图神经网络七日打卡营

小斯妹　百度PGL团队成员

2020.11.26

昨天的一些问题

Message Passing 相关函数的参数解释：

昨天的一些问题

```python
def gcn_layer(gw, feature, hidden_size, activation, name, norm=None):

    # send函数
    def send_func(src_feat, dst_feat, edge_feat):
        '''
```

度

PGL源码： https://github.com/PaddlePaddle/PGL/blob/main/examples/gcn/train.py#L38

```python
# normalize
indegree = dataset.graph.indegree()
norm = np.zeros_like(indegree, dtype="float32")
norm[indegree > 0] = np.power(indegree[indegree > 0], -0.5)
dataset.graph.node_feat["norm"] = np.expand_dims(norm, -1)
```

$D^{-\frac{1}{2}}$ ，压缩为一维向量

(num_nodes, 1)

昨天的一些问题

PGL 源码：
https://github.com/PaddlePaddle/PGL/blob/main/pgl/layers/conv.py#L27

```
        if norm is not None:
            feature = feature * norm

        msg = gw.send(send_src_copy, nfeat_list=[("h", feature)])

        if size > hidden_size:
            output = gw.recv(msg, "sum")
        else:
            output = gw.recv(msg, "sum")
            output = L.fc(output,
                          size=hidden_size,
                          bias_attr=False,
                          param_attr=fluid.ParamAttr(name=name))

        if norm is not None:
            output = output * norm
```

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right)$$

课程大纲

后续：新冠项目实战，带你助力疫情防控

参考材料：

- 斯坦福CS224W课程： http://cs224w.stanford.edu
- 图学习库 PGL：https://github.com/PaddlePaddle/PGL

飞桨

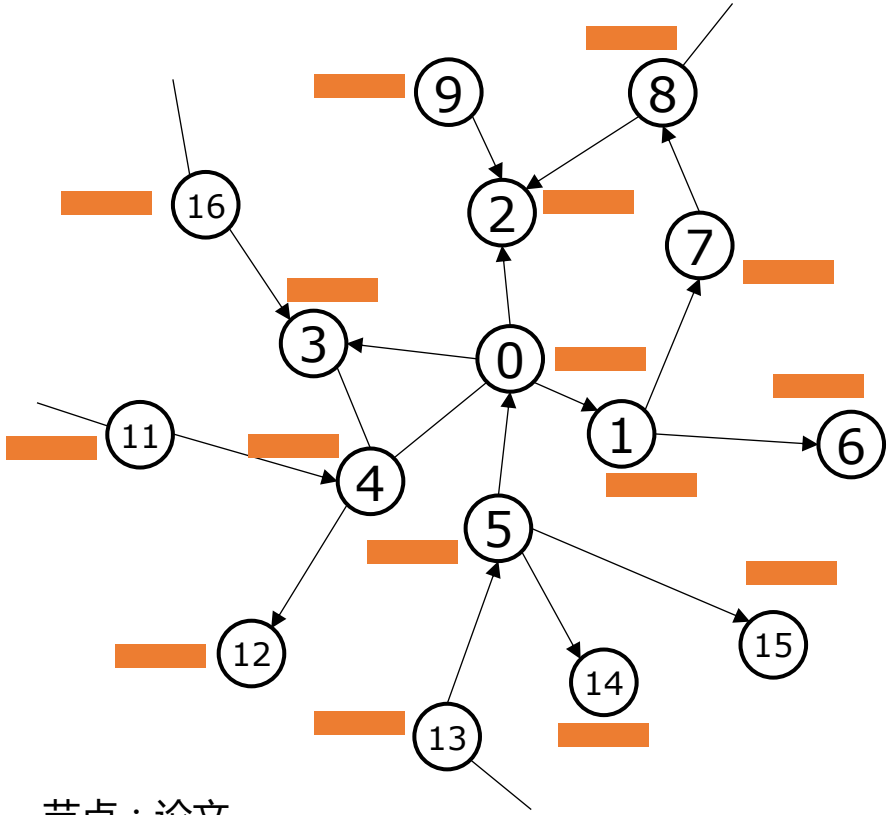
第四课 图神经网络算法(二)

小斯妹 百度PGL团队成员

2020.11.26

回顾上节课任务

Cora：引用论文节点分类数据集(7个分类类别)

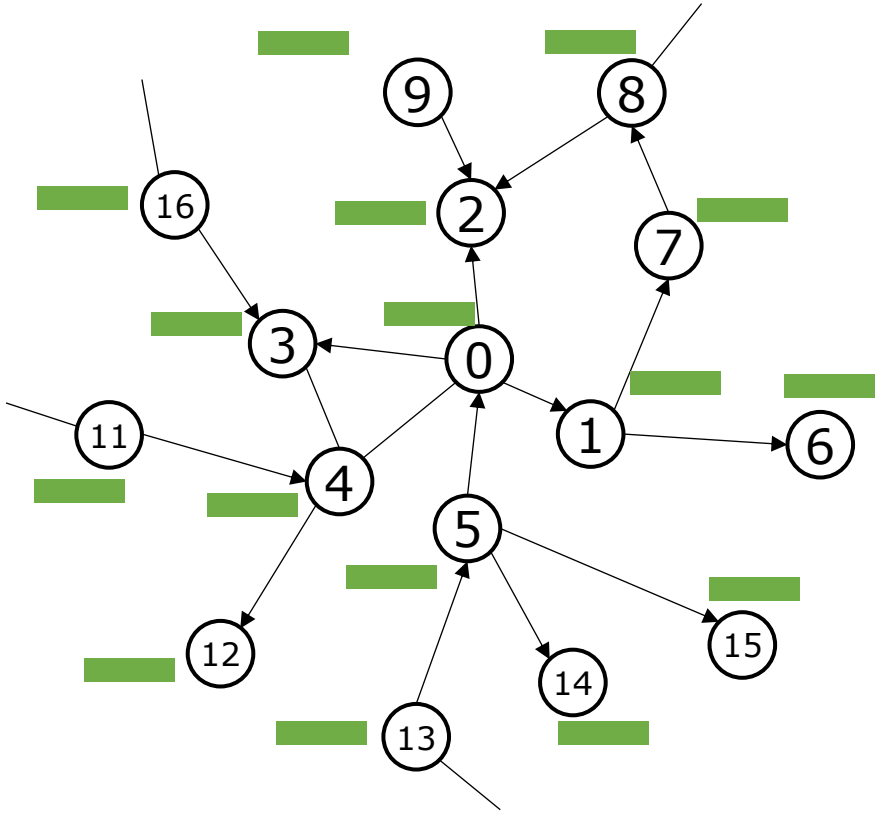


节点：论文
边：论文引用关系
节点特征：**1433维**one-hot 词向量表示

作业部分

GCN layer
GAT layer

hidden_size=64

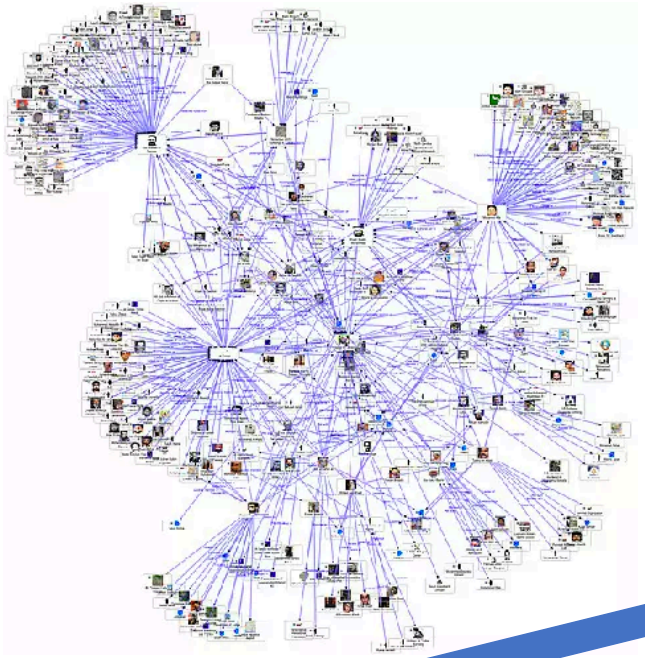


Cora：2708个节点，5429条边

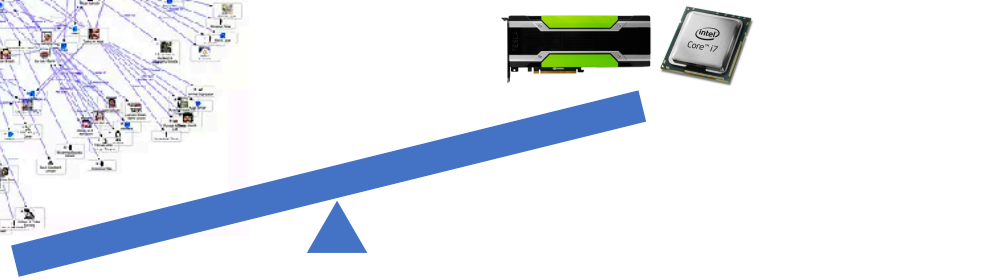
图采样

为什么要图采样

互联网中动则**亿量级**的图数据



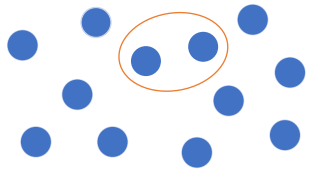
当代的GPU/CPU资源受限



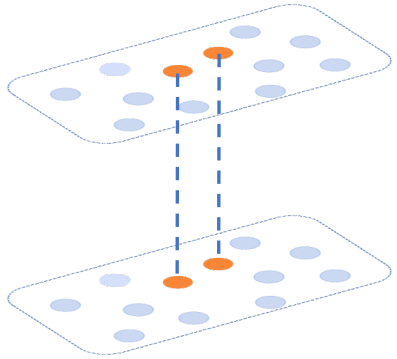
无法一次性全图送入计算资源，需要借鉴深度学习中的MiniBatch

为什么要图采样

MiniBatch训练

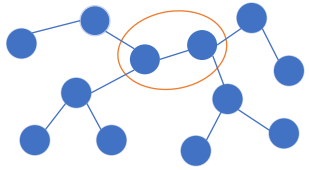


训练目标



传统深度学习

- 单batch为若干个**样本**
- 样本之间**无**依赖，多层样本计算量**固定**

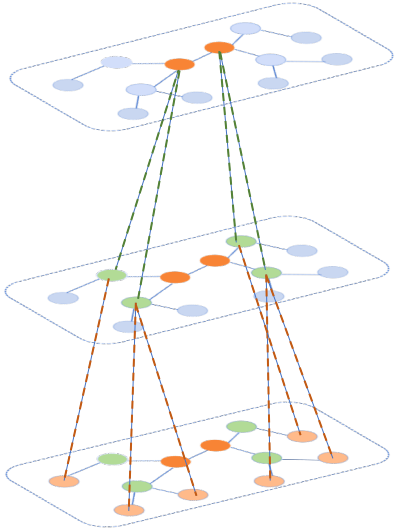


训练目标

N=2

N=6

N=11

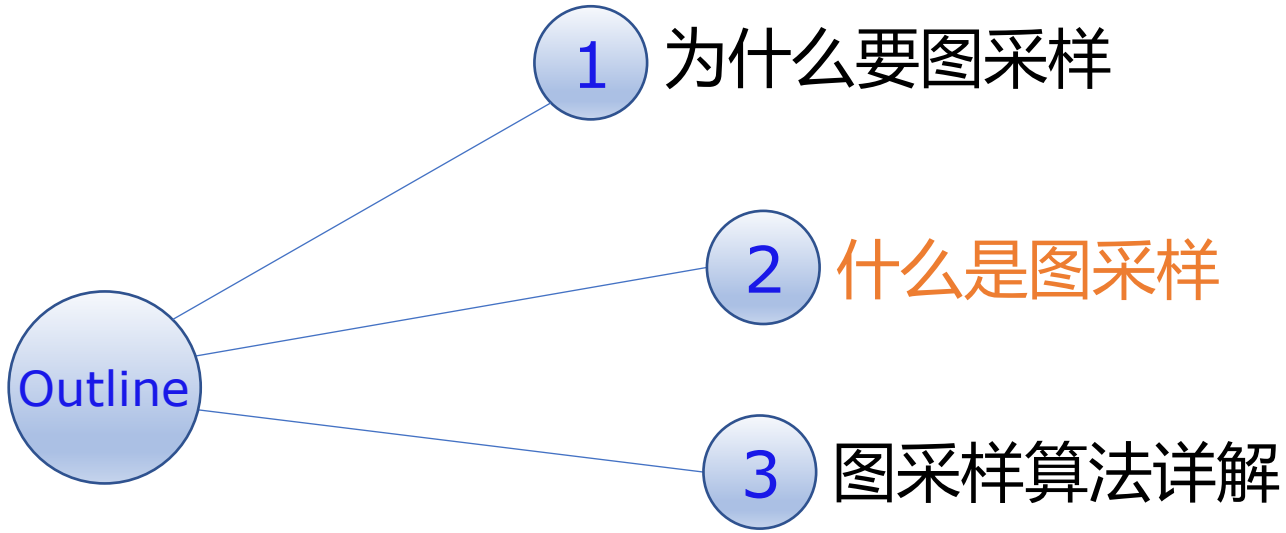


图神经网络

- 单batch为若干个**节点**
- 节点之间**相互**依赖，多层节点计算量**爆炸**

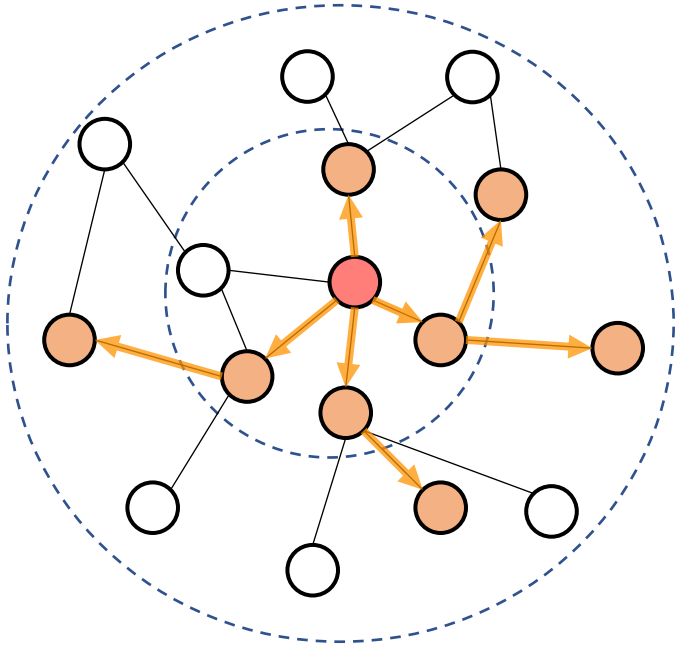
涉及计算的节点随层数增加呈指数增长

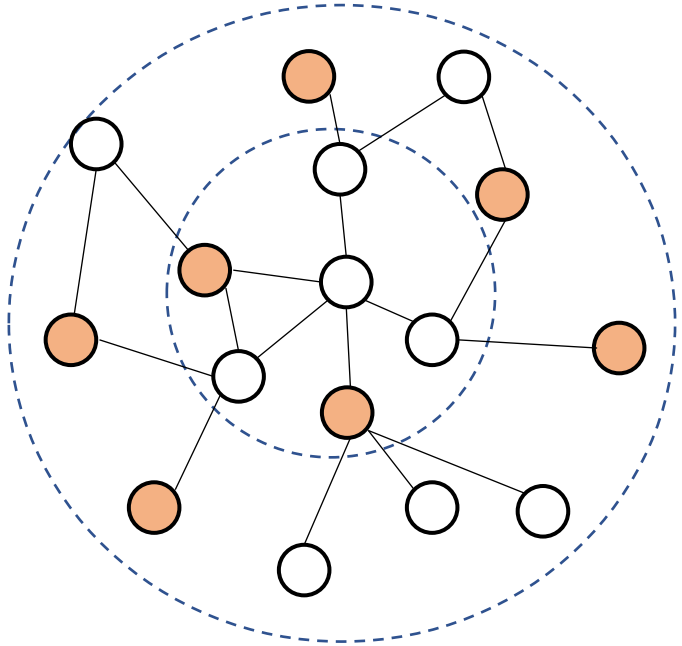
图采样



什么是图采样

子图采样而不是随机采样





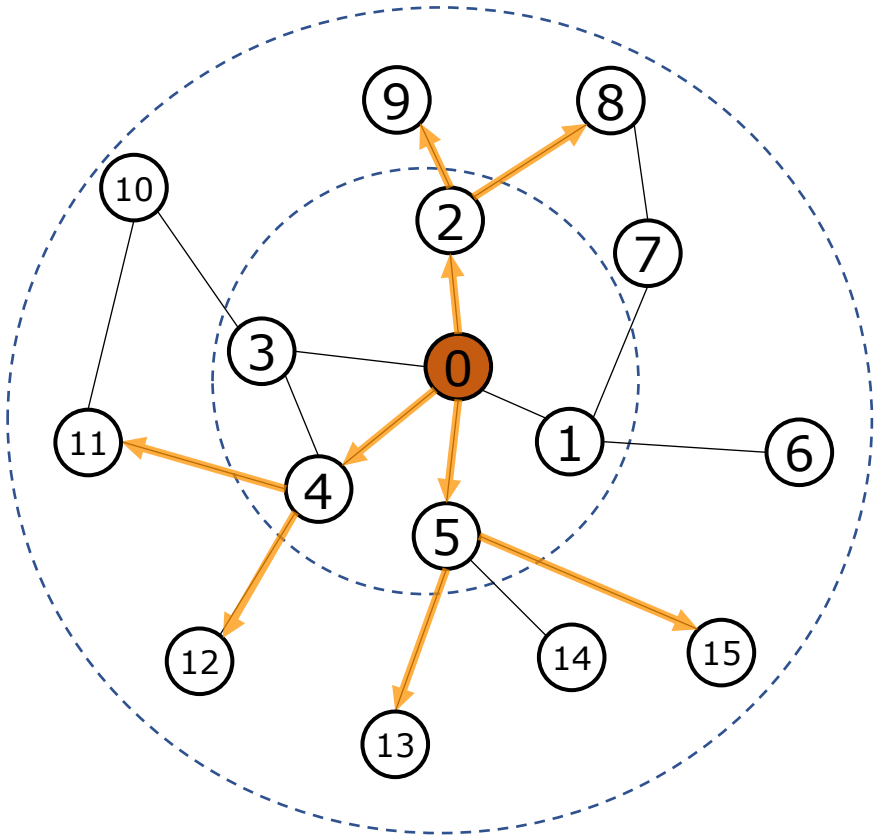
一个节点的表示由它的邻居决定

图采样

图采样算法详解

GraphSAGE (SAmple & aggreGatE)

1，邻居采样




```
graph = build_graph()
start_nodes = [0]

layer1_nodes = graph.sample_predecessor(start_nodes, 3)

layer1_edges = []
for dst_node, src_nodes in zip(start_nodes, layer1_nodes):
    for node in src_nodes:
        layer1_edges.append((node, dst_node))

print("layer1_nodes: ", layer1_nodes)
print("layer1_edges: ", layer1_edges)
# layer1_nodes: [array(2, 4, 5)]
# layer1_edges: [(2, 0), (4, 0), (5, 0)]
```

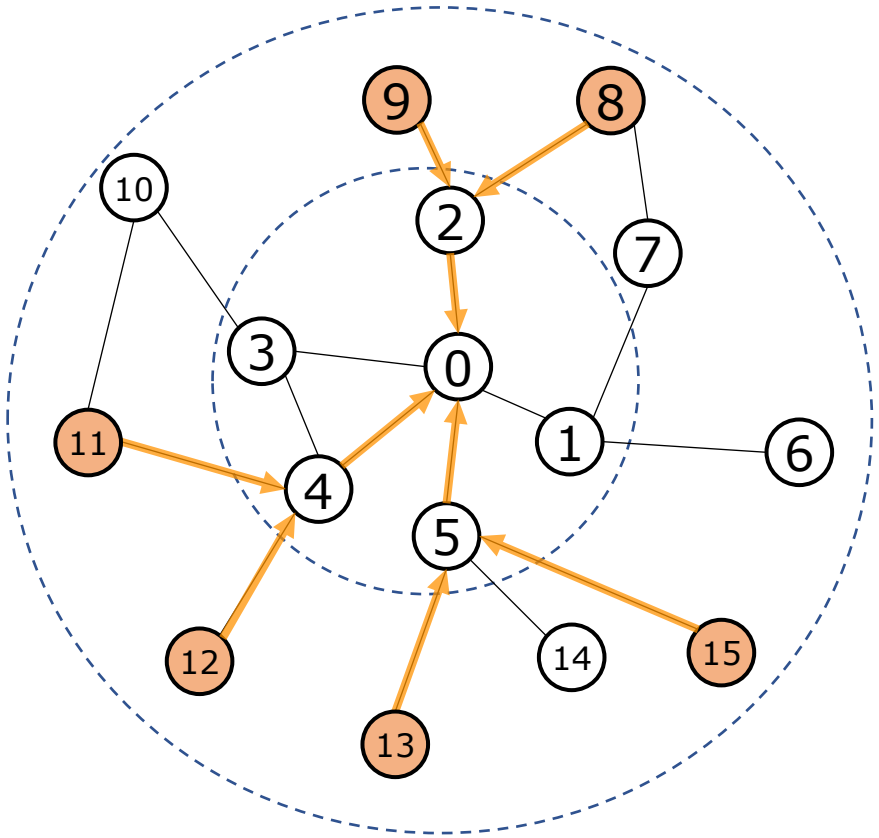
```
start_nodes = layer1_nodes[0].tolist()
layer2_nodes = graph.sample_predecessor(start_nodes, 2)

layer2_edges = []
for dst_node, src_nodes in zip(start_nodes, layer2_nodes):
    for node in src_nodes:
        layer2_edges.append((node, dst_node))

print("layer2_nodes: ", layer2_nodes)
print("layer2_edges: ", layer2_edges)
# layer2_nodes: [array([8, 9]), array([11, 12]), array([13, 15])]
# layer1_edges: [(8, 2), (9, 2), (11, 4), (12, 4), (13, 5), (15, 5)]
```

https://cs.stanford.edu/people/jure/pubs/graphsage-nips17.pdf

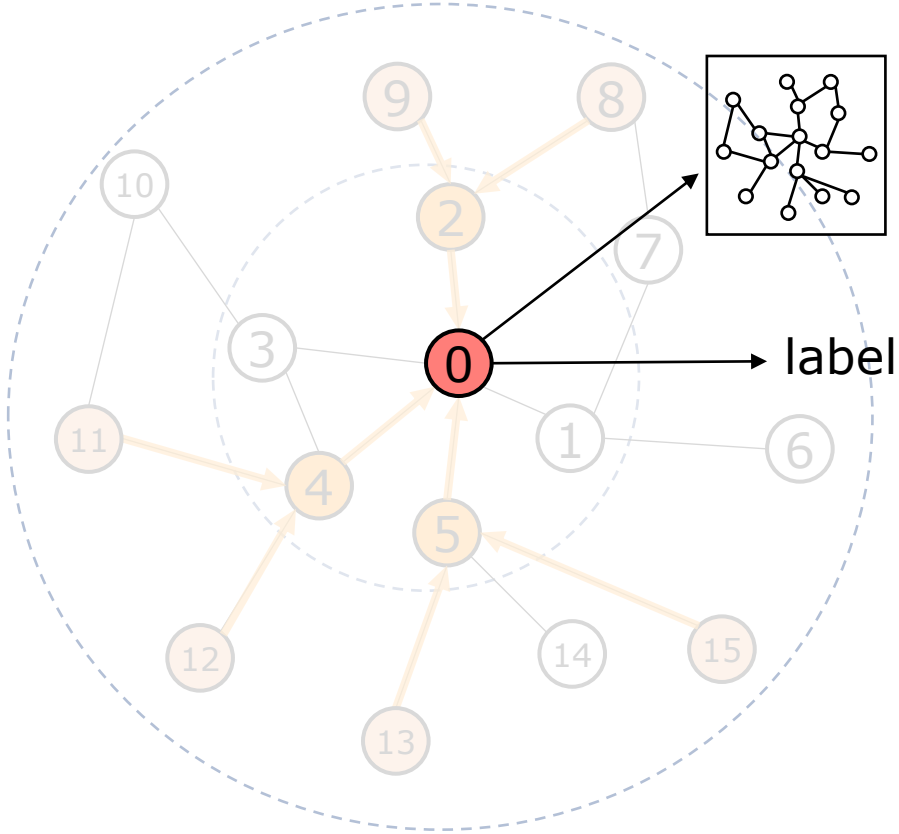
图采样算法详解

GraphSAGE (SAmple & aggreGatE)

2, 邻居聚合



3, 节点预测

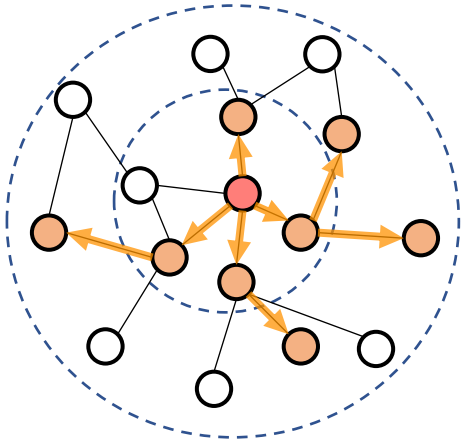


图采样算法详解

GraphSAGE (SAmple & aggreGatE)

邻居采样的优点

1、极大减少训练计算量

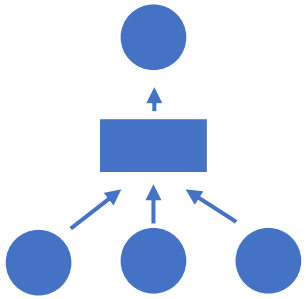


N^K -> M^K

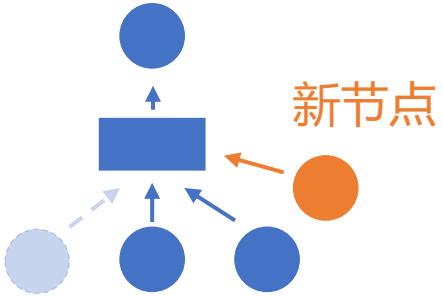
($N^K >> M^K$)

N：平均邻居数
M：采样数
K：训练层数

2、允许泛化到新连接关系



Train阶段



Infer阶段

图采样算法详解

飞桨

采样时只能选取真实的邻居节点吗？

图采样算法详解

PinSAGE (SAmple & aggreGatE)

采样时只能选取真实的邻居节点吗？

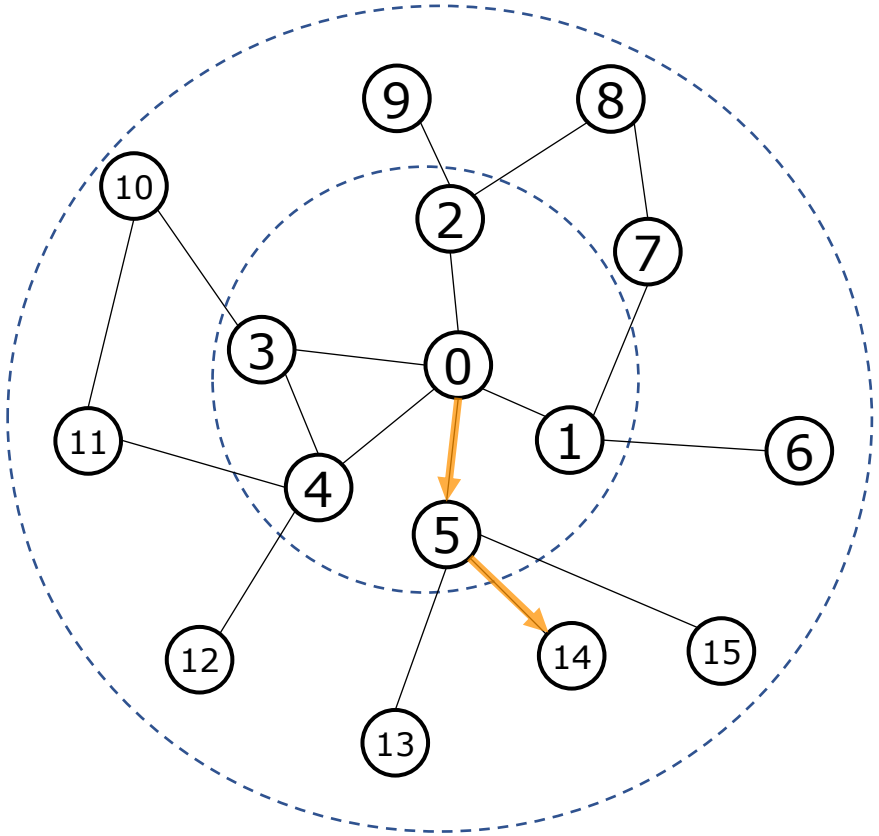
PinSAGE： 通过多次随机游走，按游走经过的频率选取邻居

图采样算法详解

PinSAGE (SAmple & aggreGatE)

采样时只能选取真实的邻居节点吗？

PinSAGE： 通过多次随机游走，按游走经过的频率选取邻居

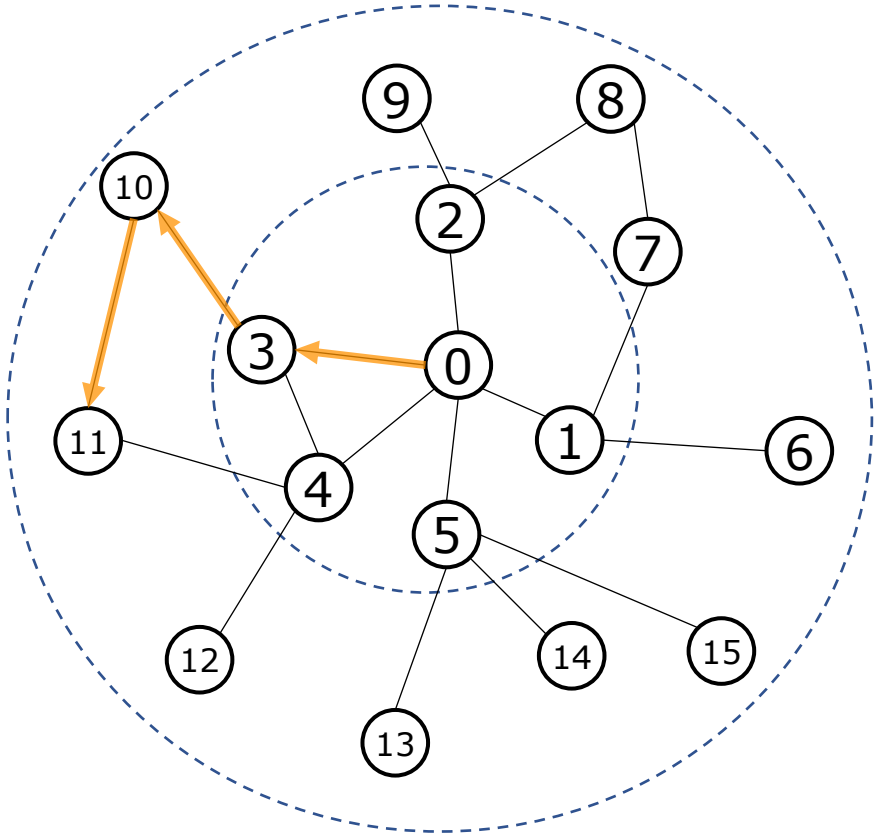


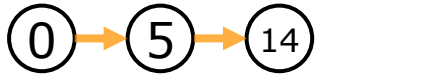
图采样算法详解

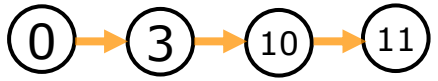
PinSAGE (SAmple & aggreGatE)

采样时只能选取真实的邻居节点吗？

PinSAGE： 通过多次随机游走，按游走经过的频率选取邻居





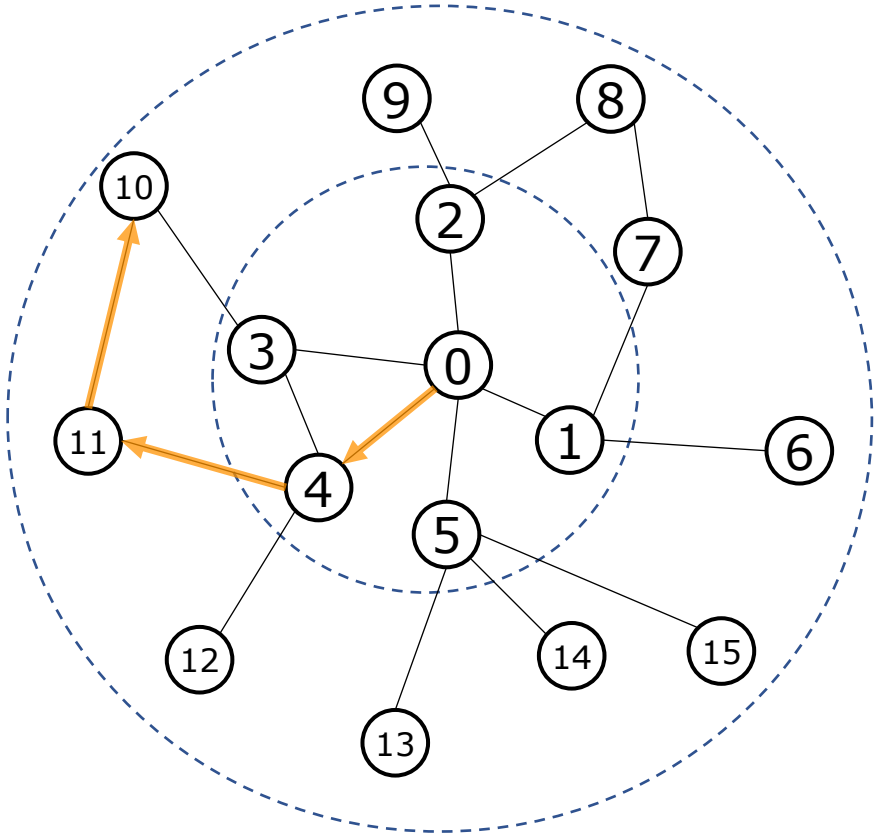


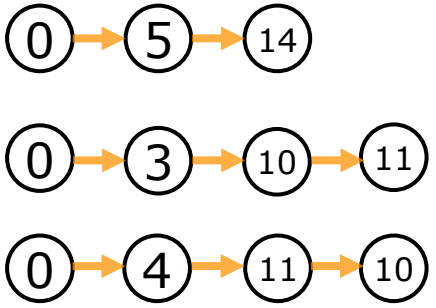
图采样算法详解

PinSAGE (SAmple & aggreGatE)

采样时只能选取真实的邻居节点吗？

PinSAGE： 通过多次随机游走，按游走经过的频率选取邻居



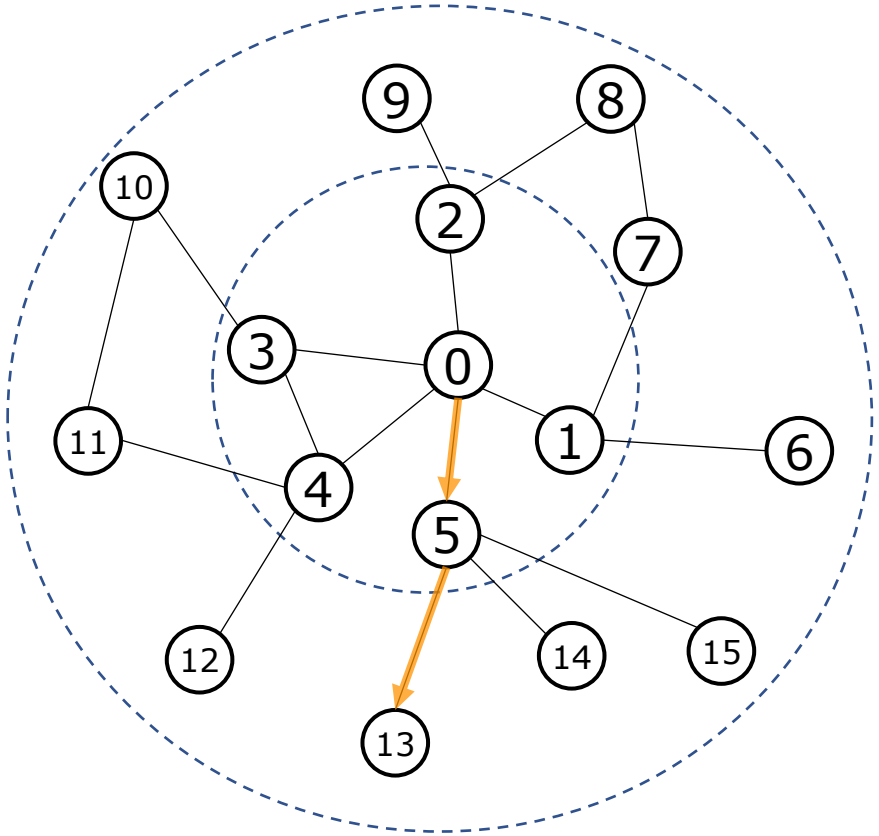


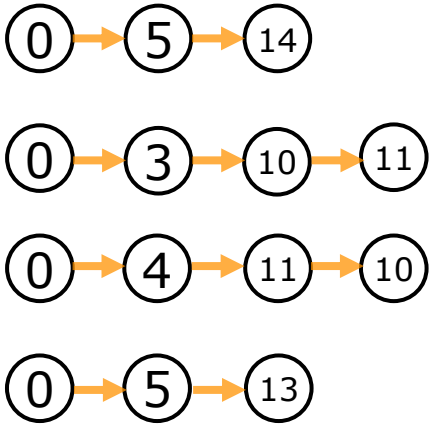
图采样算法详解

PinSAGE (SAmple & aggreGatE)

采样时只能选取真实的邻居节点吗？

PinSAGE： 通过随机游走，按游走经过的频率选取邻居



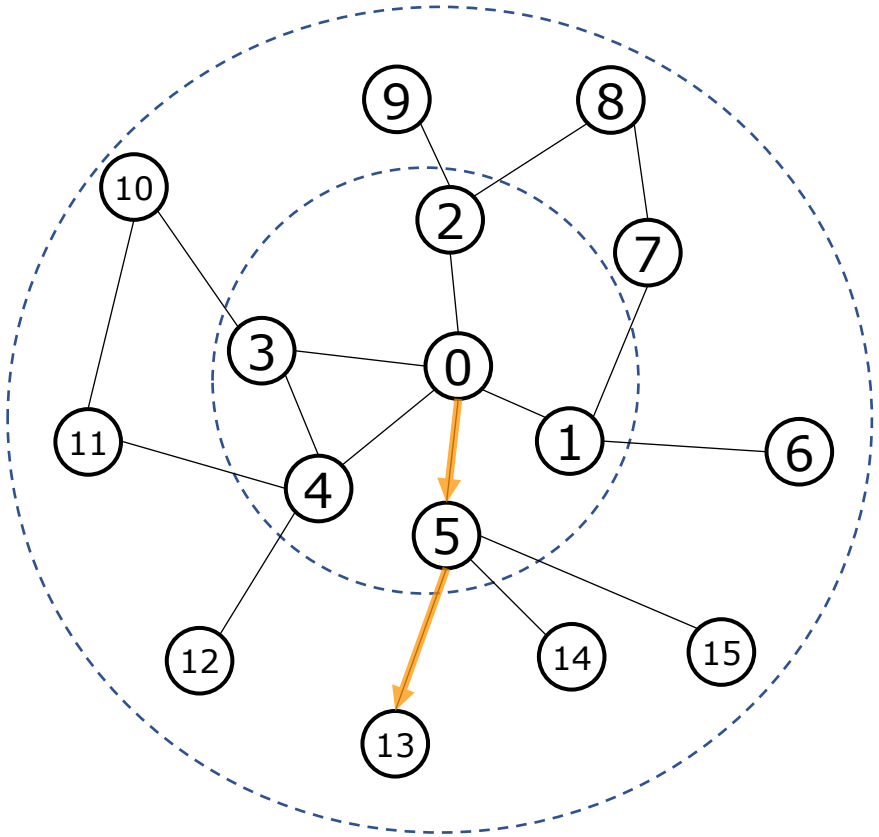


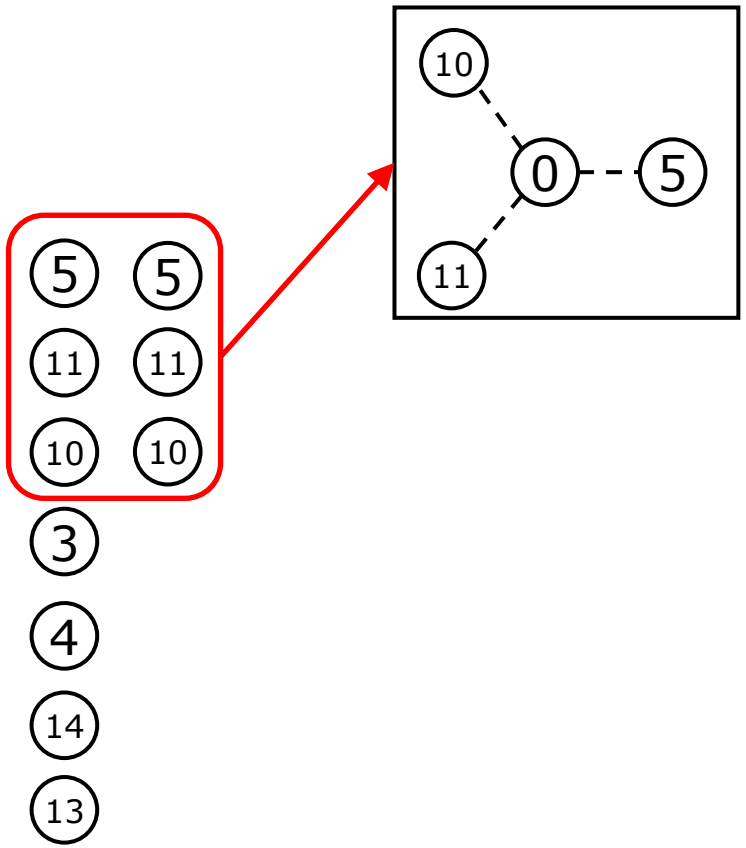
图采样算法详解

PinSAGE (SAmple & aggreGatE)

采样时只能选取真实的邻居节点吗？

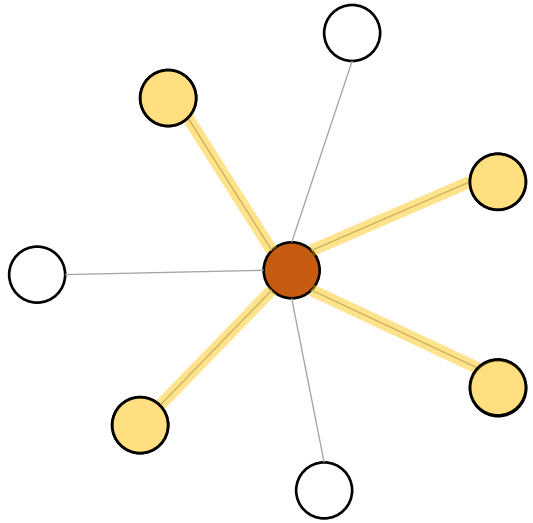
PinSAGE： 通过随机游走，按游走经过的频率选取邻居



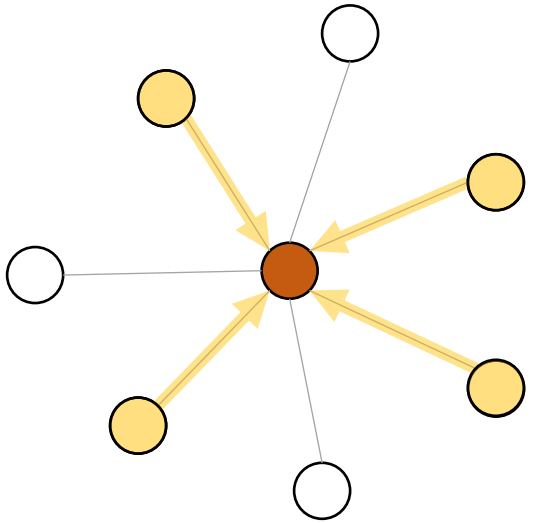


图采样小结

图采样之后——邻居聚合

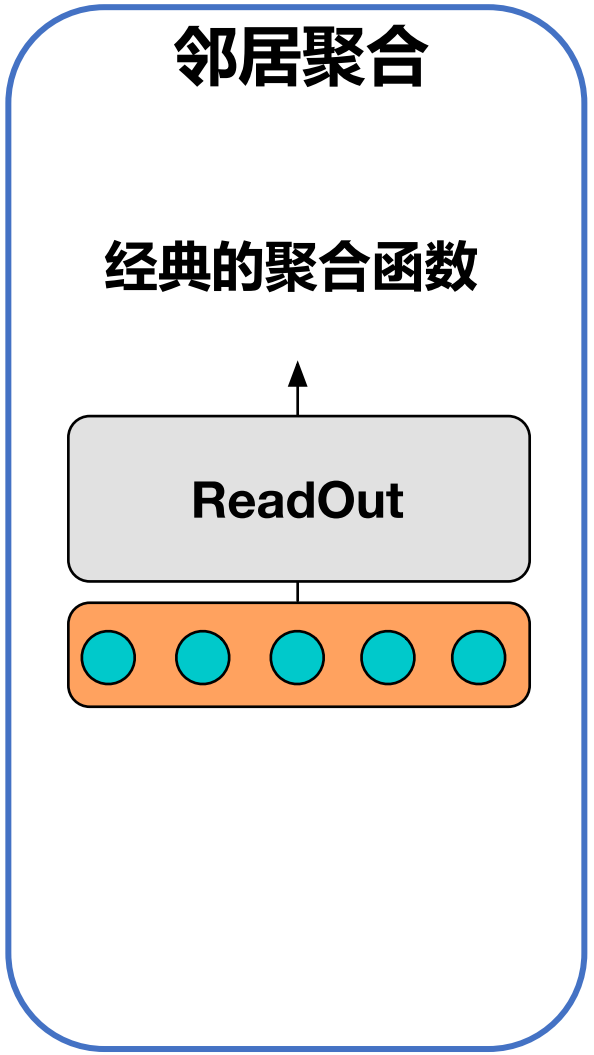


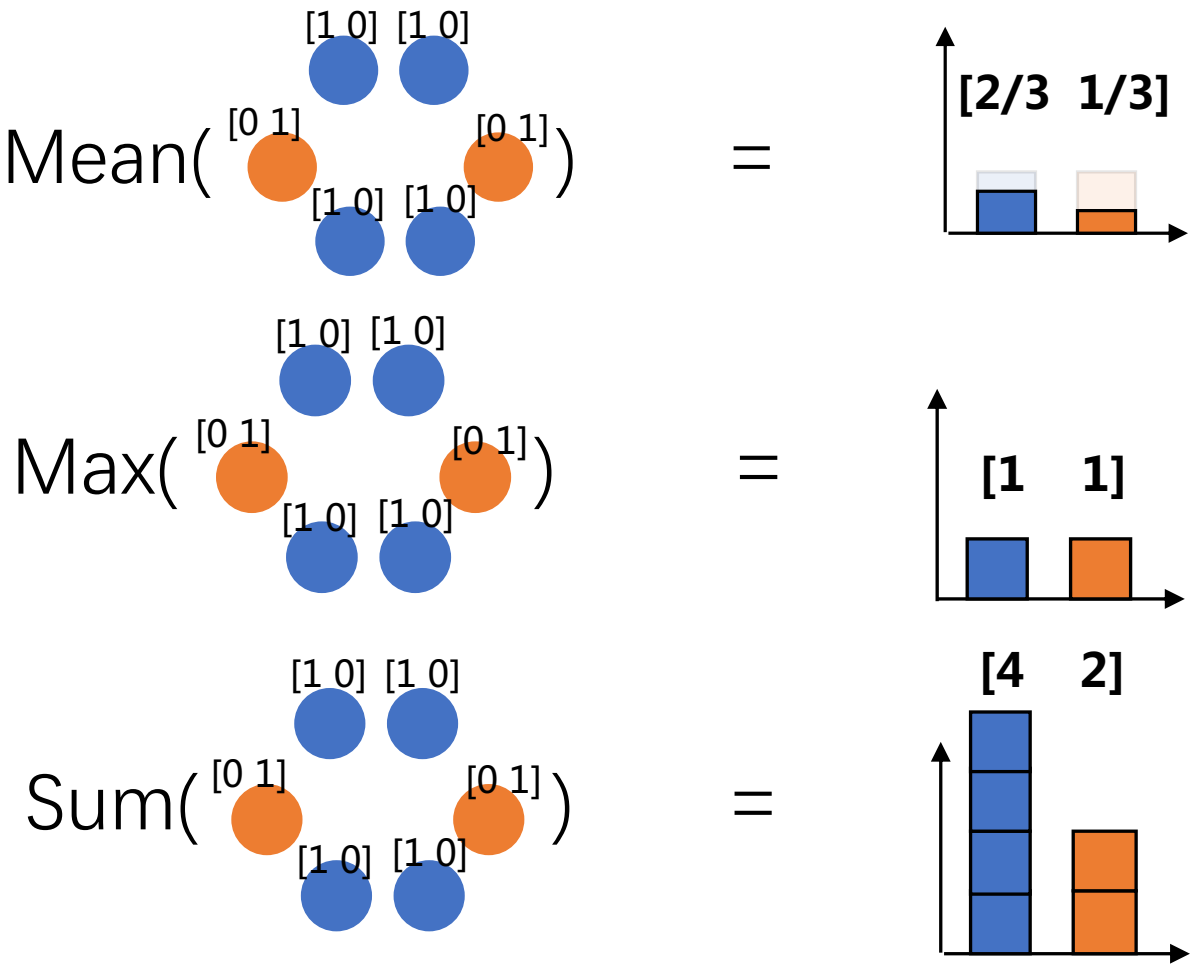
图采样



邻居聚合

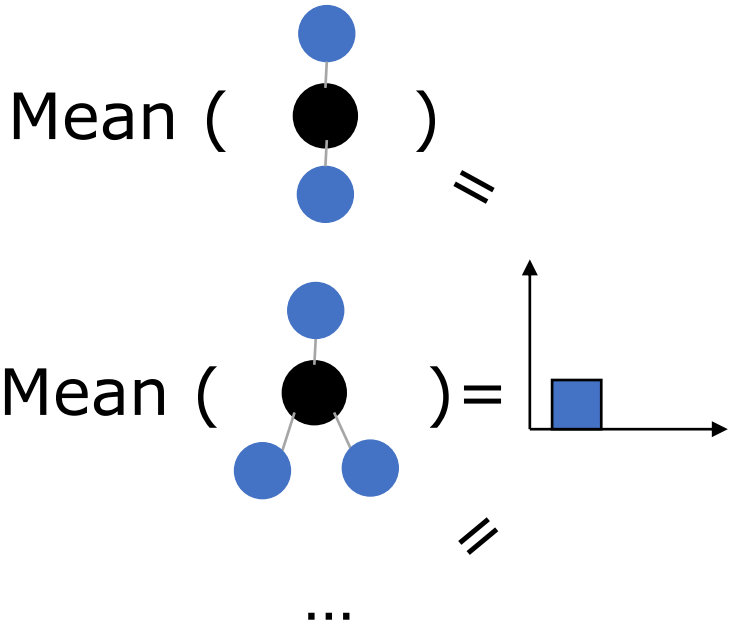
邻居聚合


邻居聚合
经典的聚合函数
ReadOut


Mean([1 0] [1 0] [0 1] [0 1] [1 0] [1 0]) = [2/3 1/3]
Max([1 0] [1 0] [0 1] [0 1] [1 0] [1 0]) = [1 1]
Sum([1 0] [1 0] [0 1] [0 1] [1 0] [1 0]) = [4 2]

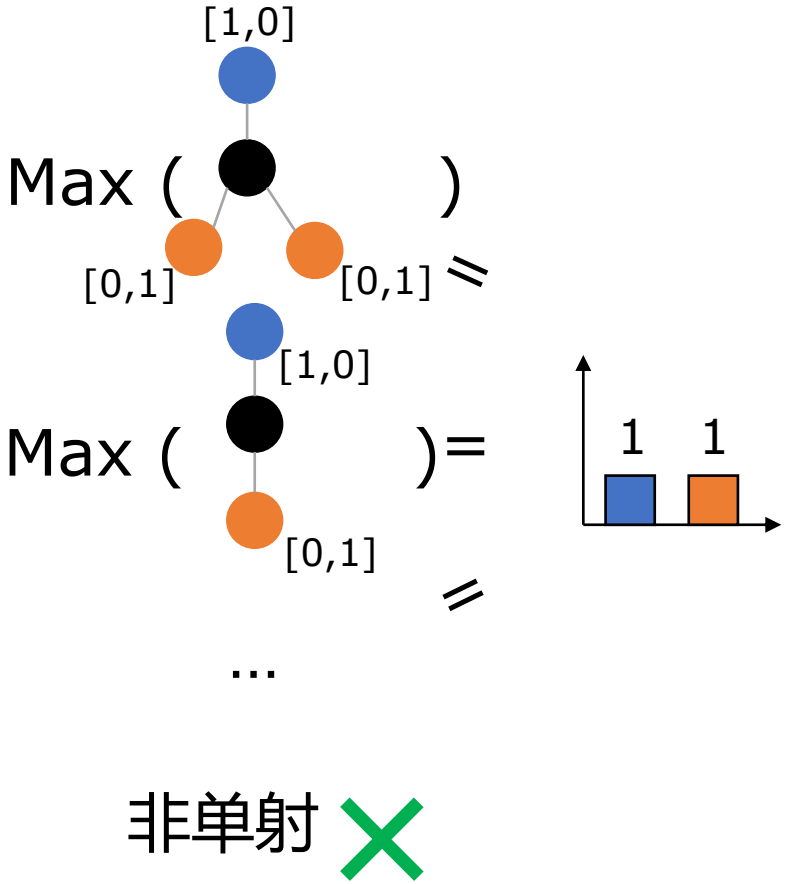
邻居聚合

评估聚合表达能力的指标**单射** (一对一映射)



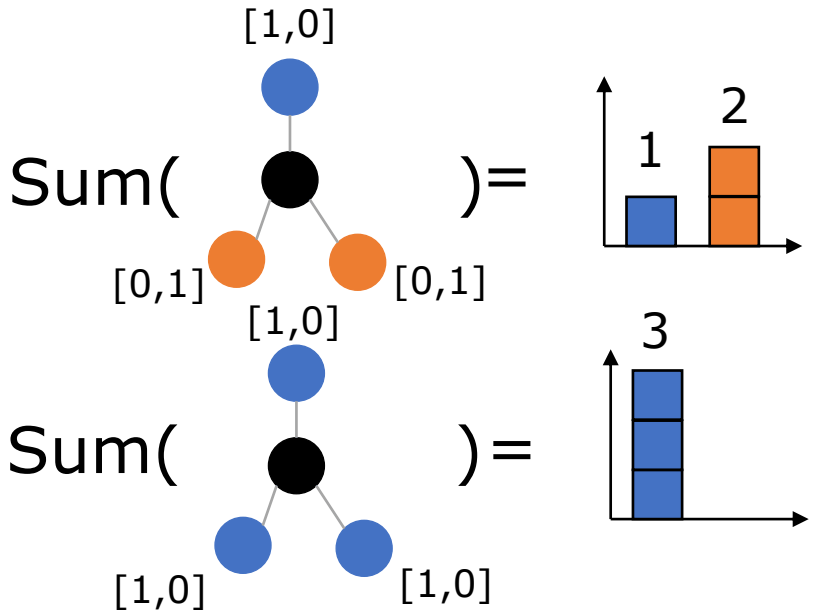
非单射 ✕

Mean倾向学分布



非单射 ✕

Max倾向忽略重复值

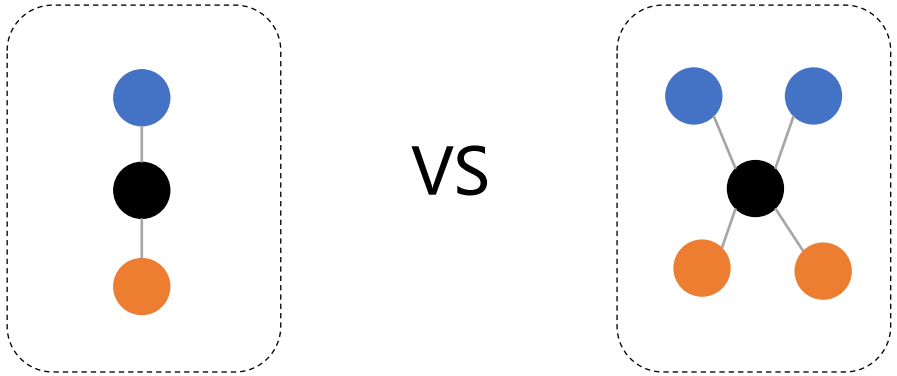


单射 O

Sum保留完整信息

邻居聚合

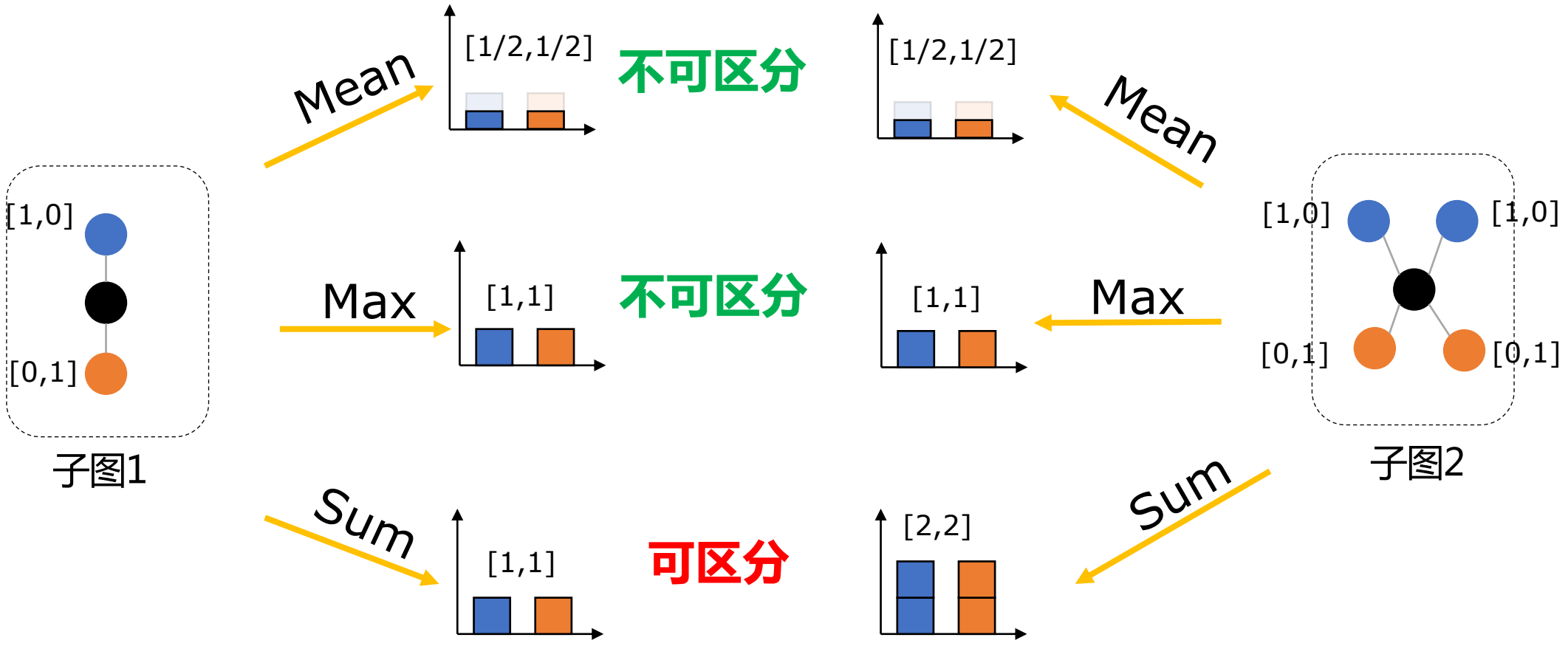
单射可以保证对聚合后的结果可区分



子图1　　子图2

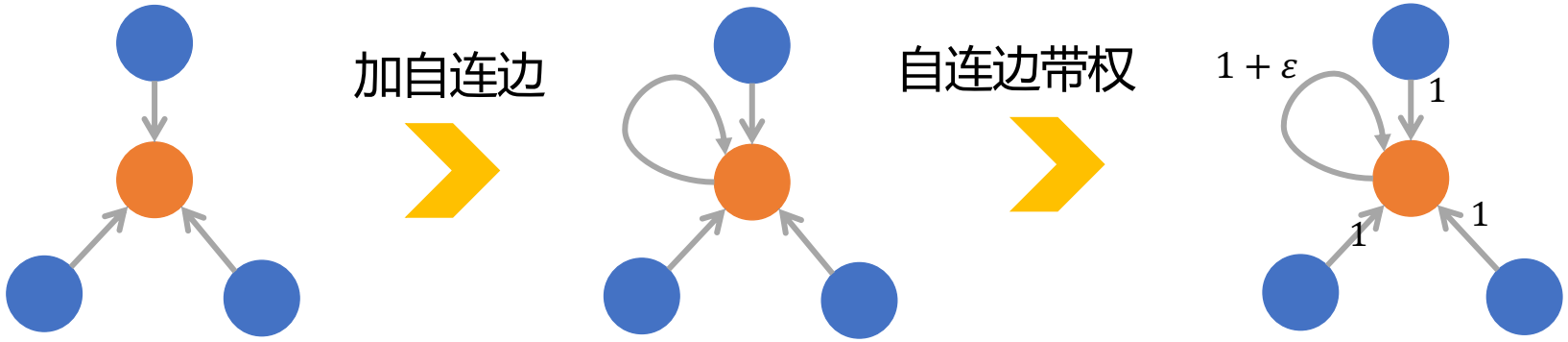
邻居聚合

单射可以保证对聚合后的结果可区分



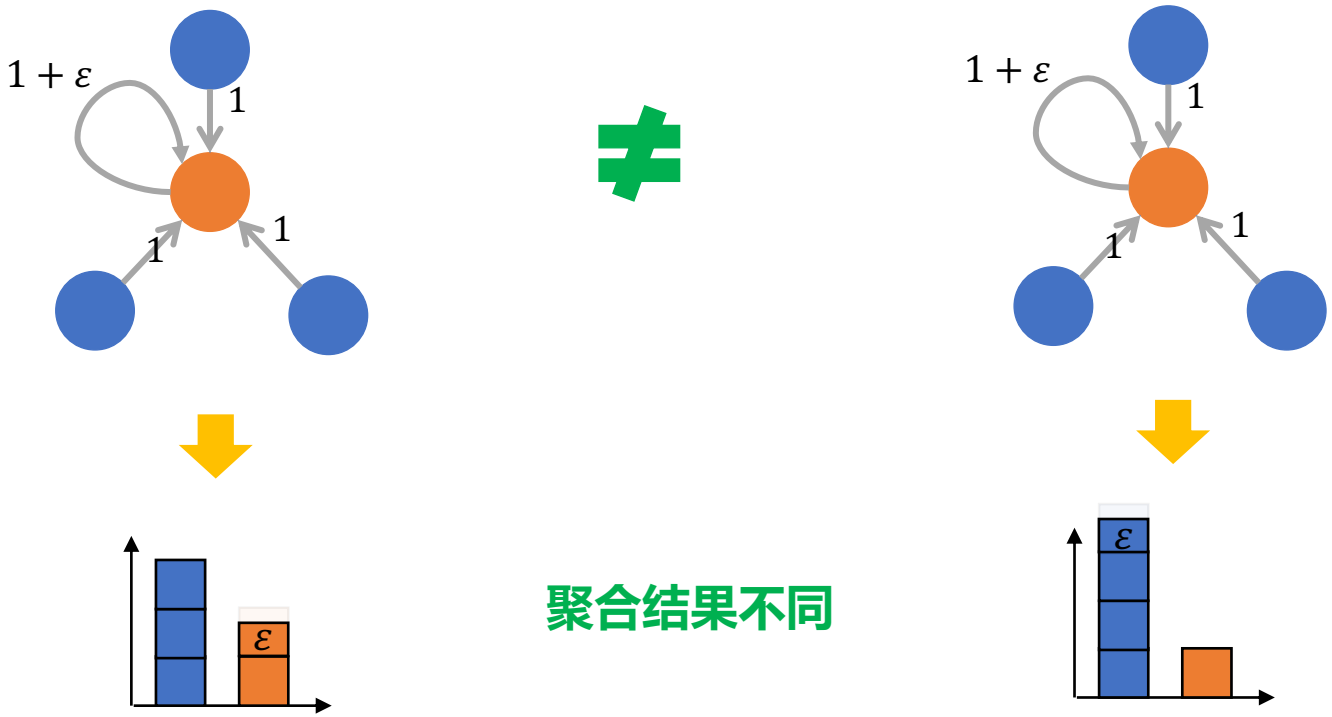
邻居聚合

基于单射的Graph Isomorphism Net (GIN)模型



邻居聚合

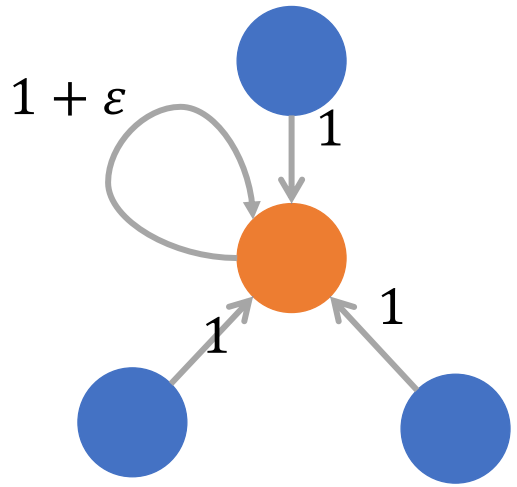
如果将中心节点与邻居节点互换



ε 保证了中心与邻居节点可区分

邻居聚合

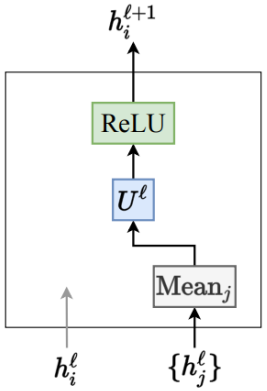
GIN代码解析



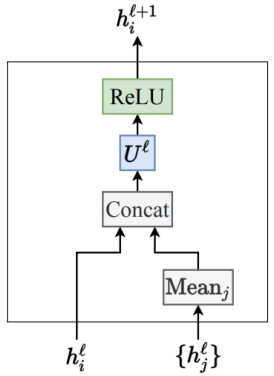

```
def send_src_copy(src_feat, dst_feat, edge_feat):
    return src_feat["h"]

msg = gw.send(send_src_copy, nfeat_list=[("h", feature)])
output = gw.recv(msg, "sum") + feature * (epsilon + 1.0)

output = L.fc(output, size=hidden_size)
```

邻居聚合

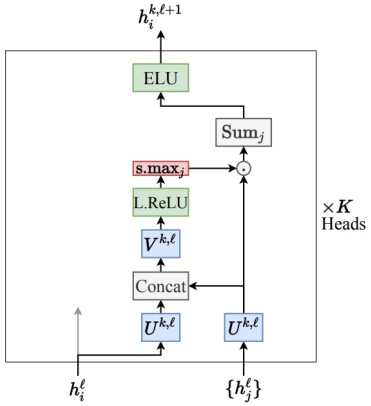
还有更复杂的聚合函数...



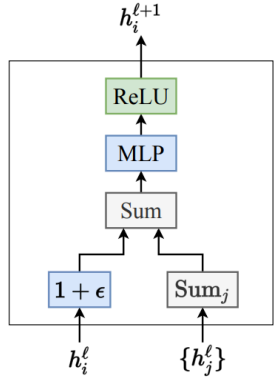
GCN



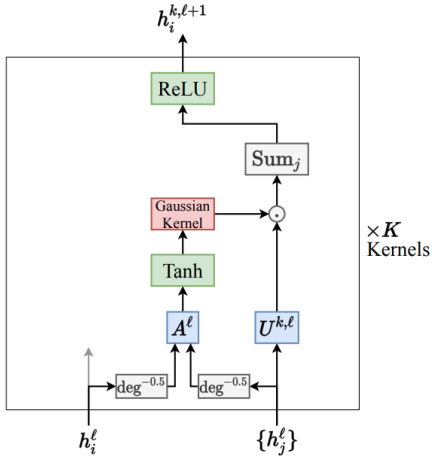
GraphSage



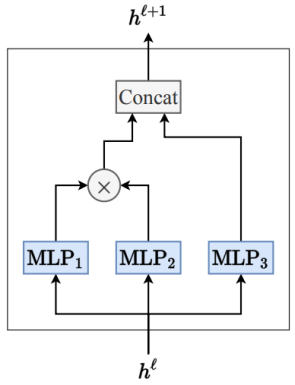
GAT



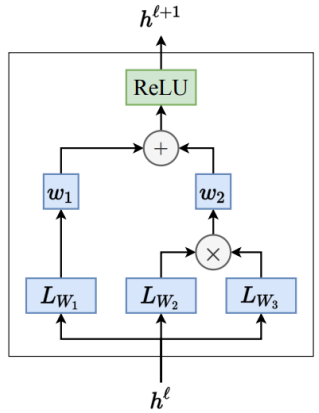
GIN



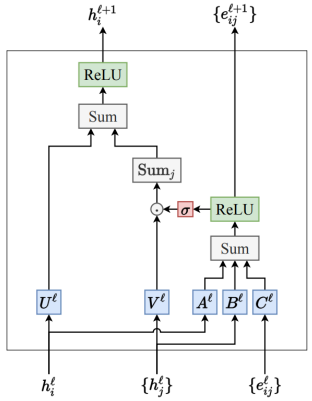
MoNet



3WL-GNN

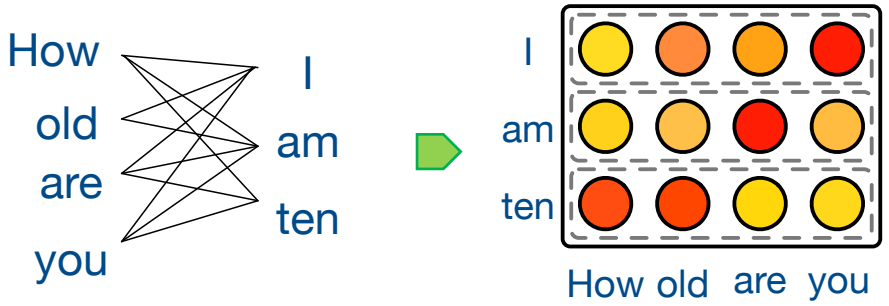


RingGNN

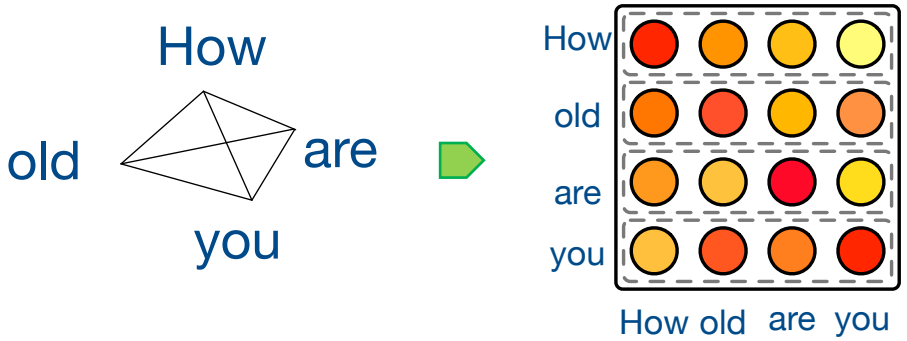


GatedGCN

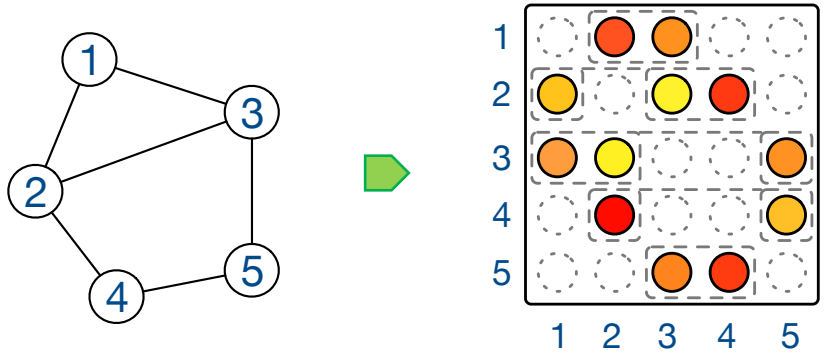
邻居聚合-语义场景



Vanilla-Attention（**二分图**）



Self-Attention（**全连通图**）

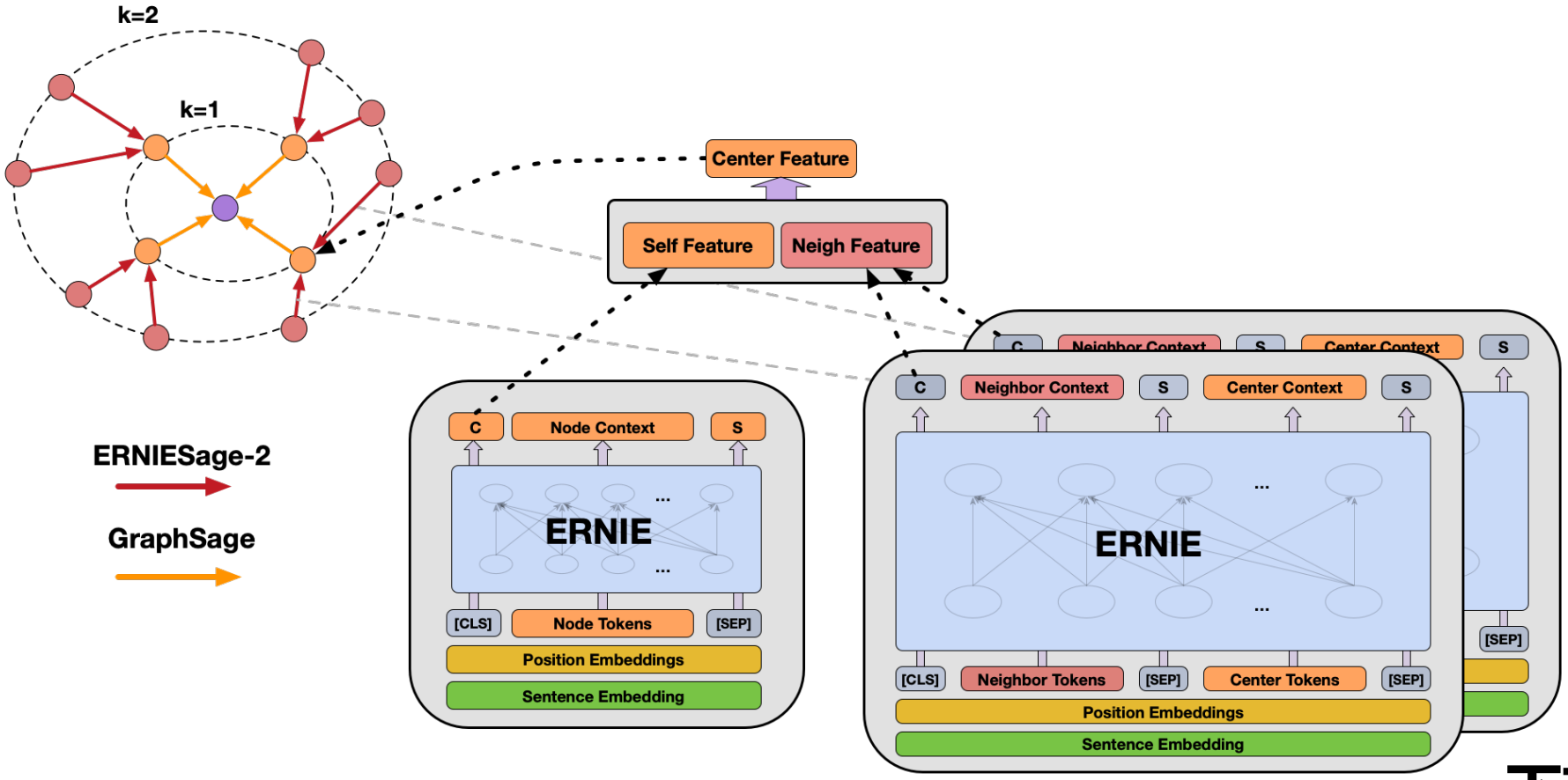


Graph-Attention（**任意图**）

传统NLP领域的Transformer就是全连通图的图模型

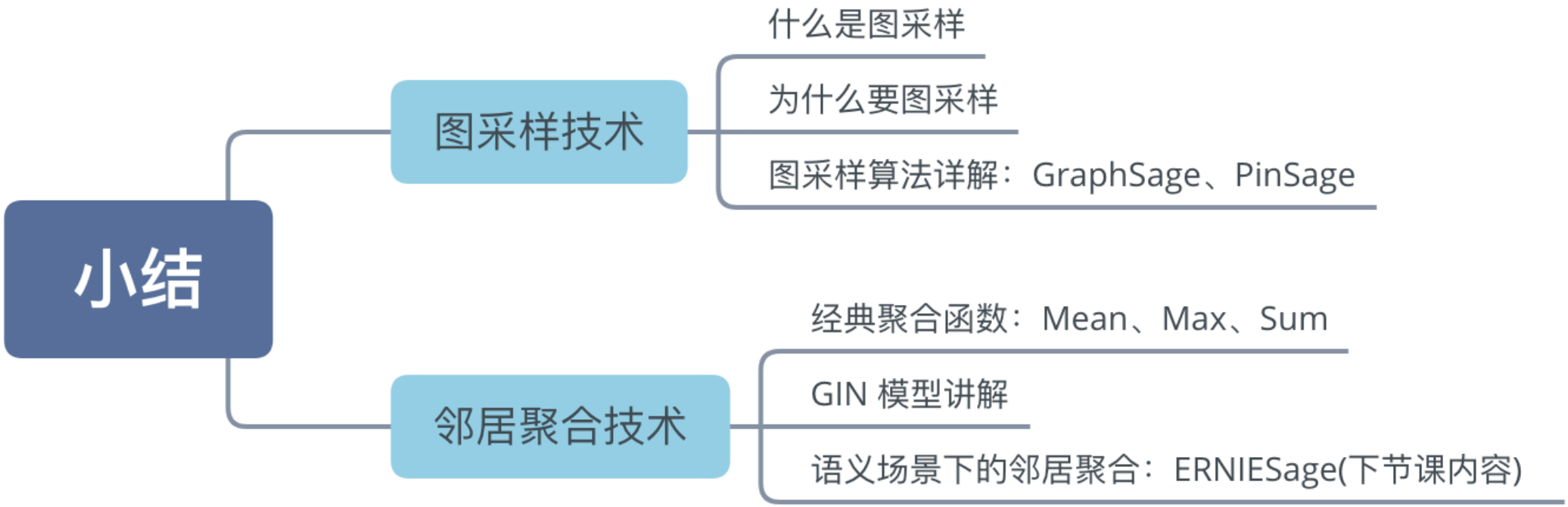
邻居聚合-语义场景

ERNIESage：在文本图（TextGraph）场景下，使用**ERNIE**做聚合函数



下节课独家解密

课程小结

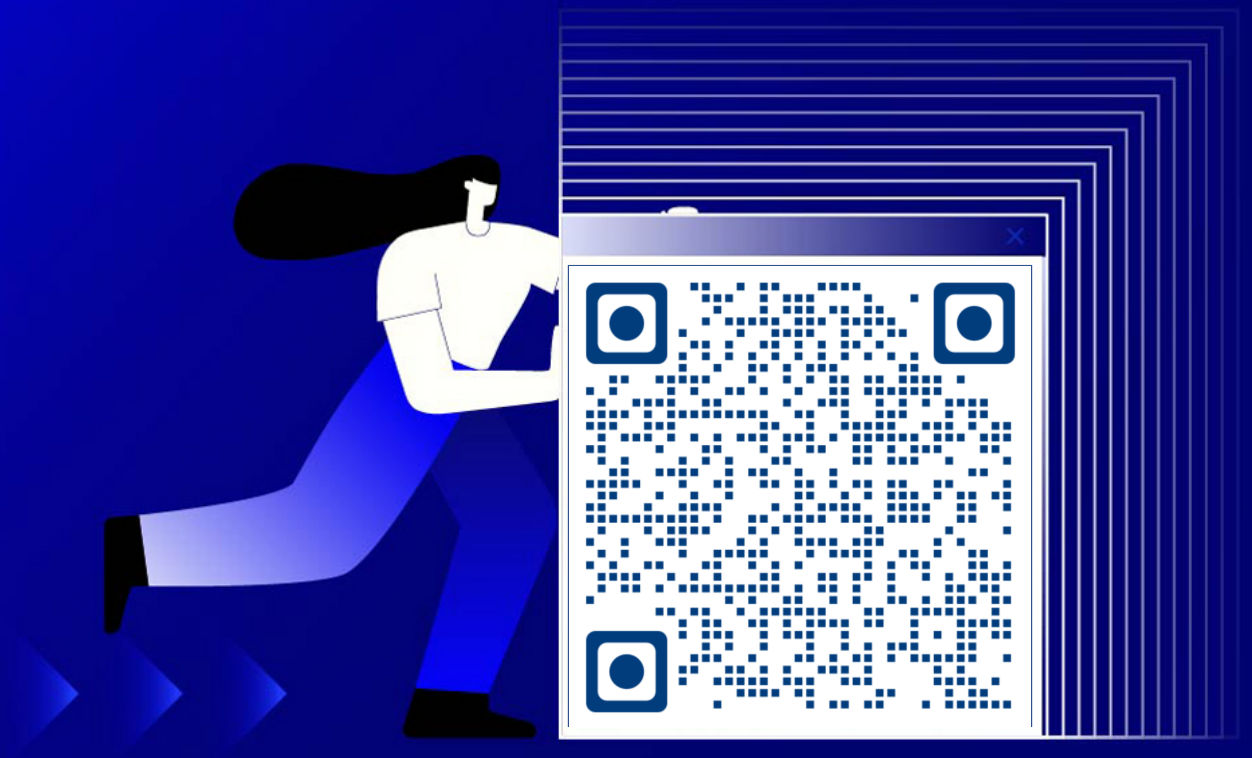

小结
图采样技术
什么是图采样
为什么要图采样
图采样算法详解：GraphSage、PinSage
邻居聚合技术
经典聚合函数：Mean、Max、Sum
GIN 模型讲解
语义场景下的邻居聚合：ERNIESage(下节课内容)

课堂实践

有请帅哥老师讲解 GraphSage 采样代码

课后作业

完成 GraphSage 模型的采样与聚合代码



PGL github

谢谢观看