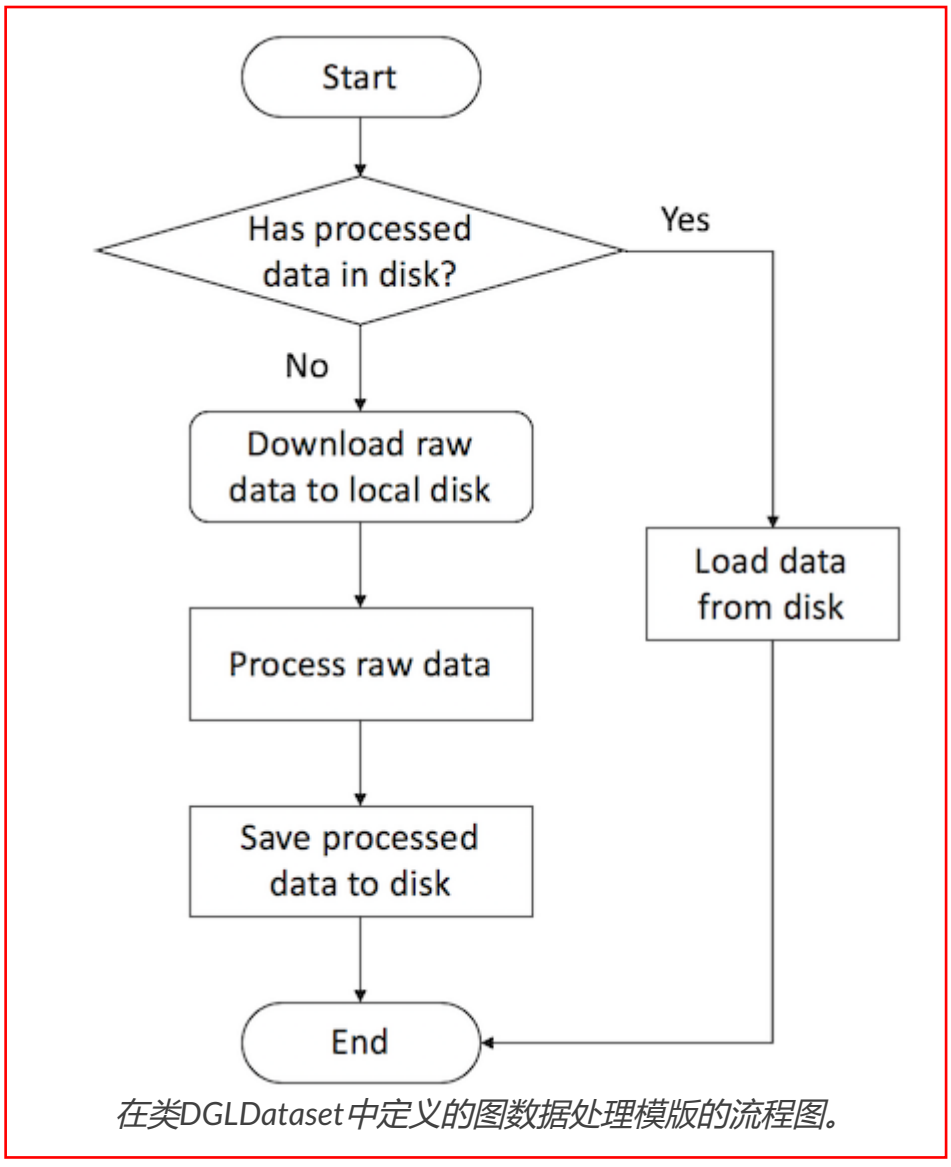
4.1 DGLDataset类

(English Version)

`DGLDataset` 是处理、导入和保存 dgl.data 中定义的图数据集的基类。 它实现了用于处理图数据的基本模版。下面的流程图展示了这个模版的工作方式。



在类DGLDataset中定义的图数据处理模版的流程图。

为了处理位于远程服务器或本地磁盘上的图数据集，下面的例子中定义了一个类，称为 `MyDataset`，它继承自 `dgl.data.DGLDataset`。

```python
from dgl.data import DGLDataset

class MyDataset(DGLDataset):
    """ 用于在DGL中自定义图数据集的模板：

    Parameters
    ----------
    url : str
        下载原始数据集的url。
    raw_dir : str
        指定下载数据的存储目录或已下载数据的存储目录。默认：~/.dgl/
    save_dir : str
        处理完成的数据集的保存目录。默认：raw_dir指定的值
    force_reload : bool
        是否重新导入数据集。默认：False
    verbose : bool
        是否打印进度信息。
    """
    def __init__(self,
                 url=None,
                 raw_dir=None,
                 save_dir=None,
                 force_reload=False,
                 verbose=False):
        super(MyDataset, self).__init__(name='dataset_name',
                                        url=url,
                                        raw_dir=raw_dir,
                                        save_dir=save_dir,
                                        force_reload=force_reload,
                                        verbose=verbose)

    def download(self):
        # 将原始数据下载到本地磁盘
        pass

    def process(self):
        # 将原始数据处理为图、标签和数据集划分的掩码
        pass

    def __getitem__(self, idx):
        # 通过idx得到与之对应的一个样本
        pass

    def __len__(self):
        # 数据样本的数量
        pass

    def save(self):
        # 将处理后的数据保存至 `self.save_path`
        pass

    def load(self):
        # 从 `self.save_path` 导入处理后的数据
        pass

    def has_cache(self):
        # 检查在 `self.save_path` 中是否存有处理后的数据
        pass
```

`DGLDataset` 类有抽象函数 `process()`， `__getitem__(idx)` 和 `__len__()`。子类必须实现这些函数。同时DGL也建议实现保存和导入函数， 因为对于处理后的大型数据集，这么做可以节省大量的时间， 并且有多个已有的API可以简化此操作(请参阅 4.4 保存和加载数据)。

请注意， `DGLDataset` 的目的是提供一种标准且方便的方式来导入图数据。 用户可以存储有关数据集的图、特征、标签、掩码，以及诸如类别数、标签数等基本信息。 诸如采样、划分或特征归一化等操作建议在 `DGLDataset` 子类之外完成。

本章的后续部分展示了实现这些函数的最佳实践。