Installing all the neccessary dependencies

```
import torch
import os
os.environ['TORCH'] = torch.__version__
print(torch.__version__)
!pip install h5py
!pip install -q torch-scatter -f https://data.pyg.org/whl/torch-$
{TORCH}.html
!pip install -q torch-sparse -f https://data.pyg.org/whl/torch-$
{TORCH}.html
!pip install -q git+https://github.com/pyg-team/pytorch_geometric.git
!pip install PyGCL
!pip install dgl
!pip install pytorch_metric_learning
```

```
2.2.1+cu121
Requirement already satisfied: h5py in /usr/local/lib/python3.10/dist-
packages (3.9.0)
Requirement already satisfied: numpy>=1.17.3 in
/usr/local/lib/python3.10/dist-packages (from h5py) (1.25.2)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 10.9/10.9 MB 41.9 MB/s eta
0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 5.0/5.0 MB 43.5 MB/s eta
0:00:00
ents to build wheel ... etadata (pyproject.toml) ... etric
(pyproject.toml) ... ent already satisfied: torch>=1.9 in
/usr/local/lib/python3.10/dist-packages (from PyGCL) (2.2.1+cu121)
Requirement already satisfied: torch-geometric>=1.7 in
/usr/local/lib/python3.10/dist-packages (from PyGCL) (2.6.0)
Requirement already satisfied: numpy in
/usr/local/lib/python3.10/dist-packages (from PyGCL) (1.25.2)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-
packages (from PyGCL) (4.66.2)
Requirement already satisfied: scipy in
/usr/local/lib/python3.10/dist-packages (from PyGCL) (1.11.4)
Requirement already satisfied: networkx in
/usr/local/lib/python3.10/dist-packages (from PyGCL) (3.2.1)
Requirement already satisfied: scikit-learn in
/usr/local/lib/python3.10/dist-packages (from PyGCL) (1.2.2)
Requirement already satisfied: filelock in
/usr/local/lib/python3.10/dist-packages (from torch>=1.9->PyGCL)
(3.13.1)
Requirement already satisfied: typing-extensions>=4.8.0 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.9->PyGCL)
(4.10.0)
Requirement already satisfied: sympy in
/usr/local/lib/python3.10/dist-packages (from torch>=1.9->PyGCL)
```

```
(1.12)
Requirement already satisfied: jinja2 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.9->PyGCL)
(3.1.3)
Requirement already satisfied: fsspec in
/usr/local/lib/python3.10/dist-packages (from torch>=1.9->PyGCL)
(2023.6.0)
Collecting nvidia-cuda-nvrtc-cu12==12.1.105 (from torch>=1.9->PyGCL)
  Downloading nvidia_cuda_nvrtc_cu12-12.1.105-py3-none-
manylinux1_x86_64.whl (23.7 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 23.7/23.7 MB 29.3 MB/s eta
0:00:00
e-cu12==12.1.105 (from torch>=1.9->PyGCL)
  Downloading nvidia_cuda_runtime_cu12-12.1.105-py3-none-
manylinux1_x86_64.whl (823 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 823.6/823.6 kB 61.1 MB/s eta
0:00:00
 torch>=1.9->PyGCL)
  Downloading nvidia_cuda_cupti_cu12-12.1.105-py3-none-
manylinux1_x86_64.whl (14.1 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 14.1/14.1 MB 52.8 MB/s eta
0:00:00
 torch>=1.9->PyGCL)
  Downloading nvidia_cudnn_cu12-8.9.2.26-py3-none-
manylinux1_x86_64.whl (731.7 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 731.7/731.7 MB 2.0 MB/s eta
0:00:00
 torch>=1.9->PyGCL)
  Downloading nvidia_cublas_cu12-12.1.3.1-py3-none-
manylinux1_x86_64.whl (410.6 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 410.6/410.6 MB 1.2 MB/s eta
0:00:00
 torch>=1.9->PyGCL)
  Downloading nvidia_cufft_cu12-11.0.2.54-py3-none-
manylinux1_x86_64.whl (121.6 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 121.6/121.6 MB 8.3 MB/s eta
0:00:00
 torch>=1.9->PyGCL)
  Downloading nvidia_curand_cu12-10.3.2.106-py3-none-
manylinux1_x86_64.whl (56.5 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 56.5/56.5 MB 9.5 MB/s eta
0:00:00
 torch>=1.9->PyGCL)
  Downloading nvidia_cusolver_cu12-11.4.5.107-py3-none-
manylinux1_x86_64.whl (124.2 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 124.2/124.2 MB 7.1 MB/s eta
0:00:00
 torch>=1.9->PyGCL)
  Downloading nvidia_cusparse_cu12-12.1.0.106-py3-none-
```

```
manylinux1_x86_64.whl (196.0 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 196.0/196.0 MB 3.7 MB/s eta
0:00:00
 torch>=1.9->PyGCL)
  Downloading nvidia_nccl_cu12-2.19.3-py3-none-manylinux1_x86_64.whl
(166.0 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 166.0/166.0 MB 5.8 MB/s eta
0:00:00
 torch>=1.9->PyGCL)
  Downloading nvidia_nvtx_cu12-12.1.105-py3-none-manylinux1_x86_64.whl
(99 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 99.1/99.1 kB 15.7 MB/s eta
0:00:00
ent already satisfied: triton==2.2.0 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.9->PyGCL)
(2.2.0)
Collecting nvidia-nvjitlink-cu12 (from nvidia-cusolver-
cu12==11.4.5.107->torch>=1.9->PyGCL)
  Downloading nvidia_nvjitlink_cu12-12.4.99-py3-none-
manylinux2014_x86_64.whl (21.1 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 21.1/21.1 MB 27.5 MB/s eta
0:00:00
ent already satisfied: aiohttp in /usr/local/lib/python3.10/dist-
packages (from torch-geometric>=1.7->PyGCL) (3.9.3)
Requirement already satisfied: psutil>=5.8.0 in
/usr/local/lib/python3.10/dist-packages (from torch-geometric>=1.7-
>PyGCL) (5.9.5)
Requirement already satisfied: pyparsing in
/usr/local/lib/python3.10/dist-packages (from torch-geometric>=1.7-
>PyGCL) (3.1.2)
Requirement already satisfied: requests in
/usr/local/lib/python3.10/dist-packages (from torch-geometric>=1.7-
>PyGCL) (2.31.0)
Requirement already satisfied: joblib>=1.1.1 in
/usr/local/lib/python3.10/dist-packages (from scikit-learn->PyGCL)
(1.3.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in
/usr/local/lib/python3.10/dist-packages (from scikit-learn->PyGCL)
(3.3.0)
Requirement already satisfied: aiosignal>=1.1.2 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->torch-
geometric>=1.7->PyGCL) (1.3.1)
Requirement already satisfied: attrs>=17.3.0 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->torch-
geometric>=1.7->PyGCL) (23.2.0)
Requirement already satisfied: frozenlist>=1.1.1 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->torch-
geometric>=1.7->PyGCL) (1.4.1)
Requirement already satisfied: multidict<7.0,>=4.5 in
```

```
/usr/local/lib/python3.10/dist-packages (from aiohttp->torch-
geometric>=1.7->PyGCL) (6.0.5)
Requirement already satisfied: yarl<2.0,>=1.0 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->torch-
geometric>=1.7->PyGCL) (1.9.4)
Requirement already satisfied: async-timeout<5.0,>=4.0 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->torch-
geometric>=1.7->PyGCL) (4.0.3)
Requirement already satisfied: MarkupSafe>=2.0 in
/usr/local/lib/python3.10/dist-packages (from jinja2->torch>=1.9-
>PyGCL) (2.1.5)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests->torch-
geometric>=1.7->PyGCL) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in
/usr/local/lib/python3.10/dist-packages (from requests->torch-
geometric>=1.7->PyGCL) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests->torch-
geometric>=1.7->PyGCL) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests->torch-
geometric>=1.7->PyGCL) (2024.2.2)
Requirement already satisfied: mpmath>=0.19 in
/usr/local/lib/python3.10/dist-packages (from sympy->torch>=1.9-
>PyGCL) (1.3.0)
Installing collected packages: nvidia-nvtx-cu12, nvidia-nvjitlink-
cu12, nvidia-nccl-cu12, nvidia-curand-cu12, nvidia-cufft-cu12, nvidia-
cuda-runtime-cu12, nvidia-cuda-nvrtc-cu12, nvidia-cuda-cupti-cu12,
nvidia-cublas-cu12, nvidia-cusparse-cu12, nvidia-cudnn-cu12, nvidia-
cusolver-cu12, PyGCL
Successfully installed PyGCL-0.1.2 nvidia-cublas-cu12-12.1.3.1 nvidia-
cuda-cupti-cu12-12.1.105 nvidia-cuda-nvrtc-cu12-12.1.105 nvidia-cuda-
runtime-cu12-12.1.105 nvidia-cudnn-cu12-8.9.2.26 nvidia-cufft-cu12-
11.0.2.54 nvidia-curand-cu12-10.3.2.106 nvidia-cusolver-cu12-
11.4.5.107 nvidia-cusparse-cu12-12.1.0.106 nvidia-nccl-cu12-2.19.3
nvidia-nvjitlink-cu12-12.4.99 nvidia-nvtx-cu12-12.1.105
Collecting dgl
  Downloading dgl-2.1.0-cp310-cp310-manylinux1_x86_64.whl (8.5 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.5/8.5 MB 24.5 MB/s eta
0:00:00
ent already satisfied: numpy>=1.14.0 in
/usr/local/lib/python3.10/dist-packages (from dgl) (1.25.2)
Requirement already satisfied: scipy>=1.1.0 in
/usr/local/lib/python3.10/dist-packages (from dgl) (1.11.4)
Requirement already satisfied: networkx>=2.1 in
/usr/local/lib/python3.10/dist-packages (from dgl) (3.2.1)
Requirement already satisfied: requests>=2.19.0 in
/usr/local/lib/python3.10/dist-packages (from dgl) (2.31.0)
```

```
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from dgl) (4.66.2)
Requirement already satisfied: psutil>=5.8.0 in
/usr/local/lib/python3.10/dist-packages (from dgl) (5.9.5)
Requirement already satisfied: torchdata>=0.5.0 in
/usr/local/lib/python3.10/dist-packages (from dgl) (0.7.1)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.19.0->dgl)
(3.3.2)
Requirement already satisfied: idna<4,>=2.5 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.19.0->dgl)
(3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.19.0->dgl)
(2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.19.0->dgl)
(2024.2.2)
Requirement already satisfied: torch>=2 in
/usr/local/lib/python3.10/dist-packages (from torchdata>=0.5.0->dgl)
(2.2.1+cu121)
Requirement already satisfied: filelock in
/usr/local/lib/python3.10/dist-packages (from torch>=2-
>torchdata>=0.5.0->dgl) (3.13.1)
Requirement already satisfied: typing-extensions>=4.8.0 in
/usr/local/lib/python3.10/dist-packages (from torch>=2-
>torchdata>=0.5.0->dgl) (4.10.0)
Requirement already satisfied: sympy in
/usr/local/lib/python3.10/dist-packages (from torch>=2-
>torchdata>=0.5.0->dgl) (1.12)
Requirement already satisfied: jinja2 in
/usr/local/lib/python3.10/dist-packages (from torch>=2-
>torchdata>=0.5.0->dgl) (3.1.3)
Requirement already satisfied: fsspec in
/usr/local/lib/python3.10/dist-packages (from torch>=2-
>torchdata>=0.5.0->dgl) (2023.6.0)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.1.105 in
/usr/local/lib/python3.10/dist-packages (from torch>=2-
>torchdata>=0.5.0->dgl) (12.1.105)
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.1.105
in /usr/local/lib/python3.10/dist-packages (from torch>=2-
>torchdata>=0.5.0->dgl) (12.1.105)
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.1.105 in
/usr/local/lib/python3.10/dist-packages (from torch>=2-
>torchdata>=0.5.0->dgl) (12.1.105)
Requirement already satisfied: nvidia-cudnn-cu12==8.9.2.26 in
/usr/local/lib/python3.10/dist-packages (from torch>=2-
>torchdata>=0.5.0->dgl) (8.9.2.26)
Requirement already satisfied: nvidia-cublas-cu12==12.1.3.1 in
```

```
/usr/local/lib/python3.10/dist-packages (from torch>=2-
>torchdata>=0.5.0->dgl) (12.1.3.1)
Requirement already satisfied: nvidia-cufft-cu12==11.0.2.54 in
/usr/local/lib/python3.10/dist-packages (from torch>=2-
>torchdata>=0.5.0->dgl) (11.0.2.54)
Requirement already satisfied: nvidia-curand-cu12==10.3.2.106 in
/usr/local/lib/python3.10/dist-packages (from torch>=2-
>torchdata>=0.5.0->dgl) (10.3.2.106)
Requirement already satisfied: nvidia-cusolver-cu12==11.4.5.107 in
/usr/local/lib/python3.10/dist-packages (from torch>=2-
>torchdata>=0.5.0->dgl) (11.4.5.107)
Requirement already satisfied: nvidia-cusparse-cu12==12.1.0.106 in
/usr/local/lib/python3.10/dist-packages (from torch>=2-
>torchdata>=0.5.0->dgl) (12.1.0.106)
Requirement already satisfied: nvidia-nccl-cu12==2.19.3 in
/usr/local/lib/python3.10/dist-packages (from torch>=2-
>torchdata>=0.5.0->dgl) (2.19.3)
Requirement already satisfied: nvidia-nvtx-cu12==12.1.105 in
/usr/local/lib/python3.10/dist-packages (from torch>=2-
>torchdata>=0.5.0->dgl) (12.1.105)
Requirement already satisfied: triton==2.2.0 in
/usr/local/lib/python3.10/dist-packages (from torch>=2-
>torchdata>=0.5.0->dgl) (2.2.0)
Requirement already satisfied: nvidia-nvjitlink-cu12 in
/usr/local/lib/python3.10/dist-packages (from nvidia-cusolver-
cu12==11.4.5.107->torch>=2->torchdata>=0.5.0->dgl) (12.4.99)
Requirement already satisfied: MarkupSafe>=2.0 in
/usr/local/lib/python3.10/dist-packages (from jinja2->torch>=2-
>torchdata>=0.5.0->dgl) (2.1.5)
Requirement already satisfied: mpmath>=0.19 in
/usr/local/lib/python3.10/dist-packages (from sympy->torch>=2-
>torchdata>=0.5.0->dgl) (1.3.0)
Installing collected packages: dgl
Successfully installed dgl-2.1.0
Collecting pytorch_metric_learning
  Downloading pytorch_metric_learning-2.4.1-py3-none-any.whl (118 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 118.6/118.6 kB 1.3 MB/s eta
0:00:00
ent already satisfied: numpy in /usr/local/lib/python3.10/dist-
packages (from pytorch_metric_learning) (1.25.2)
Requirement already satisfied: scikit-learn in
/usr/local/lib/python3.10/dist-packages (from pytorch_metric_learning)
(1.2.2)
Requirement already satisfied: torch>=1.6.0 in
/usr/local/lib/python3.10/dist-packages (from pytorch_metric_learning)
(2.2.1+cu121)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-
packages (from pytorch_metric_learning) (4.66.2)
Requirement already satisfied: filelock in
```

```
/usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->pytorch_metric_learning) (3.13.1)
Requirement already satisfied: typing-extensions>=4.8.0 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->pytorch_metric_learning) (4.10.0)
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->pytorch_metric_learning) (1.12)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->pytorch_metric_learning) (3.2.1)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->pytorch_metric_learning) (3.1.3)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->pytorch_metric_learning) (2023.6.0)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->pytorch_metric_learning) (12.1.105)
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->pytorch_metric_learning) (12.1.105)
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->pytorch_metric_learning) (12.1.105)
Requirement already satisfied: nvidia-cudnn-cu12==8.9.2.26 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->pytorch_metric_learning) (8.9.2.26)
Requirement already satisfied: nvidia-cublas-cu12==12.1.3.1 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->pytorch_metric_learning) (12.1.3.1)
Requirement already satisfied: nvidia-cufft-cu12==11.0.2.54 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->pytorch_metric_learning) (11.0.2.54)
Requirement already satisfied: nvidia-curand-cu12==10.3.2.106 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->pytorch_metric_learning) (10.3.2.106)
Requirement already satisfied: nvidia-cusolver-cu12==11.4.5.107 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->pytorch_metric_learning) (11.4.5.107)
Requirement already satisfied: nvidia-cusparse-cu12==12.1.0.106 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->pytorch_metric_learning) (12.1.0.106)
Requirement already satisfied: nvidia-nccl-cu12==2.19.3 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->pytorch_metric_learning) (2.19.3)
Requirement already satisfied: nvidia-nvtx-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0-
```

```
>pytorch_metric_learning) (12.1.105)
Requirement already satisfied: triton==2.2.0 in
/usr/local/lib/python3.10/dist-packages (from torch>=1.6.0-
>pytorch_metric_learning) (2.2.0)
Requirement already satisfied: nvidia-nvjitlink-cu12 in
/usr/local/lib/python3.10/dist-packages (from nvidia-cusolver-
cu12==11.4.5.107->torch>=1.6.0->pytorch_metric_learning) (12.4.99)
Requirement already satisfied: scipy>=1.3.2 in
/usr/local/lib/python3.10/dist-packages (from scikit-learn-
>pytorch_metric_learning) (1.11.4)
Requirement already satisfied: joblib>=1.1.1 in
/usr/local/lib/python3.10/dist-packages (from scikit-learn-
>pytorch_metric_learning) (1.3.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in
/usr/local/lib/python3.10/dist-packages (from scikit-learn-
>pytorch_metric_learning) (3.3.0)
Requirement already satisfied: MarkupSafe>=2.0 in
/usr/local/lib/python3.10/dist-packages (from jinja2->torch>=1.6.0-
>pytorch_metric_learning) (2.1.5)
Requirement already satisfied: mpmath>=0.19 in
/usr/local/lib/python3.10/dist-packages (from sympy->torch>=1.6.0-
>pytorch_metric_learning) (1.3.0)
Installing collected packages: pytorch_metric_learning
Successfully installed pytorch_metric_learning-2.4.1
```

Importing all the neccessary dependencies

Note : This project uses PyTorch Geometric Contrastive Learning(PyGCL), a PyTorch-based, library for all the Graph Contrastive learning task.

```python
import numpy as np
import h5py
import tqdm
import matplotlib.pyplot as plt
from sklearn.neighbors import kneighbors_graph

import torch.nn as nn
import torch.nn.functional as F
from torch.nn import Linear, ReLU
from torch.optim import Adam

from torch_geometric.nn import GCNConv, global_mean_pool
from torch_geometric.data import Data
from torch_geometric.loader import DataLoader

import GCL.augmentors as A
```

```
import GCL.losses as L
from GCL.models import DualBranchContrast
```

```
DGL backend not selected or invalid.  Assuming PyTorch for now.
```

```
Setting the default backend to "pytorch". You can change it in the
~/.dgl/config.json file or export the DGLBACKEND environment variable.
Valid options are: pytorch, mxnet, tensorflow (all lowercase)
```

Mounting and Loading the data from Drive

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
path = "/content/drive/MyDrive/quark-gluon_data-set_n139306.hdf5"
#Path to the dataset on my google drive

with h5py.File(path, 'r') as f:
  X_jets = np.array(f['X_jets'][:8000])
  labels = np.array(f['y'][:8000])
```

Converting the data to Graph format and doing preprocessing

```
dataset = []
for i, x in enumerate(X_jets):
  flattened = x.reshape(-1,3)
  non_zero = np.any(flattened != (0,0,0), axis = -1) # Removing any
zero element by considering only non zero ones
  node = flattened[non_zero]
  edges = kneighbors_graph(node, 2, mode = 'connectivity',include_self
= True)
  edges = edges.tocoo()
  y = torch.tensor([int(labels[i])], dtype=torch.long)
  data = Data(x=torch.from_numpy(node),
edge_index=torch.from_numpy(np.vstack((edges.row,edges.col))).type(tor
ch.long), edge_attr=torch.from_numpy(edges.data.reshape(-1,1)), y=y)
  dataset.append(data)

print(f'Number of graphs: {len(dataset)}')
print(f'Number of nodes: {dataset[0].num_nodes}')
print(f'Number of edges: {dataset[0].num_edges}')
print(f'Number of node features: {dataset[0].num_node_features}')
```

```
print(f'Number of edges features: {dataset[0].num_edge_features}')
print(dataset[0])
```

```
Number of graphs: 8000
Number of nodes: 884
Number of edges: 1768
Number of node features: 3
Number of edges features: 1
Data(x=[884, 3], edge_index=[2, 1768], edge_attr=[1768, 1], y=[1])
```

```
train_loader = DataLoader(dataset[:5000], batch_size=8, shuffle=True)
#Creating the train loader with batch = 8
test_loader = DataLoader(dataset[5000:], batch_size=8, shuffle=False)
# Creating the test loader with batch = 8

aug = A.Compose([A.EdgeRemoving(pe=0.3), A.FeatureMasking(pf=0.3)]) #
Selecing the graph augmentations

device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
```

Creating the Contrastive model

```
class GCN(nn.Module):
    def __init__(self, xavier=True):
        super(GCN, self).__init__()

        self.conv1 = GCNConv(3, 32)
        self.conv2 = GCNConv(32, 32)
        self.fc1 = Linear(32, 32)
        self.fc2 = Linear(32, 32)
        self.act = ReLU()

    def forward(self, data):
          # Performing the augmentaion twice as we use dual branch
contrastive learning
          augm_1 = aug(data.x, data.edge_index)
          augm_2 = aug(data.x, data.edge_index)

          x1 = self.conv1(augm_1[0], augm_1[1])
          x1 = self.act(x1)
          x2 = self.conv2(x1, augm_1[1])
          z1 = self.act(x2)

          x1 = self.conv1(augm_2[0], augm_2[1])
          x1 = self.act(x1)
          x2 = self.conv2(x2, augm_2[1])
          z2 = self.act(x2)

          x1 = self.conv1(data.x, data.edge_index)
```

```
        x1 = self.act(x1)
        x2 = self.conv2(x1, data.edge_index)
        z = self.act(x2)

        return z, z1, z2

    def project(self, z: torch.Tensor) -> torch.Tensor:
        #Projection head to reduce the size of the embeddings
        z = F.elu(self.fc1(z))
        return self.fc2(z)
```

Training the contrastive learning model

```
def train(encoder_model, contrast_model, data, optimizer):
    encoder_model.train()
    optimizer.zero_grad()
    z, z1, z2 = encoder_model(data)
    h1, h2 = [encoder_model.project(x) for x in [z1, z2]] # Creating
the reduced embeddings for the contrastive learning
    loss = contrast_model(h1, h2)
    loss.backward()
    optimizer.step()
    return loss.item()

def test(encoder_model, contrast_mocel, data, optimizer):
    encoder_model.eval()
    z, z1, z2 = encoder_model(data)
    h1, h2 = [encoder_model.project(x) for x in [z1, z2]] # Creating
the reduced embeddings for the contrastive learning
    loss = contrast_model(h1, h2)
    return loss.item()

encoder_model = GCN().to(device)
#Using Dual Branch Contrastive Learning with InfoNCE loss and using
local-to-local mode[to learn local representation]
contrast_model = DualBranchContrast(loss=L.InfoNCE(tau=0.2),
mode='L2L').to(device)
optimizer = Adam(encoder_model.parameters(), lr=0.01)

for epoch in range(30):
  total_loss = 0
  for _, data in enumerate(tqdm.tqdm(train_loader)):
      data = data.to(device)
      train_loss = train(encoder_model, contrast_model, data,
optimizer)
  for _, data in enumerate(tqdm.tqdm(test_loader)):
      data = data.to(device)
      test_loss = test(encoder_model, contrast_model, data, optimizer)
```

```
  log = "Epoch {}, Train Loss: {:.3f}, Test Loss: {:.3f}"
  print(log.format(epoch, train_loss, test_loss))

#Save Model
torch.save(encoder_model.state_dict(), 'autoencoder_weights.pth')
print("Encoder weights saved successfully!")
```

```
   0%|          | 0/625 [00:00<?, 
?it/s]/usr/local/lib/python3.10/dist-packages/torch_geometric/deprecat
ion.py:26: UserWarning: 'dropout_adj' is deprecated, use
'dropout_edge' instead
  warnings.warn(out)
100%|██████████| 625/625 [00:35<00:00, 17.83it/s]
100%|██████████| 375/375 [00:12<00:00, 30.08it/s]

Epoch 0, Train Loss: 8.071, Test Loss: 7.537

100%|██████████| 625/625 [00:32<00:00, 19.04it/s]
100%|██████████| 375/375 [00:12<00:00, 29.49it/s]

Epoch 1, Train Loss: 7.330, Test Loss: 6.931

100%|██████████| 625/625 [00:32<00:00, 19.10it/s]
100%|██████████| 375/375 [00:12<00:00, 30.08it/s]

Epoch 2, Train Loss: 6.880, Test Loss: 7.830

100%|██████████| 625/625 [00:32<00:00, 19.11it/s]
100%|██████████| 375/375 [00:12<00:00, 30.03it/s]

Epoch 3, Train Loss: 6.609, Test Loss: 6.658

100%|██████████| 625/625 [00:32<00:00, 19.14it/s]
100%|██████████| 375/375 [00:18<00:00, 20.75it/s]

Epoch 4, Train Loss: 6.332, Test Loss: 6.410

100%|██████████| 625/625 [00:33<00:00, 18.76it/s]
100%|██████████| 375/375 [00:12<00:00, 30.07it/s]

Epoch 5, Train Loss: 6.679, Test Loss: 7.155

100%|██████████| 625/625 [00:33<00:00, 18.80it/s]
100%|██████████| 375/375 [00:12<00:00, 29.93it/s]

Epoch 6, Train Loss: 6.109, Test Loss: 6.331

100%|██████████| 625/625 [00:32<00:00, 19.08it/s]
100%|██████████| 375/375 [00:12<00:00, 30.03it/s]

Epoch 7, Train Loss: 7.140, Test Loss: 6.191
```

```
100%|██████████| 625/625 [00:32<00:00, 18.96it/s]
100%|██████████| 375/375 [00:12<00:00, 30.09it/s]

Epoch 8, Train Loss: 6.235, Test Loss: 6.098

100%|██████████| 625/625 [00:32<00:00, 19.15it/s]
100%|██████████| 375/375 [00:12<00:00, 30.28it/s]

Epoch 9, Train Loss: 5.790, Test Loss: 5.789

100%|██████████| 625/625 [00:32<00:00, 18.94it/s]
100%|██████████| 375/375 [00:12<00:00, 29.97it/s]

Epoch 10, Train Loss: 6.492, Test Loss: 6.982

100%|██████████| 625/625 [00:32<00:00, 19.23it/s]
100%|██████████| 375/375 [00:12<00:00, 30.22it/s]

Epoch 11, Train Loss: 6.332, Test Loss: 6.913

100%|██████████| 625/625 [00:32<00:00, 19.07it/s]
100%|██████████| 375/375 [00:12<00:00, 30.26it/s]

Epoch 12, Train Loss: 6.874, Test Loss: 5.855

100%|██████████| 625/625 [00:32<00:00, 19.21it/s]
100%|██████████| 375/375 [00:12<00:00, 29.86it/s]

Epoch 13, Train Loss: 5.663, Test Loss: 6.481

100%|██████████| 625/625 [00:33<00:00, 18.64it/s]
100%|██████████| 375/375 [00:12<00:00, 29.50it/s]

Epoch 14, Train Loss: 7.655, Test Loss: 6.290

100%|██████████| 625/625 [00:32<00:00, 18.96it/s]
100%|██████████| 375/375 [00:12<00:00, 29.86it/s]

Epoch 15, Train Loss: 5.518, Test Loss: 5.478

100%|██████████| 625/625 [00:33<00:00, 18.90it/s]
100%|██████████| 375/375 [00:12<00:00, 29.48it/s]

Epoch 16, Train Loss: 5.534, Test Loss: 5.860

100%|██████████| 625/625 [00:32<00:00, 19.06it/s]
100%|██████████| 375/375 [00:12<00:00, 29.68it/s]

Epoch 17, Train Loss: 6.940, Test Loss: 5.489

100%|██████████| 625/625 [00:33<00:00, 18.84it/s]
100%|██████████| 375/375 [00:12<00:00, 29.80it/s]

Epoch 18, Train Loss: 5.568, Test Loss: 5.293
```

```
100%|██████████| 625/625 [00:32<00:00, 18.96it/s]
100%|██████████| 375/375 [00:12<00:00, 29.60it/s]

Epoch 19, Train Loss: 5.250, Test Loss: 5.292

100%|██████████| 625/625 [00:33<00:00, 18.72it/s]
100%|██████████| 375/375 [00:12<00:00, 30.05it/s]

Epoch 20, Train Loss: 6.022, Test Loss: 5.888

100%|██████████| 625/625 [00:33<00:00, 18.94it/s]
100%|██████████| 375/375 [00:12<00:00, 30.16it/s]

Epoch 21, Train Loss: 5.067, Test Loss: 5.232

100%|██████████| 625/625 [00:32<00:00, 18.94it/s]
100%|██████████| 375/375 [00:12<00:00, 30.32it/s]

Epoch 22, Train Loss: 5.423, Test Loss: 6.408

100%|██████████| 625/625 [00:32<00:00, 19.22it/s]
100%|██████████| 375/375 [00:12<00:00, 30.31it/s]

Epoch 23, Train Loss: 6.304, Test Loss: 5.331

100%|██████████| 625/625 [00:32<00:00, 19.08it/s]
100%|██████████| 375/375 [00:12<00:00, 30.29it/s]

Epoch 24, Train Loss: 5.226, Test Loss: 5.331

100%|██████████| 625/625 [00:32<00:00, 19.23it/s]
100%|██████████| 375/375 [00:12<00:00, 30.25it/s]

Epoch 25, Train Loss: 5.049, Test Loss: 5.108

100%|██████████| 625/625 [00:32<00:00, 19.12it/s]
100%|██████████| 375/375 [00:12<00:00, 29.65it/s]

Epoch 26, Train Loss: 5.934, Test Loss: 5.471

100%|██████████| 625/625 [00:32<00:00, 19.22it/s]
100%|██████████| 375/375 [00:12<00:00, 30.27it/s]

Epoch 27, Train Loss: 5.156, Test Loss: 5.101

100%|██████████| 625/625 [00:32<00:00, 19.17it/s]
100%|██████████| 375/375 [00:12<00:00, 30.36it/s]

Epoch 28, Train Loss: 5.307, Test Loss: 5.306

100%|██████████| 625/625 [00:32<00:00, 19.25it/s]
100%|██████████| 375/375 [00:12<00:00, 30.01it/s]
```

```
Epoch 29, Train Loss: 5.166, Test Loss: 6.304
Encoder weights saved successfully!
```

Defining the classifcation model

Here we use the model defined for learning representation before but without the projection head as we only need the learned represntation

```
class GraphClassificationModel(nn.Module):
    def __init__(self, load=True):
        super(GraphClassificationModel, self).__init__()

        self.encoder = GCN().to(device)

        if load:
            pth = 
self.encoder.load_state_dict(torch.load('autoencoder_weights.pth'))
            for param in self.encoder.parameters():
                param.requires_grad = False # Freezing the learned 
weights of encoder

        self.classifier = nn.Linear(32, 2)


    def forward(self, data):
        x, edge_index, batch = data.x, data.edge_index, data.batch
        embeddings,_,_ = self.encoder(data)
        z = global_mean_pool(embeddings, batch)
        pred = self.classifier(z)
        return pred
```

Training and Testing of the Classification model

```
def train_classification(model, loader, optimizer, criterion):
  model.train()
  total_loss = 0
  correct = 0
  total_samples = 0
  for _, data in enumerate(tqdm.tqdm(train_loader)):
    # print(data.batch.size)
    data = data.to(device)
    optimizer.zero_grad()
    out = model(data)
    #print(out.shape)
```

```
    loss = criterion(out, data.y)
    loss.backward()
    optimizer.step()
    total_loss += loss.item() * data.num_graphs
    # Calculate train accuracy
    pred = out.argmax(dim=1)
    correct += (pred == data.y).sum().item()
    total_samples += data.num_graphs
  train_accuracy = correct / total_samples
  return total_loss / len(loader.dataset), train_accuracy


def test_classification(model, loader):
    model.eval()
    correct = 0
    total = 0
    with torch.no_grad():
        for data in loader:
            data = data.to(device)
            out = model(data)
            pred = out.argmax(dim=1)
            correct += (pred == data.y).sum().item() #Calculating the
correct predictions
            total += data.num_graphs
    accuracy = correct / total
    return accuracy

model = GraphClassificationModel().to(device)
optimizer_2 = Adam(model.parameters(), lr=0.01)
criterion = nn.CrossEntropyLoss()

for epoch in range(20):
    train_loss, train_accuracy = train_classification(model,
train_loader, optimizer_2, criterion)
    test_accuracy = test_classification(model, test_loader)
    print(f'Epoch {epoch+1}, Train Loss: {train_loss:.4f}, Train
Accuracy: {train_accuracy:.4f}, Test Accuracy: {test_accuracy:.4f}')
```

```
100%|██████████| 625/625 [00:06<00:00, 103.42it/s]

Epoch 1, Train Loss: 0.6547, Train Accuracy: 0.6292, Test Accuracy:
0.6143

100%|██████████| 625/625 [00:05<00:00, 113.81it/s]

Epoch 2, Train Loss: 0.6370, Train Accuracy: 0.6598, Test Accuracy:
0.6507

100%|██████████| 625/625 [00:06<00:00, 102.85it/s]
```

```
Epoch 3, Train Loss: 0.6361, Train Accuracy: 0.6620, Test Accuracy:
0.5960

100%|██████████| 625/625 [00:05<00:00, 119.59it/s]

Epoch 4, Train Loss: 0.6309, Train Accuracy: 0.6676, Test Accuracy:
0.6873

100%|██████████| 625/625 [00:05<00:00, 122.45it/s]

Epoch 5, Train Loss: 0.6304, Train Accuracy: 0.6682, Test Accuracy:
0.6447

100%|██████████| 625/625 [00:05<00:00, 107.31it/s]

Epoch 6, Train Loss: 0.6360, Train Accuracy: 0.6688, Test Accuracy:
0.6750

100%|██████████| 625/625 [00:05<00:00, 123.73it/s]

Epoch 7, Train Loss: 0.6305, Train Accuracy: 0.6658, Test Accuracy:
0.6917

100%|██████████| 625/625 [00:05<00:00, 119.19it/s]

Epoch 8, Train Loss: 0.6343, Train Accuracy: 0.6710, Test Accuracy:
0.6930

100%|██████████| 625/625 [00:06<00:00, 98.48it/s]

Epoch 9, Train Loss: 0.6294, Train Accuracy: 0.6692, Test Accuracy:
0.6357

100%|██████████| 625/625 [00:05<00:00, 113.41it/s]

Epoch 10, Train Loss: 0.6240, Train Accuracy: 0.6686, Test Accuracy:
0.6013

100%|██████████| 625/625 [00:05<00:00, 114.18it/s]

Epoch 11, Train Loss: 0.6328, Train Accuracy: 0.6702, Test Accuracy:
0.6217

100%|██████████| 625/625 [00:05<00:00, 104.69it/s]

Epoch 12, Train Loss: 0.6187, Train Accuracy: 0.6754, Test Accuracy:
0.6733

100%|██████████| 625/625 [00:05<00:00, 117.21it/s]

Epoch 13, Train Loss: 0.6382, Train Accuracy: 0.6680, Test Accuracy:
0.6500

100%|██████████| 625/625 [00:05<00:00, 108.69it/s]
```

```
Epoch 14, Train Loss: 0.6324, Train Accuracy: 0.6718, Test Accuracy: 0.6480

100%|██████████| 625/625 [00:05<00:00, 107.79it/s]

Epoch 15, Train Loss: 0.6332, Train Accuracy: 0.6734, Test Accuracy: 0.6827

100%|██████████| 625/625 [00:05<00:00, 122.53it/s]

Epoch 16, Train Loss: 0.6249, Train Accuracy: 0.6762, Test Accuracy: 0.5250

100%|██████████| 625/625 [00:05<00:00, 117.88it/s]

Epoch 17, Train Loss: 0.6292, Train Accuracy: 0.6746, Test Accuracy: 0.6940

100%|██████████| 625/625 [00:05<00:00, 108.11it/s]

Epoch 18, Train Loss: 0.6268, Train Accuracy: 0.6764, Test Accuracy: 0.6627

100%|██████████| 625/625 [00:05<00:00, 123.18it/s]

Epoch 19, Train Loss: 0.6281, Train Accuracy: 0.6730, Test Accuracy: 0.7013

100%|██████████| 625/625 [00:05<00:00, 110.04it/s]

Epoch 20, Train Loss: 0.6274, Train Accuracy: 0.6752, Test Accuracy: 0.7027
```

Conclusion

The model's accuracy is 70% which is not the best. There are a multitude of reasons for that.

- One big problem is graph-level representation. Although, I have used global pooling to get a graph-level representation that is not the best way.
- We only consider an extremely small subset of the actual data due to memory issues which may cause data imbalance which stops the model from learning properly.
- Another problem is the graph representation isn't being learned well. Many possible reasons can be for this such as the architecture may not be right, the parameter tuning needs to be done well, etc. Further Research into this is required.
- When constructing the contrastive learning architecture other Graph models may be used such as GAT, GraphSage, etc to learn the representation. Each of these models will learn a different representation for the node which may be better or worse but

may increase the complexity of the model which may be computationally inefficient for larger datasets and graphs or also decrease.