4.4 Save and load data

(中文版)

DGL recommends implementing saving and loading functions to cache the processed data in local disk. This saves a lot of data processing time in most cases. DGL provides four functions to make things simple:

- `dgl.save_graphs()` and `dgl.load_graphs()`: save/load DGLGraph objects and labels to/from local disk.
- `dgl.data.utils.save_info()` and `dgl.data.utils.load_info()`: save/load useful information of the dataset (python `dict` object) to/from local disk.

The following example shows how to save and load a list of graphs and dataset information.

```python
import os
from dgl import save_graphs, load_graphs
from dgl.data.utils import makedirs, save_info, load_info

def save(self):
    # save graphs and labels
    graph_path = os.path.join(self.save_path, self.mode + '_dgl_graph.bin')
    save_graphs(graph_path, self.graphs, {'labels': self.labels})
    # save other information in python dict
    info_path = os.path.join(self.save_path, self.mode + '_info.pkl')
    save_info(info_path, {'num_classes': self.num_classes})

def load(self):
    # load processed data from directory `self.save_path`
    graph_path = os.path.join(self.save_path, self.mode + '_dgl_graph.bin')
    self.graphs, label_dict = load_graphs(graph_path)
    self.labels = label_dict['labels']
    info_path = os.path.join(self.save_path, self.mode + '_info.pkl')
    self.num_classes = load_info(info_path)['num_classes']

def has_cache(self):
    # check whether there are processed data in `self.save_path`
    graph_path = os.path.join(self.save_path, self.mode + '_dgl_graph.bin')
    info_path = os.path.join(self.save_path, self.mode + '_info.pkl')
    return os.path.exists(graph_path) and os.path.exists(info_path)
```

Note that there are cases not suitable to save processed data. For example, in the builtin dataset `GDELTDataset`, the processed data is quite large, so it's more effective to process each data example in `__getitem__(idx)`.