GNN-Assignment-1-Group-20

1 Graph Nural Network Group 20 Assignment Submission:

1. Hemant Kumar Parakh (2023AA05741)
2. Sushil Kumar (2023aa05849)
3. Jitendra Kumar (2023aa05198)
4. MAREEDU RAVI KISHORE VARMA (2023aa05278)
5. K. KAMALAHASAN (2023ab05086)

BITS Lab link: https://cloudlabs.nuvepro.com/subscriptions/launch?id=2927862

Path: http://localhost:8888/notebooks/Desktop/Persistent_Folder/GNN-Assignment-1-Group-35.ipynb

1.1 Graph Analysis and Subgraph Generation Using GCNs in PyTorch Geometric:

This assignment involves creating and analyzing a graph neural network (GCN) using PyTorch Geometric. It covers loading the OGB dataset, computing graph metrics, visualizing the graph, generating subgraphs, and extracting node embeddings, providing comprehensive insights into graph-based machine learning.

1.1.1 Import requird libraries and packages

[4]:
```
# Install required libraries
!pip install torch torchvision torchaudio
!pip install torch-geometric
!pip install ogb
!pip install --upgrade ogb
!pip install networkx matplotlib
!pip install pycairo
!pip install cairocffi
!pip install python-igraph



import torch

!pip uninstall torch-scatter torch-sparse torch-geometric torch-cluster  --y
!pip install torch-scatter -f https://data.pyg.org/whl/torch-{torch.
↪__version__}.html
!pip install torch-sparse -f https://data.pyg.org/whl/torch-{torch.__version__}.
↪html
```

```
!pip install torch-cluster -f https://data.pyg.org/whl/torch-{torch.
 ↪__version__}.html
!pip install git+https://github.com/pyg-team/pytorch_geometric.git

# Import libraries
from ogb.nodeproppred import PygNodePropPredDataset
from torch_geometric.data import DataLoader
from torch_geometric.transforms import ToUndirected
from torch_geometric.nn import MessagePassing, GCNConv
from torch_geometric.utils import add_self_loops, degree

import torch.nn.functional as F
import networkx as nx
import numpy as np
from ogb.nodeproppred import Evaluator
import matplotlib.pyplot as plt

from torch_geometric.utils import to_networkx, to_undirected

print("Import done")
```

```
Requirement already satisfied: torch in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (2.5.1)
Requirement already satisfied: torchvision in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (0.20.1)
Requirement already satisfied: torchaudio in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (2.5.1)
Requirement already satisfied: filelock in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch) (3.13.1)
Requirement already satisfied: typing-extensions>=4.8.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch) (4.11.0)
Requirement already satisfied: networkx in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch) (3.2.1)
Requirement already satisfied: jinja2 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch) (3.1.4)
Requirement already satisfied: fsspec in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch) (2024.3.1)
Requirement already satisfied: setuptools in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch) (69.5.1)
Requirement already satisfied: sympy==1.13.1 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch) (1.13.1)
```

```
Requirement already satisfied: mpmath<1.4,>=1.1.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
sympy==1.13.1->torch) (1.3.0)
Requirement already satisfied: numpy in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torchvision) (1.26.4)
Requirement already satisfied: pillow!=8.3.*,>=5.3.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torchvision) (10.3.0)
Requirement already satisfied: MarkupSafe>=2.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
jinja2->torch) (2.1.3)
Requirement already satisfied: torch-geometric in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (2.7.0)
Requirement already satisfied: aiohttp in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch-geometric) (3.9.5)
Requirement already satisfied: fsspec in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch-geometric) (2024.3.1)
Requirement already satisfied: jinja2 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch-geometric) (3.1.4)
Requirement already satisfied: numpy in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch-geometric) (1.26.4)
Requirement already satisfied: psutil>=5.8.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch-geometric) (5.9.0)
Requirement already satisfied: pyparsing in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch-geometric) (3.0.9)
Requirement already satisfied: requests in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch-geometric) (2.32.2)
Requirement already satisfied: tqdm in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch-geometric) (4.66.4)
Requirement already satisfied: aiosignal>=1.1.2 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
aiohttp->torch-geometric) (1.2.0)
Requirement already satisfied: attrs>=17.3.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
aiohttp->torch-geometric) (23.1.0)
Requirement already satisfied: frozenlist>=1.1.1 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
aiohttp->torch-geometric) (1.4.0)
Requirement already satisfied: multidict<7.0,>=4.5 in
```

```
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
aiohttp->torch-geometric) (6.0.4)
Requirement already satisfied: yarl<2.0,>=1.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
aiohttp->torch-geometric) (1.9.3)
Requirement already satisfied: MarkupSafe>=2.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
jinja2->torch-geometric) (2.1.3)
Requirement already satisfied: charset-normalizer<4,>=2 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
requests->torch-geometric) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
requests->torch-geometric) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
requests->torch-geometric) (2.2.2)
Requirement already satisfied: certifi>=2017.4.17 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
requests->torch-geometric) (2024.7.4)
Requirement already satisfied: colorama in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
tqdm->torch-geometric) (0.4.6)
Requirement already satisfied: ogb in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (1.3.6)
Requirement already satisfied: torch>=1.6.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from ogb)
(2.5.1)
Requirement already satisfied: numpy>=1.16.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from ogb)
(1.26.4)
Requirement already satisfied: tqdm>=4.29.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from ogb)
(4.66.4)
Requirement already satisfied: scikit-learn>=0.20.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from ogb)
(1.4.2)
Requirement already satisfied: pandas>=0.24.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from ogb)
(2.2.2)
Requirement already satisfied: six>=1.12.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from ogb)
(1.16.0)
Requirement already satisfied: urllib3>=1.24.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from ogb)
(2.2.2)
Requirement already satisfied: outdated>=0.2.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from ogb)
```

```
(0.2.2)
Requirement already satisfied: setuptools>=44 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
outdated>=0.2.0->ogb) (69.5.1)
Requirement already satisfied: littleutils in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
outdated>=0.2.0->ogb) (0.2.4)
Requirement already satisfied: requests in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
outdated>=0.2.0->ogb) (2.32.2)
Requirement already satisfied: python-dateutil>=2.8.2 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
pandas>=0.24.0->ogb) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
pandas>=0.24.0->ogb) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
pandas>=0.24.0->ogb) (2023.3)
Requirement already satisfied: scipy>=1.6.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
scikit-learn>=0.20.0->ogb) (1.13.1)
Requirement already satisfied: joblib>=1.2.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
scikit-learn>=0.20.0->ogb) (1.4.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
scikit-learn>=0.20.0->ogb) (2.2.0)
Requirement already satisfied: filelock in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch>=1.6.0->ogb) (3.13.1)
Requirement already satisfied: typing-extensions>=4.8.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch>=1.6.0->ogb) (4.11.0)
Requirement already satisfied: networkx in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch>=1.6.0->ogb) (3.2.1)
Requirement already satisfied: jinja2 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch>=1.6.0->ogb) (3.1.4)
Requirement already satisfied: fsspec in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch>=1.6.0->ogb) (2024.3.1)
Requirement already satisfied: sympy==1.13.1 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch>=1.6.0->ogb) (1.13.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
```

```
sympy==1.13.1->torch>=1.6.0->ogb) (1.3.0)
Requirement already satisfied: colorama in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
tqdm>=4.29.0->ogb) (0.4.6)
Requirement already satisfied: MarkupSafe>=2.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
jinja2->torch>=1.6.0->ogb) (2.1.3)
Requirement already satisfied: charset-normalizer<4,>=2 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
requests->outdated>=0.2.0->ogb) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
requests->outdated>=0.2.0->ogb) (3.7)
Requirement already satisfied: certifi>=2017.4.17 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
requests->outdated>=0.2.0->ogb) (2024.7.4)
Requirement already satisfied: ogb in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (1.3.6)
Requirement already satisfied: torch>=1.6.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from ogb)
(2.5.1)
Requirement already satisfied: numpy>=1.16.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from ogb)
(1.26.4)
Requirement already satisfied: tqdm>=4.29.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from ogb)
(4.66.4)
Requirement already satisfied: scikit-learn>=0.20.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from ogb)
(1.4.2)
Requirement already satisfied: pandas>=0.24.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from ogb)
(2.2.2)
Requirement already satisfied: six>=1.12.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from ogb)
(1.16.0)
Requirement already satisfied: urllib3>=1.24.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from ogb)
(2.2.2)
Requirement already satisfied: outdated>=0.2.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from ogb)
(0.2.2)
Requirement already satisfied: setuptools>=44 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
outdated>=0.2.0->ogb) (69.5.1)
Requirement already satisfied: littleutils in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
outdated>=0.2.0->ogb) (0.2.4)
```

```
Requirement already satisfied: requests in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
outdated>=0.2.0->ogb) (2.32.2)
Requirement already satisfied: python-dateutil>=2.8.2 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
pandas>=0.24.0->ogb) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
pandas>=0.24.0->ogb) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
pandas>=0.24.0->ogb) (2023.3)
Requirement already satisfied: scipy>=1.6.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
scikit-learn>=0.20.0->ogb) (1.13.1)
Requirement already satisfied: joblib>=1.2.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
scikit-learn>=0.20.0->ogb) (1.4.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
scikit-learn>=0.20.0->ogb) (2.2.0)
Requirement already satisfied: filelock in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch>=1.6.0->ogb) (3.13.1)
Requirement already satisfied: typing-extensions>=4.8.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch>=1.6.0->ogb) (4.11.0)
Requirement already satisfied: networkx in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch>=1.6.0->ogb) (3.2.1)
Requirement already satisfied: jinja2 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch>=1.6.0->ogb) (3.1.4)
Requirement already satisfied: fsspec in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch>=1.6.0->ogb) (2024.3.1)
Requirement already satisfied: sympy==1.13.1 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch>=1.6.0->ogb) (1.13.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
sympy==1.13.1->torch>=1.6.0->ogb) (1.3.0)
Requirement already satisfied: colorama in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
tqdm>=4.29.0->ogb) (0.4.6)
Requirement already satisfied: MarkupSafe>=2.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
jinja2->torch>=1.6.0->ogb) (2.1.3)
```

```
Requirement already satisfied: charset-normalizer<4,>=2 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
requests->outdated>=0.2.0->ogb) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
requests->outdated>=0.2.0->ogb) (3.7)
Requirement already satisfied: certifi>=2017.4.17 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
requests->outdated>=0.2.0->ogb) (2024.7.4)
Requirement already satisfied: networkx in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (3.2.1)
Requirement already satisfied: matplotlib in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (3.8.4)
Requirement already satisfied: contourpy>=1.0.1 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
matplotlib) (1.2.0)
Requirement already satisfied: cycler>=0.10 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
matplotlib) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
matplotlib) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
matplotlib) (1.4.4)
Requirement already satisfied: numpy>=1.21 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
matplotlib) (1.26.4)
Requirement already satisfied: packaging>=20.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
matplotlib) (23.2)
Requirement already satisfied: pillow>=8 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
matplotlib) (10.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
matplotlib) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
matplotlib) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
python-dateutil>=2.7->matplotlib) (1.16.0)
Requirement already satisfied: pycairo in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (1.27.0)
Requirement already satisfied: cairocffi in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (1.7.1)
Requirement already satisfied: cffi>=1.1.0 in
```

```
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
cairocffi) (1.16.0)
Requirement already satisfied: pycparser in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
cffi>=1.1.0->cairocffi) (2.21)
Requirement already satisfied: python-igraph in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (0.11.8)
Requirement already satisfied: igraph==0.11.8 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
python-igraph) (0.11.8)
Requirement already satisfied: texttable>=1.6.2 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
igraph==0.11.8->python-igraph) (1.7.0)
Found existing installation: torch_scatter 2.1.2+pt25cpu
Uninstalling torch_scatter-2.1.2+pt25cpu:
  Successfully uninstalled torch_scatter-2.1.2+pt25cpu
Found existing installation: torch_sparse 0.6.18+pt25cpu
Uninstalling torch_sparse-0.6.18+pt25cpu:
  Successfully uninstalled torch_sparse-0.6.18+pt25cpu
Found existing installation: torch-geometric 2.7.0
Uninstalling torch-geometric-2.7.0:
  Successfully uninstalled torch-geometric-2.7.0
Found existing installation: torch_cluster 1.6.3+pt25cpu
Uninstalling torch_cluster-1.6.3+pt25cpu:
  Successfully uninstalled torch_cluster-1.6.3+pt25cpu
Looking in links: https://data.pyg.org/whl/torch-2.5.1+cpu.html
Collecting torch-scatter
  Using cached
https://data.pyg.org/whl/torch-2.5.0%2Bcpu/torch_scatter-2.1.2%2Bpt25cpu-
cp312-cp312-win_amd64.whl (366 kB)
Installing collected packages: torch-scatter
Successfully installed torch-scatter-2.1.2+pt25cpu
Looking in links: https://data.pyg.org/whl/torch-2.5.1+cpu.html
Collecting torch-sparse
  Using cached
https://data.pyg.org/whl/torch-2.5.0%2Bcpu/torch_sparse-0.6.18%2Bpt25cpu-
cp312-cp312-win_amd64.whl (802 kB)
Requirement already satisfied: scipy in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch-sparse) (1.13.1)
Requirement already satisfied: numpy<2.3,>=1.22.4 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
scipy->torch-sparse) (1.26.4)
Installing collected packages: torch-sparse
Successfully installed torch-sparse-0.6.18+pt25cpu
Looking in links: https://data.pyg.org/whl/torch-2.5.1+cpu.html
Collecting torch-cluster
  Using cached
```

```
https://data.pyg.org/whl/torch-2.5.0%2Bcpu/torch_cluster-1.6.3%2Bpt25cpu-
cp312-cp312-win_amd64.whl (507 kB)
Requirement already satisfied: scipy in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch-cluster) (1.13.1)
Requirement already satisfied: numpy<2.3,>=1.22.4 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
scipy->torch-cluster) (1.26.4)
Installing collected packages: torch-cluster
Successfully installed torch-cluster-1.6.3+pt25cpu
```

```
  Running command git clone --filter=blob:none --quiet https://github.com/pyg-
team/pytorch_geometric.git 'C:\Users\hemant.parakh.CORP\AppData\Local\Temp\pip-
req-build-3oag2gks'
```

```
Collecting git+https://github.com/pyg-team/pytorch_geometric.git
  Cloning https://github.com/pyg-team/pytorch_geometric.git to
c:\users\hemant.parakh.corp\appdata\local\temp\pip-req-build-3oag2gks
  Resolved https://github.com/pyg-team/pytorch_geometric.git to commit
ef028547ff4459f6e98fe429d1564bd1d513fc31
  Installing build dependencies: started
  Installing build dependencies: finished with status 'done'
  Getting requirements to build wheel: started
  Getting requirements to build wheel: finished with status 'done'
  Preparing metadata (pyproject.toml): started
  Preparing metadata (pyproject.toml): finished with status 'done'
Requirement already satisfied: aiohttp in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch-geometric==2.7.0) (3.9.5)
Requirement already satisfied: fsspec in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch-geometric==2.7.0) (2024.3.1)
Requirement already satisfied: jinja2 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch-geometric==2.7.0) (3.1.4)
Requirement already satisfied: numpy in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch-geometric==2.7.0) (1.26.4)
Requirement already satisfied: psutil>=5.8.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch-geometric==2.7.0) (5.9.0)
Requirement already satisfied: pyparsing in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch-geometric==2.7.0) (3.0.9)
Requirement already satisfied: requests in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
torch-geometric==2.7.0) (2.32.2)
Requirement already satisfied: tqdm in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
```

```
torch-geometric==2.7.0) (4.66.4)
Requirement already satisfied: aiosignal>=1.1.2 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
aiohttp->torch-geometric==2.7.0) (1.2.0)
Requirement already satisfied: attrs>=17.3.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
aiohttp->torch-geometric==2.7.0) (23.1.0)
Requirement already satisfied: frozenlist>=1.1.1 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
aiohttp->torch-geometric==2.7.0) (1.4.0)
Requirement already satisfied: multidict<7.0,>=4.5 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
aiohttp->torch-geometric==2.7.0) (6.0.4)
Requirement already satisfied: yarl<2.0,>=1.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
aiohttp->torch-geometric==2.7.0) (1.9.3)
Requirement already satisfied: MarkupSafe>=2.0 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
jinja2->torch-geometric==2.7.0) (2.1.3)
Requirement already satisfied: charset-normalizer<4,>=2 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
requests->torch-geometric==2.7.0) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
requests->torch-geometric==2.7.0) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
requests->torch-geometric==2.7.0) (2.2.2)
Requirement already satisfied: certifi>=2017.4.17 in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
requests->torch-geometric==2.7.0) (2024.7.4)
Requirement already satisfied: colorama in
c:\users\hemant.parakh.corp\appdata\local\anaconda3\lib\site-packages (from
tqdm->torch-geometric==2.7.0) (0.4.6)
Building wheels for collected packages: torch-geometric
  Building wheel for torch-geometric (pyproject.toml): started
  Building wheel for torch-geometric (pyproject.toml): finished with status
'done'
  Created wheel for torch-geometric: filename=torch_geometric-2.7.0-py3-none-
any.whl size=1178459
sha256=838f806295447daf4ef8532d313b300f6c2f0cc78374bccc4240b59febfaa53a
  Stored in directory: C:\Users\hemant.parakh.CORP\AppData\Local\Temp\pip-ephem-
wheel-cache-
ocb4u_0g\wheels\96\ab\80\5e43250505a6e639df59a3d89c6b45ed5511f70db8d0ac39c7
Successfully built torch-geometric
Installing collected packages: torch-geometric
Successfully installed torch-geometric-2.7.0
Import done
```

Dataset Loading

The ogbn-products dataset is an undirected and unweighted graph, representing an Amazon product co-purchasing network [1]. Nodes represent products sold in Amazon, and edges between two products indicate that the products are purchased together. We follow [2] to process node features and target categories. Specifically, node features are generated by extracting bag-of-words features from the product descriptions followed by a Principal Component Analysis to reduce the dimension to 100.

[6]:
```
# Load ogbn-products dataset
#dataset = PygNodePropPredDataset(name='ogbn-arxiv', root='/tmp') # Use small␣
↪dataset to work faster
dataset = PygNodePropPredDataset(name='ogbn-products', root='/tmp')
data = dataset[0]  # Get the graph data
#data = ToUndirected()(data)  # Convert the graph to undirected

from torch_geometric.utils import remove_self_loops
# Check for self-loops
print(f"Number of self-loops before removal: {(data.edge_index[0] == data.
↪edge_index[1]).sum().item()}")

# Remove self-loops
data.edge_index, _ = remove_self_loops(data.edge_index)

# Verify removal
print(f"Number of self-loops after removal: {(data.edge_index[0] == data.
↪edge_index[1]).sum().item()}")
print("printing data...")
print(f"Number of nodes: {data.num_nodes}")
print(f"Number of edges: {data.edge_index.size(1) // 2}")  # Divide by 2 if␣
↪undirected
print(f"Node feature shape: {data.x.shape}")
print(f"Number of classes: {dataset.num_classes}")
print(f"Edge index (first 5 edges): {data.edge_index[:, :5]}")
print(f"Node features (first 5 nodes): {data.x[:5]}")
print(f"Labels (first 5 nodes): {data.y[:5]}")
print(f"Data representation: {repr(data)}")

print("printing done...")
```

```
C:\Users\hemant.parakh.CORP\AppData\Local\anaconda3\Lib\site-
packages\ogb\nodeproppred\dataset_pyg.py:69: FutureWarning: You are using
`torch.load` with `weights_only=False` (the current default value), which uses
the default pickle module implicitly. It is possible to construct malicious
pickle data which will execute arbitrary code during unpickling (See
https://github.com/pytorch/pytorch/blob/main/SECURITY.md#untrusted-models for
more details). In a future release, the default value for `weights_only` will be
flipped to `True`. This limits the functions that could be executed during
```

```
unpickling. Arbitrary objects will no longer be allowed to be loaded via this
mode unless they are explicitly allowlisted by the user via
`torch.serialization.add_safe_globals`. We recommend you start setting
`weights_only=True` for any use case where you don't have full control of the
loaded file. Please open an issue on GitHub for any issues related to this
experimental feature.
  self.data, self.slices = torch.load(self.processed_paths[0])

Number of self-loops before removal: 256
Number of self-loops after removal: 0
printing data…
Number of nodes: 2449029
Number of edges: 61859012
Node feature shape: torch.Size([2449029, 100])
Number of classes: 47
Edge index (first 5 edges): tensor([[     0, 152857,      0,  32104,      0],
        [152857,      0,  32104,      0,  23158]])
Node features (first 5 nodes): tensor([[ 3.1933e-02, -1.9586e-01,  5.1996e-02,
-6.3349e-02, -2.2987e-01,
         -2.2130e-02,  4.0465e-01, -1.0794e-01,  3.2562e-02,  6.0270e-02,
          1.3270e-01,  4.5856e-01, -9.5493e-02,  2.5118e-01, -2.7464e-02,
          2.0437e-01, -6.5092e-02,  2.8799e-01,  1.5266e-02,  1.3919e-01,
         -2.7391e-01, -1.0493e-01, -2.1358e-02,  2.7579e-01,  4.5625e-02,
         -3.1325e-01, -2.0205e-01, -2.0238e-01, -3.1769e-01,  7.9290e-02,
         -1.0984e-01,  2.1498e-01, -3.4563e-01, -2.2236e-01, -4.0740e-01,
         -1.0198e-01, -4.0942e-01, -5.0210e-03,  4.8593e-01,  3.5642e-01,
          4.4607e-02, -5.4105e-02,  1.4025e-01,  3.2521e-01,  2.1967e-02,
         -3.0192e-01,  2.0702e-01,  2.7724e-01,  1.2430e-04,  2.1450e-01,
         -1.0187e-01, -1.4705e-02,  4.4537e-01, -1.2550e-01, -8.7204e-02,
         -6.3675e-02, -8.2962e-02, -3.8798e-02,  1.9939e-01,  4.3434e-01,
         -1.5574e-01,  1.0861e-01, -2.8592e-01, -7.1163e-01, -2.2232e-02,
         -1.1628e-01, -3.1844e-01, -5.5476e-02,  3.0896e-02,  3.5956e-01,
          2.5517e-01,  2.1755e-01,  2.1767e-01, -1.7638e-01, -1.3238e-02,
         -2.6137e-01,  6.2384e-03,  1.6235e-01, -1.2374e-01, -1.3847e-01,
         -4.7884e-01,  9.0116e-03,  8.4543e-02, -2.5821e-01, -2.6492e-01,
          2.8033e-01, -2.2818e-01,  8.7880e-02, -3.5572e-01,  6.7761e-02,
         -2.9939e-01, -1.8311e-01,  5.0098e-01,  4.0224e-01,  1.1225e-01,
         -1.1269e-01,  1.4176e-01,  7.6696e-02, -3.9295e-01, -6.4784e-02],
        [-2.4058e-02,  6.3032e-01,  1.0606e+00, -7.9183e-02, -3.2062e-01,
          1.3723e+00, -1.4605e+00,  3.9458e-01,  4.8687e-01, -2.6310e+00,
          6.1941e-01, -1.4335e-01,  2.0034e-01,  9.6304e-01, -1.7939e-01,
          3.4026e-01,  3.2152e-01, -1.8167e+00, -5.8037e-01,  2.8239e-01,
         -7.1155e-01,  1.2476e-01, -2.0734e+00, -5.8871e-01, -6.2062e-01,
         -7.2479e-01, -8.0023e-02, -1.4579e-01,  3.5878e-01, -9.5739e-01,
         -8.0171e-01, -1.2759e+00,  5.4423e-01,  1.5880e+00, -8.9738e-01,
          9.5927e-01,  1.2718e+00, -1.3204e+00, -5.6687e-01, -1.2433e+00,
         -1.8770e-01, -6.2439e-01,  6.8238e-01, -1.4801e+00, -1.2868e+00,
         -2.1820e+00, -1.6109e+00,  1.4456e-01, -5.6290e-01, -1.2691e+00,
```

```
  2.0068e-01, -1.2292e+00,  4.4382e-01,  2.8553e-01,  6.3349e-01,
 -2.9132e-01,  2.3911e-01,  2.6819e+00, -8.9025e-01,  2.3306e+00,
 -1.6056e+00,  8.9716e-01, -9.7976e-01,  3.5086e-01,  1.4573e+00,
 -9.8415e-02,  1.2122e+00,  1.9668e+00, -7.6289e-01, -1.9312e+00,
  1.1985e+00, -1.2284e+00, -1.7076e+00, -1.3423e-01,  9.7463e-01,
  7.7464e-01, -5.2571e-02, -1.4664e+00,  1.3394e+00, -5.3983e-01,
  1.1983e+00, -2.2499e+00, -1.5825e-01,  1.5906e+00, -1.4592e-01,
  1.9539e-01,  1.1294e+00,  8.2542e-01,  5.1861e-01,  5.2653e-01,
  2.6601e+00,  8.3046e-01,  3.2782e+00,  9.2244e-01, -5.6272e-01,
 -5.1659e-01, -5.3100e-02, -1.6875e+00,  3.5867e+00,  8.1822e-01],
[ 3.3269e-01, -5.5860e-01, -2.8861e-01,  2.1552e-01,  3.6495e-01,
  1.8257e-01, -9.7908e-02, -2.5049e-01,  5.3144e-01,  2.7853e-01,
  2.8432e-01, -1.3282e-01, -4.5808e-02, -1.7047e-01, -3.0150e-01,
  1.5755e-01, -5.3379e-01,  4.4447e-01,  2.1066e-01,  6.4368e-02,
 -3.8159e-02, -3.7573e-01, -3.0348e-01, -8.9196e-02, -2.2821e-01,
  1.3811e-01,  1.7351e-01,  3.9079e-01, -5.4295e-02,  2.7811e-01,
  5.7655e-02, -4.0649e-01,  3.9022e-01, -1.5365e-01, -3.7700e-01,
  4.5267e-01,  4.4652e-01, -4.9942e-02,  1.2088e-01,  1.9008e-01,
 -3.6541e-01,  4.4363e-02, -6.7469e-02, -4.5332e-01, -7.4105e-01,
 -3.0548e-01,  2.5848e-01,  2.6698e-01,  4.1946e-01, -2.5558e-01,
  4.8985e-01, -3.4307e-02,  1.7017e-01, -5.4782e-01, -1.2719e-01,
  4.5171e-01, -2.6762e-01,  3.0792e-01, -7.4064e-01, -2.0778e-01,
  8.7415e-01,  9.9945e-02,  9.2112e-03, -3.4322e-01, -3.3743e-02,
  5.4478e-02, -1.0254e-01, -4.4112e-01, -2.8270e-01,  2.5495e-02,
 -5.8151e-01,  4.3318e-02, -2.8995e-01,  3.2427e-01, -4.5689e-01,
  1.0203e-01,  3.4146e-01, -2.6178e-01, -2.4286e-01, -2.1914e-01,
 -1.4639e-01,  4.5815e-02, -1.3819e-01,  6.5855e-01, -1.0953e-01,
 -3.5585e-03,  5.3334e-01,  4.9375e-01,  7.8734e-02, -2.4455e-01,
  4.5448e-01, -1.9042e-01,  4.3034e-01,  3.0199e-01,  1.7448e-01,
 -3.3457e-01,  2.9385e-02, -3.7157e-01,  2.5206e-01,  4.1532e-02],
[-2.2079e-01,  4.7938e-01,  2.9859e-01,  1.1589e+00,  3.7441e-01,
 -1.4335e-03,  5.2553e-01, -5.6309e-01, -2.3326e-01, -2.2007e-01,
 -5.1632e-01,  3.1902e-01, -4.8400e-01, -5.1031e-01,  1.0170e+00,
 -6.1652e-01,  7.5738e-01, -7.6972e-01, -5.9704e-01,  6.4940e-02,
 -1.1655e+00,  9.0938e-01,  1.7427e-01,  1.1393e+00, -7.6400e-02,
  1.5401e+00,  1.8013e-01, -4.2013e-03, -3.5327e-01, -1.2614e-01,
  1.7872e-01, -8.8890e-01, -6.5576e-01,  7.8876e-01,  2.9145e-01,
  3.9107e-02, -2.0338e-01, -7.6228e-01, -5.4982e-01,  8.2315e-01,
 -1.8166e-01,  3.6151e-03, -9.2522e-01, -5.4644e-01,  8.1995e-02,
  1.6795e-01,  4.0838e-01, -1.6539e+00, -5.9818e-01,  1.1024e-02,
  1.5706e+00,  1.9453e-01,  3.3051e-01, -1.0885e+00, -1.8881e-01,
 -1.7460e-01, -9.6734e-01,  6.8738e-01, -1.3662e+00, -7.8843e-01,
  5.5509e-01,  1.0842e+00,  2.9162e-01, -1.1672e+00,  5.0599e-01,
 -3.5149e-01,  5.0870e-01,  6.8492e-01, -9.0212e-02, -1.0080e+00,
  4.5793e-01, -2.1920e-01, -2.1796e-01,  8.6871e-03, -4.7979e-01,
 -4.9734e-01,  1.5404e-01, -1.7627e+00, -5.0614e-01,  5.0527e-01,
  6.6560e-01, -5.9715e-01, -1.2055e+00,  1.2821e-01, -1.9179e-01,
 -1.5012e-01,  6.4959e-01, -2.7624e-01,  8.7947e-01,  7.0973e-01,
```

```
          1.4016e+00, -2.2328e-01, -7.9496e-02,  4.4955e-01,  4.0634e-01,
          1.1538e-02,  3.4065e-02,  4.5608e-01,  6.5496e-01, -1.0544e-01],
        [-1.2294e-01,  7.4850e-01, -2.4815e-01,  6.3324e-01,  3.2393e-01,
         -2.0612e-01, -9.0714e-02, -4.1749e-01,  1.6775e+00,  3.3086e-01,
          5.1199e-01, -4.9361e-01,  3.6294e-02,  3.6904e-01,  5.5728e-01,
         -3.7430e-01, -1.3459e+00, -2.8989e-01, -5.1558e-02, -1.1448e+00,
         -4.2478e-01, -4.9677e-01,  1.2526e+00,  7.5927e-01, -9.9442e-01,
          1.6015e-01,  1.0311e+00,  5.8884e-01,  1.2785e+00,  7.9789e-01,
          8.9351e-01, -5.8659e-02, -3.2933e-01, -3.7147e-01,  5.5327e-01,
         -1.2374e-03,  9.6847e-01, -1.9130e-01, -6.3718e-01,  5.0880e-01,
         -7.8397e-01,  9.5903e-01, -6.3904e-01,  2.3388e-01,  4.7308e-01,
         -1.0540e-01, -2.2563e-01, -5.7432e-01, -1.6020e+00, -1.9842e-01,
          6.9138e-01, -5.4280e-01, -8.4146e-02,  9.1433e-01, -1.5322e-01,
          5.5961e-01,  1.4389e-01,  8.5483e-01, -1.9744e-01,  1.9194e-01,
         -1.9398e+00,  1.8611e-01, -4.4823e-01,  7.4822e-01,  9.9097e-01,
         -3.9505e-03,  5.7359e-01,  2.6399e-02, -3.4092e-01,  1.3141e-01,
         -3.5033e-01, -1.1004e+00, -6.5422e-02,  4.4337e-01, -1.3527e+00,
          1.5685e+00, -1.4441e-01, -7.9155e-01,  9.3888e-01, -6.6552e-01,
          2.3192e-01, -2.3889e+00, -5.9441e-01,  1.6821e+00,  8.1150e-01,
          4.9227e-01,  5.2118e-01,  2.5105e-01,  3.8933e-01, -1.0273e+00,
          1.3896e+00, -4.4019e-01, -6.4065e-02,  5.8786e-01,  2.4355e-01,
         -5.6231e-01, -6.1447e-01, -3.2601e-01,  1.0584e-01, -1.5714e+00]])
Labels (first 5 nodes): tensor([[0],
        [1],
        [2],
        [3],
        [3]])
Data representation: Data(num_nodes=2449029, edge_index=[2, 123718024],
x=[2449029, 100], y=[2449029, 1])
printing done…
```

1.2 Model Creation Using PyTorch Geometric GCN with NeighborLoader for Node Classification

This code defines a Graph Convolutional Network (GCN) model using PyTorch Geometric, processes the dataset with NeighborLoader for efficient mini-batch training, and trains the model for node classification with dropout and validation.

[8]:
```python
from torch_geometric.loader import NeighborLoader
from sklearn.model_selection import train_test_split
import torch
from torch_geometric.nn import GCNConv
import torch.nn.functional as F
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns

# Define the GCN model with Dropout
```

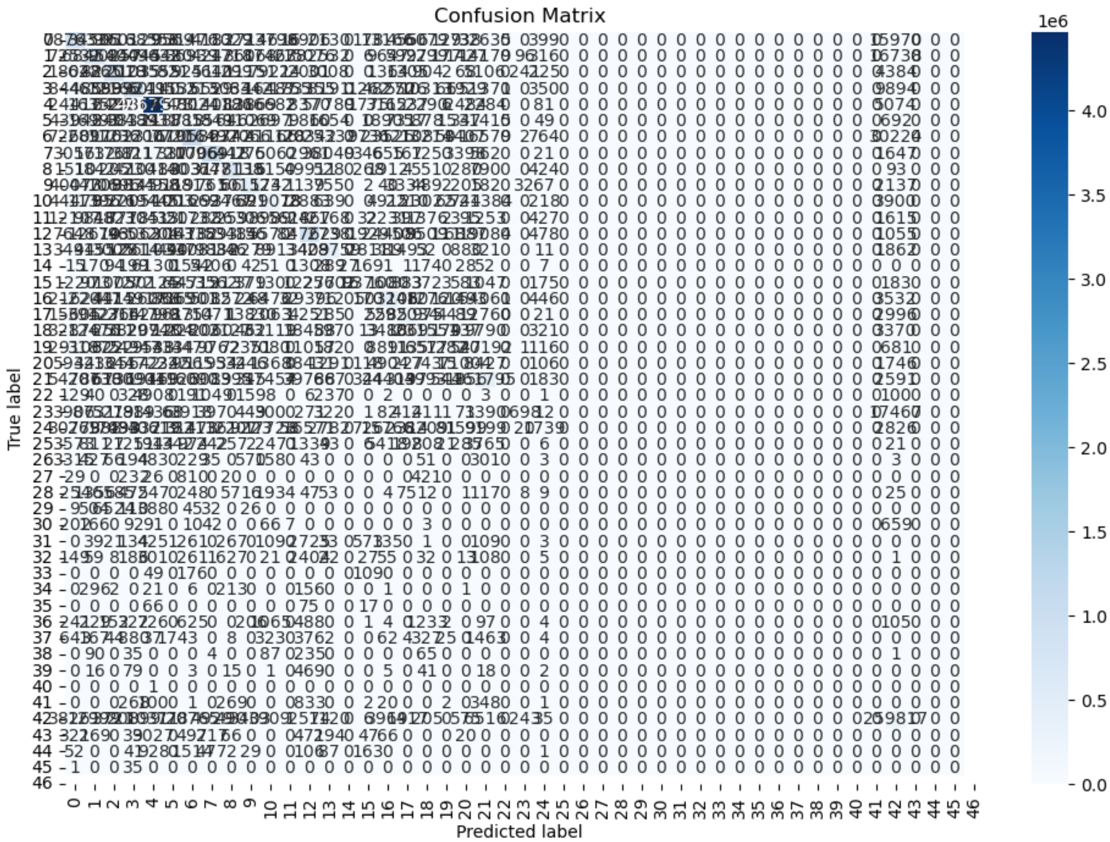
```
class GCN(torch.nn.Module):
    def __init__(self, in_channels, hidden_channels, out_channels, num_layers,␣
 ↪dropout=0.5):
        super(GCN, self).__init__()
        self.convs = torch.nn.ModuleList()
        self.convs.append(GCNConv(in_channels, hidden_channels))
        for _ in range(num_layers - 2):
            self.convs.append(GCNConv(hidden_channels, hidden_channels))
        self.convs.append(GCNConv(hidden_channels, out_channels))
        self.dropout = dropout

    def forward(self, x, edge_index):
        for conv in self.convs[:-1]:
            x = F.dropout(conv(x, edge_index).relu(), p=self.dropout,␣
 ↪training=self.training)
        x = self.convs[-1](x, edge_index)
        return x


# Data Splitting
num_nodes = data.num_nodes
train_idx, test_idx = train_test_split(range(num_nodes), test_size=0.2,␣
 ↪random_state=42)
train_idx, val_idx = train_test_split(train_idx, test_size=0.2, random_state=42)

data.train_mask = torch.zeros(num_nodes, dtype=torch.bool)
data.val_mask = torch.zeros(num_nodes, dtype=torch.bool)
data.test_mask = torch.zeros(num_nodes, dtype=torch.bool)

data.train_mask[train_idx] = True
data.val_mask[val_idx] = True
data.test_mask[test_idx] = True

# Normalize features
data.x = (data.x - data.x.mean(dim=0)) / data.x.std(dim=0)

# Define NeighborLoader for batching
train_loader = NeighborLoader(
    data,
    input_nodes=data.train_mask,
    num_neighbors=[25, 20],
    shuffle=True,
    batch_size=1024,
    pin_memory=True
)

val_loader = NeighborLoader(
```

```
    data,
    input_nodes=data.val_mask,
    num_neighbors=[25, 20],
    shuffle=False,
    batch_size=1024,
    pin_memory=True,
)

test_loader = NeighborLoader(
    data,
    input_nodes=data.test_mask,
    num_neighbors=[25, 20],
    shuffle=False,
    batch_size=1024,
    pin_memory=True
)

# Define model parameters
device = 'cuda' if torch.cuda.is_available() else 'cpu'

# Move data to the GPU
data.x = data.x.to(device)
data.edge_index = data.edge_index.to(device)
data.y = data.y.to(device)

in_channels = data.x.size(1)
hidden_channels = 256  # Increased for richer representations
out_channels = dataset.num_classes
num_layers = 4  # Reduced slightly for stability
learning_rate = 0.005  # Adjusted for better optimization
dropout = 0.5

# Initialize model, optimizer, scheduler, and loss function
model = GCN(in_channels, hidden_channels, out_channels, num_layers, dropout).
 ↪to(device)
optimizer = torch.optim.Adam(model.parameters(), lr=learning_rate,␣
 ↪weight_decay=5e-4)
scheduler = torch.optim.lr_scheduler.StepLR(optimizer, step_size=5, gamma=0.5)
criterion = torch.nn.CrossEntropyLoss()

# Move the model to the GPU
model.to(device)

# Training function
def train_with_loader(model, loader, optimizer, criterion):
    model.train()
    total_loss = 0
```

```python
    for batch in loader:
        batch = batch.to(device)
        optimizer.zero_grad()
        pred = model(batch.x, batch.edge_index)
        loss = criterion(pred[batch.train_mask], batch.y[batch.train_mask].squeeze())
        loss.backward()
        optimizer.step()
        total_loss += loss.item()
    return total_loss / len(loader)


# Validation and Testing function
def evaluate_with_loader(model, loader, criterion):
    model.eval()
    total_loss = 0
    correct = 0
    total = 0
    with torch.no_grad():
        for batch in loader:
            batch = batch.to(device)
            pred = model(batch.x, batch.edge_index)
            loss = criterion(pred[batch.train_mask], batch.y[batch.train_mask].squeeze())
            total_loss += loss.item()
            correct += (pred.argmax(dim=1)[batch.train_mask] == batch.y[batch.train_mask].squeeze()).sum().item()
            total += batch.train_mask.sum().item()
    return total_loss / len(loader), correct / total

# Function to plot confusion matrix
def plot_confusion_matrix(cm, class_names):
    plt.figure(figsize=(12, 8))
    sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=class_names, yticklabels=class_names)
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
    plt.title('Confusion Matrix')
    plt.show()

# Evaluation function to collect predictions for confusion matrix
def get_predictions_for_confusion_matrix(model, loader):
    model.eval()
    all_preds = []
    all_labels = []
    with torch.no_grad():
        for batch in loader:
```

```
            batch = batch.to(device)
            pred = model(batch.x, batch.edge_index)
            all_preds.append(pred.argmax(dim=1)[batch.test_mask].cpu().numpy())
            all_labels.append(batch.y[batch.test_mask].cpu().numpy())
    all_preds = np.concatenate(all_preds, axis=0)
    all_labels = np.concatenate(all_labels, axis=0)
    return all_preds, all_labels

# Training and Evaluation loop
epochs = 5
best_val_acc = 0
for epoch in range(epochs):
    train_loss = train_with_loader(model, train_loader, optimizer, criterion)
    val_loss, val_acc = evaluate_with_loader(model, val_loader, criterion)
    scheduler.step()
    if val_acc > best_val_acc:
        best_val_acc = val_acc
        best_model_state = model.state_dict()
    print(f"Epoch: {epoch+1}, Train Loss: {train_loss:.4f}, Val Loss: {val_loss:
 ↪.4f}, Val Acc: {val_acc:.4f}")

# Load the best model and test
model.load_state_dict(best_model_state)
test_loss, test_acc = evaluate_with_loader(model, test_loader, criterion)
print(f"Test Loss: {test_loss:.4f}, Test Accuracy: {test_acc:.4f}")

# Generate predictions for confusion matrix
test_preds, test_labels = get_predictions_for_confusion_matrix(model,␣
 ↪test_loader)

# Compute the confusion matrix
cm = confusion_matrix(test_labels, test_preds)

# Plot the confusion matrix
class_names = [str(i) for i in range(out_channels)]  # Adjust based on your␣
 ↪number of classes
plot_confusion_matrix(cm, class_names)
```

```
Epoch: 1, Train Loss: 1.3547, Val Loss: 1.0810, Val Acc: 0.7047
Epoch: 2, Train Loss: 1.2767, Val Loss: 1.0797, Val Acc: 0.7026
Epoch: 3, Train Loss: 1.2737, Val Loss: 1.0817, Val Acc: 0.7034
Epoch: 4, Train Loss: 1.2718, Val Loss: 1.0771, Val Acc: 0.7042
Epoch: 5, Train Loss: 1.2715, Val Loss: 1.0794, Val Acc: 0.7043
Test Loss: 1.0801, Test Accuracy: 0.7039
```



2 Plot the main graph with label

Plotting the whole graph require a good amount of GPU resources, due to lack of resources we adopted a conversion from PyTorch Geometric graph to an igraph object for visualization. This approach use Fruchterman-Reingold layout to arrange nodes evenly and saves the graph as a PNG file. This approach was chosen for its efficiency in handling graphs and the appealing layout provided by the Fruchterman-Reingold algorithm.

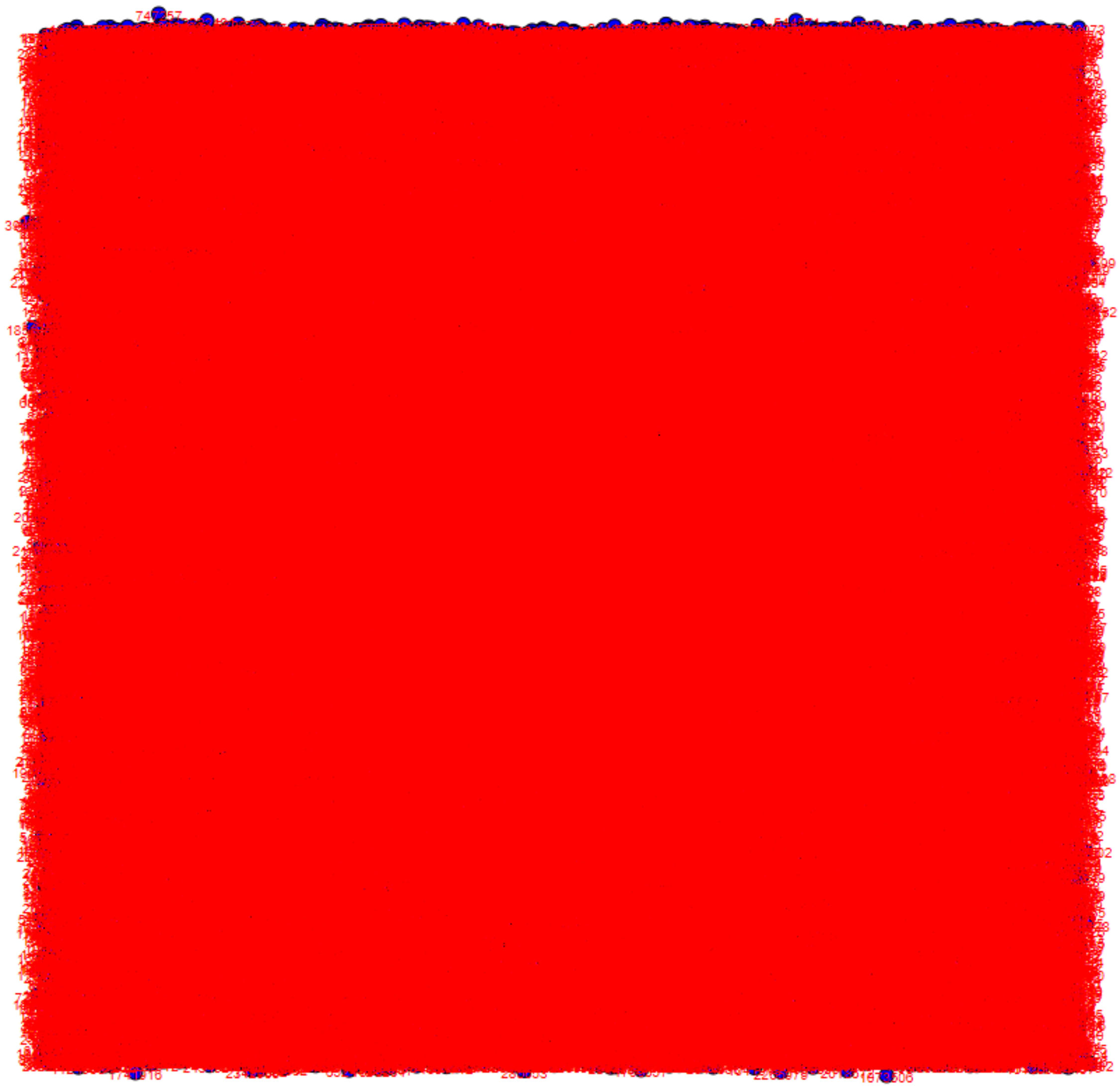
[10]:
```
from igraph import Graph, plot
import cairo

# Convert PyTorch Geometric graph to igraph
def to_igraph(data):
    edges = data.edge_index.cpu().numpy().T.tolist()  # Convert edges to a list
 ↪of tuples
    num_nodes = data.num_nodes
    G = Graph(n=num_nodes, edges=edges, directed=False)
    # Set default node labels as node indices
    G.vs["label"] = list(range(num_nodes))  # Add node labels as vertex
 ↪attributes
```

```
    return G

# Visualize with igraph and export to file
def visualize_with_igraph(data, output_file="graph.png"):
    G = to_igraph(data)
    layout = G.layout("fr")  # Fruchterman-Reingold layout
    visual_style = {
        "vertex_size": 10,
        "vertex_color": "blue",
        "edge_color": "gray",
        "vertex_label": G.vs["label"],  # Specify node labels
        "vertex_label_size": 10,       # Set label size
        "vertex_label_color": "red",        # Set label color
        "bbox": (800, 800),
        "margin": 20
    }
    # Save the graph to a file
    plot(G, target=output_file, layout=layout, **visual_style)
    print(f"Graph saved to {output_file}")

# Usage
visualize_with_igraph(data, output_file="graph.png")
```

Graph saved to graph.png

[11]:

```
from IPython.display import Image, display

# Display the generated PNG image
output_file = "graph.png"
display(Image(filename=output_file))
```



2.1 Generate Node Induced Subgraph

Justification for Subgraph Generation

2.1.1 Relevant Material

Subgraph generation involves sampling and creating a subset of nodes and edges while preserving relationships.

2.1.2 Explanation

1. **Node Sampling**: Randomly selects a subset of nodes.

2. **Subgraph Extraction**: Uses PyTorch Geometric's `subgraph()` to create a node-induced subgraph, including edges connecting the sampled nodes.

3. **Reindexing**: Reindexes nodes for independent structure.

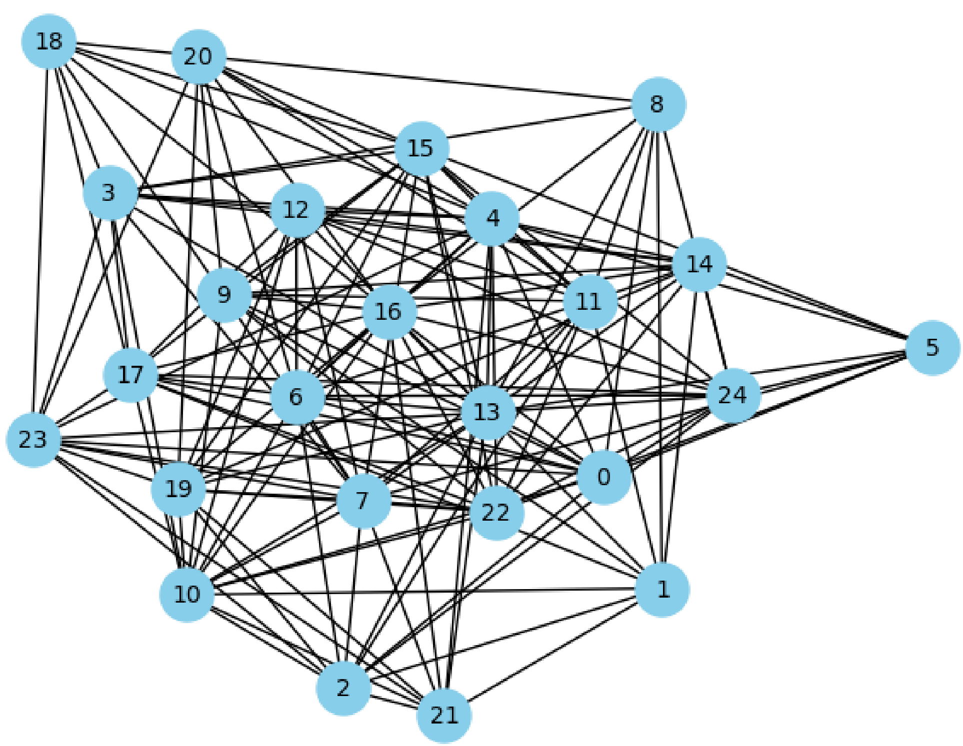
4. **Analysis**: Converts the subgraph to NetworkX for visualization and analysis.

NOTE: This method is efficient and practical for large-scale graphs like `ogbn-products`.

[13]:
```
from torch_geometric.utils import to_scipy_sparse_matrix, subgraph
import random
import networkx as nx
import matplotlib.pyplot as plt
from networkx.algorithms.cluster import average_clustering

print("Draw subgraph")
# Parameters for sampling
num_sample_nodes = 20000  # Adjust this value based on available RAM
sample_nodes = random.sample(range(data.num_nodes), num_sample_nodes)

# Create a subgraph using sampled nodes
subset_edge_index, _ = subgraph(sample_nodes, data.edge_index,␣
 ↪relabel_nodes=True)

# Convert the subgraph edges to a NetworkX graph
print("Convert subgraph")
G_sub = nx.Graph()
G_sub.add_edges_from(subset_edge_index.t().tolist())

# Ensure the subgraph is connected, otherwise work on the largest connected␣
 ↪component
if not nx.is_connected(G_sub):
    largest_cc = max(nx.connected_components(G_sub), key=len)
    G_sub = G_sub.subgraph(largest_cc)

# Calculate the diameter for the largest connected component
print("Calculate the diameter")
diameter_approx = nx.diameter(G_sub)
print(f"Approximated Diameter (based on largest connected subgraph):␣
 ↪{diameter_approx}")

# Convert the subgraph to a sparse adjacency matrix for clustering coefficient
adj_matrix = to_scipy_sparse_matrix(subset_edge_index,␣
 ↪num_nodes=num_sample_nodes)

# Use NetworkX to compute clustering coefficient on the sampled graph
sampled_graph = nx.Graph(adj_matrix)
```

```
global_clustering_coeff_approx = average_clustering(sampled_graph,␣
    ↪count_zeros=True)
print(f"Approximated Global Clustering Coefficient:␣
    ↪{global_clustering_coeff_approx}")

# Create a sample graph
print("create sample graph")
G = nx.erdos_renyi_graph(25, 0.5)  # A random graph with 10 nodes and edge␣
    ↪probability of 0.5

# Draw the graph

nx.draw(G, with_labels=True, node_color='skyblue', node_size=500, font_size=10)

# Show the plot
#plt.show()
```

```
Draw subgraph
Convert subgraph
Calculate the diameter
Approximated Diameter (based on largest connected subgraph): 37
Approximated Global Clustering Coefficient: 0.020064591380841382
create sample graph
```



2.2 Node embedding using 2-hop method for all nodes in subgraph using MP-GNN library in PyTorch Geometric

The flow would be, - Extract features and edges for the subgraph (subgraph_x, subgraph_edge_index). - Define the TwoHopGCN architecture with the required input, hidden, and output dimensions. - Initialize an optimizer to train the model.

[15]:
```
from torch_geometric.nn import GCNConv
import torch.nn as nn
import torch_geometric.data as geom_data

# Define a TwoHopGCN Model
class TwoHopGCN(nn.Module):
    def __init__(self, in_channels, hidden_channels, out_channels, num_layers):
        super(TwoHopGCN, self).__init__()
        self.convs = nn.ModuleList()
        self.convs.append(GCNConv(in_channels, hidden_channels))
        for _ in range(num_layers - 2):
            self.convs.append(GCNConv(hidden_channels, hidden_channels))
        self.convs.append(GCNConv(hidden_channels, out_channels))
        self.relu = nn.ReLU()

    def forward(self, x, edge_index):
        for conv in self.convs[:-1]:
            x = conv(x, edge_index)
            x = self.relu(x)
        x = self.convs[-1](x, edge_index)
        return x

# Extract features and edge index for the subgraph
subgraph_x = data.x[sample_nodes]  # Subgraph node features
subgraph_edge_index = subset_edge_index  # Subgraph edges

# Define model parameters
sub_in_channels = subgraph_x.size(1)
sub_hidden_channels = 64
sub_out_channels = 64
sub_num_layers = 2  # Use 2-hop aggregation
learning_rate = 0.01

# Initialize TwoHopGCN model, optimizer, and loss function
subgraph_model = TwoHopGCN(sub_in_channels, sub_hidden_channels,␣
 ↪sub_out_channels, sub_num_layers)
sub_optimizer = torch.optim.Adam(subgraph_model.parameters(), lr=learning_rate)
```

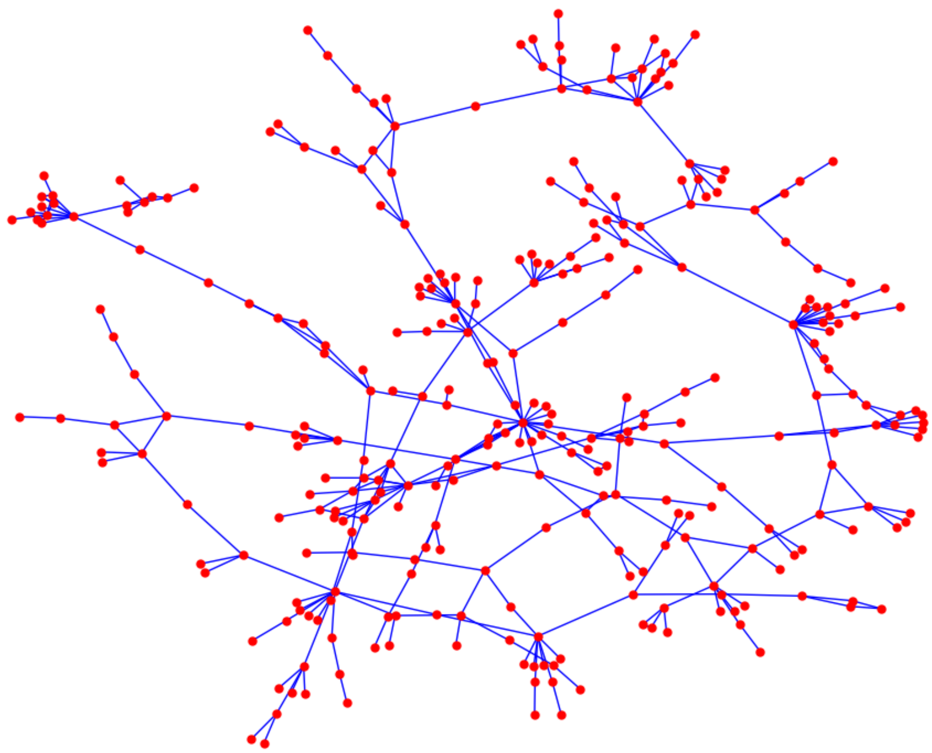
2.3 Plot Subgrpah and compute their Diameter

Generate node embeddings for the subgraph using the trained TwoHopGCN model.

[17]:
```
# Generate Node Embeddings for Subgraph
subgraph_model.eval()
with torch.no_grad():
    node_embeddings = subgraph_model(subgraph_x, subgraph_edge_index)
print("Generated Node Embeddings Shape:", node_embeddings.shape)

# Plot the Subgraph
plt.figure(figsize=(10, 8))
pos = nx.spring_layout(G_sub)  # Layout for better visualization
nx.draw(G_sub, pos, with_labels=False, node_color='red', edge_color='blue',␣
 ↪node_size=25, font_size=10)
plt.title("Subgraph Visualization")
plt.show()

# Re-compute Diameter for Updated Subgraph
diameter_approx = nx.diameter(G_sub)
print(f"Updated Diameter (based on largest connected subgraph):␣
 ↪{diameter_approx}")
```

```
Generated Node Embeddings Shape: torch.Size([20000, 64])
```

Subgraph Visualization



```
Updated Diameter (based on largest connected subgraph): 37
```

2.3.1 Justification:

We have taken only 20K nodes to create sub graph due to memory constraint. This can be adjusted into code with higher number of nodes.

2.3.2 Observation:

The subgraph's diameter of 37 indicates that the farthest two nodes are separated by 37 hops, suggesting sparse connectivity or a widely spread structure. This could impact tasks requiring long-range information propagation, such as message passing in GCNs.

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```