Chem 277B Spring 2024 Tutorial 10

Outline

1. Graph Neural Netork - Manipulating Graph Data with `torch_geometric`
2. Get started with Savio (Berkeley HPC platform)
3. Setting up ANI project

1. Graph Neural Network

No description has been provided for this image

- Nodes: $\boldsymbol{v}_i$
- Edges: $\boldsymbol{e}_{ij}$
- An example of message passing:

$$\boldsymbol{e}_{ij}^{(l+1)} = f_e(\boldsymbol{e}_{ij}^{(l)}, \boldsymbol{v}_i^{(l)}, \boldsymbol{v}_j^{(l)})$$

$$\boldsymbol{v}_i^{(l+1)} = f_v(\boldsymbol{v}_i^{(l)}, \{\boldsymbol{e}_{ij}^{(l)}\})$$

Manipulating Graph Data in PyTorch: PyG

- Documentation: https://pytorch-geometric.readthedocs.io/en/latest/
- Installation: https://pytorch-geometric.readthedocs.io/en/latest/install/installation.html

`pip install torch_geometric`

`conda install pyg -c pyg`

Usage: Take QM9 as an example

QM9 is a dataset with 130,000 molecules with 19 regression targets, including dipole moments, atomization enthalpy, etc.

In [ ]:
```
import itertools

import torch
import torch.nn as nn
```

```
from torch_geometric.datasets import QM9

import numpy as np
from sklearn.model_selection import train_test_split
```

The `load_qm9` does the following things:

1. Download the QM9 dataset
2. Re-build the molecular graph: the original datasets add edges only for atoms connected by a chemical bond, however, here we create an edge between every pair of atoms
3. Calculate edge feature: $1/r$
4. Extract only atomization enthalpies as the target.

In [ ]:
```
def load_qm9(path="./QM9"):

    def transform(data):
        # re-build molecular graph
        edge_index = torch.tensor(
            list(itertools.permutations(range(data.x.shape[0]), 2)),
            dtype=torch.long
        ).T
        data.edge_index = edge_index
        # use 1/r as edge features
        edge_feature = 1 / torch.sqrt(
            torch.sum(
                (data.pos[edge_index[0]] - data.pos[edge_index[1]]) ** 2,
                axis=1, keepdim=True
            )
        )
        data.edge_attr = edge_feature
        # extract atomization enthalpies
        data.y = data.y[:, [-7]]
        return data

    qm9 = QM9(path, transform=transform)
    return qm9

qm9 = load_qm9("../../Datasets/QM9")
qm9
```

```
Downloading https://deepchemdata.s3-us-west-1.amazonaws.com/datasets/molnet_
publish/qm9.zip
Extracting ../../Datasets/QM9/raw/qm9.zip
Downloading https://ndownloader.figshare.com/files/3195404
Processing...
100%|██████████| 133885/133885 [00:46<00:00, 2866.80it/s]
Done!
```

Out[ ]: QM9(130831)

The dataset can be sliced.

In [ ]:
```
train_index, test_index = train_test_split(np.arange(len(qm9)), test_size=0.
train_data = qm9[train_index]
test_data = qm9[test_index]
train_data
```

Out[ ]:
```
QM9(104664)
```

The dataset can be batched with data loader.

In [ ]:
```
from torch_geometric.loader import DataLoader as GraphDataLoader

dataloader = GraphDataLoader(qm9, batch_size=1)
for data in dataloader:
    print(data)
    break
```

```
DataBatch(x=[5, 11], edge_index=[2, 20], edge_attr=[20, 1], y=[1, 1], pos=
[5, 3], z=[5], smiles=[1], name=[1], idx=[1], batch=[5], ptr=[2])
```

Node features

In [ ]:
```
data.x.shape
```

Out[ ]:
```
torch.Size([5, 11])
```

Edge features

In [ ]:
```
data.edge_attr.shape
```

Out[ ]:
```
torch.Size([20, 1])
```

Edge index: a tensor with shape `(n_edge, 2)`

In [ ]:
```
data.edge_index
```

Out[ ]:
```
tensor([[0, 0, 0, 0, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4],
        [1, 2, 3, 4, 0, 2, 3, 4, 0, 1, 3, 4, 0, 1, 2, 4, 0, 1, 2, 3]])
```

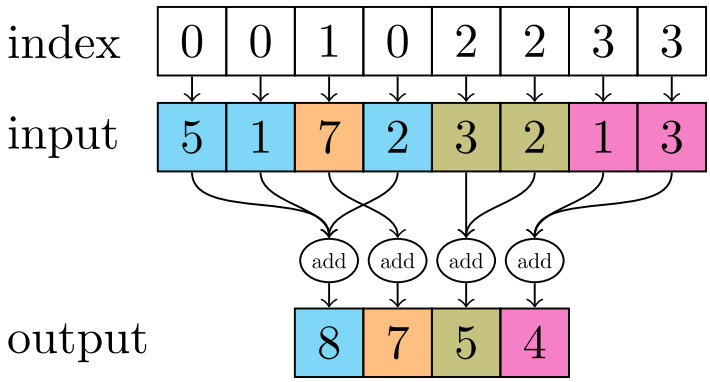
Batch: the node belongs to which graph

In [ ]:
```
data.batch
```

Out[ ]:
```
tensor([0, 0, 0, 0, 0])
```

Useful function: `scatter`

- Documentation: https://pytorch-scatter.readthedocs.io/en/latest/functions/scatter.html



$$\mathrm{out}_i = \mathrm{out}_i + \sum_j \mathrm{src}_j$$

where $\sum_j$ is over j such that $\mathrm{index}_j = i$.

In [ ]:
```
from torch_geometric.utils import scatter

inp = torch.tensor([5, 1, 7, 2, 3, 2, 1, 3], dtype=torch.float)
index = torch.tensor([0, 0, 1, 0, 2, 2, 3, 3], dtype=torch.long)
out = scatter(inp, index)
out
```

Out[ ]:
```
tensor([8., 7., 5., 4.])
```

Example: aggregate edge features and concatenate with node features. i.e.

$$v_i' = v_i \oplus \sum_{j \in N(i)} e_{ij}$$

$N(i)$ means the set of nodes that is directly connected with node i

In [ ]:
```
data.edge_attr.shape
```

Out[ ]:
```
torch.Size([20, 1])
```

In [ ]:
```
data.edge_index[0]
```

Out[ ]:
```
tensor([0, 0, 0, 0, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4])
```

In [ ]:
```
edge_aggr = scatter(data.edge_attr, data.edge_index[0])
edge_aggr.shape
```

Out[ ]:
```
torch.Size([5, 1])
```

In [ ]:
```
edge_aggr = scatter(data.edge_attr, data.edge_index[0])
new_node = torch.cat([data.x, edge_aggr], dim=1)
print(new_node.shape)
new_node
```

```
torch.Size([5, 12])
```

```
Out[ ]: tensor([[0.0000, 1.0000, 0.0000, 0.0000, 0.0000, 6.0000, 0.0000, 0.0000, 0.
        0000,
                 0.0000, 4.0000, 3.6633],
                [1.0000, 0.0000, 0.0000, 0.0000, 0.0000, 1.0000, 0.0000, 0.0000, 0.
        0000,
                 0.0000, 0.0000, 2.5983],
                [1.0000, 0.0000, 0.0000, 0.0000, 0.0000, 1.0000, 0.0000, 0.0000, 0.
        0000,
                 0.0000, 0.0000, 2.5983],
                [1.0000, 0.0000, 0.0000, 0.0000, 0.0000, 1.0000, 0.0000, 0.0000, 0.
        0000,
                 0.0000, 0.0000, 2.5983],
                [1.0000, 0.0000, 0.0000, 0.0000, 0.0000, 1.0000, 0.0000, 0.0000, 0.
        0000,
                 0.0000, 0.0000, 2.5983]])
```

2. Savio

Important: always use scratch directory to avoid disk quota issues

`cd /global/scratch/users/[USER_NAME]`

`conda init bash`

Replace `[USER_NAME]` with yours

Use premade env

It has all required dependencies for the ANI project (torch, torchani, numpy):

`conda activate /global/scratch/users/honamnguyen/chem277b/ani-env`

`python -m ipykernel install --user --name=ani-env`

Set up your own env

`conda create -p /global/scratch/users/[USER_NAME]/[ENV_NAME] python=3.10`

`conda activate /global/scratch/users/[USER_NAME]/[ENV_NAME]`

`conda install [PACKAGE_NAMES] # don't forget to install `ipykernel` here`

`python -m ipykernel install --user --name=[ENV_NAME]`

For example:

`conda create -p /global/scratch/users/honamnguyen/chem277b/ani-env python=3.10`

`conda activate /global/scratch/users/honamnguyen/chem277b/ani-env`

`pip install numpy scipy scikit-learn matplotlib pandas seaborn ipykernel h5py torchani`

`python -m ipykernel install --user --name=ani-env`

ANI Project

See: bCourses > Final Project for checkpoint templates

In [ ]:

```
```