Research and Programming Assignment

Task

The task is related to the G2P2 model as described in the paper link given below. Extend the G2P2 model to zero-shot prompt tuning. The basic idea is to use the zero-shot module in G2P2 to assign pseudo labels to each node in the unlabelled pre-training graph. Then, employ the few-shot module in G2P2 to perform prompt tuning. The paper and code can be found at the links below.

Paper: https://smufang.github.io/paper/SIGIR23_G2P2.pdf
Code: https://github.com/smufang/G2P2

Dataset

You only need to use our provided Cora dataset, and its five different testing splits (download here).

Sample code to load the data and splits:

Loading the text data and corresponding sample labels:

```
num_nodes = 0
tit_list = []
lab_list = []
with open('data/cora_train_text.txt', 'r') as f:
    lines = f.readlines()
    for line in lines:
        line = line.strip().split('\t')
        tit_list.append(line[2])
        lab_list.append(line[3])
        num_nodes += 1
```

Selecting only the ids that have annotated labels:

```
labeled_ids = []

for i in range(len(lab_list)):
    if lab_list[i] != 'nan':
        labeled_ids.append(i)
```

Loading the edges:

```
raw_edge_index = [[], []]
with open('data/cora_mapped_edges.txt', 'r') as f:
    lines = f.readlines()
    for line in lines:
        line = line.strip().split()
        raw_edge_index[0].append(int(line[0]))
        raw_edge_index[1].append(int(line[1]))

edge_index = [raw_edge_index[0] + raw_edge_index[1], raw_edge_index[1] +
raw_edge_index[0]]
```

```
arr_edge_index = np.array(edge_index)
```

Loading the node features:

```
node_f = np.load('data/cora_node_f.npy')
node_f = preprocessing.StandardScaler().fit_transform(node_f)
```

Loading the list of classes:

```
with open('data/cora_lab_list.txt', 'r') as f:
    line = f.readline().strip().split('\t')
    label_texts = line

labels = []
for i in label_texts:
    if i != 'nan':
        labels.append(i)
```

Loading the splits as follows. Different seeds refers to different splits. `text_idx` contains the indices of the testing nodes for each zero-shot node classification task.

```
checkpoint_org =
torch.load('./org_seed_data/cora_org_data_seed{}'.format(seed))
task_list, train_idx, val_idx, test_idx = checkpoint_org['task_list'],
checkpoint_org['train_idx'], checkpoint_org['val_idx'],
checkpoint_org['test_idx']
```

Pretrained model

The pre-trained model for Cora is provided (download here).

Sample code to load the pre-trained model:

```
model = CLIP(args).to(device)
model.load_state_dict(torch.load('./res/{}/node_ttgt_8&12_0.1.pkl'.format(data
_name), map_location=device))
```

Implementation instructions

1. Use the G2P2 approach to label the nodes in the pre-training graph in a zero-shot fashion, using a discrete prompt “a + [ClassName]”, for all the 70 classes in the Cora dataset.

2. For each class in a given downstream classification task, we sample 200 nodes with the corresponding pseudo labels obtained from the first step. In case a class does not have 200 pseudo labelled nodes, we choose all the nodes available for the class instead.

3. For each classification task in each split, perform prompt tuning using the sampled 200 nodes per class. You may try different values for other hyperparameters (such as batch size, learning rate, number of epochs) as you see fit.

4. For each classification task in each split, perform inference on the test nodes using the learned prompt, and evaluate the prediction accuracy.

Minimum Hardware Requirements

It is preferable if you have access to an GPU with minimum 5GB memory. Otherwise, it is also possible to run on a reasonably good CPU with multiple threads, requiring minimum 8GB memory.

Deliverables

1. Code (a well-documented python script or Jupyter Notebook)
2. Pseudocode describing the high-level steps of your algorithm.
3. Detailed hyperparameters tried and used.
4. Visualization and interpretation of the training curves for prompt tuning. You may pick 3 to 5 classification tasks for illustration.
5. Report the average and standard deviation of prediction accuracy over the five splits.

G2P2	63.52 ± 2.89
G2P2 + d	65.28 ± 3.12
Pseudo-label approach	