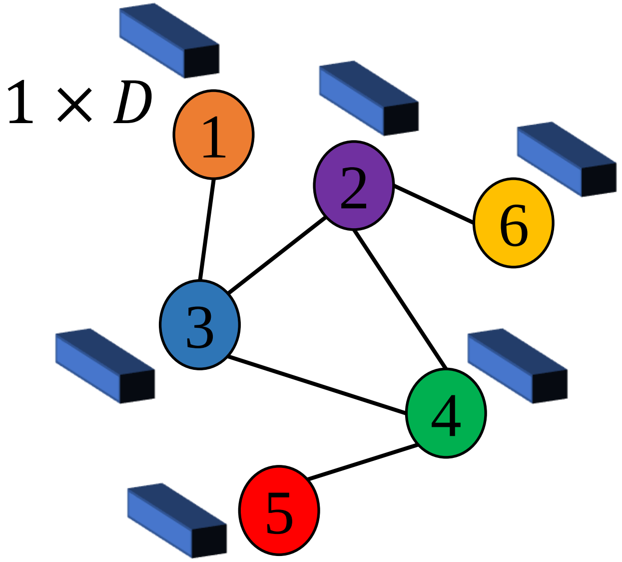
Outline

1. Graph Neural Networks
2. Graph Classification
3. U2GNN and Transformer
4. Papers

What is GNN?

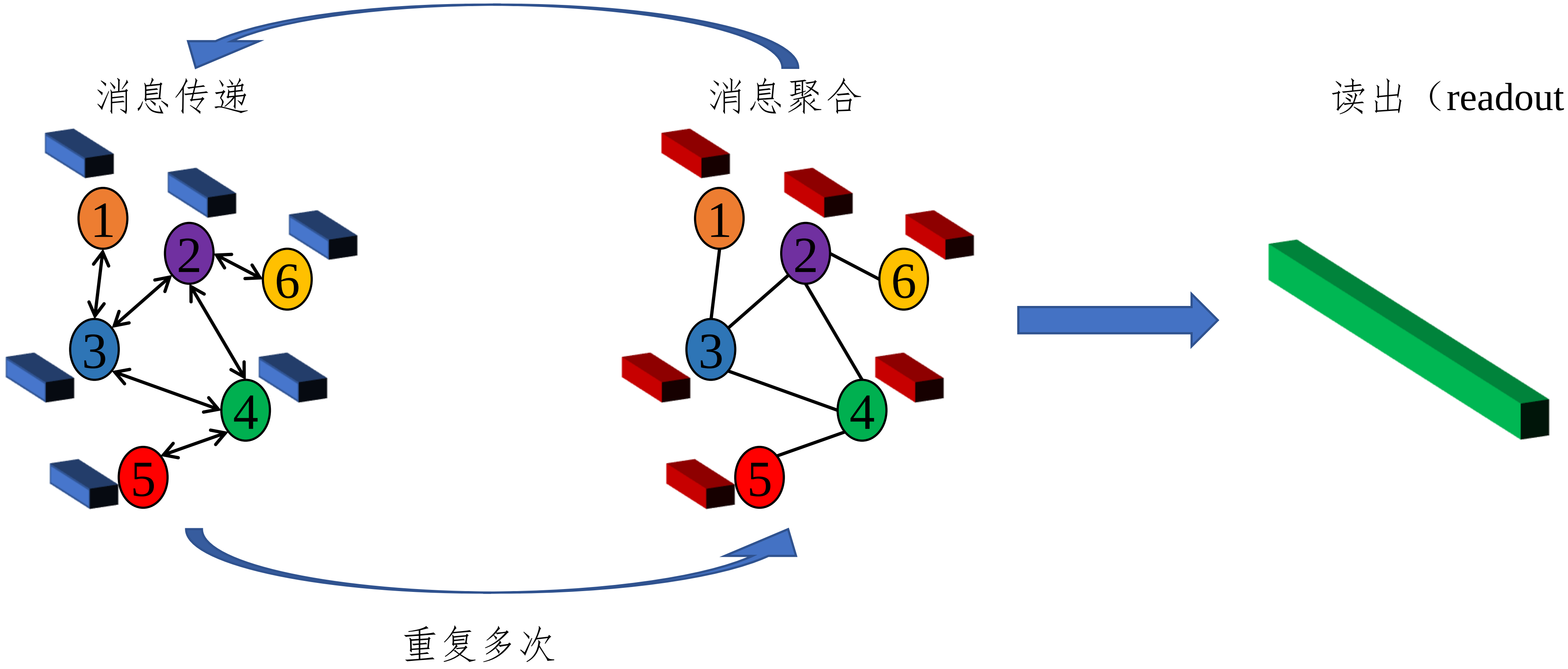


- GNN是在可以图（Graph）上定义卷积(特征聚合)操作，进行反向传播，从而实现端到端的图表示学习的一类神经网络模型
- GNN分为两类：基于空间和基于谱
- GNN是一种图表示学习的方法，其它方法还包括图嵌入等

Spatial-based and spectral based

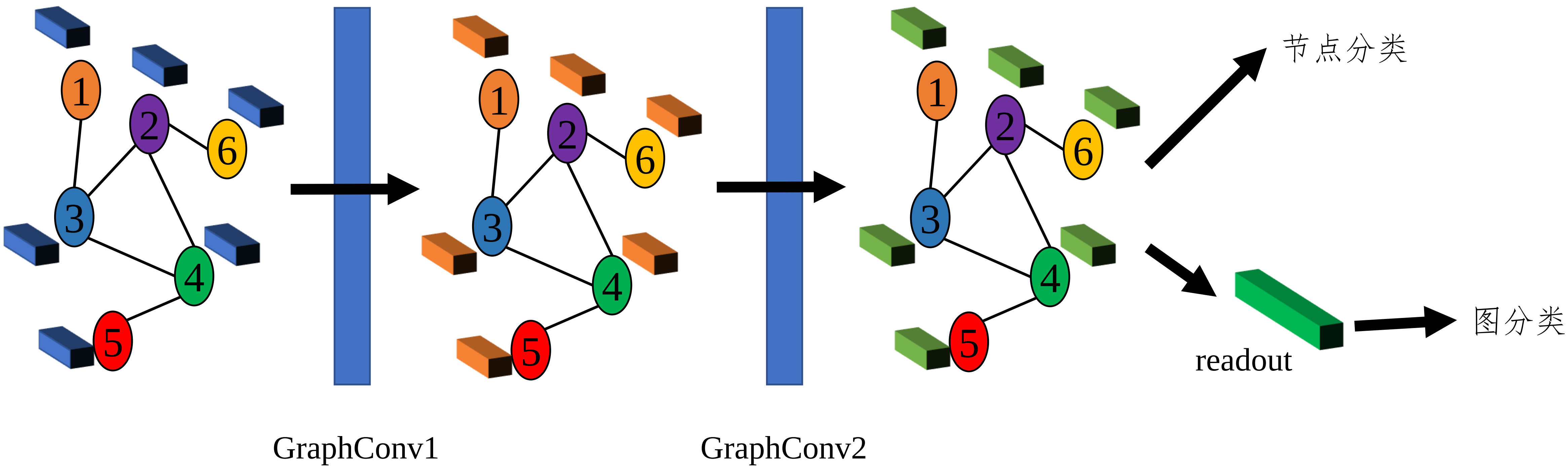
spatial-based	spectral-based
在图的空间拓扑结构上进行，也叫Message Passing Neural Network (MPNN)	运用谱图理论，在**频域**进行
可以处理大型图，可以在部分节点中执行，而不是整个图，还可以引入节点采样来提高效率	计算成本随着图的大小增加而急剧增加，难以处理大型图
各种图都行	只可以在无向图上工作，因为有向图上拉普拉斯矩阵没有定义。在有向图上应用，就只能化有向图为无向图。
在每个节点执行本地卷积，可以在不同的位置和结构之间共享权重	假定一个固定的图，难以添加新的节点
GAT、GraphSAGE…	GCN

GNN framework

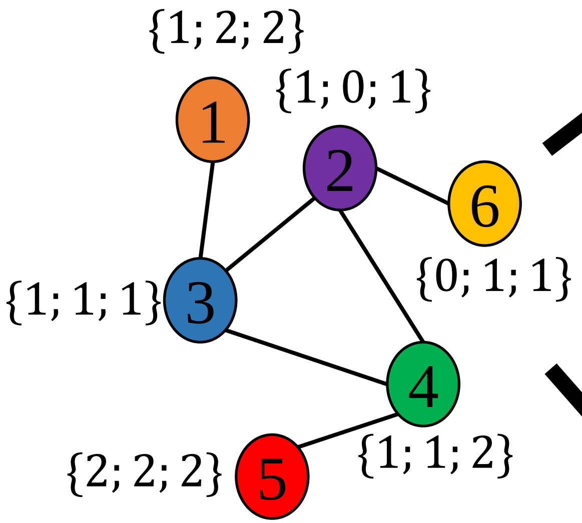


ICML2017, Gilmer J, et al. Neural message passing for quantum chemistry

GNN framework



How is a simplest GNN implemented?



$A: N \times N$

	1	2	3	4	5	6
1	0	0	1	0	0	0
2	0	0	1	1	0	1
3	1	1	0	1	0	0
4	0	1	1	0	1	0
5	0	0	0	1	0	0
6	0	1	0	0	0	0

$X^0: N \times D^0$

1	1	2	2
2	1	0	1
3	1	1	1
4	1	1	2
5	2	2	2
6	0	1	1

实现邻居sum聚合更新：

$$X^* = AX$$

1	1	1	1
2	3	4	5
3	3	3	5
4	4	3	4
5	1	1	2
6	1	0	1

$X^*: N \times D^0$

引入训练参数$W: D^0 \times D^1$

$$X^1 = AXW$$

1	…	…	…	…
2	…	…	…	…
3	…	…	…	…
4	…	…	…	…
5	…	…	…	…
6	…	…	…	…

$X^1: N \times D^1$

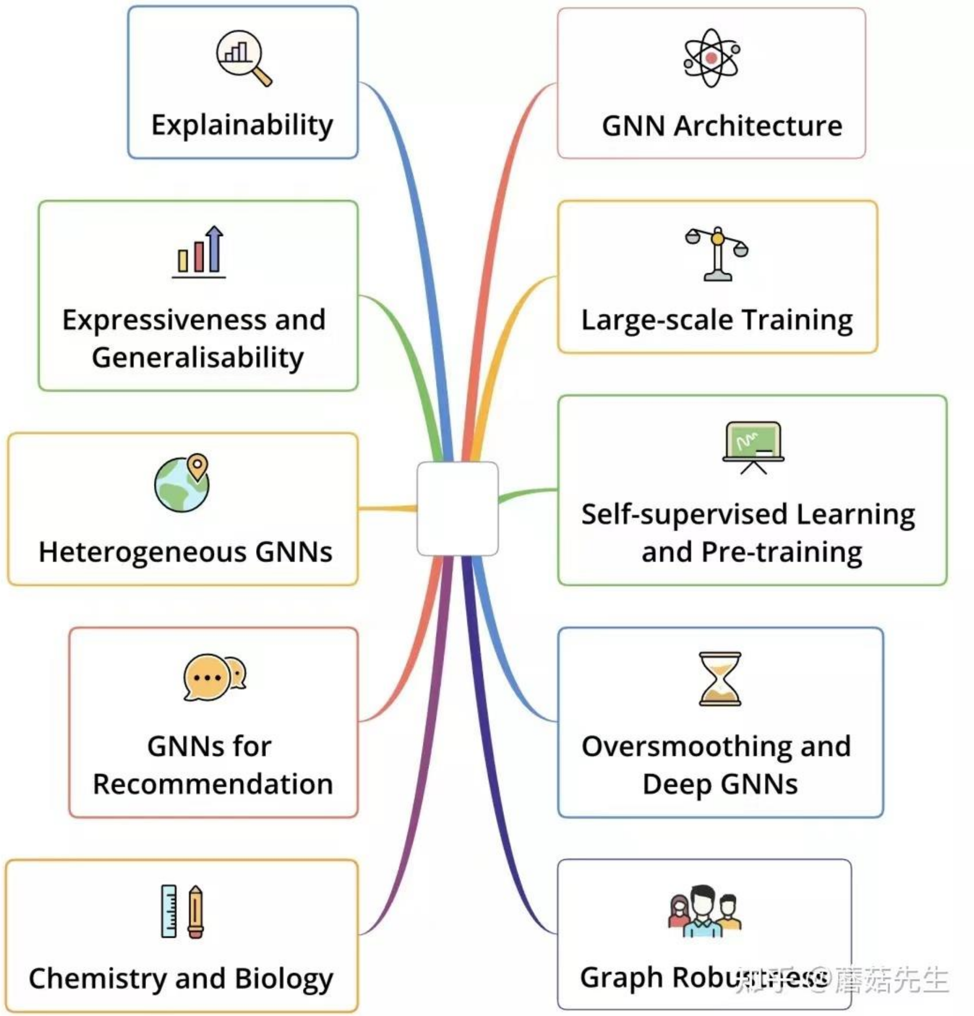
$$\boldsymbol{H_i^*} = \sum_{j \in N(i)} \boldsymbol{H_j W_j}$$

$$\boldsymbol{H^{k+1}} = \boldsymbol{A H^k W^k}$$

Common GNN variants

1. GAT:
   a) $e_{ij} = a(WH_i|WH_j)$
   b) $\alpha_{ij} = softmax(e_{ij})$
   c) $H_i^* = \sigma\left(\sum_{j \in N(i)} \alpha_{ij} WH_j\right)$
2. GraphSAGE:
   a) Mean aggregator: $H_i^* = \sigma(W \cdot mean(\{H_i\} \cup \{H_j, \forall j \in N(i)\}))$
   b) GCN aggregator: $H_i^* = \sigma(W \cdot concate(H_i, H_{N(i)})))$
   c) …
3. GIN: $H_i^* = MLP(\sum_{j \in N(i) \cup \{i\}} H_j)$
4. GCN: $H^* = \sigma(\widetilde{D}^{-\frac{1}{2}} \tilde{A} \widetilde{D}^{-\frac{1}{2}} HW)$

GNN-related research



GNN Python frameworks

Deep Graph library (DGL)

https://github.com/dmlc/dgl

PyTorch Geometric

https://github.com/pyg-team/pytorch_geometric

GNN Researchers

Jure Leskovec
(莱斯科维奇)

https://cs.stanford.edu/~jure/

Node2Vec, GraphSAGE,
GIN, DiffPool, PyG, OGB

William L. Hamilton

https://williamleif.github.io/

Graph Representation
Learning, GraphSAGE

Thomas N. Kipf
(托马斯·基普夫)

https://tkipf.github.io/

GCN

Petar Veličković
(维利克科维奇)

https://petar-v.com/

GAT, DGI

GNN Researchers

Muhan Zhang (张牧涵)

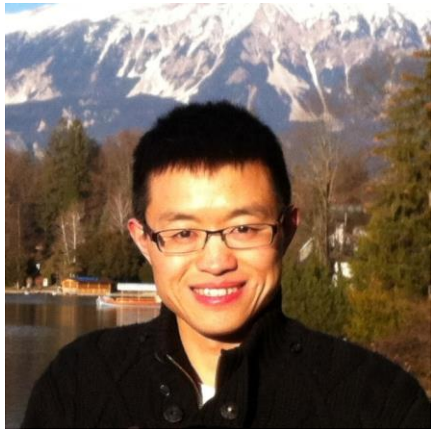
https://muhanzhang.github.io/

DGCNN, SEAL

Peng Cui (崔鹏)

https://pengcui.thumedialab.com/

SDNE



Jie Tang (唐杰)

https://keg.cs.tsinghua.edu.cn/jietang/

Graph Classification benchmark

Open Graph Benchmark (OGB)

https://ogb.stanford.edu/

https://ogb.stanford.edu/docs/leader_graphprop/

TUDataset

https://chrsmrrs.github.io/datasets/docs/home/

https://paperswithcode.com/task/graph-classification

Graph Classification experimental setting

图分类的衡量指标通常是准确率（±标准差），但对于实验设置，目前存在两套主流的方法：

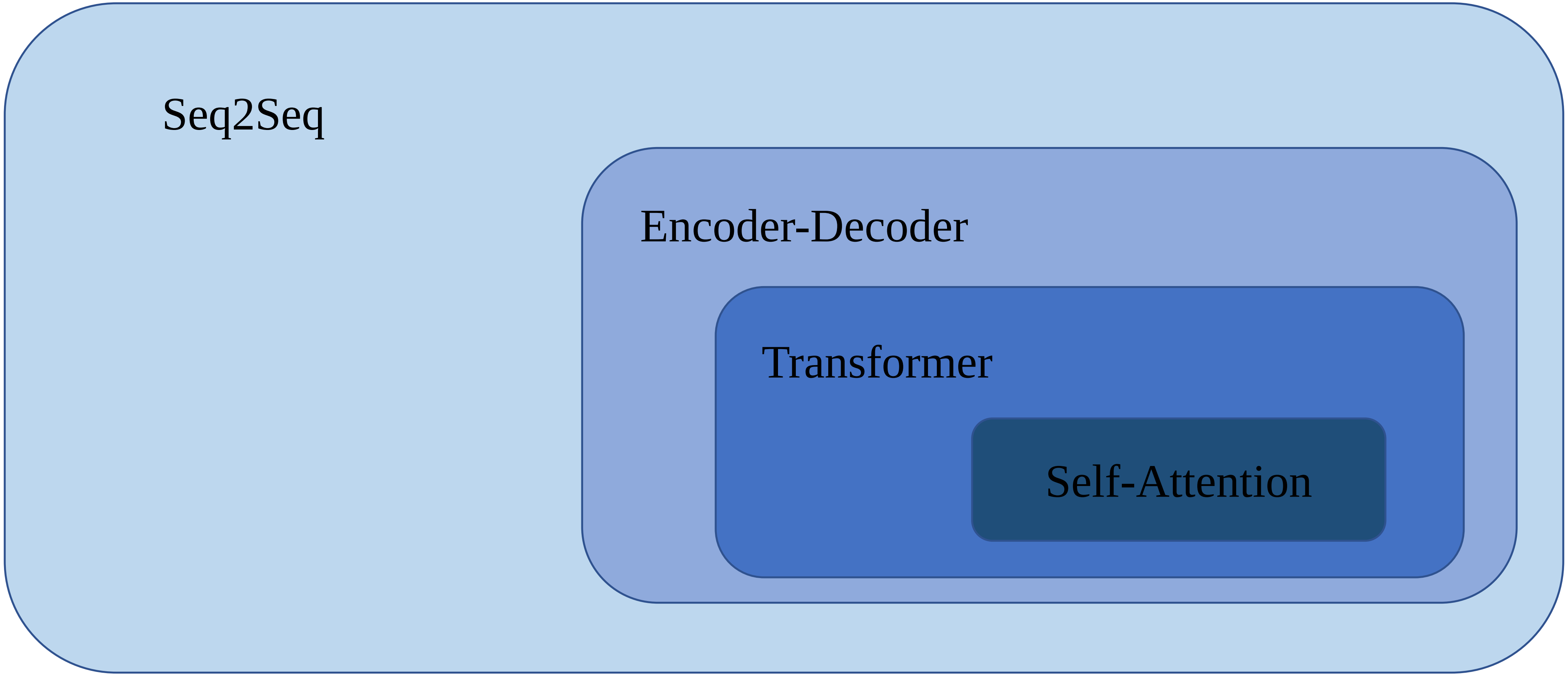
1. 10次十折交叉验证，每次十折交叉中，9折作为训练集，在训练集中划分验证集，用于参数搜索；1折作为测试集，用于验证模型性能，实验结果取100次实验模型的均值及标准差。(PATCHY-SAN, DGCNN)
2. 1次十折交叉验证，9折作为训练集，1折作为验证集，没有测试集，在每次训练过程中，记录验证集在每个epoch的准确率，一共得到10条验证集准确率曲线，将10条曲线做平均得到一条平均曲线，取平均曲线的最高点，作为最终的准确率。(GIN, DiffPool, U2GNN…)

ICLR 2020: A Fair Comparison of Graph Neural Networks for Graph Classification

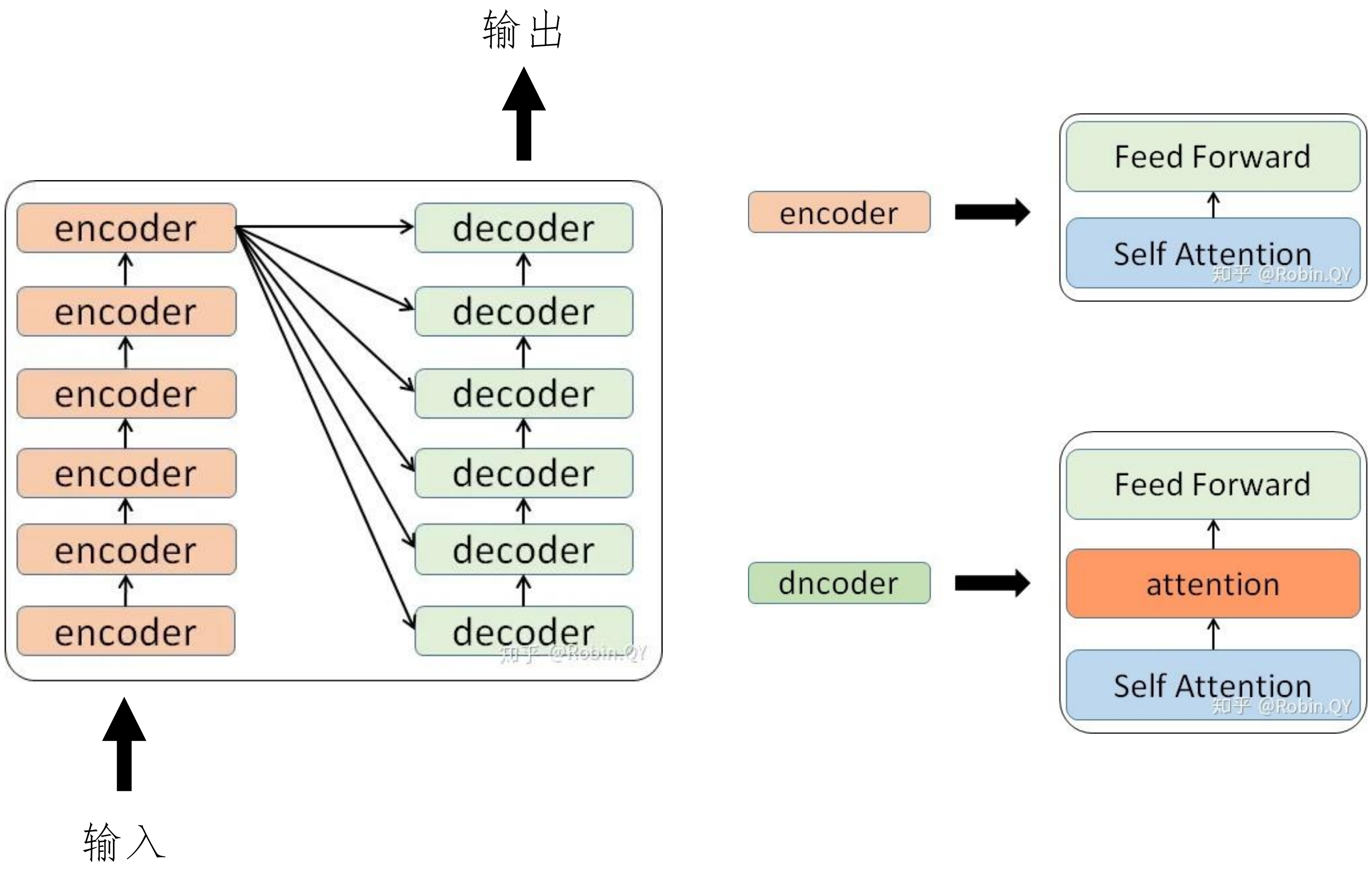
Graph Classification baselines

1. Graph kernels:
   1. GK
   2. SP
   3. RW
   4. WL
2. Deep learning
   1. DGK
   2. DGCNN
   3. PATCHY-SAN
   4. DiffPool
   5. ECC

Transformer



Transformer & Self-attention



U2GNN

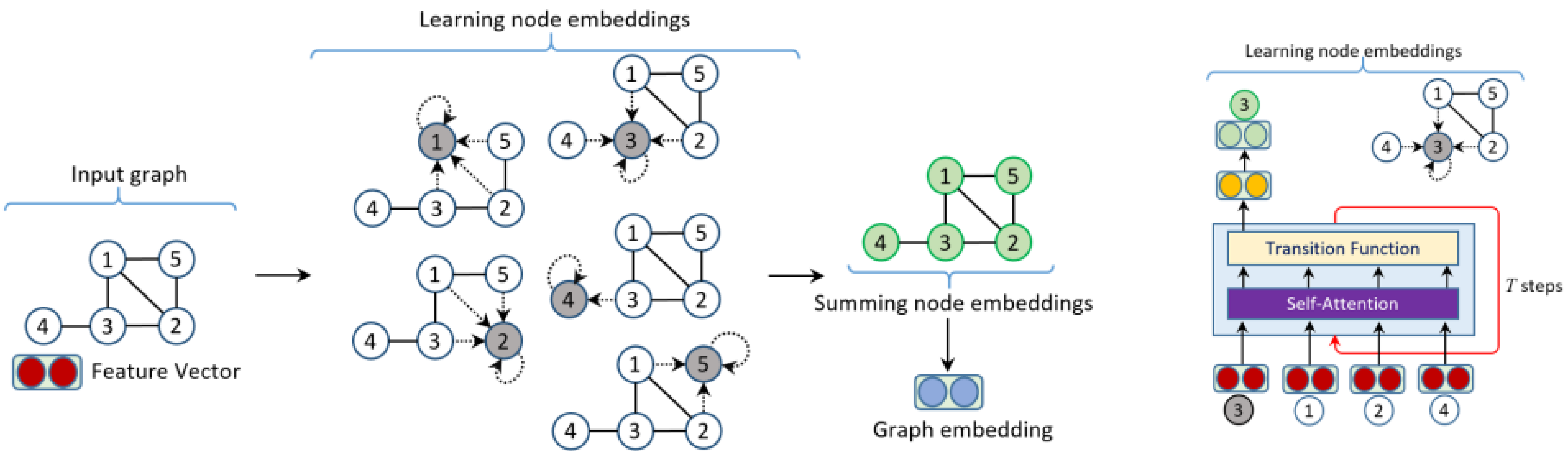


Figure 1: Illustration of our U2GNN.

U2GNN

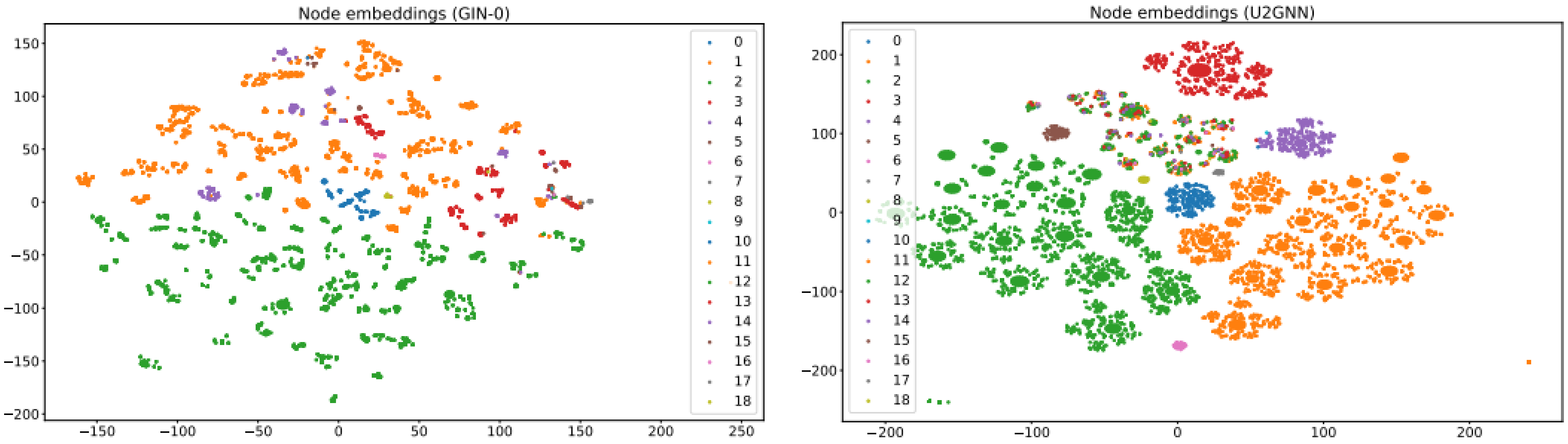


Figure 2: A t-SNE visualization of the node embeddings learned by GIN-0 and our U2GNN on the PTC dataset.

U2GNN

Table 2: Graph classification results (% accuracy). The best scores are in bold.

Model	COLLAB	IMDB-B	IMDB-M	DD	PROTEINS	MUTAG	PTC
GK [39]	72.84 ± 0.28	65.87 ± 0.98	43.89 ± 0.38	78.45 ± 0.26	71.67 ± 0.55	81.58 ± 2.11	57.26 ± 1.41
WL [38]	79.02 ± 1.77	73.40 ± 4.63	49.33 ± 4.75	79.78 ± 0.36	74.68 ± 0.49	82.05 ± 0.36	57.97 ± 0.49
PSCN [34]	72.60 ± 2.15	71.00 ± 2.29	45.23 ± 2.84	77.12 ± 2.41	75.89 ± 2.76	**92.63 ± 4.21**	62.29 ± 5.68
GCN [24]	**81.72 ± 1.64**	73.30 ± 5.29	51.20 ± 5.13	79.12 ± 3.07	75.65 ± 3.24	87.20 ± 5.11	–
GFN [7]	81.50 ± 2.42	73.00 ± 4.35	51.80 ± 5.16	78.78 ± 3.49	76.46 ± 4.06	90.84 ± 7.22	–
GraphSAGE [16]	79.70 ± 1.70	72.40 ± 3.60	49.90 ± 5.00	65.80 ± 4.90	65.90 ± 2.70	79.80 ± 13.9	–
GAT [42]	75.80 ± 1.60	70.50 ± 2.30	47.80 ± 3.10	–	74.70 ± 2.20	89.40 ± 6.10	66.70 ± 5.10
DGCNN [54]	73.76 ± 0.49	70.03 ± 0.86	47.83 ± 0.85	79.37 ± 0.94	75.54 ± 0.94	85.83 ± 1.66	58.59 ± 2.47
SAGPool [27]	–	–	–	76.45 ± 0.97	71.86 ± 0.97	–	–
PPGN [30]	81.38 ± 1.42	73.00 ± 5.77	50.46 ± 3.59	–	77.20 ± 4.73	90.55 ± 8.70	66.17 ± 6.54
CapsGNN [47]	79.62 ± 0.91	73.10 ± 4.83	50.27 ± 2.65	75.38 ± 4.17	76.28 ± 3.63	86.67 ± 6.88	–
DSGC [37]	79.20 ± 1.60	73.20 ± 4.90	48.50 ± 4.80	77.40 ± 6.40	74.20 ± 3.80	86.70 ± 7.60	–
GIN-0 [48]	80.20 ± 1.90	75.10 ± 5.10	52.30 ± 2.80	–	76.20 ± 2.80	89.40 ± 5.60	64.60 ± 7.00
GCAPS [43]	77.71 ± 2.51	71.69 ± 3.40	48.50 ± 4.10	77.62 ± 4.99	76.40 ± 4.17	–	66.01 ± 5.91
IEGN [31]	77.92 ± 1.70	71.27 ± 4.50	48.55 ± 3.90	–	75.19 ± 4.30	84.61 ± 10.0	59.47 ± 7.30
U2GNN	77.84 ± 1.48	**77.04 ± 3.45**	**53.60 ± 3.53**	**80.23 ± 1.48**	**78.53 ± 4.07**	89.97 ± 3.65	**69.63 ± 3.60**

Unsupervised U2GNN

$$\mathcal{L}_{\mathsf{U2GNN}}(\mathsf{v}) = -\log \frac{\exp(\mathbf{o}_{\mathsf{v}} \cdot \mathbf{e}_{\mathsf{v}})}{\sum_{\mathsf{v}' \in \mathcal{V}'} \exp(\mathbf{o}_{\mathsf{v}'} \cdot \mathbf{e}_{\mathsf{v}})}$$

Model	COLLAB	IMDB-B	IMDB-M	DD	PROTEINS	MUTAG	PTC
DGK [49]	73.09 ± 0.25	66.96 ± 0.56	44.55 ± 0.52	73.50 ± 1.01	75.68 ± 0.54	87.44 ± 2.72	60.08 ± 2.55
AWE [20]	73.93 ± 1.94	74.45 ± 5.83	51.54 ± 3.61	71.51 ± 4.02	–	87.87 ± 9.76	–
uGCN	93.28 ± 0.99	94.50 ± 2.79	81.66 ± 3.16	94.31 ± 1.71	**89.09 ± 3.25**	**95.36 ± 2.64**	**92.67 ± 4.60**
U2GNN	**95.62 ± 0.92**	**96.41 ± 1.94**	**89.20 ± 2.52**	**95.67 ± 1.89**	80.01 ± 3.21	88.47 ± 7.13	91.81 ± 6.61

Papers

https://github.com/thunlp/GNNPapers

https://github.com/benedekrozemberczki/awesome-graph-classification