Visual
Intelligence
Lab.

Graph U-nets

VILab winter study
2021.01.19

이은혜

Contents

- Introduction
- Motivation
- Methods
- Experiments
- Q&A

Contents

- **Introduction**
  - Graph U-Nets(ICML19)
- Motivation
- Methods
- Experiments
- Q&A

Introduction

- Graph U-Nets (ICML19)
  - U-Net architecture for Graph
  - Proposed graph pooling and unpooling methods

Graph u-nets

H Gao, S Ji - arXiv preprint arXiv:1905.05178, 2019 - arxiv.org

We consider the problem of representation learning for graph data. Convolutional neural networks can naturally operate on images, but have significant challenges in dealing with graph data. Given images are special cases of graphs with nodes lie on 2D lattices, graph .

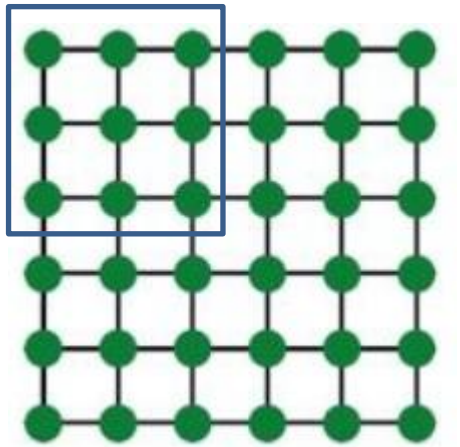
☆ 99 170회 인용 관련 학술자료 전체 12개의 버전 »

Contents

- Introduction
- **Motivation**
  - U-nets(2015)
  - Image vs Graph
- Methods
- Experiments
- Q&A

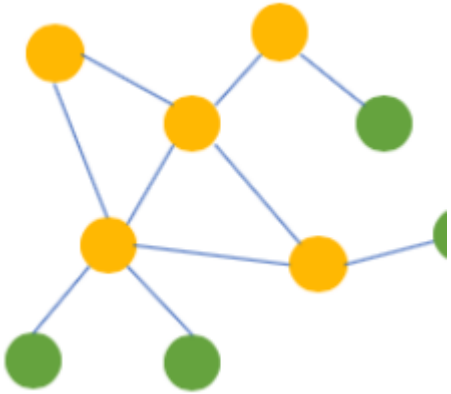
Motivation

- Image vs Graph



Image

- grid-like data
- special graph with well-defined locality

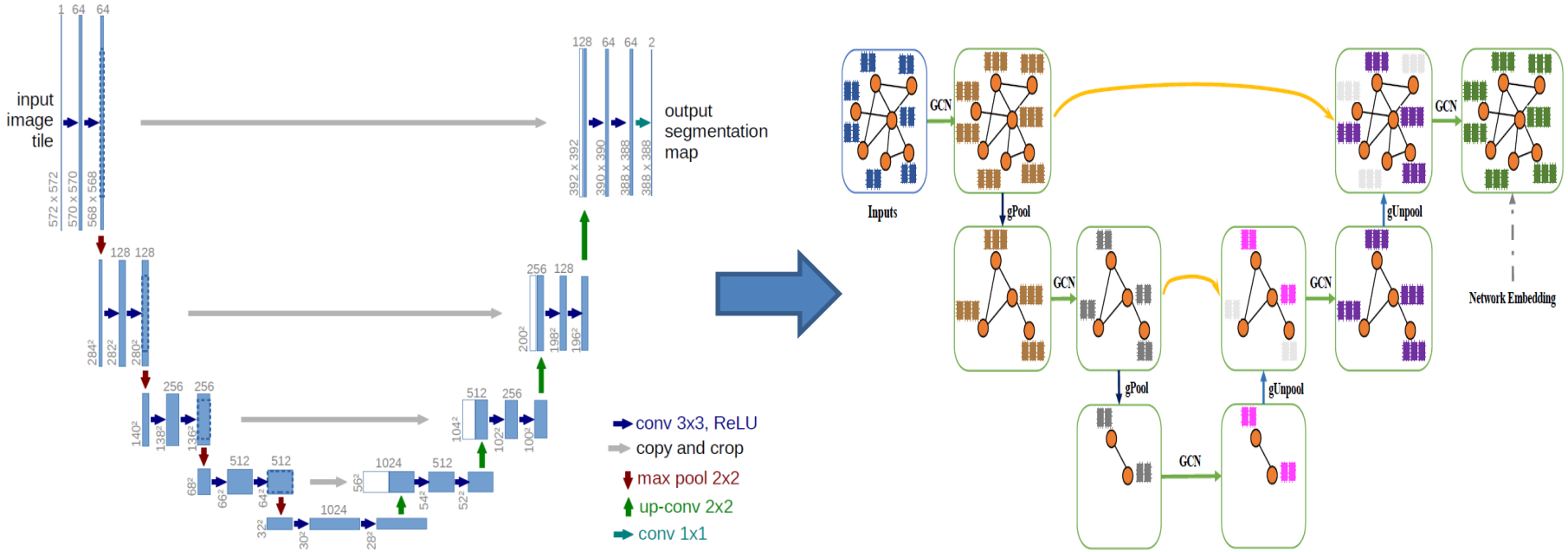


graph

- No spatial locality and order information among nodes
- **cannot directly apply pooling and unpooling operation**

Motivation

- Image vs Graph
  - Node classification and Image segmentation tasks are similar. Both predict for each input unit, pixel on images and node on graphs.



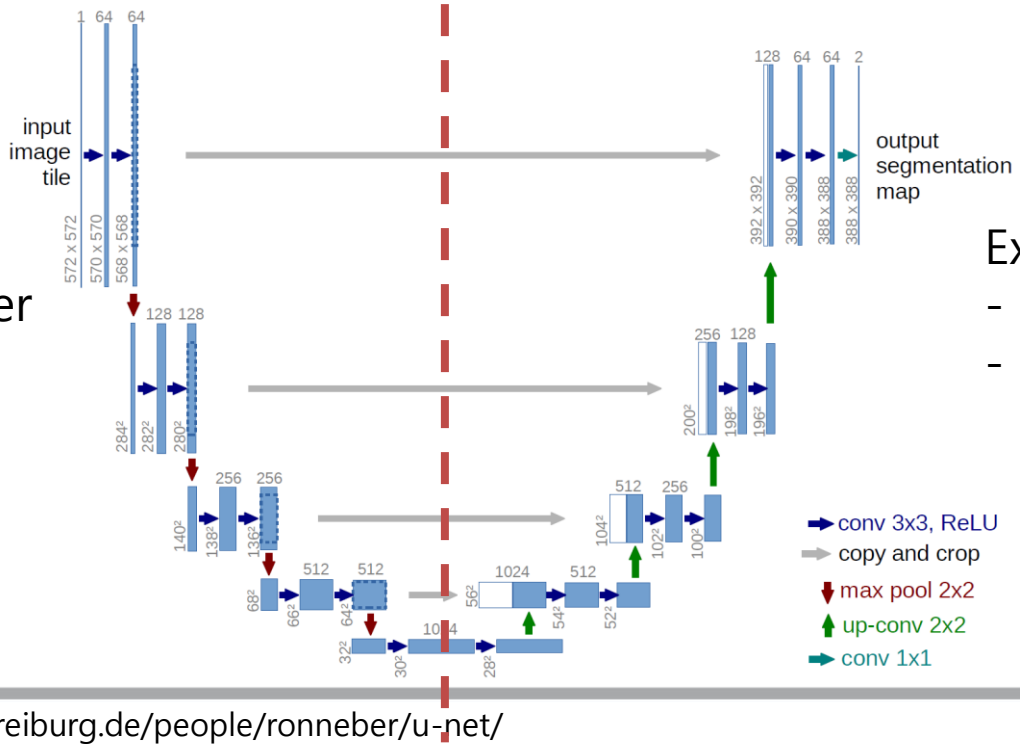
U-Nets
For image segmentation

Graph U-Nets
For node classification

Motivation

- U-Net
  - Contracting path: feature map downsapling, catch the context of input image.
  - Expanding path: feature map upsampling + corresponding feature map of contracting path, reflect localization

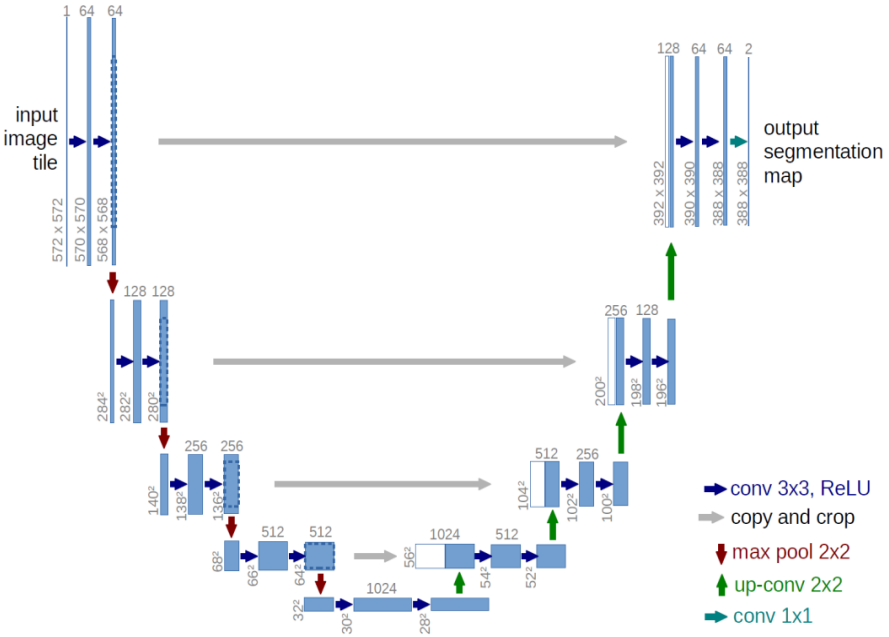
Contracting path
- Convolution layer
- Pooling layer



Expanding path
- Convolution layer
- Upsampling layer

https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/

Motivation



U-Nets

- Conv layer
- Pooling layer
- Unpooling layer

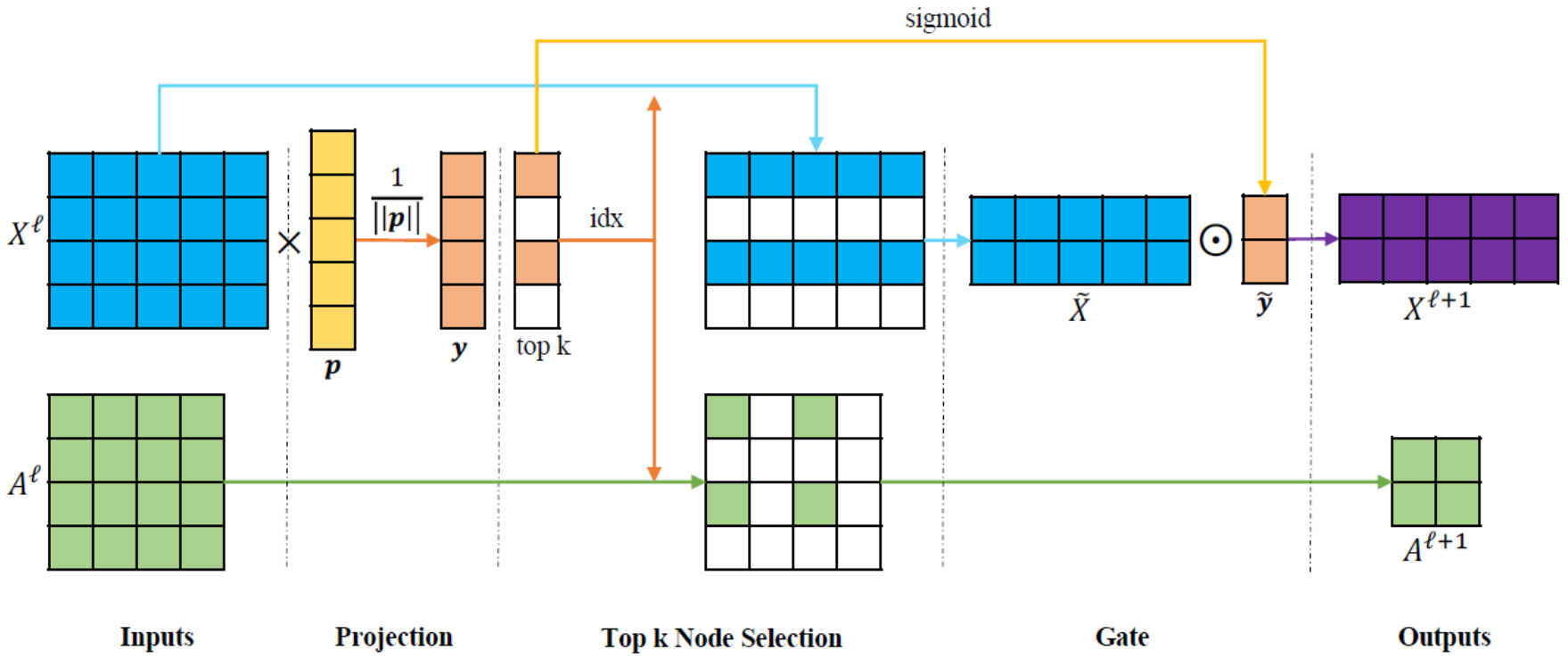
Graph U-Nets

- GCN layer
- gPool layer
- gUnpool layer

Contents

- Introduction
- Motivation
- **Methods**
  - gPool Layer
  - gUnpool Layer
  - Graph U-Nets
- Experiments
- Q&A

Graph Pooling Layer(gPool Layer)



- The graph can be represented by two matrices
  - the feature matrix $X^\ell \in \mathbb{R}^{N \times C}$.
  - the adjacency matrix $A^\ell \in \mathbb{R}^{N \times N}$
- **p**: trainable projection vector
- There are N nodes in a graph G and each node contains C features.
  - In this example, N = 4, C = 5, k = 2

Graph Pooling Layer(gPool Layer)

- Projection
  - 현 layer의 feature matrix X를 trainable parameter인 projection vector **p** 위로 projection시켜서 각 노드의 scalar projection value를 갖는 y를 구한다.

$$\mathbf{y} = X^{\ell}\mathbf{p}^{\ell}/\|\mathbf{p}^{\ell}\|,$$

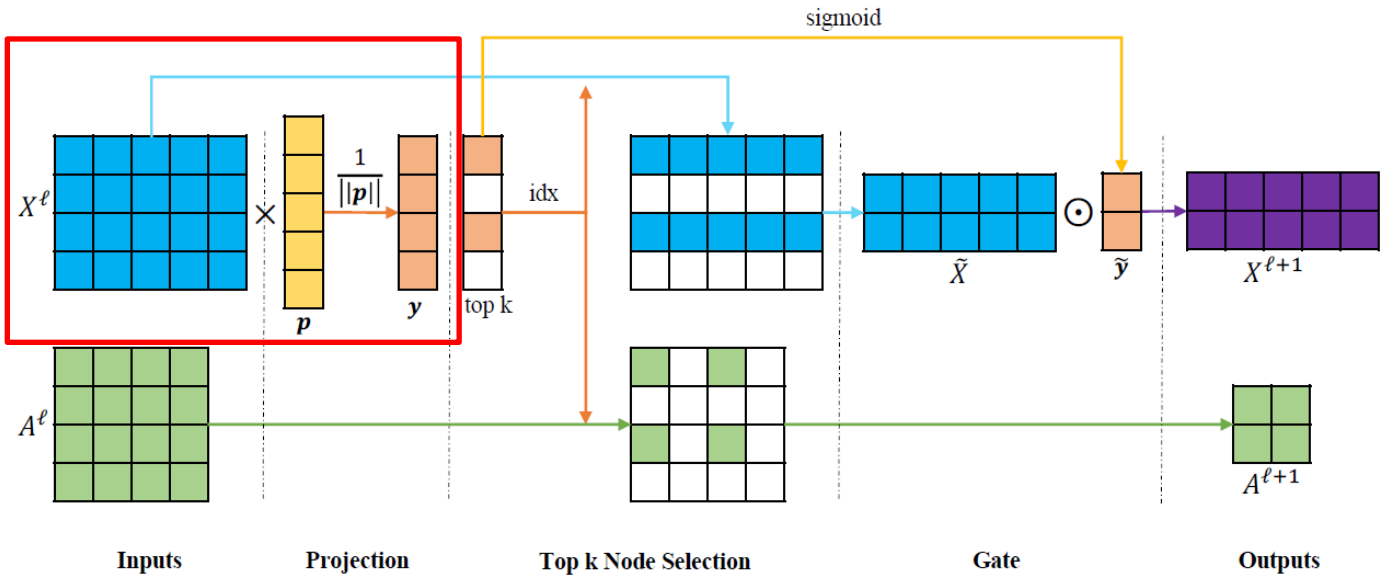
$$\text{idx} = \text{rank}(\mathbf{y}, k),$$

$$\tilde{\mathbf{y}} = \text{sigmoid}(\mathbf{y}(\text{idx})),$$

$$\tilde{X}^{\ell} = X^{\ell}(\text{idx}, :),$$

$$A^{\ell+1} = A^{\ell}(\text{idx}, \text{idx}),$$

$$X^{\ell+1} = \tilde{X}^{\ell} \odot (\tilde{\mathbf{y}}\mathbf{1}_C^T),$$



Graph Pooling Layer(gPool Layer)

- Top k Node Selection
  - Return the k-largest values in y and their indices.
    -> selected node for the new graph.

$$\mathbf{y} = X^{\ell}\mathbf{p}^{\ell}/\|\mathbf{p}^{\ell}\|,$$

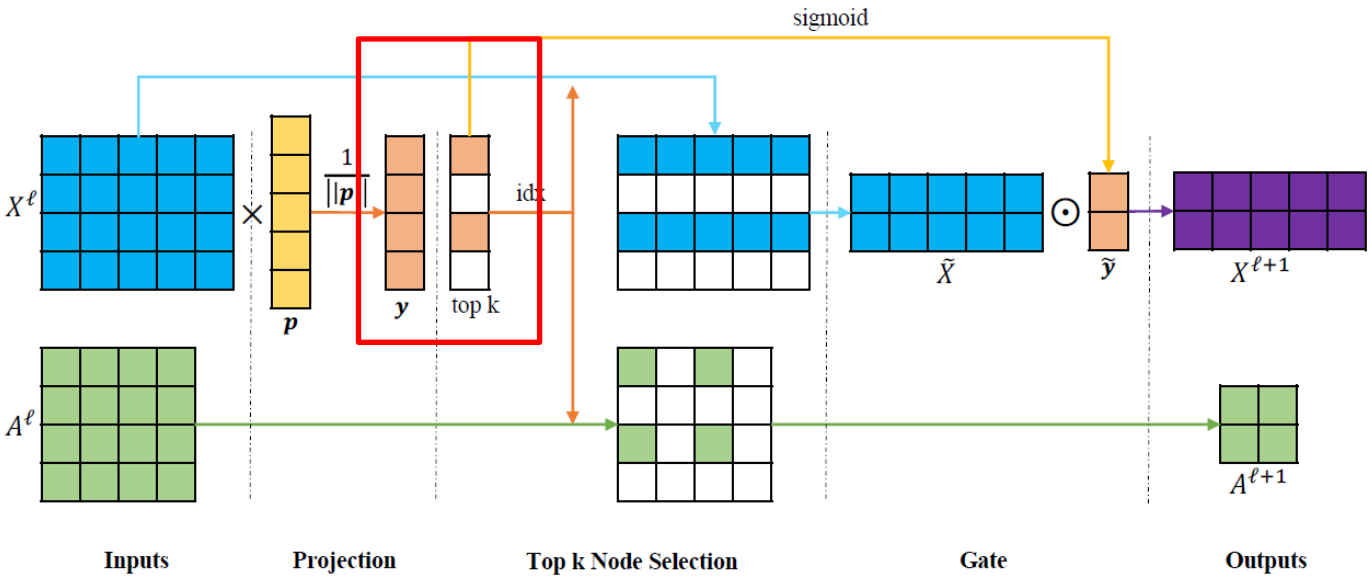
$$\text{idx} = \text{rank}(\mathbf{y}, k),$$

$$\tilde{\mathbf{y}} = \text{sigmoid}(\mathbf{y}(\text{idx})),$$

$$\tilde{X}^{\ell} = X^{\ell}(\text{idx}, :),$$

$$A^{\ell+1} = A^{\ell}(\text{idx}, \text{idx}),$$

$$X^{\ell+1} = \tilde{X}^{\ell} \odot (\tilde{\mathbf{y}}\mathbf{1}_C^T),$$



Graph Pooling Layer(gPool Layer)

- Top k Node Selection
  - Make new feature map X~ and adjacency matrix A with Selected nodes.
  - Adjacency matrix is finished.

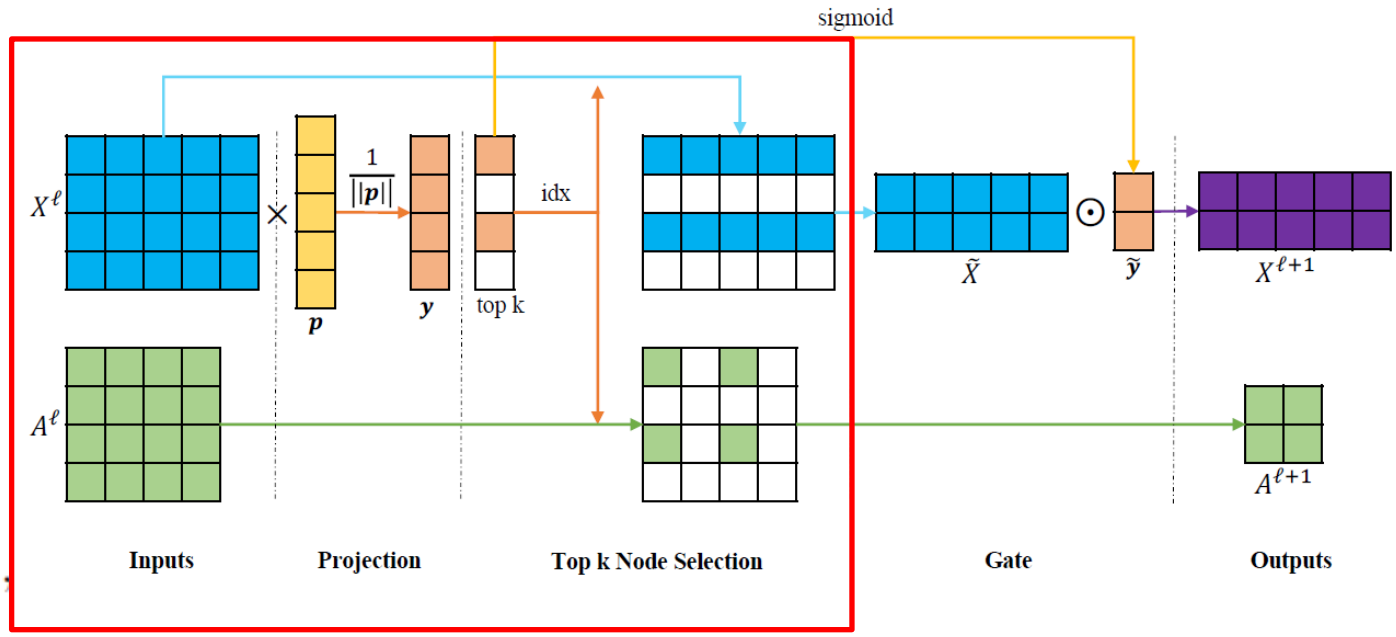
$$\mathbf{y} = X^{\ell}\mathbf{p}^{\ell}/\|\mathbf{p}^{\ell}\|,$$

$$\text{idx} = \text{rank}(\mathbf{y}, k),$$

$$\tilde{\mathbf{y}} = \text{sigmoid}(\mathbf{y}(\text{idx})),$$

$$\tilde{X}^{\ell} = X^{\ell}(\text{idx}, :),$$
$$A^{\ell+1} = A^{\ell}(\text{idx}, \text{idx}),$$

$$X^{\ell+1} = \tilde{X}^{\ell} \odot (\tilde{\mathbf{y}}\mathbf{1}_C^T),$$



Graph Pooling Layer(gPool Layer)

- Gate operation
  - Apply sigmoid function to y and return gate vector y~.
  - Element-wise matrix product of X~ and y~ -> gate
  - It makes projection vector **p** trainable by back-propagation.

$$\mathbf{y} = X^{\ell}\mathbf{p}^{\ell}/\|\mathbf{p}^{\ell}\|,$$

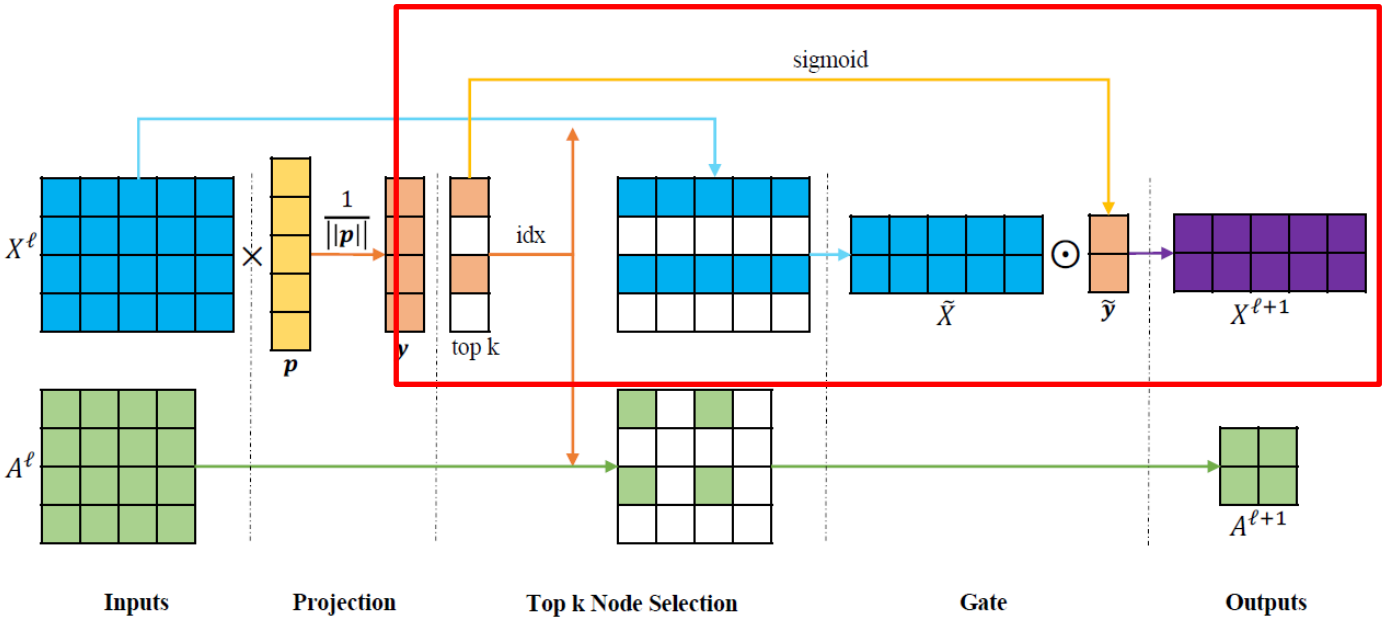
$$\text{idx} = \text{rank}(\mathbf{y}, k),$$

$$\tilde{\mathbf{y}} = \text{sigmoid}(\mathbf{y}(\text{idx})),$$

$$\tilde{X}^{\ell} = X^{\ell}(\text{idx}, :),$$

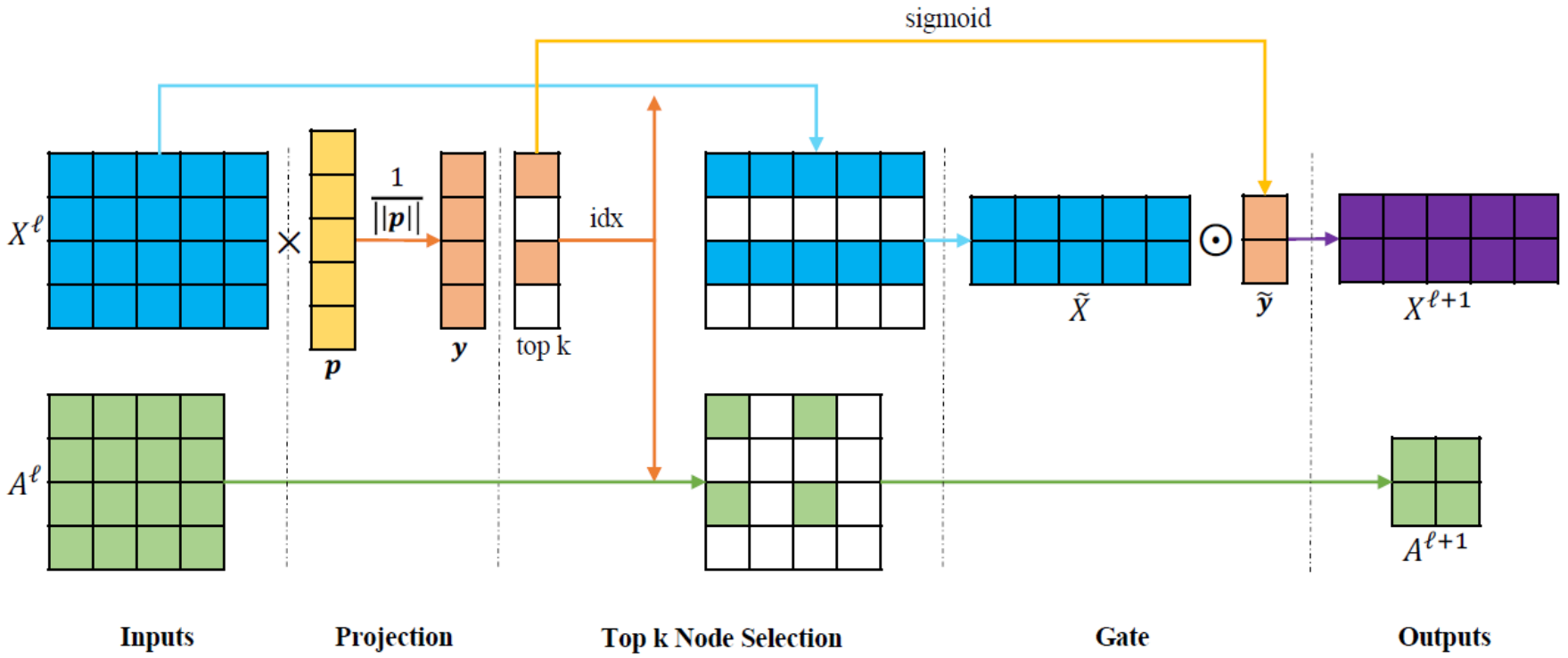
$$A^{\ell+1} = A^{\ell}(\text{idx}, \text{idx}),$$

$$X^{\ell+1} = \tilde{X}^{\ell} \odot (\tilde{\mathbf{y}}\mathbf{1}_C^T),$$



* A vector of size C with all components bring 1 곱셈: element-wise 연산을 위해 y~ 차원 늘려줌.

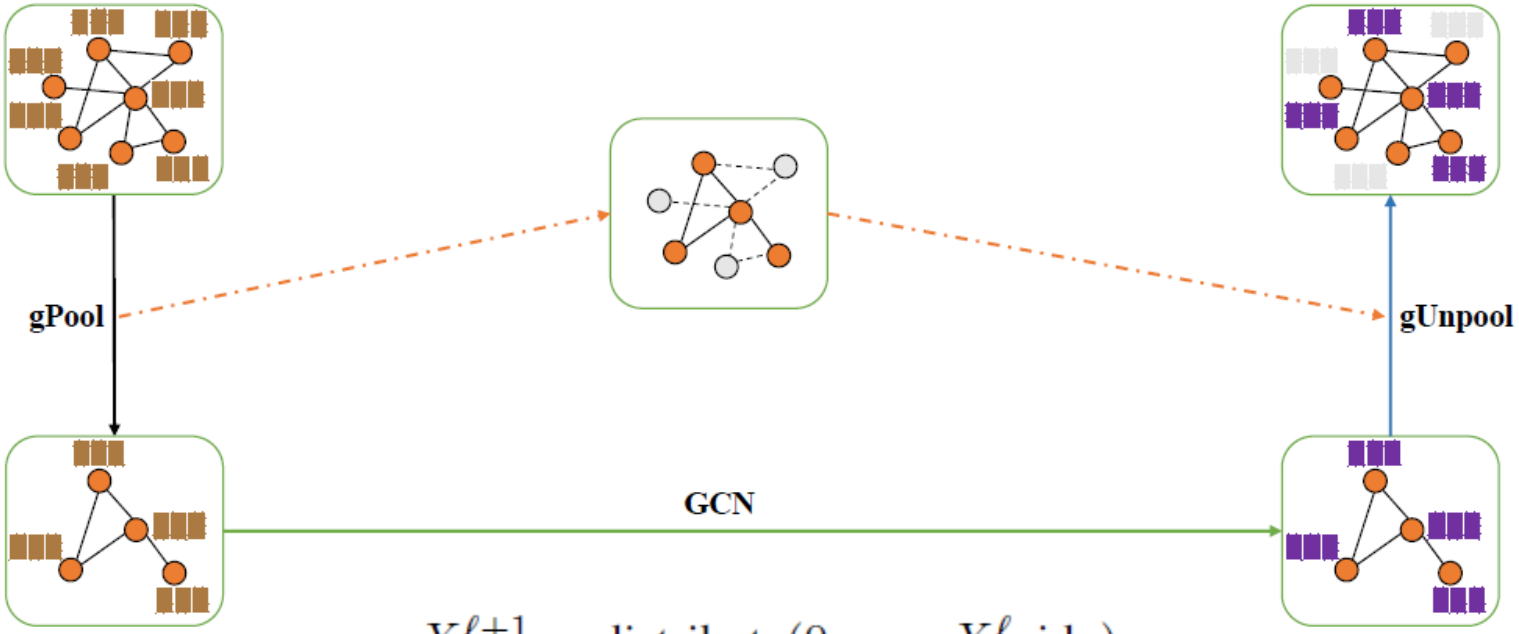
Graph Pooling Layer(gPool Layer)



- Output graph
  - The feature matrix dimension : N x C -> k x C
  - The adjacency matrix dimension : N x N -> k x k

Graph Unpooling Layer(gUnpool Layer)

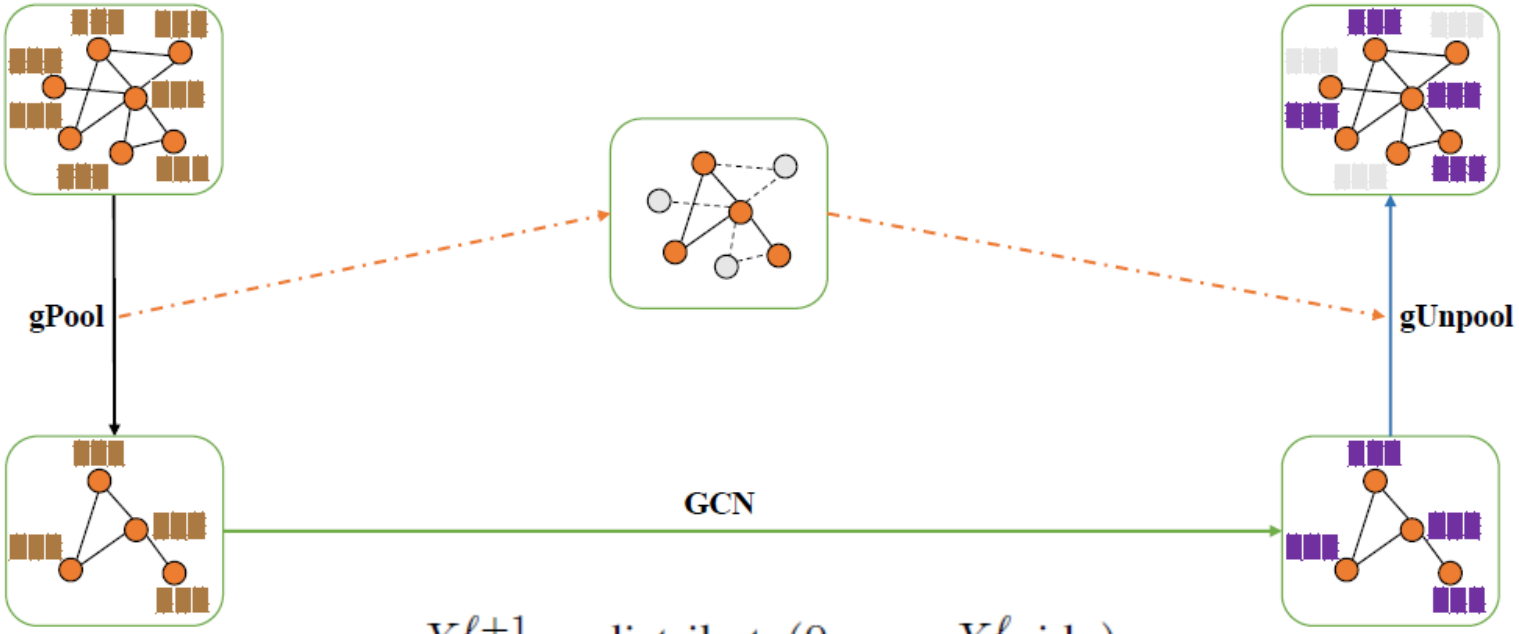
- Inverse operation of the gPool layer
  - Restore graph into its original structure.
  - For this, record the locations of nodes selected in the corresponding gPool layer.



$$X^{\ell+1} = \text{distribute}(0_{N\times C}, X^{\ell}, \text{idx}),$$

Graph Unpooling Layer(gUnpool Layer)

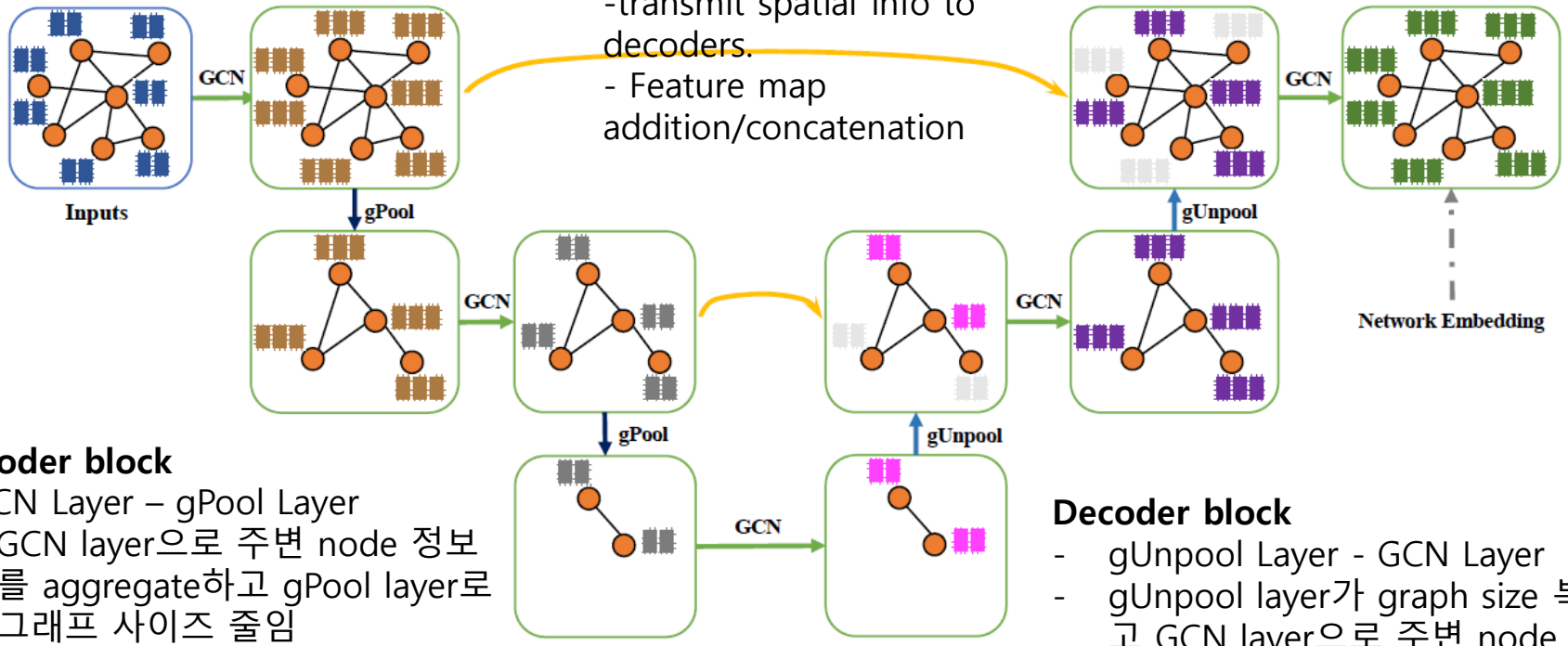
- Inverse operation of the gPool layer
  - 0NxC : initial empty feature matrix for the new graph
  - distribute operation: Selected nodes의 feature values는 현 레이어의 feature matrix에서 그대로 가져오고 선택되지 않은 노드들의 feature values은 0으로 놔둔다.



$$X^{\ell+1} = \text{distribute}(0_{N\times C}, X^{\ell}, \text{idx}),$$

Graph U-nets

- Apply graph embedding layer to input graph.
  - Convert nodes into low-dimensional representations.
- Encoder – Decoder

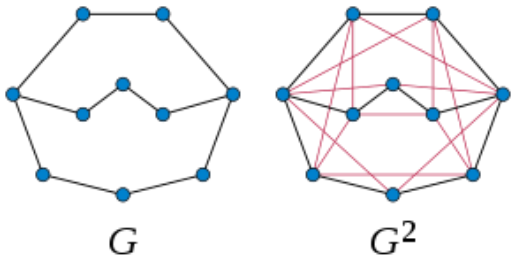


Encoder block
- GCN Layer – gPool Layer
- GCN layer으로 주변 node 정보를 aggregate하고 gPool layer로 그래프 사이즈 줄임

Decoder block
- gUnpool Layer - GCN Layer
- gUnpool layer가 graph size 복구하고 GCN layer으로 주변 node 정보 aggregate

Graph U-nets

- Graph connectivity augmentation
  - Pooling 할 때 선택되지 않은 node들이 제거되면서 connectivity가 약화되어 주변 노드에서 얻을 수 있는 정보가 줄어든다.
  - Connectivity augmentation를 위해 GCN layer에 k=2인 graph power을 적용한다.



G G^2

$$A^2 = A^\ell A^\ell, \qquad A^{\ell+1} = A^2(\text{idx}, \text{idx}),$$

- Improved GCN Layer
  - Information aggregation을 수행할 때 주변 노드의 feature vector보다 자기 feature vector에 높은 weight를 주기 위해 self loop에 가중치를 준다.

$$X_{\ell+1} = \sigma(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}X_\ell W_\ell),$$

$$\hat{A} = A + I \quad \Rightarrow \quad \hat{A} = A + 2I$$

Contents

- Introduction
- Motivation
- Methods
- **Experiments**
- Q&A

Experiments

Table 3. Results of transductive learning experiments in terms of node classification accuracies on Cora, Citeseer, and Pubmed datasets. g-U-Nets denotes our proposed graph U-Nets model.

Models	Cora	Citeseer	Pubmed
DeepWalk (Perozzi et al., 2014)	67.2%	43.2%	65.3%
Planetoid (Yang et al., 2016)	75.7%	64.7%	77.2%
Chebyshev (Defferrard et al., 2016)	81.2%	69.8%	74.4%
GCN (Kipf & Welling, 2017)	81.5%	70.3%	79.0%
GAT (Veličković et al., 2017)	$83.0 \pm 0.7\%$	$72.5 \pm 0.7\%$	$79.0 \pm 0.3\%$
g-U-Nets (Ours)	$\mathbf{84.4 \pm 0.6\%}$	$\mathbf{73.2 \pm 0.5\%}$	$\mathbf{79.6 \pm 0.2\%}$

Table 4. Results of inductive learning experiments in terms of graph classification accuracies on D&D, PROTEINS, and COLLAB datasets. g-U-Nets denotes our proposed graph U-Nets model.

Models	D&D	PROTEINS	COLLAB
PSCN (Niepert et al., 2016)	76.27%	75.00%	72.60%
DGCNN (Zhang et al., 2018)	79.37%	76.26%	73.76%
DiffPool-DET (Ying et al., 2018)	75.47%	75.62%	**82.13%**
DiffPool-NOLP (Ying et al., 2018)	79.98%	76.22%	75.58%
DiffPool (Ying et al., 2018)	80.64%	76.25%	75.48%
g-U-Nets (Ours)	**82.43%**	**77.68%**	77.56%

Experiments

- Without gPool and gUnpool Layer

Table 5. Comparison of g-U-Nets with and without gPool or gUnpool layers in terms of node classification accuracy on Cora, Citeseer, and Pubmed datasets.

Models	Cora	Citeseer	Pubmed
g-U-Nets without gPool or gUnpool	$82.1 \pm 0.6\%$	$71.6 \pm 0.5\%$	$79.1 \pm 0.2\%$
g-U-Nets (Ours)	$\mathbf{84.4 \pm 0.6\%}$	$\mathbf{73.2 \pm 0.5\%}$	$\mathbf{79.6 \pm 0.2\%}$

- Comparison with Network depths

Table 7. Comparison of different network depths in terms of node classification accuracy on Cora, Citeseer, and Pubmed datasets. Based on g-U-Nets, we experiment with different network depths in terms of the number of blocks in encoder and decoder parts.

Depth	Cora	Citeseer	Pubmed
2	$82.6 \pm 0.6\%$	$71.8 \pm 0.5\%$	$79.1 \pm 0.3\%$
3	$83.8 \pm 0.7\%$	$72.7 \pm 0.7\%$	$79.4 \pm 0.4\%$
4	$\mathbf{84.4 \pm 0.6\%}$	$\mathbf{73.2 \pm 0.5\%}$	$\mathbf{79.6 \pm 0.2\%}$
5	$84.1 \pm 0.5\%$	$72.8 \pm 0.6\%$	$79.5 \pm 0.3\%$

Q & A