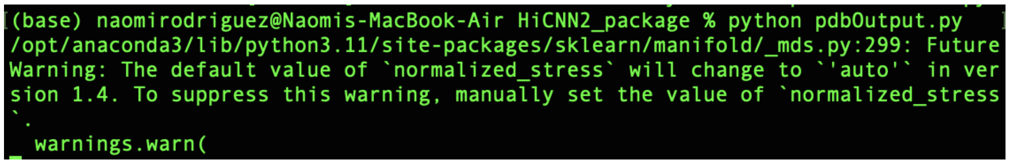
```
import numpy as np
from sklearn.manifold import MDS

predicted_matrix = np.load('data/chr15.predictedMat.npy')

mds = MDS(n_components=3, dissimilarity='precomputed')
spatial_coordinates = mds.fit_transform(predicted_matrix)

with open('output_structure.pdb', 'w') as f:
    for i, coords in enumerate(spatial_coordinates):
        f.write(f'ATOM  {i+1:5}  CA  ALA A   1    {coords[0]:8.3f}
{coords[1]:8.3f}{coords[2]:8.3f}  1.00  0.00\n')
```

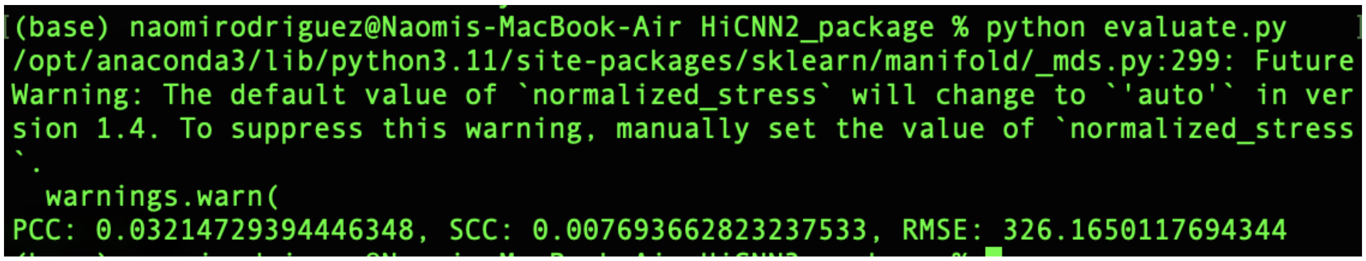
I ignored this warning:



```
(base) naomirodriguez@Naomis-MacBook-Air HiCNN2_package % python pdbOutput.py
/opt/anaconda3/lib/python3.11/site-packages/sklearn/manifold/_mds.py:299: Future
Warning: The default value of `normalized_stress` will change to `'auto'` in ver
sion 1.4. To suppress this warning, manually set the value of `normalized_stress
`.
  warnings.warn(
```

python evaluate.py

PCC: 0.03214729394446348, SCC: 0.007693662823237533, RMSE: 326.1650117694344



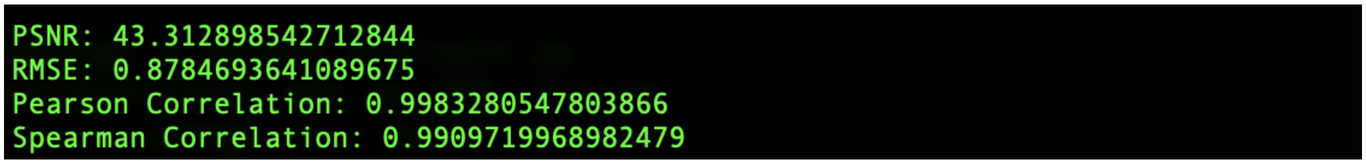
```
(base) naomirodriguez@Naomis-MacBook-Air HiCNN2_package % python evaluate.py
/opt/anaconda3/lib/python3.11/site-packages/sklearn/manifold/_mds.py:299: Future
Warning: The default value of `normalized_stress` will change to `'auto'` in ver
sion 1.4. To suppress this warning, manually set the value of `normalized_stress
`.
  warnings.warn(
PCC: 0.03214729394446348, SCC: 0.007693662823237533, RMSE: 326.1650117694344
```

PSNR: 43.312898542712844

RMSE: 0.8784693641089675

Pearson Correlation: 0.9983280547803866

Spearman Correlation: 0.9909719968982479



```
PSNR: 43.312898542712844
RMSE: 0.8784693641089675
Pearson Correlation: 0.9983280547803866
Spearman Correlation: 0.9909719968982479
```