DGL

Deep Graph Learning

主流的图学习框架（库）

The Tools of the GraphNeuralNetwork

名称	类型	适用场景	Github
OpenNE	图表示学习	图节点表示学习，预训练	https://github.com/thunlp/OpenNE
Graph_nets	图神经网络	基于关系模糊的图数据推理	https://github.com/deepmind/graph_nets
DGL	图神经网络	建立图数据（可以无需通过networkx）并加载常用图神经网络	https://github.com/jermainewang/dgl
GPF	训练流程	基于关系数据的数据预测（节点分类、关系预测）	https://github.com/xchadesi/GPF
networkx	图数据预处理	非大规模图数据预处理	https://github.com/networkx/networkx
Euler	工业级图深度学习框架	工业级图数据的用户研究快速进行算法创新与定制	https://github.com/alibaba/euler
PyG	几何深度学习	适合于图、点云、流形数据的深度学习，速度比DGL快	https://github.com/rusty1s/pytorch_geometric

DGL的优势

- 支持后端： Pytorch,MXNET,Numpy(no autograd)
- 接口友好
- 能够在巨大的图上（5亿节点，250亿边）训练图神经网络。
- 消息融合（Fused Message Passing)
- ICLR workshop(https://rlgm.github.io/papers/49.pdf)

大图训练的性能瓶颈

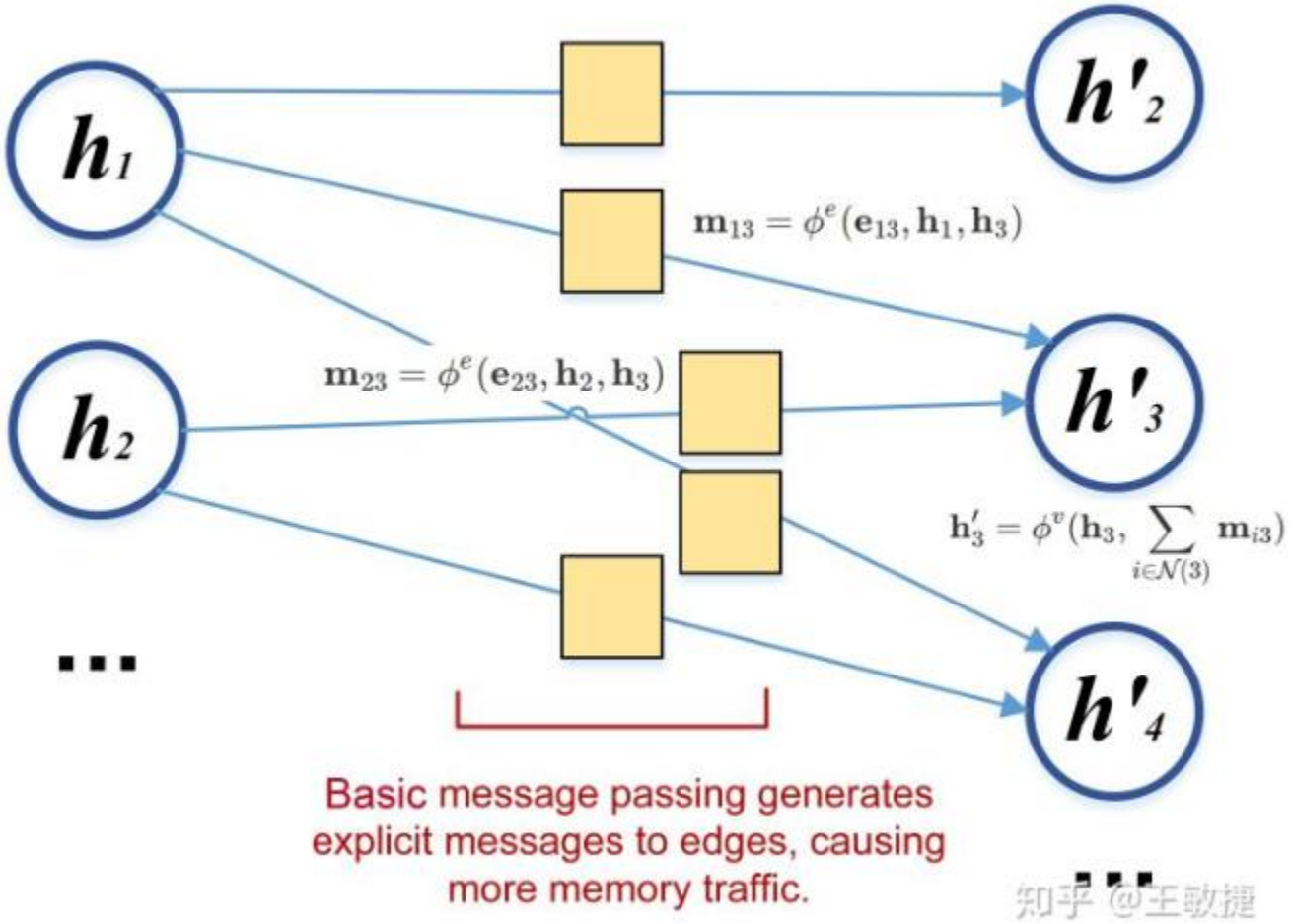
- 绝大多数图神经网络模型遵循消息传递的计算范式，用户需要提供两个函数：

1.消息函数：在边上触发，定义了如何计算发送给相邻节点的消息。

2.累和函数：在点上触发，定义了如果在点上累和收到的消息。

消息传递机制

```
# 使用自定义消息函数和累和函数计算图卷积
G.update_all(lambda edges: {'m' : edges.src['h']},
             lambda nodes: {'h' : sum(nodes.mailbox['m'], axis=1)})
```



消息张量的大小正比于图中边的数量，因而当图增大时，消息张量消耗的内存空间也会显著上升

消息融合解决大图训练难题

$$\mathbf{M} = \text{send}(E, \phi^e, \mathbf{H}_v) \qquad \mathbf{H}'_v = \text{recv}(V, \Sigma, \mathbf{M})$$

Basic message passing

$$\mathbf{H}'_v = \text{send\_and\_recv}(E, \phi^e, \Sigma, \mathbf{H}_v)$$

DGL **fuses** the computation as `send_and_recv` to avoid explicit message storage, thus is **faster** and more **scalable** for large graphs.

DGL将 send 和 recv 接口合并成 send_and_recv（见下图）。DGL的后端通过自己的CUDA代码，在每个GPU线程中将源节点特征载入其本地内存并计算消息函数，然后将计算结果直接累和到目标节点，从而避免生成消息张量。

性能表现

Dataset			Model	Accuracy	Time		Memory	
$\|V\|$	$\|E\|$				PyG	DGL	PyG	DGL
Cora			GCN	81.31 ± 0.88	**0.478**	0.666	1.1	1.1
3K	11K		GAT	83.98 ± 0.52	1.608	**1.399**	1.2	**1.1**
CiteSeer			GCN	70.98 ± 0.68	**0.490**	0.674	1.1	1.1
3K	9K		GAT	69.96 ± 0.53	1.606	**1.399**	1.3	**1.2**
PubMed			GCN	79.00 ± 0.41	**0.491**	0.690	1.1	1.1
20K	889K		GAT	77.65 ± 0.32	1.946	**1.393**	1.6	**1.2**
Reddit			GCN	93.46 ± 0.06	*OOM*	**28.6**	*OOM*	**11.7**
232K	114M							
Reddit-S			GCN	N/A	29.12	**9.44**	15.7	**3.6**
232K	23M							

Table 2: Training time (in seconds) for 200 epochs and memory consumption (GB).

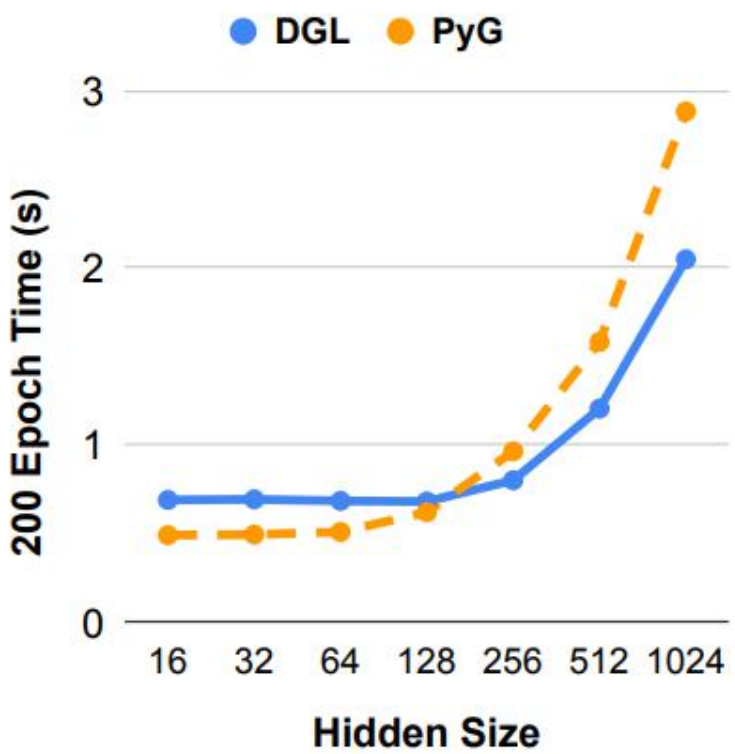


Figure 1: GCN training time on Pubmed with varying hidden size.