

Programming Assignment

Q1

Model Structure

My model has 4 convolution layers decreasing in size gradually from the input number of features to 1. I have a combination of relu and gelu.

```
def __init__(self, n_features):  
    """  
    n_features: number of features from dataset, should be 37  
    """  
    super(GraphNet, self).__init__()  
    # define your GNN model here  
    self.conv1 = GCNConv(n_features, 25)  
    self.conv2 = GCNConv(25, 18)  
    self.conv3 = GCNConv(18, 12)  
    self.conv4 = GCNConv(12, 1)  
    #self.conv5 = GCNConv(5, 1)  
    #raise NotImplementedError  
  
def forward(self, data):  
    # define the forward pass here  
    x, edge_index, edge_weight = data.x, data.edge_index, data.edge_attr  
    x = F.gelu(self.conv1(x, edge_index))  
    #x = F.dropout(x, training=self.training, p=0.1)  
    x = F.relu(self.conv2(x, edge_index))  
    #x = F.dropout(x, training=self.training, p=0.2)  
    x = F.gelu(self.conv3(x, edge_index))  
    #x = F.dropout(x, training=self.training, p=0.2)  
    x = F.relu(self.conv4(x, edge_index))  
    x = self.conv4(x, edge_index)  
    #x = F.log_softmax(x, dim=0)  
    return scatter_mean(torch.squeeze(x), data.batch)  
    #raise NotImplementedError
```

Hyperparameters

After multiple iterations with the hyperparameters by changing the optimizer, scheduler, lr and the number of epochs I found the following combination of parameters that worked well for my case

Optimizer: RMSprop

Loss: MSE Loss

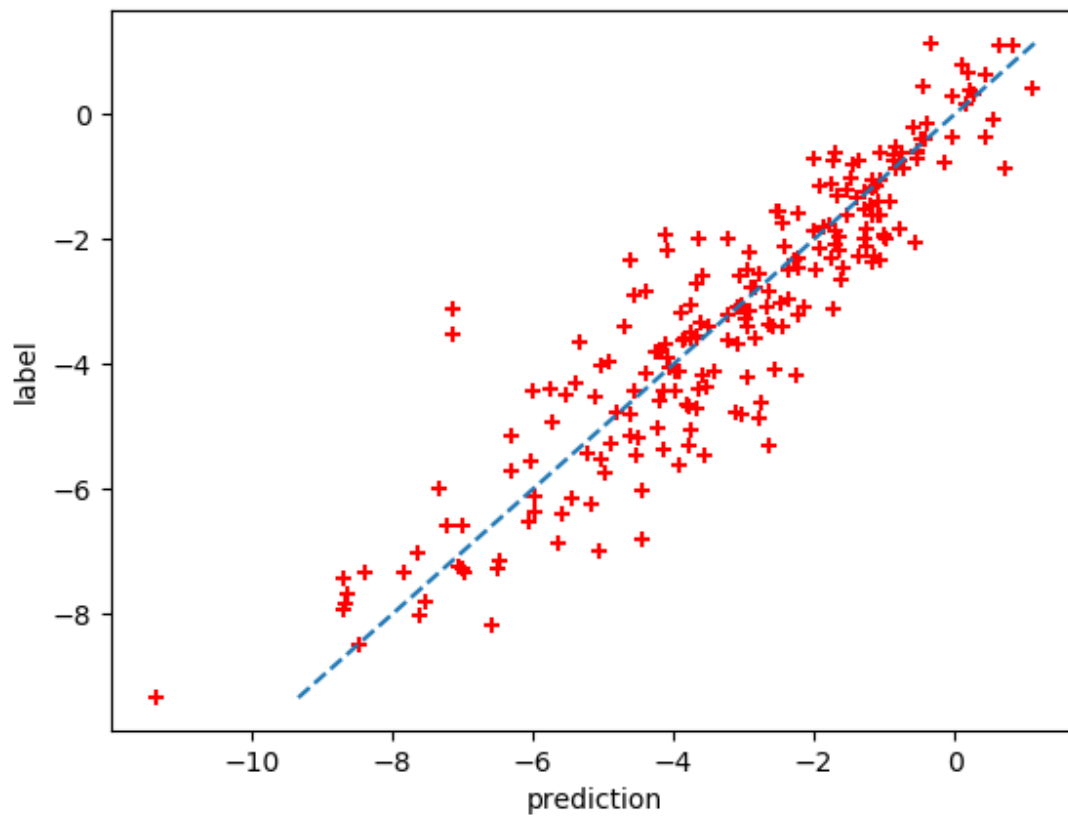
Lr: $3e-3$

Number of Epochs: 300

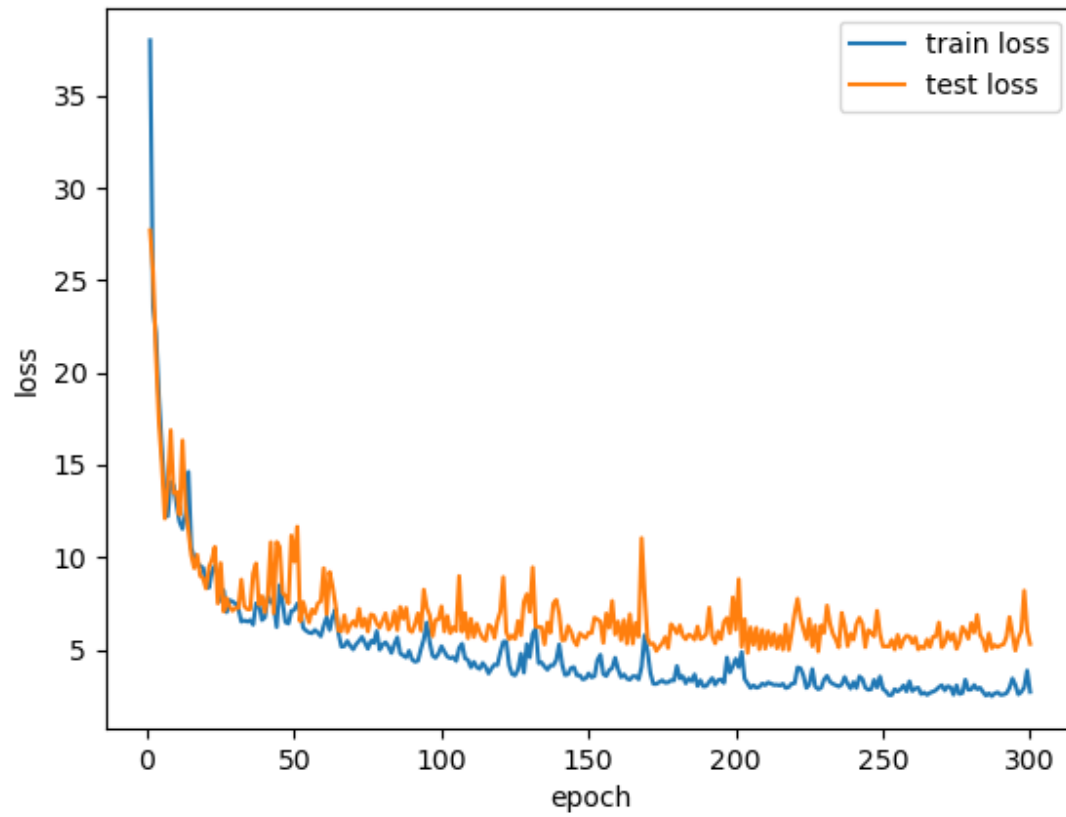
Scheduler: CosineAnnealingLR with $\eta_{\min} = 1e-3$

I tried Adam, and SGD with momentum and weight decay and also varying number of epochs and learning rates but they did not work well.

Performance on test set

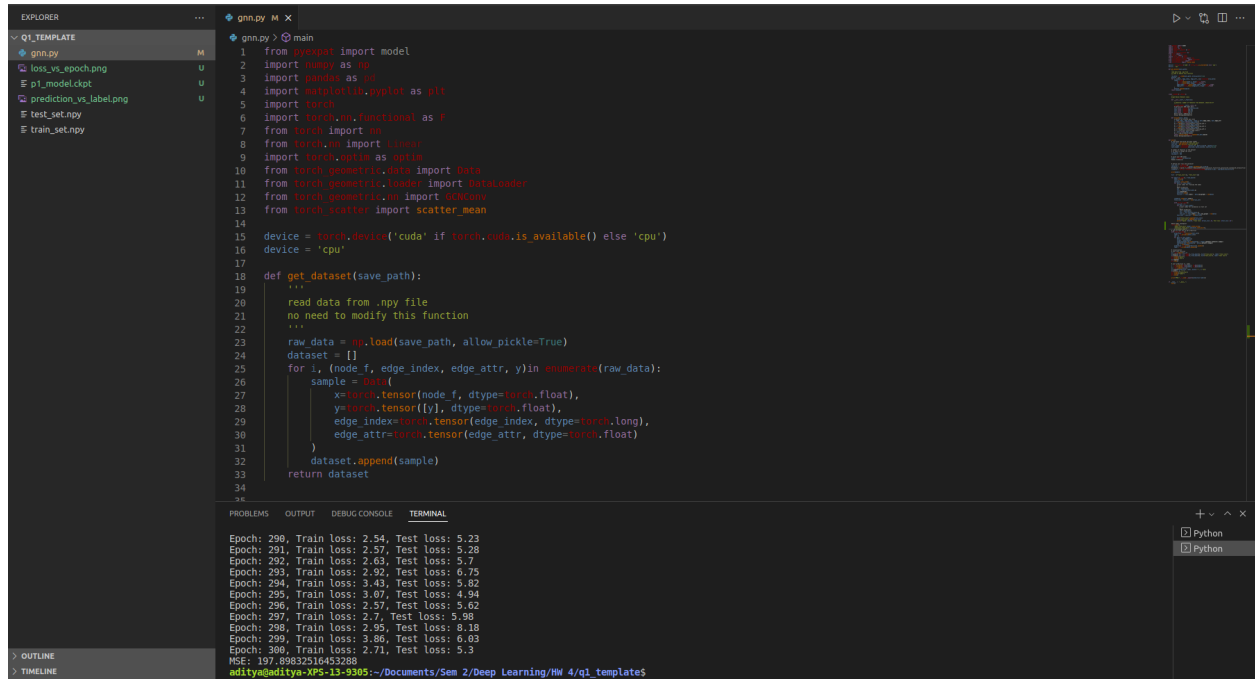


Training Process



MSE Loss

The final MSE loss that I got on the test set was 197.89



The image shows a VS Code editor with a Python file named `gnn.py` and its output in the terminal. The script defines a `get_dataset` function and a training loop. The terminal output shows the training progress from epoch 290 to 300, with the final MSE loss of 197.89.

```
1 from pytorch import model
2 import numpy as np
3 import random as r
4 import matplotlib.pyplot as plt
5 import torch
6 import torch.nn.functional as F
7 from torch import nn
8 from torch.nn import Linear
9 import torch.nn as nn
10 from torch_geometric.data import Data
11 from torch_geometric.loader import DataLoader
12 from torch_geometric.nn import GCNConv
13 from torch.nn import scatter_mean
14
15 device = 'cuda' if torch.cuda.is_available() else 'cpu'
16 device = 'cpu'
17
18 def get_dataset(save_path):
19     """
20     read data from .npz file
21     no need to modify this function
22     """
23     raw_data = np.load(save_path, allow_pickle=True)
24     dataset = []
25     for i, (node_f, edge_index, edge_attr, y) in enumerate(raw_data):
26         sample = {}
27         x=torch.tensor(node_f, dtype=torch.float),
28         y=torch.tensor(y, dtype=torch.float),
29         edge_index=torch.tensor(edge_index, dtype=torch.long),
30         edge_attr=torch.tensor(edge_attr, dtype=torch.float)
31     }
32     dataset.append(sample)
33     return dataset
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

Epoch: 290, Train loss: 2.54, Test loss: 5.23
Epoch: 291, Train loss: 2.57, Test loss: 5.28
Epoch: 292, Train loss: 2.63, Test loss: 5.7
Epoch: 293, Train loss: 2.92, Test loss: 6.75
Epoch: 294, Train loss: 3.43, Test loss: 5.82
Epoch: 295, Train loss: 3.07, Test loss: 4.94
Epoch: 296, Train loss: 2.57, Test loss: 5.62
Epoch: 297, Train loss: 2.7, Test loss: 5.98
Epoch: 298, Train loss: 2.95, Test loss: 8.18
Epoch: 299, Train loss: 3.86, Test loss: 6.03
Epoch: 300, Train loss: 2.71, Test loss: 5.3
MSE: 197.89832516453288

nditya@nditya-XPS-13-9305:~/Documents/Sem 2/Deep Learning/IM 4/q1 templates