

3.1 DGL NN模块的构造函数

(English Version)

构造函数完成以下几个任务：

1. 设置选项。
2. 注册可学习的参数或者子模块。
3. 初始化参数。

```
import torch.nn as nn

from dgl.utils import expand_as_pair

class SAGEConv(nn.Module):
    def __init__(self,
                  in_feats,
                  out_feats,
                  aggregator_type,
                  bias=True,
                  norm=None,
                  activation=None):
        super(SAGEConv, self).__init__()

        self._in_src_feats, self._in_dst_feats = expand_as_pair(in_feats)
        self._out_feats = out_feats
        self._aggre_type = aggregator_type
        self.norm = norm
        self.activation = activation
```

在构造函数中，用户首先需要设置数据的维度。对于一般的PyTorch模块，维度通常包括输入的维度、输出的维度和隐层的维度。对于图神经网络，输入维度可被分为源节点特征维度和目标节点特征维度。

除了数据维度，图神经网络的一个典型选项是聚合类型(`self._aggre_type`)。对于特定目标节点，聚合类型决定了如何聚合不同边上的信息。常用的聚合类型包括 `mean`、`sum`、`max` 和 `min`。一些模块可能会使用更加复杂的聚合函数，比如 `lstm`。

上面代码里的 `norm` 是用于特征归一化的可调用函数。在SAGEConv论文里，归一化可以是L2归一化: $h_v = h_v / \|h_v\|_2$ 。

```
# 聚合类型: mean、pool、lstm、gcn
if aggregator_type not in ['mean', 'pool', 'lstm', 'gcn']:
    raise KeyError('Aggregator type {} not supported.'.format(aggregator_type))
if aggregator_type == 'pool':
    self.fc_pool = nn.Linear(self._in_src_feats, self._in_src_feats)
if aggregator_type == 'lstm':
    self.lstm = nn.LSTM(self._in_src_feats, self._in_src_feats, batch_first=True)
if aggregator_type in ['mean', 'pool', 'lstm']:
    self.fc_self = nn.Linear(self._in_dst_feats, out_feats, bias=bias)
self.fc_neigh = nn.Linear(self._in_src_feats, out_feats, bias=bias)
self.reset_parameters()
```

注册参数和子模块。在SAGEConv中，子模块根据聚合类型而有所不同。这些模块是纯PyTorch NN模块，例如 `nn.Linear`、`nn.LSTM` 等。构造函数的最后调用了 `reset_parameters()` 进行权重初始化。

```
def reset_parameters(self):
    """重新初始化可学习的参数"""
    gain = nn.init.calculate_gain('relu')
    if self._aggre_type == 'pool':
        nn.init.xavier_uniform_(self.fc_pool.weight, gain=gain)
    if self._aggre_type == 'lstm':
        self.lstm.reset_parameters()
    if self._aggre_type != 'gcn':
        nn.init.xavier_uniform_(self.fc_self.weight, gain=gain)
    nn.init.xavier_uniform_(self.fc_neigh.weight, gain=gain)
```