2019.08.09 Seminar

Graph Convolutional Networks

이 민정

Contents

Introduction

Graph Convolutional Networks

1489회 인용

Published as a conference paper at ICLR 2017

SEMI-SUPERVISED CLASSIFICATION WITH GRAPH CONVOLUTIONAL NETWORKS

Thomas N. Kipf
University of Amsterdam
T.N.Kipf@uva.nl

Max Welling
University of Amsterdam
Canadian Institute for Advanced Research (CIFAR)
M.Welling@uva.nl

ABSTRACT

We present a scalable approach for semi-supervised learning on graph-structured data that is based on an efficient variant of convolutional neural networks which operate directly on graphs. We motivate the choice of our convolutional architecture via a localized first-order approximation of spectral graph convolutions. Our model scales linearly in the number of graph edges and learns hidden layer representations that encode both local graph structure and features of nodes. In a number of experiments on citation networks and on a knowledge graph dataset we demonstrate that our approach outperforms related methods by a significant margin.

Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.

Introduction

Graph Convolutional Networks

564회 인용

Inductive Representation Learning on Large Graphs

William L. Hamilton* wleif@stanford.edu

Rex Ying* rexying@stanford.edu

Jure Leskovec jure@cs.stanford.edu

Department of Computer Science
Stanford University
Stanford, CA, 94305

Abstract

Low-dimensional embeddings of nodes in large graphs have proved extremely useful in a variety of prediction tasks, from content recommendation to identifying protein functions. However, most existing approaches require that all nodes in the graph are present during training of the embeddings; these previous approaches are inherently *transductive* and do not naturally generalize to unseen nodes. Here we present GraphSAGE, a general *inductive* framework that leverages node feature information (e.g., text attributes) to efficiently generate node embeddings for previously unseen data. Instead of training individual embeddings for each node, we learn a function that generates embeddings by sampling and aggregating features from a node's local neighborhood. Our algorithm outperforms strong baselines on three inductive node-classification benchmarks: we classify the category of unseen nodes in evolving information graphs based on citation and Reddit post data, and we show that our algorithm generalizes to completely unseen graphs using a multi-graph dataset of protein-protein interactions.

Hamilton, W., Ying, Z., & Leskovec, J. (2017). Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems* (pp. 1024-1034).

Introduction

Graph Convolutional Networks

398회 인용

Published as a conference paper at ICLR 2018

Graph Attention Networks

Petar Veličković*
Department of Computer Science and Technology
University of Cambridge
petar.velickovic@cst.cam.ac.uk

Guillem Cucurull*
Centre de Visió per Computador, UAB
gcucurull@gmail.com

Arantxa Casanova*
Centre de Visió per Computador, UAB
ar.casanova.8@gmail.com

Adriana Romero
Montréal Institute for Learning Algorithms
adriana.romero.soriano@umontreal.ca

Pietro Liò
Department of Computer Science and Technology
University of Cambridge
pietro.lio@cst.cam.ac.uk

Yoshua Bengio
Montréal Institute for Learning Algorithms
yoshua.umontreal@gmail.com

Abstract

We present graph attention networks (GATs), novel neural network architectures that operate on graph-structured data, leveraging masked self-attentional layers to address the shortcomings of prior methods based on graph convolutions or their approximations. By stacking layers in which nodes are able to attend over their neighborhoods' features, we enable (implicitly) specifying different weights to different nodes in a neighborhood, without requiring any kind of costly matrix operation (such as inversion) or depending on knowing the graph structure upfront. In this way, we address several key challenges of spectral-based graph neural networks simultaneously, and make our model readily applicable to inductive as well as transductive problems. Our GAT models have achieved or matched state-of-the-art results across four established transductive and inductive graph benchmarks: the *Cora*, *Citeseer* and *Pubmed* citation network datasets, as well as a *protein-protein interaction* dataset (wherein test graphs remain unseen during training).

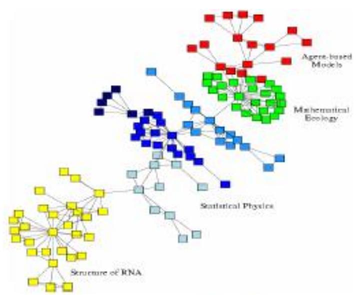
Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2017). Graph attention networks. arXiv preprint arXiv:1710.10903.

Introduction

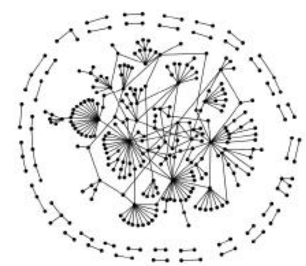
Graph Convolutional Networks

Graph Convolutional Networks

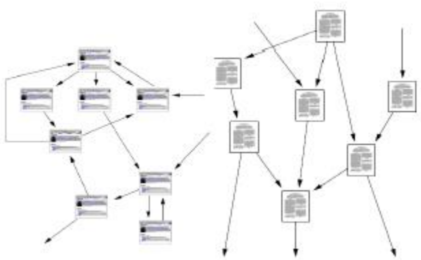
Social networks



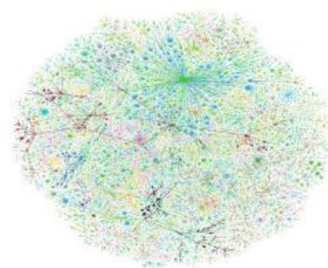
Economic networks



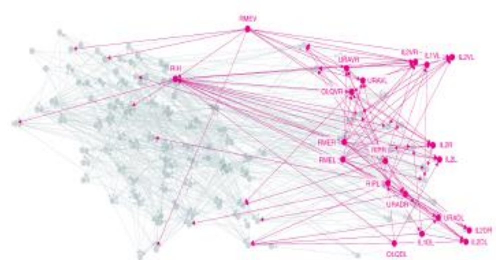
Biomedical networks



Information networks:
Web & citations



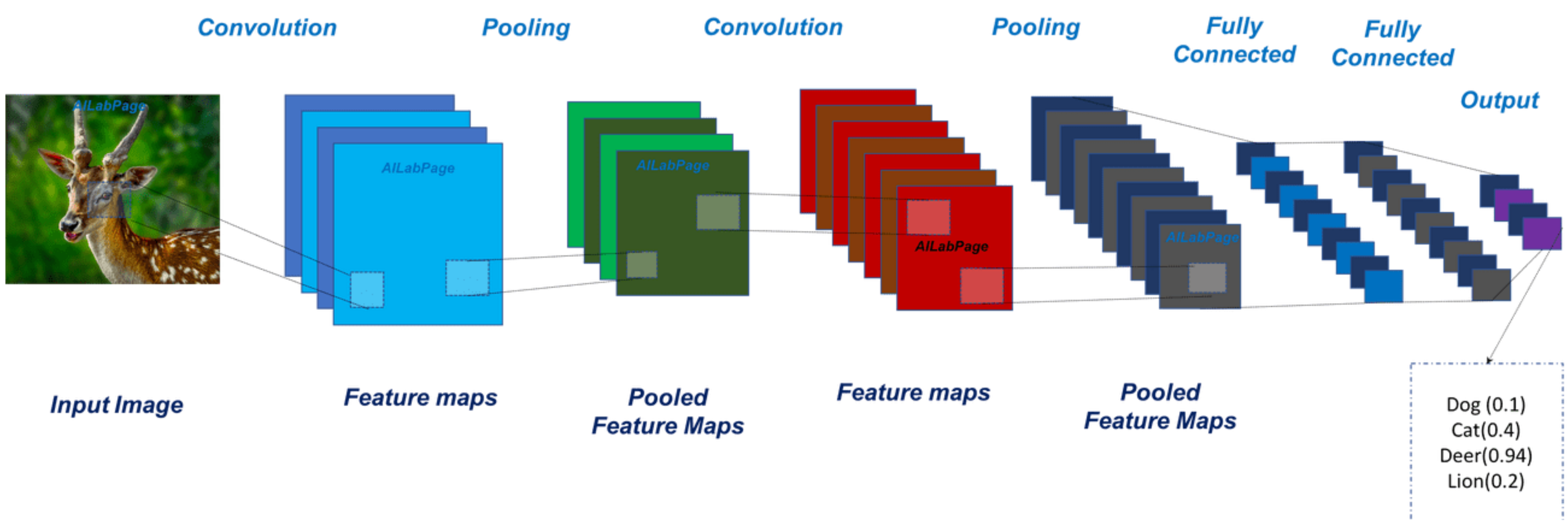
Internet



Networks of neurons

http://snap.stanford.edu/proj/embeddings-www/

Graph Convolutional Networks

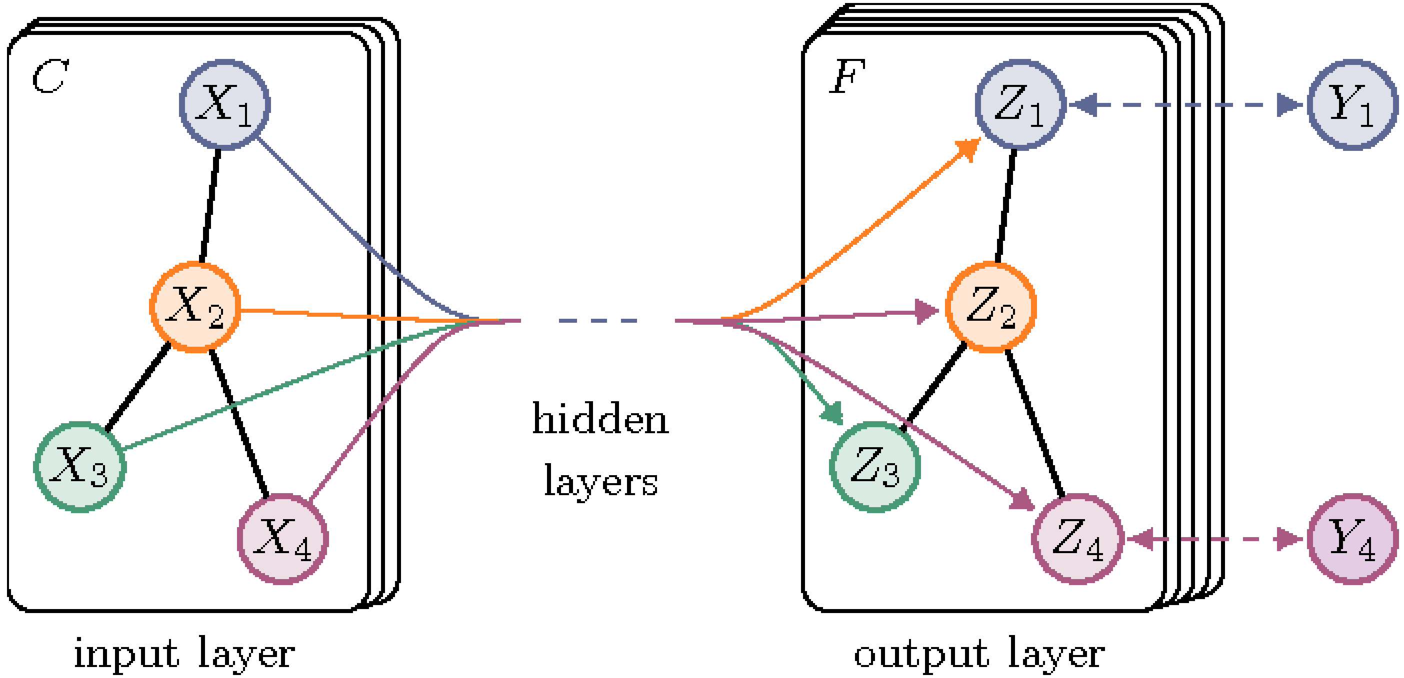


https://vinodsblog.com/2018/10/15/everything-you-need-to-know-about-convolutional-neural-networks/

Introduction

Graph Convolutional Networks

Graph Convolutional Networks



Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.

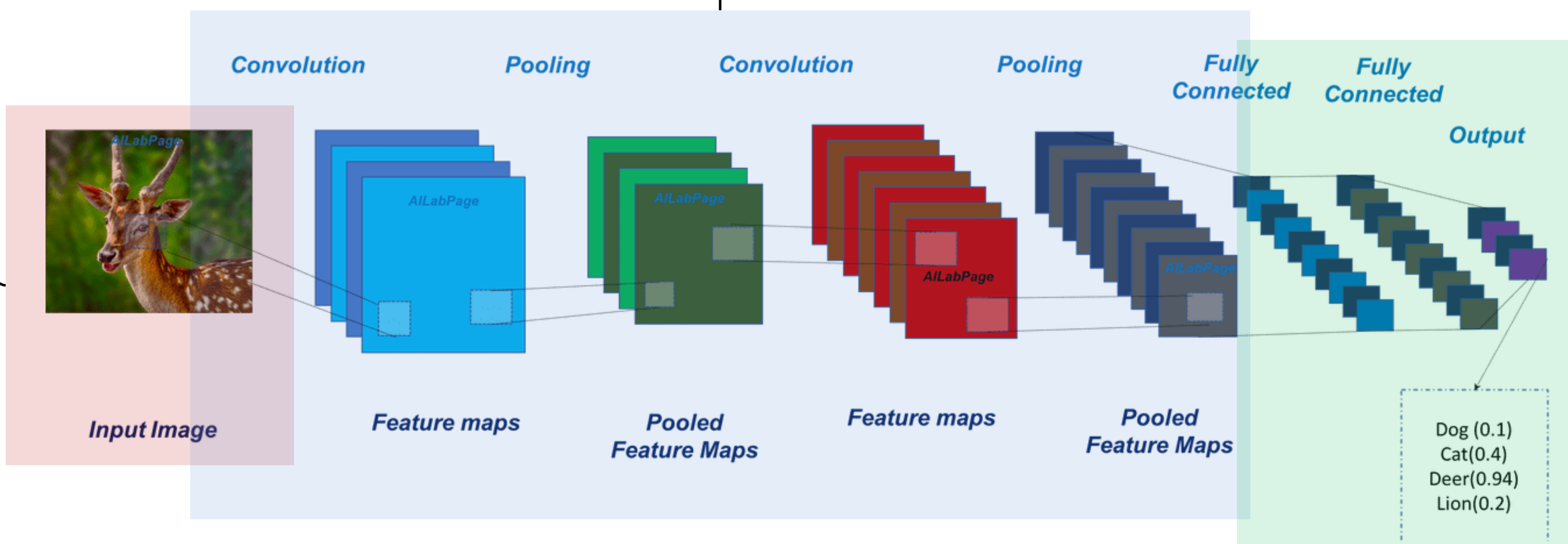
Introduction

Graph Convolutional Networks

이미지 **데이터**

Feature extraction
이미지 **데이터 특징에 적합한**
NN 구조

Task해결
Classification
Object detection
Segmentation



https://vinodsblog.com/2018/10/15/everything-you-need-to-know-about-convolutional-neural-networks/

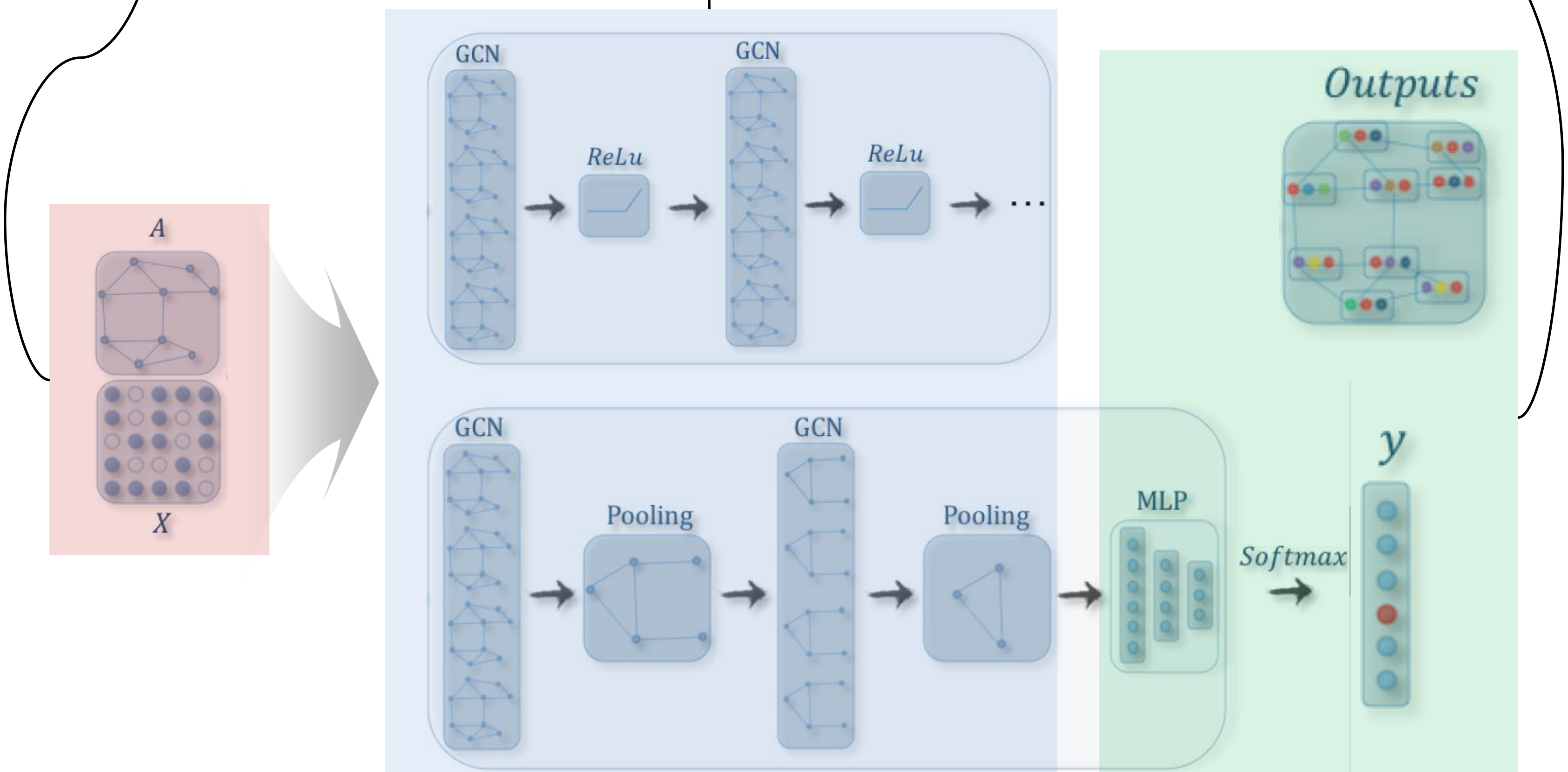
Introduction

Graph Convolutional Networks

그래프 **데이터**

Feature extraction
그래프 **데이터 특징에 적합한 NN 구조**

Task해결
Graph-level
Edge-level
Node-level

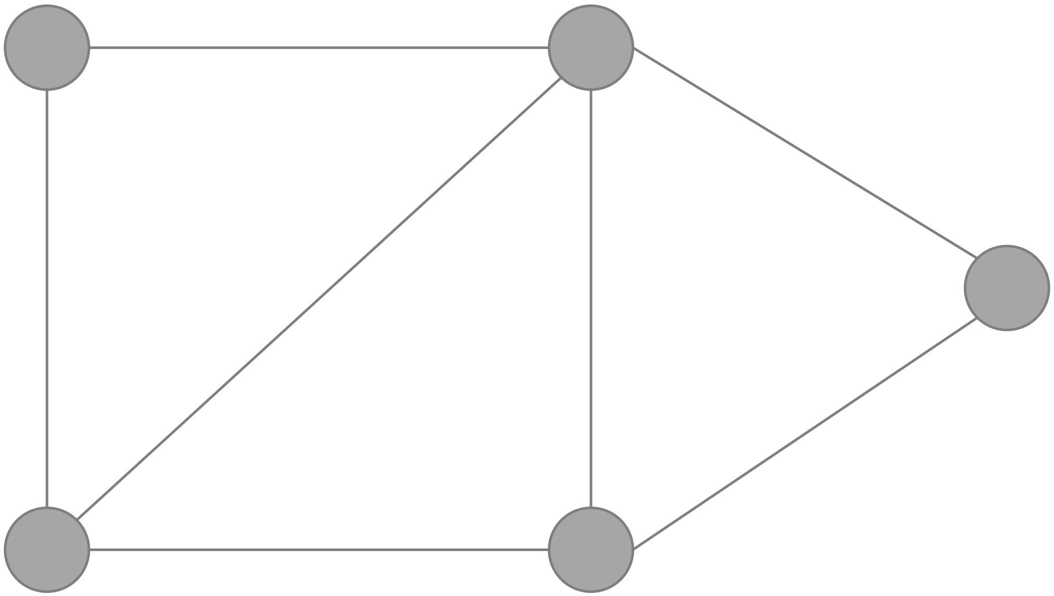


Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2019). A comprehensive survey on graph neural networks. *arXiv preprint arXiv:1901.00596*.

Graph

Graph structure

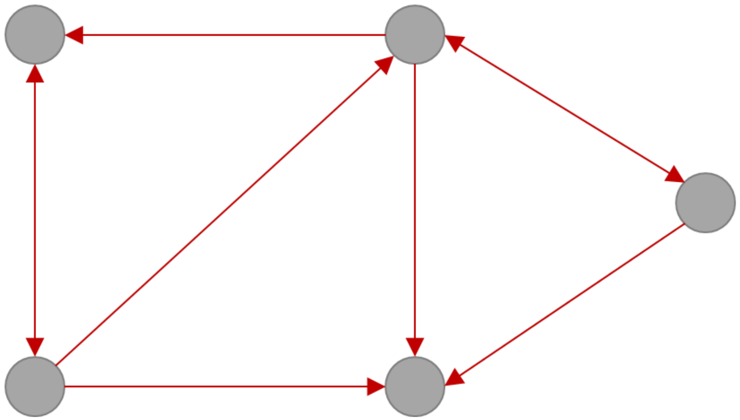
- ❖ 그래프(graph)란 노드(node)와 그 노드를 연결하는 간선(edge)을 하나로 모아놓은 구조
- ❖ Node (Vertex)
- ❖ Edge: directed/undirected, weighted/unweighted

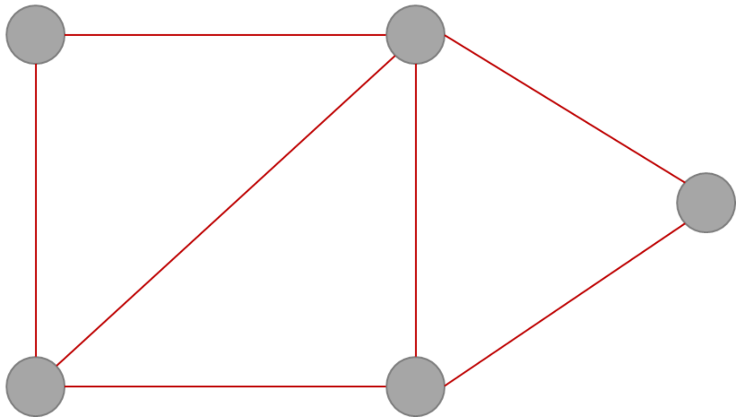


Graph

Graph structure

- ❖ 그래프(graph)란 노드(node)와 그 노드를 연결하는 간선(edge)을 하나로 모아놓은 구조
- ❖ Node (Vertex)
- ❖ Edge: directed/undirected, weighted/unweighted





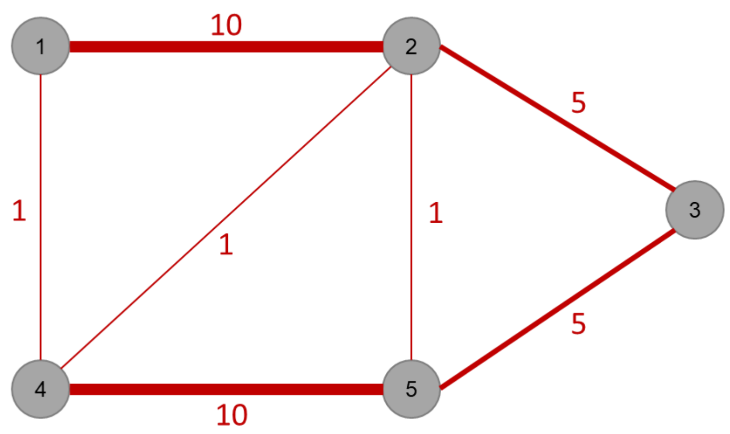
Graph

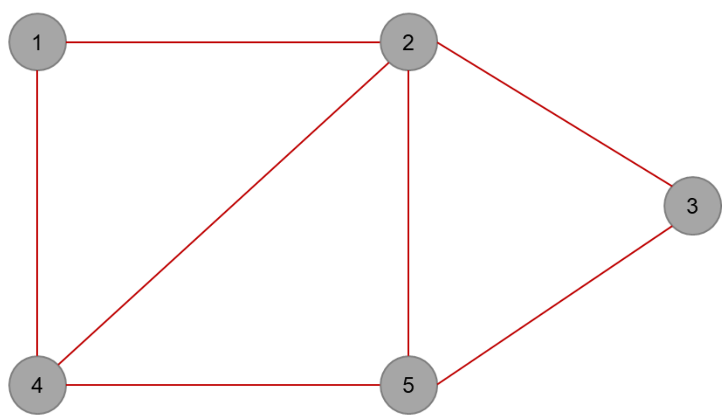
Graph structure

❖ 그래프(graph)란 노드(node)와 그 노드를 연결하는 간선(edge)을 하나로 모아놓은 구조

❖ Node (Vertex)

❖ Edge: directed/undirected, weighted/unweighted

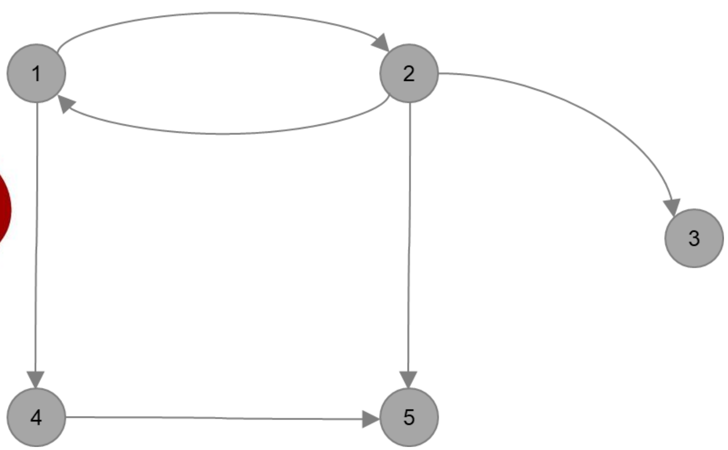




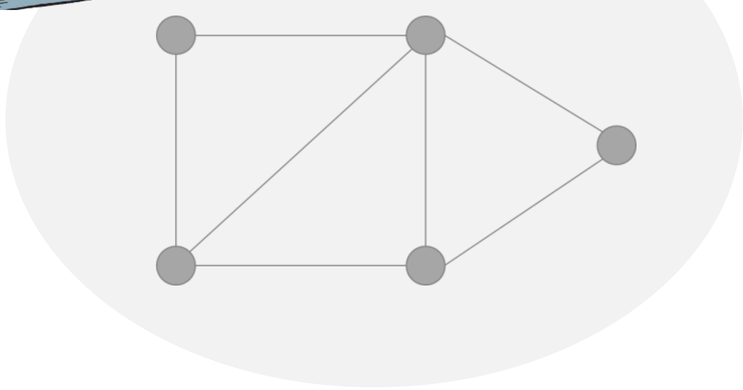
Graph

Graph structure

- ❖ 그래프(graph)란 노드(node)와 그 노드를 연결하는 간선(edge)을 하나로 모아놓은 구조
- ❖ Node (Vertex)
- ❖ Edge: directed/undirected, weighted/unweighted



undirected + unweighted



Graph

Types of Graph

- ❖ Social network graph
- ❖ Molecular graph
- ❖ 3D mesh graph
- ❖ Citation graph

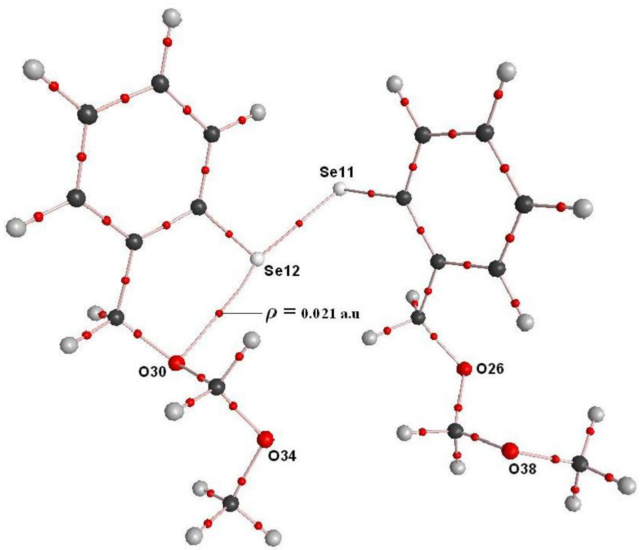
……

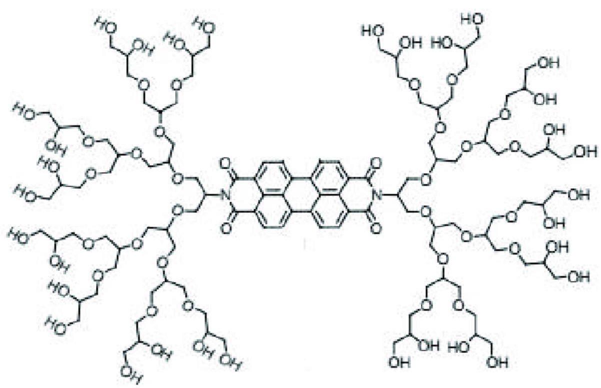
Graph

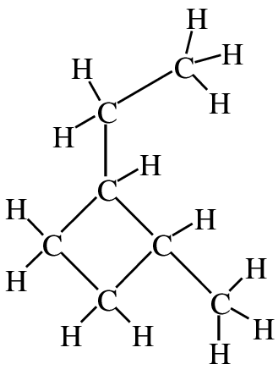
Types of Graph

- ❖ Social network graph
- ❖ Molecular graph
- ❖ 3D mesh graph
- ❖ Citation graph

……





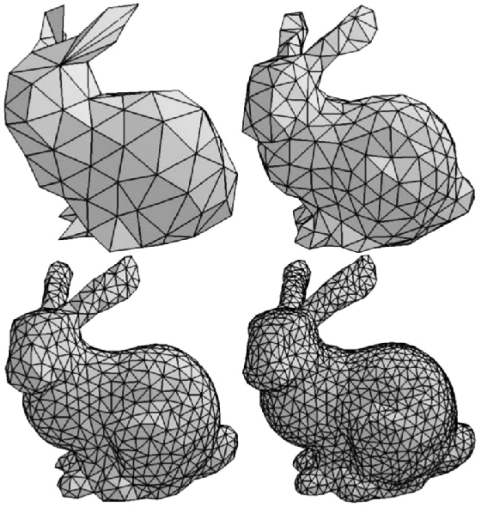


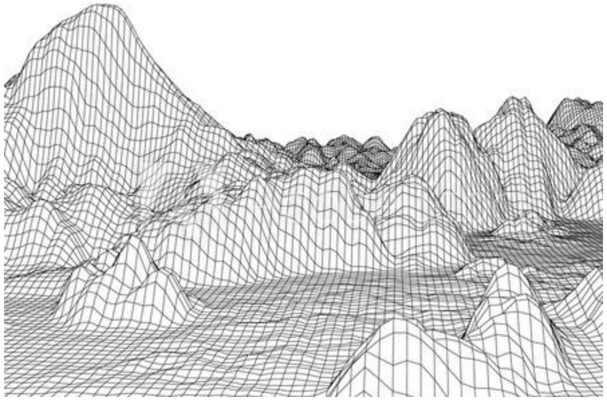
Graph

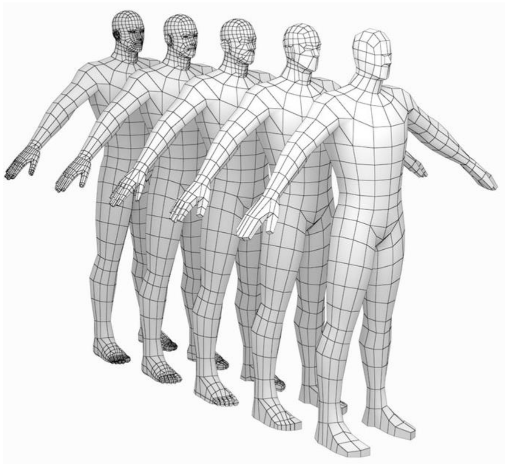
Types of Graph

- ❖ Social network graph
- ❖ Molecular graph
- ❖ 3D mesh graph
- ❖ Citation graph

……







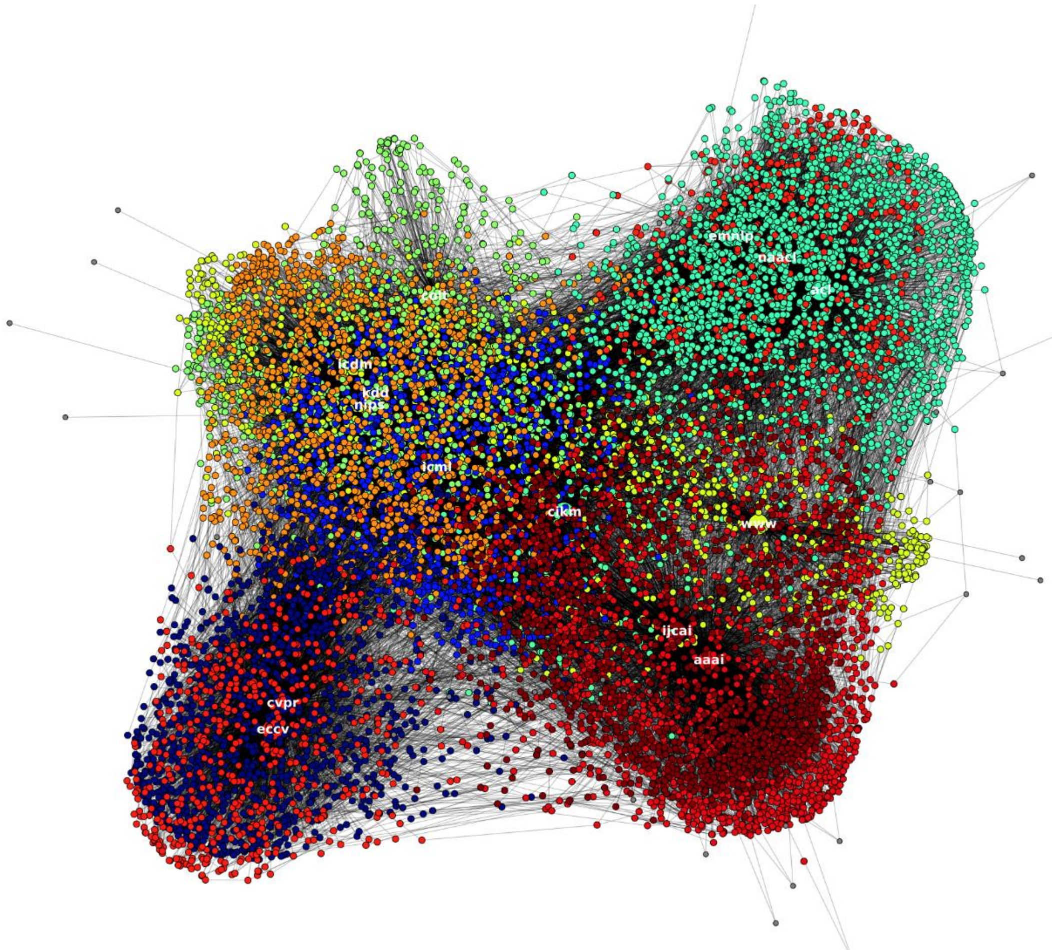
Graph

Types of Graph

- ❖ Social network graph
- ❖ Molecular graph
- ❖ 3D mesh graph
- ❖ Citation graph

……

- **Machine Learning** - ICML, NIPS,
- **Data Mining** - KDD, CIKM, WSDM
- **NLP** - ACL, NAACL, EMNLP
- **Vision** - CVPR, EECV
- **AI** - AAAI, IJCAI



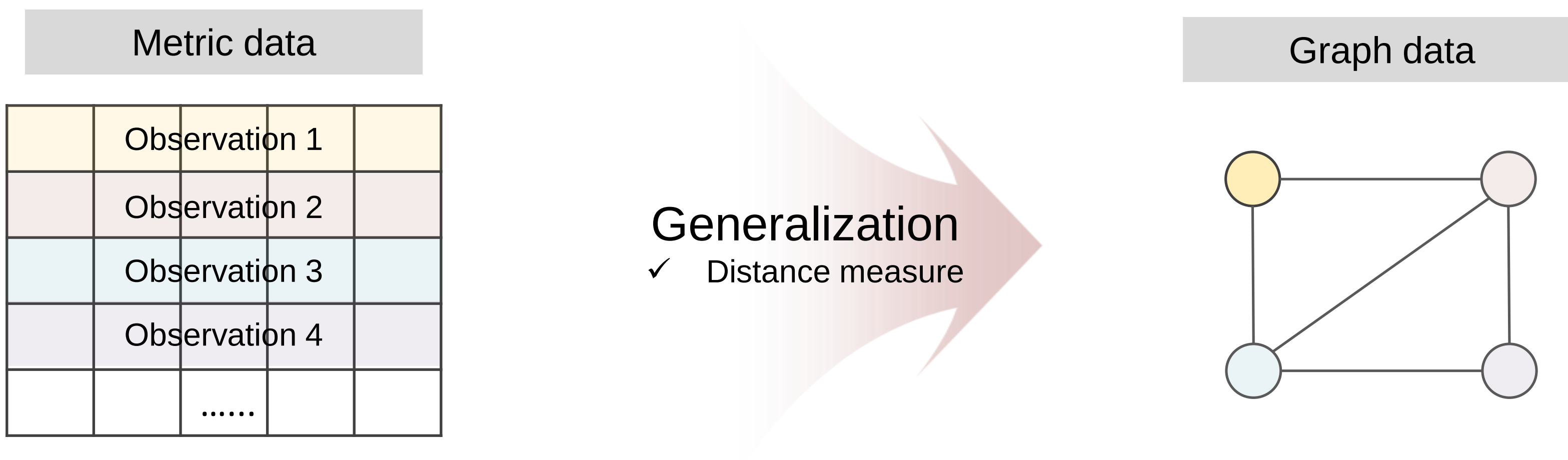
https://gisellezeno.com/academic-work/exploring-citeseerx-dataset.html

Graph

Types of Graph

- ❖ Social network graph
- ❖ Molecular graph
- ❖ 3D mesh graph
- ❖ Citation graph

……



Hammond, D. K., Vandergheynst, P., & Gribonval, R. (2011). Wavelets on graphs via spectral graph theory. Applied and Computational Harmonic Analysis, 30(2), 129-150.

Graph

Types of Graph

- ❖ Social network graph
- ❖ Molecular graph
- ❖ 3D mesh graph
- ❖ Citation graph

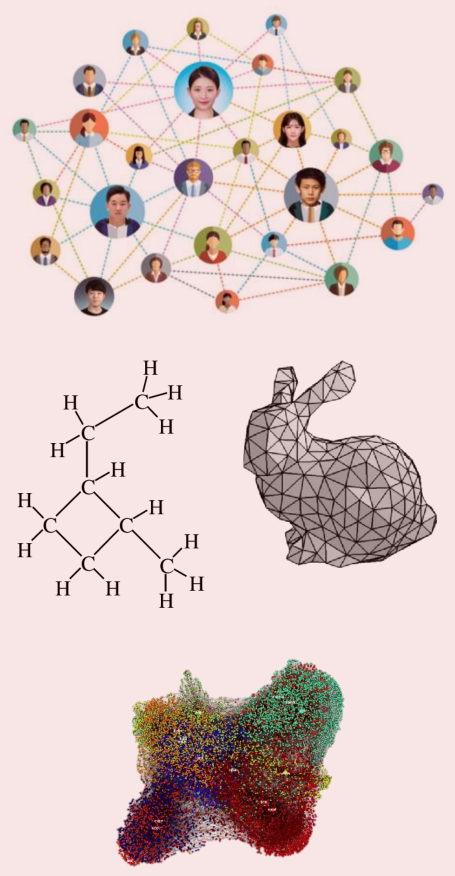
……

Hammond, D. K., Vandergheynst, P., & Gribonval, R. (2011). Wavelets on graphs via spectral graph theory. Applied and Computational Harmonic Analysis, 30(2), 129-150.

Graph

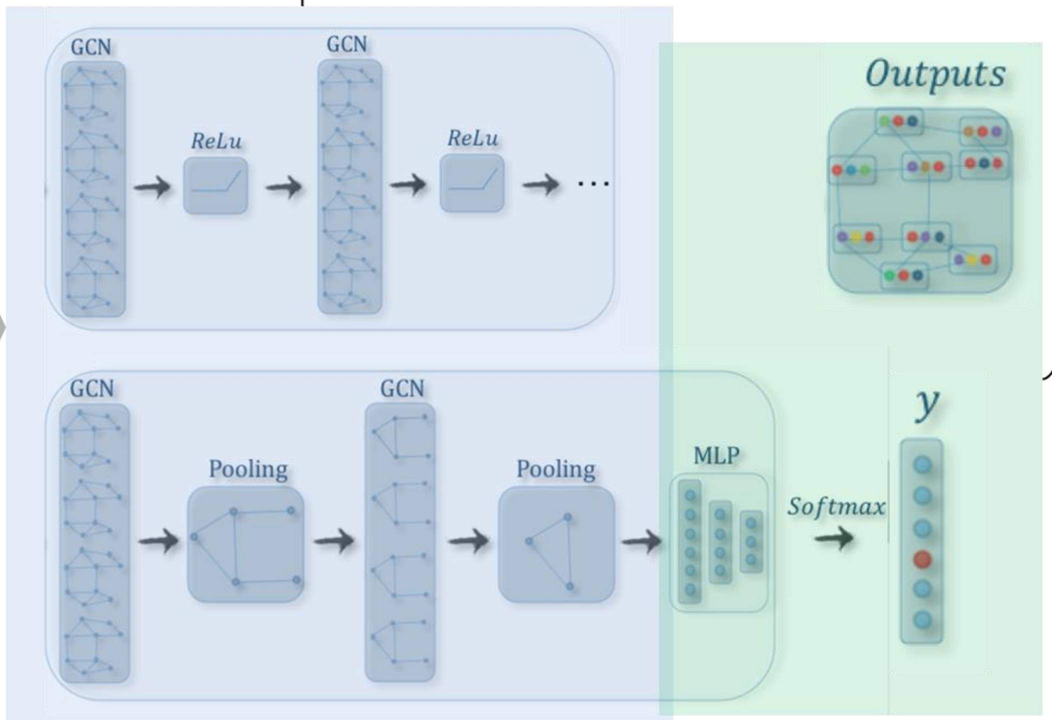
Data Representation

그래프 데이터



Feature extraction
그래프 데이터 특징에 적합한
NN 구조

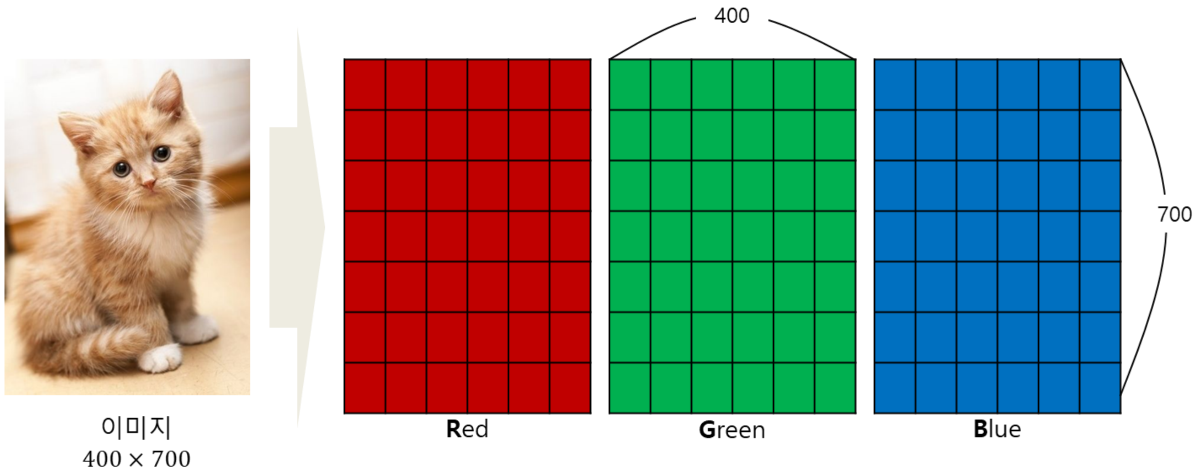
Task해결
Graph-level
Edge-level
Node-level

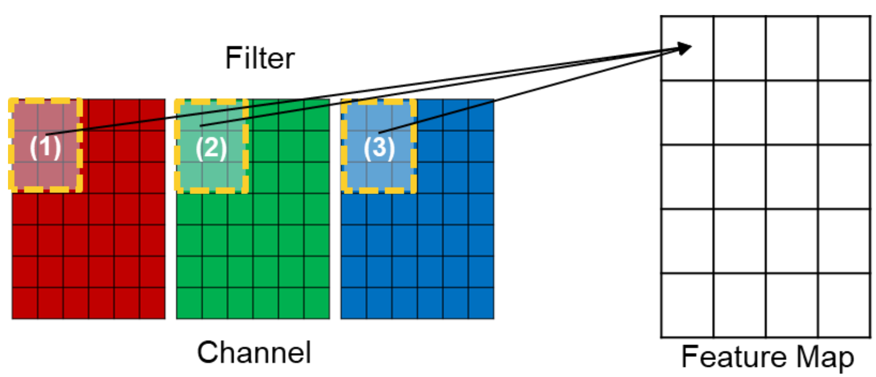


Graph

Data Representation

❖ 이미지 : W × H × C





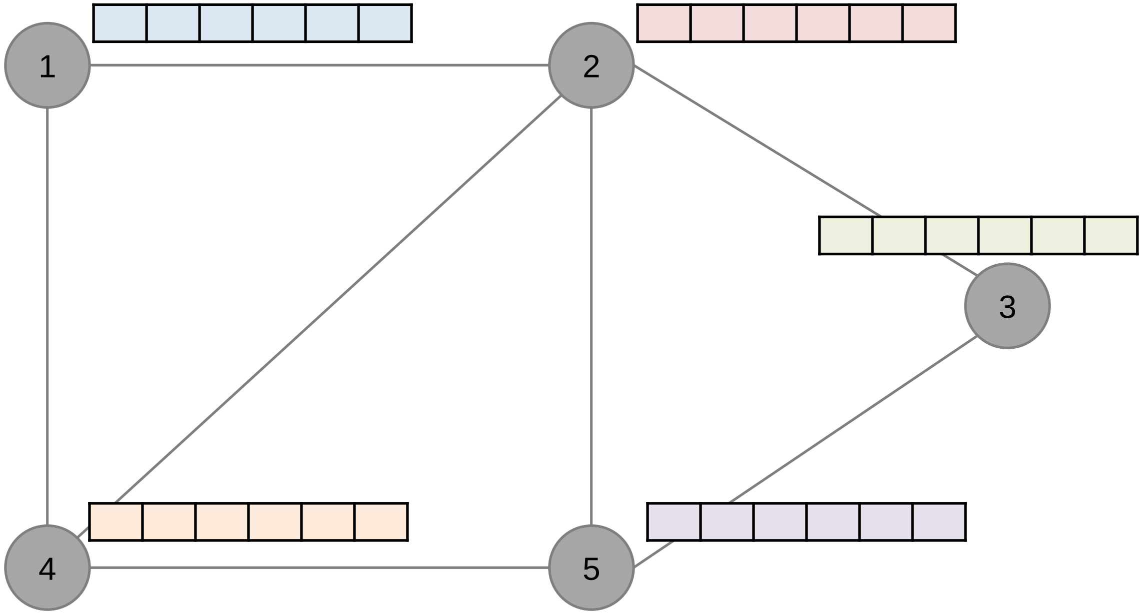
Graph

Data Representation

Graph structure

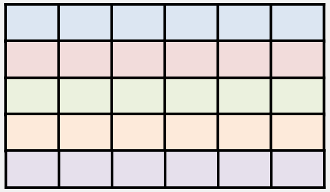
❖ 그래프 : node-feature matrix, adjacency matrix, degree matrix, Laplacian matrix

- $g = \{v, \varepsilon\}$
- Vertex set : $v(g) = \{v_1, \ldots, v_n\}, n = |v|$
- Edge set $\varepsilon(g) = \{e_{ij}\}, m = |\varepsilon|$



$Node - feature\ matrix$

$X \in R^{n \times F}$



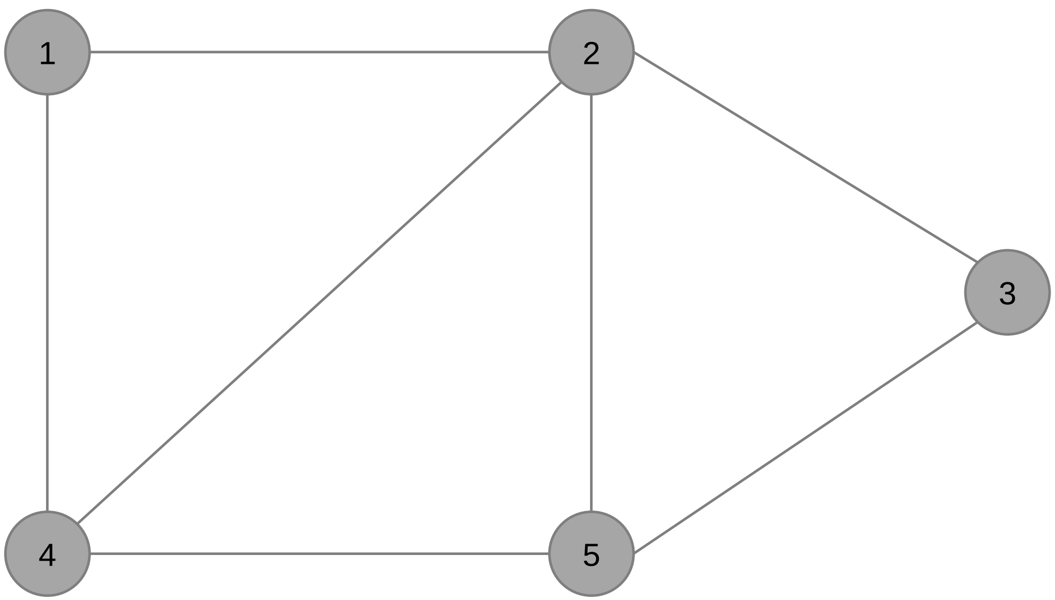
Graph

Data Representation

Graph structure

❖ 그래프 : node-feature matrix, adjacency matrix, degree matrix, Laplacian matrix

- $g = \{v, \varepsilon\}$
- Vertex set : $v(g) = \{v_1, \ldots, v_n\}, n = |v|$
- Edge set $\varepsilon(g) = \{e_{ij}\}, m = |\varepsilon|$



$$Adjacency\ matrix$$
$$A \in R^{n \times n}$$

$$A = \begin{cases} A_{ij} = 1 & if\ there\ is\ an\ edge\ e_{ij} \\ A_{ij} = 0 & if\ there\ is\ no\ edge \\ A_{ii} = 0 & \end{cases}$$

0	1	0	1	0
1	0	1	1	1
0	1	0	0	1
1	1	0	0	1
0	1	1	1	0

Graph

Data Representation

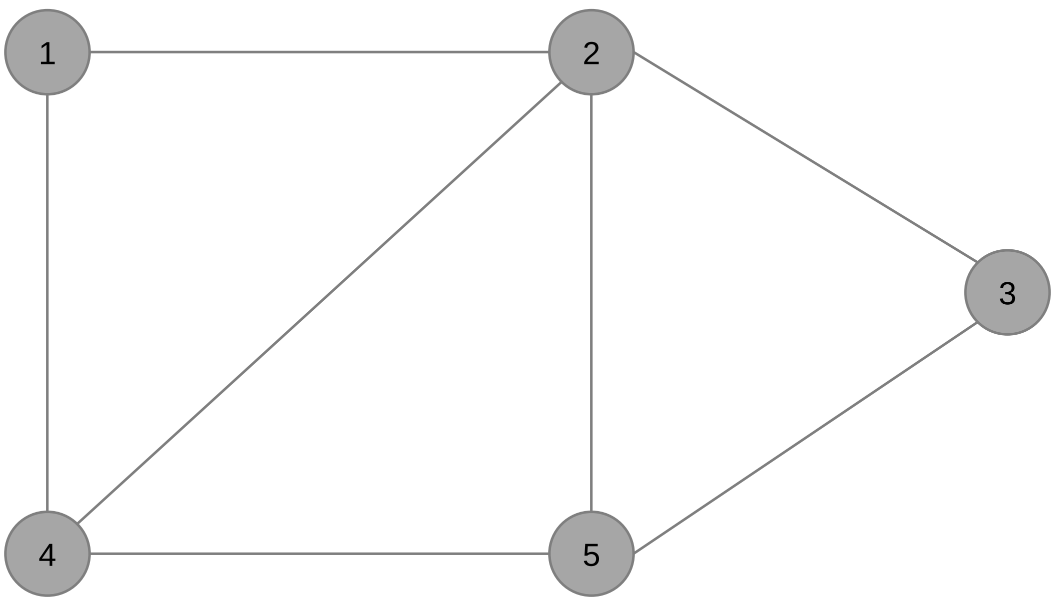
Graph structure

❖ 그래프 : node-feature matrix, adjacency matrix, degree matrix, Laplacian matrix

- $g = \{v, \varepsilon\}$
- Vertex set : $v(g) = \{v_1, \ldots, v_n\}, n = |v|$
- Edge set $\varepsilon(g) = \{e_{ij}\}, m = |\varepsilon|$

Adjacency matrix

0	1	0	1	0
1	0	1	1	1
0	1	0	0	1
1	1	0	0	1
0	1	1	1	0



Degree matrix

$$D \in R^{n \times n}$$

$$D_{ii} = \sum_{i \sim j} A_{ij}$$

2	0	0	0	0
0	4	0	0	0
0	0	2	0	0
0	0	0	3	0
0	0	0	0	3

Graph

Data Representation

Graph structure

❖ 그래프 : node-feature matrix, adjacency matrix, degree matrix, Laplacian matrix

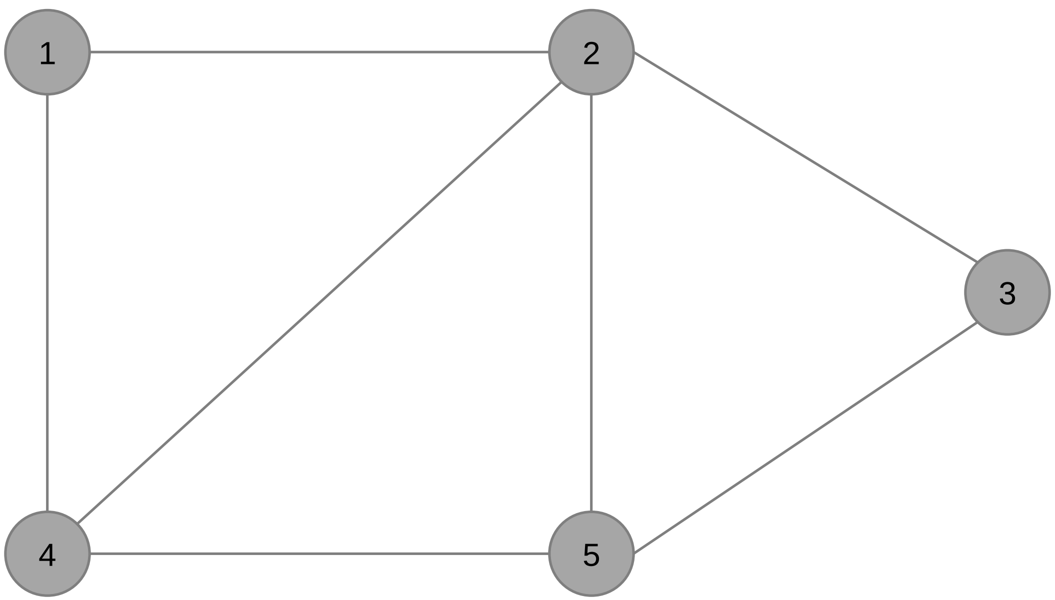
- $g = \{v, \varepsilon\}$
- Vertex set : $v(g) = \{v_1, \dots, v_n\}, n = |v|$
- Edge set $\varepsilon(g) = \{e_{ij}\}, m = |\varepsilon|$

Degree matrix

2	0	0	0	0
0	4	0	0	0
0	0	2	0	0
0	0	0	3	0
0	0	0	0	3

Adjacency matrix

0	1	0	1	0
1	0	1	1	1
0	1	0	0	1
1	1	0	0	1
0	1	1	1	0



Laplacian matrix

$$L \in R^{n \times n}$$

$$L = D - A$$

2	-1	0	-1	0
-1	4	-1	-1	-1
0	-1	2	0	-1
-1	-1	0	3	-1
0	-1	-1	-1	3

Graph

Tasks

그래프 데이터

Node − feature matrix

Graph structure

Degree matrix

2	0	0	0	0
0	4	0	0	0
0	0	2	0	0
0	0	0	3	0
0	0	0	0	3

Adjacency matrix

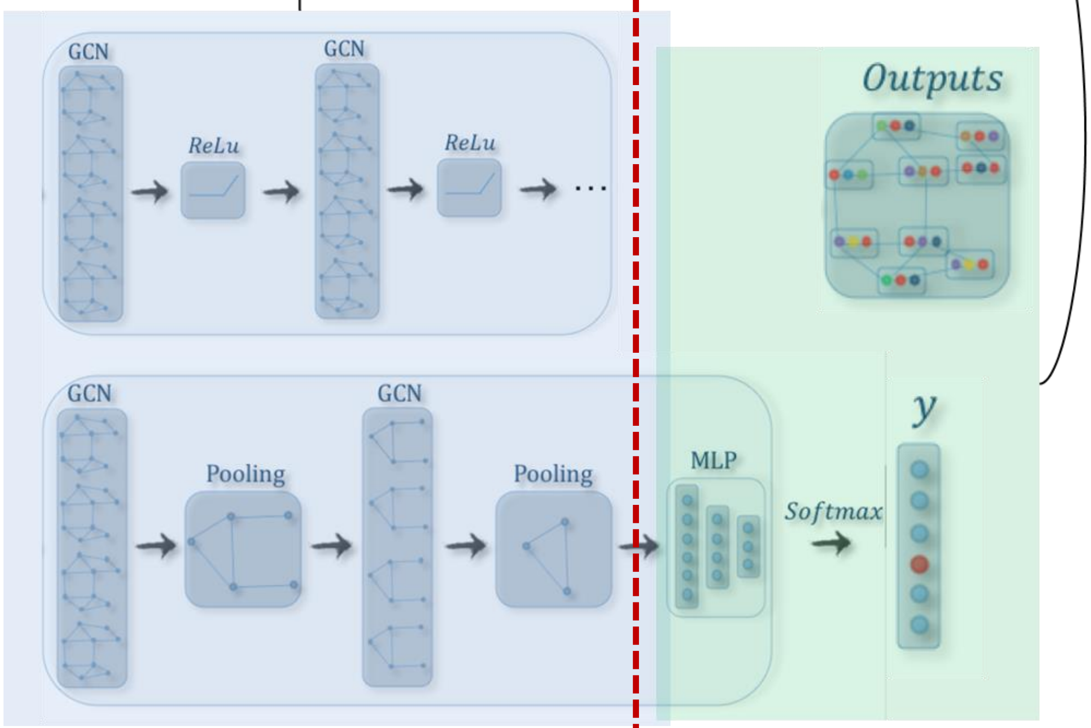
0	1	0	1	0
1	0	1	1	1
0	1	0	0	1
1	1	0	0	1
0	1	1	1	0

Laplacian matrix

2	-1	0	-1	0
-1	4	-1	-1	-1
0	-1	2	0	-1
-1	-1	0	3	-1
0	-1	-1	-1	3

Feature extraction
그래프 데이터 특징에 적합한
NN 구조

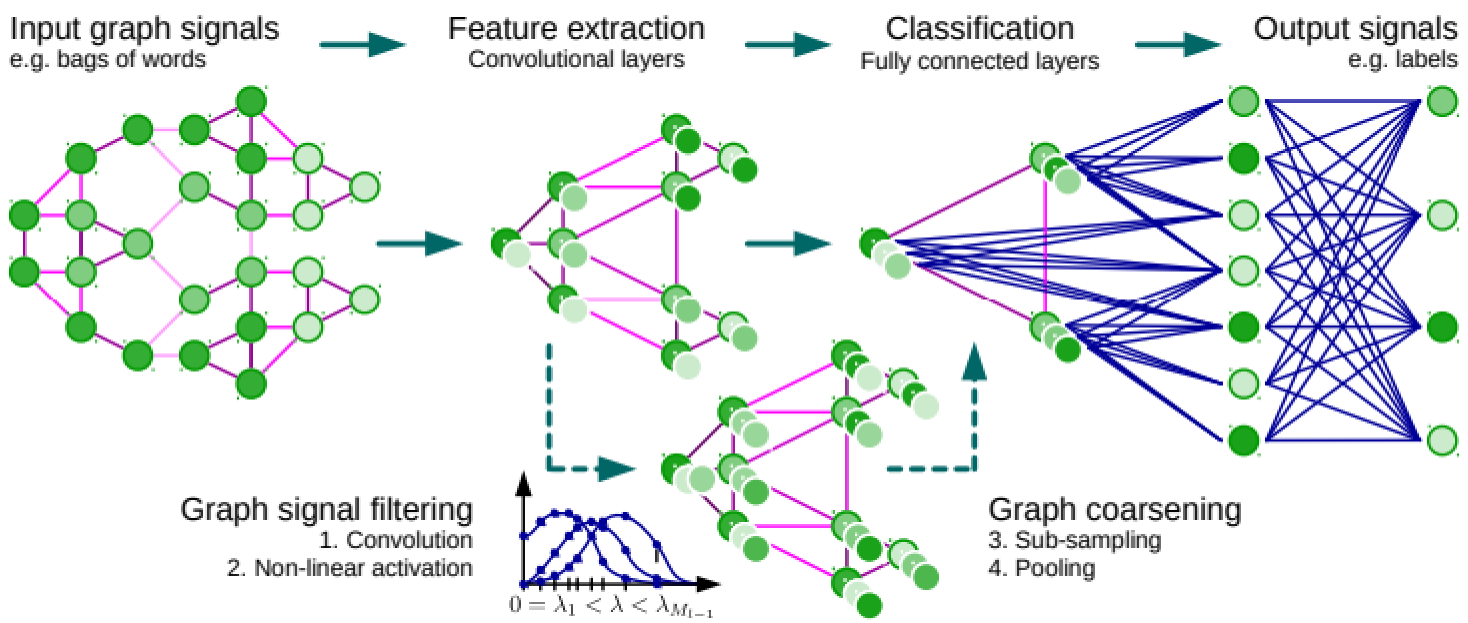
Task해결
Graph-level
Edge-level
Node-level



Graph

Tasks

- ❖ **Graph-level : graph classification**
- ❖ Edge-level : edge classification and link prediction
- ❖ Node-level : node regression and classification



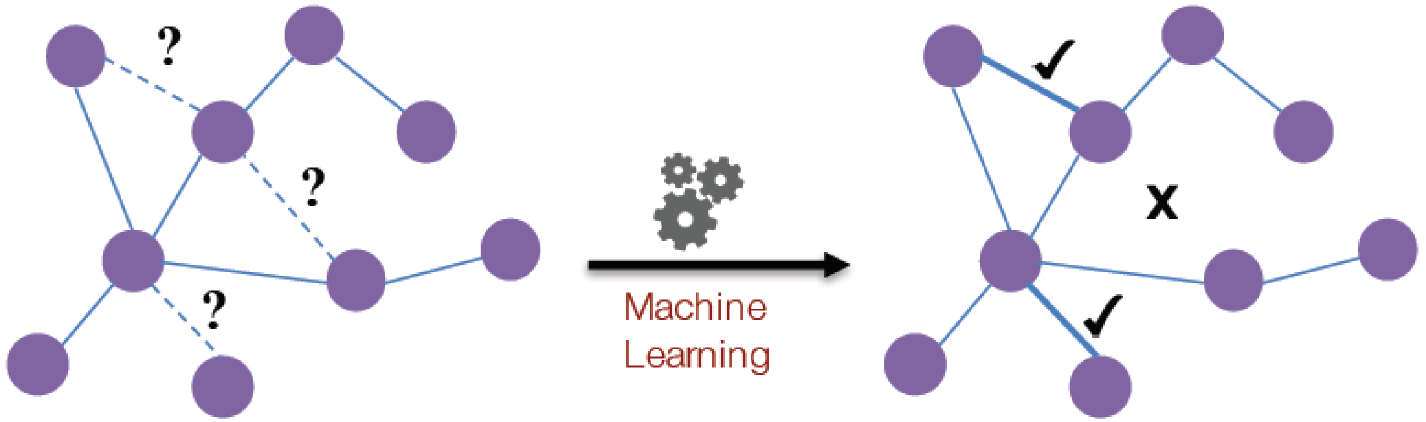
Defferrard, M., Bresson, X., & Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems* (pp. 3844-3852).

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2019). A comprehensive survey on graph neural networks. arXiv preprint arXiv:1901.00596.

Graph

Tasks

- ❖ Graph-level : graph classification
- ❖ **Edge-level : edge classification and link prediction**
- ❖ Node-level : node regression and classification



http://snap.stanford.edu/proj/embeddings-www/

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2019). A comprehensive survey on graph neural networks. arXiv preprint arXiv:1901.00596.

Graph

Tasks

- ❖ Graph-level : graph classification
- ❖ **Edge-level : edge classification and link prediction**
- ❖ Node-level : node regression and classification

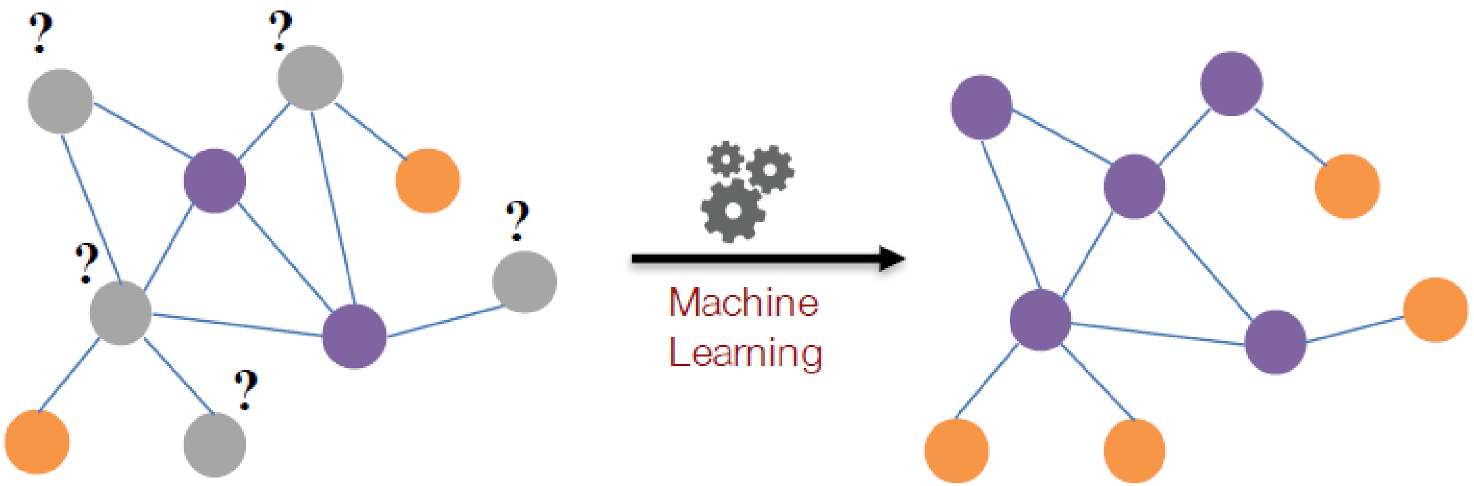
http://snap.stanford.edu/proj/embeddings-www/

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2019). A comprehensive survey on graph neural networks. arXiv preprint arXiv:1901.00596.

Graph

Tasks

- ❖ Graph-level : graph classification
- ❖ Edge-level : edge classification and link prediction
- ❖ **Node-level** : node regression and **classification**
  - **Graph-based semi-supervised learning**



http://snap.stanford.edu/proj/embeddings-www/

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2019). A comprehensive survey on graph neural networks. arXiv preprint arXiv:1901.00596.

Graph

Tasks

- ❖ Graph-level : graph classification
- ❖ Edge-level : edge classification and link prediction
- ❖ **Node-level** : node regression and **classification**
    - **Graph-based semi-supervised learning**

Classifying the function of proteins in the interactome!

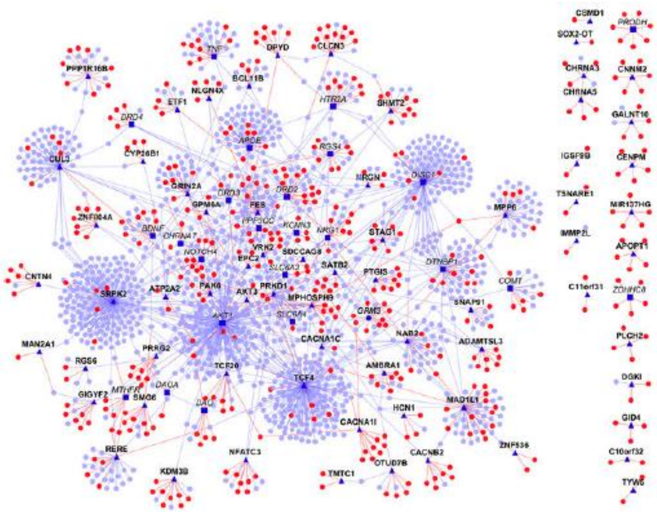


Image from: Ganapathiraju et al. 2016. [Schizophrenia interactome with 504 novel protein–protein interactions](). *Nature*.

http://snap.stanford.edu/proj/embeddings-www/

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2019). A comprehensive survey on graph neural networks. arXiv preprint arXiv:1901.00596.

Graph

Tasks

- Graph-level : graph classification
- Edge-level : edge classification and link prediction
- **Node-level** : node regression and **classification**
  - **Graph-based semi-supervised learning**

Transductive	Inductive	Transductive + Inductive

Published as a conference paper at ICLR 2017

SEMI-SUPERVISED CLASSIFICATION WITH GRAPH CONVOLUTIONAL NETWORKS

Thomas N. Kipf
University of Amsterdam
T.N.Kipf@uva.nl

Max Welling
University of Amsterdam
Canadian Institute for Advanced Research (CIFAR)
M.Welling@uva.nl

ABSTRACT

We present a scalable approach for semi-supervised learning on graph-structured data that is based on an efficient variant of convolutional neural networks which operate directly on graphs. We motivate the choice of our convolutional architecture via a localized first-order approximation of spectral graph convolutions. Our model scales linearly in the number of graph edges and learns hidden layer representations that encode both local graph structure and features of nodes. In a number of experiments on citation networks and on a knowledge graph dataset we demonstrate that our approach outperforms related methods by a significant margin.

Inductive Representation Learning on Large Graphs

William L. Hamilton*
wleif@stanford.edu

Rex Ying*
rexying@stanford.edu

Jure Leskovec
jure@cs.stanford.edu

Department of Computer Science
Stanford University
Stanford, CA, 94305

Abstract

Low-dimensional embeddings of nodes in large graphs have proved extremely useful in a variety of prediction tasks, from content recommendation to identifying protein functions. However, most existing approaches require that all nodes in the graph are present during training of the embeddings; these previous approaches are inherently *transductive* and do not naturally generalize to unseen nodes. Here we present GraphSAGE, a general *inductive* framework that leverages node feature information (e.g., text attributes) to efficiently generate node embeddings for previously unseen data. Instead of training individual embeddings for each node, we learn a function that generates embeddings by sampling and aggregating features from a node's local neighborhood. Our algorithm outperforms strong baselines on three inductive node-classification benchmarks: we classify the category of unseen nodes in evolving information graphs based on citation and Reddit post data, and we show that our algorithm generalizes to completely unseen graphs using a multi-graph dataset of protein-protein interactions.

Published as a conference paper at ICLR 2018

GRAPH ATTENTION NETWORKS

Petar Veličković*
Department of Computer Science and Technology
University of Cambridge
petar.velickovic@cst.cam.ac.uk

Guillem Cucurull*
Centre de Visió per Computador, UAB
gcucurull@gmail.com

Arantxa Casanova*
Centre de Visió per Computador, UAB
ar.casanova.8@gmail.com

Adriana Romero
Montréal Institute for Learning Algorithms
adriana.romero.soriano@umontreal.ca

Pietro Liò
Department of Computer Science and Technology
University of Cambridge
pietro.lio@cst.cam.ac.uk

Yoshua Bengio
Montréal Institute for Learning Algorithms
yoshua.umontreal@gmail.com

ABSTRACT

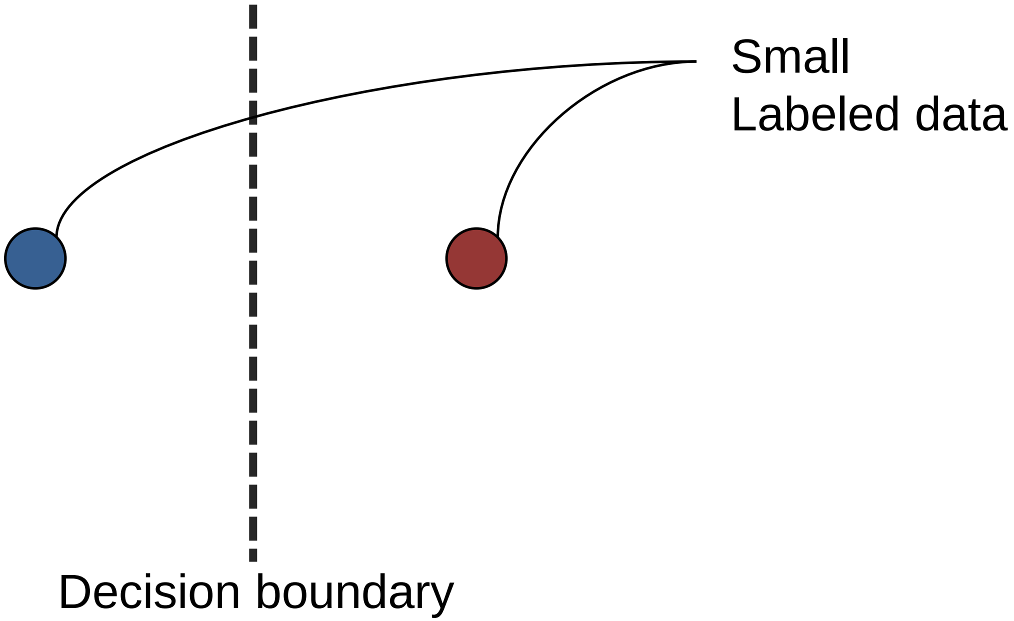
We present graph attention networks (GATs), novel neural network architectures that operate on graph-structured data, leveraging masked self-attentional layers to address the shortcomings of prior methods based on graph convolutions or their approximations. By stacking layers in which nodes are able to attend over their neighborhoods' features, we enable (implicitly) specifying different weights to different nodes in a neighborhood, without requiring any kind of costly matrix operation (such as inversion) or depending on knowing the graph structure upfront. In this way, we address several key challenges of spectral-based graph neural networks simultaneously, and make our model readily applicable to inductive as well as transductive problems. Our GAT models have achieved or matched state-of-the-art results across four established transductive and inductive graph benchmarks: the *Cora*, *Citeseer* and *Pubmed* citation network datasets, as well as a *protein-protein interaction* dataset (wherein test graphs remain unseen during training).

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2019). A comprehensive survey on graph neural networks. arXiv preprint arXiv:1901.00596.

Graph

Tasks

- ❖ Graph-level : graph classification
- ❖ Edge-level : edge classification and link prediction
- ❖ **Node-level** : node regression and **classification**
  - **Graph-based semi-supervised learning**



https://en.wikipedia.org/wiki/Semi-supervised_learning

Graph

Tasks

- ❖ Graph-level : graph classification
- ❖ Edge-level : edge classification and link prediction
- ❖ **Node-level** : node regression and **classification**
    - **Graph-based semi-supervised learning**

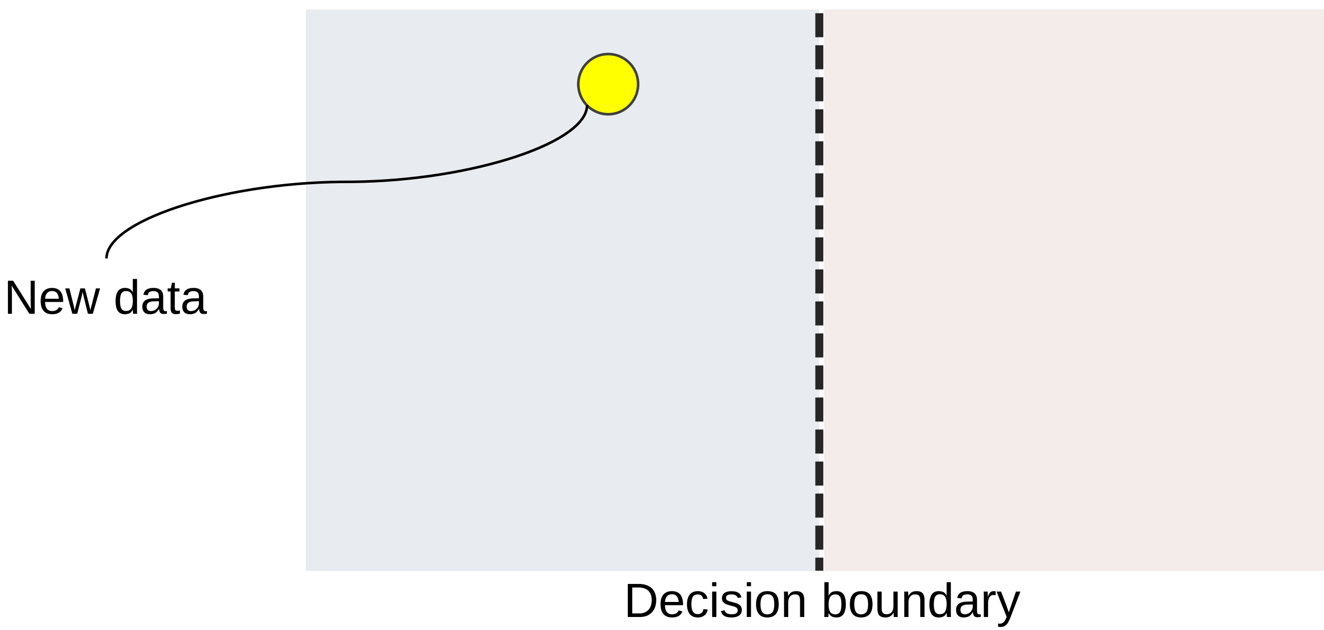
Decision boundary

https://en.wikipedia.org/wiki/Semi-supervised_learning

Graph

Tasks

- ❖ Graph-level : graph classification
- ❖ Edge-level : edge classification and link prediction
- ❖ **Node-level** : node regression and **classification**
    - **Graph-based semi-supervised learning**

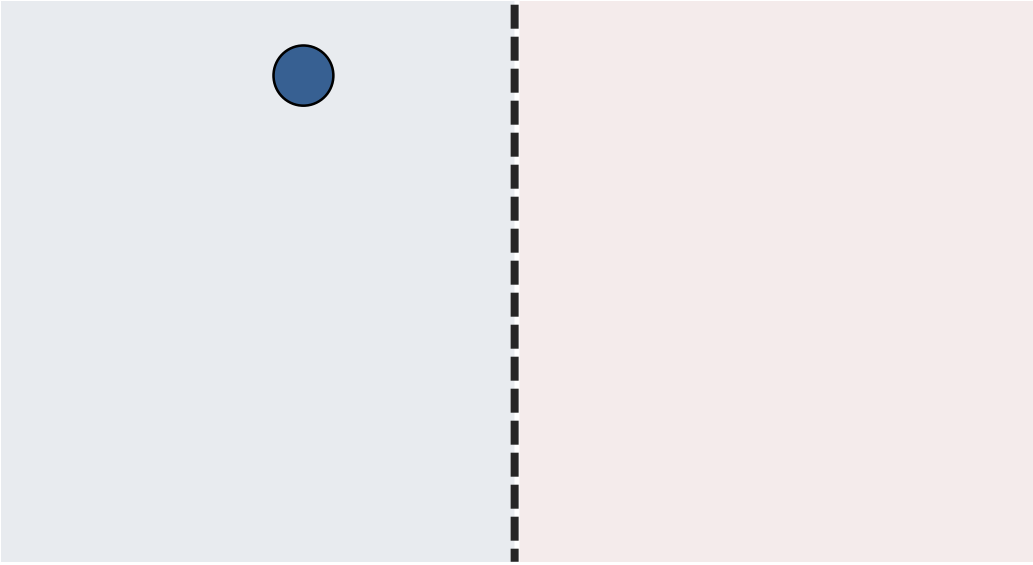


https://en.wikipedia.org/wiki/Semi-supervised_learning

Graph

Tasks

- ❖ Graph-level : graph classification
- ❖ Edge-level : edge classification and link prediction
- ❖ **Node-level** : node regression and **classification**
  - **Graph-based semi-supervised learning**



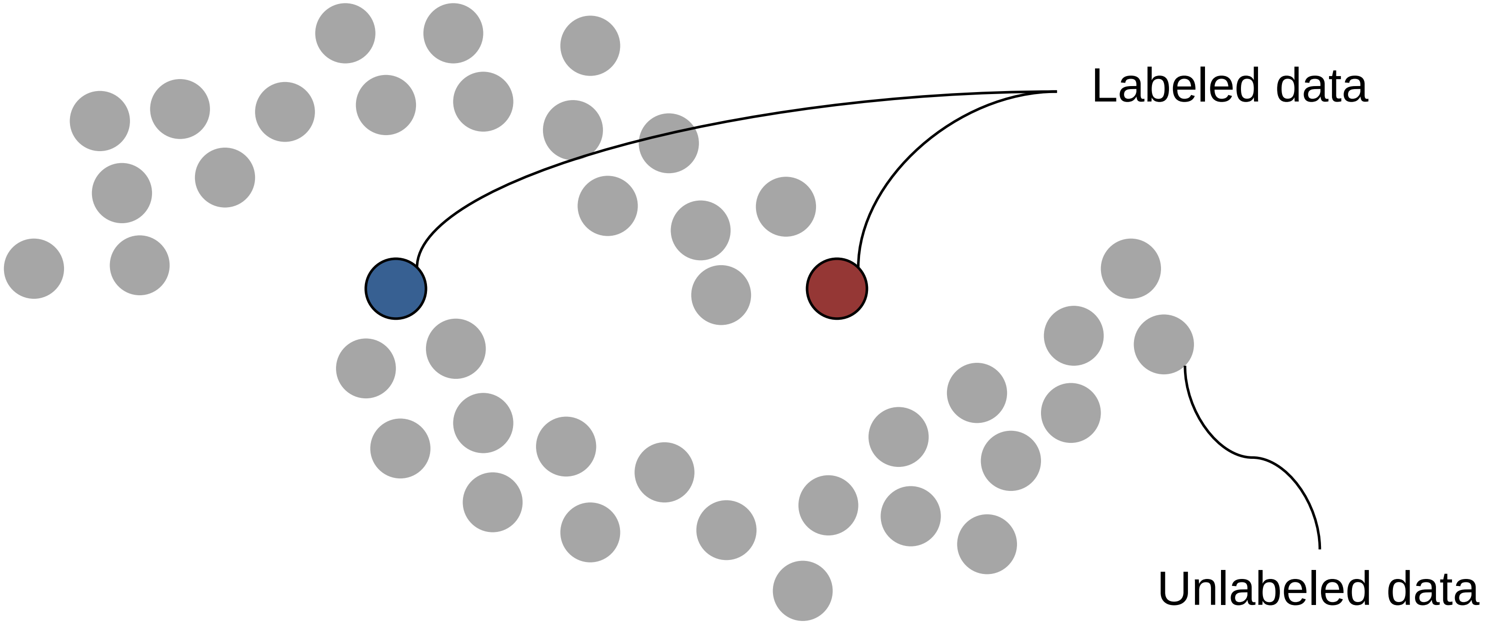
Decision boundary

https://en.wikipedia.org/wiki/Semi-supervised_learning

Graph

Tasks

- ❖ Graph-level : graph classification
- ❖ Edge-level : edge classification and link prediction
- ❖ **Node-level** : node regression and **classification**
  - **Graph-based semi-supervised learning**

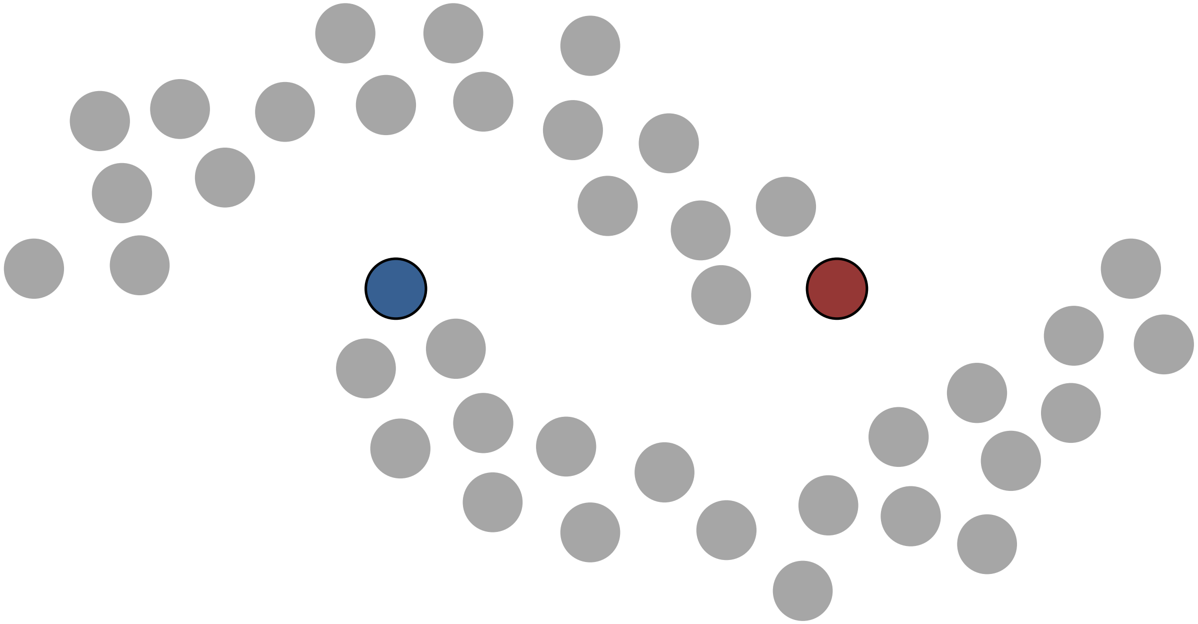


https://en.wikipedia.org/wiki/Semi-supervised_learning

Graph

Tasks

- ❖ Graph-level : graph classification
- ❖ Edge-level : edge classification and link prediction
- ❖ **Node-level** : node regression and **classification**
    - **Graph-based semi-supervised learning**

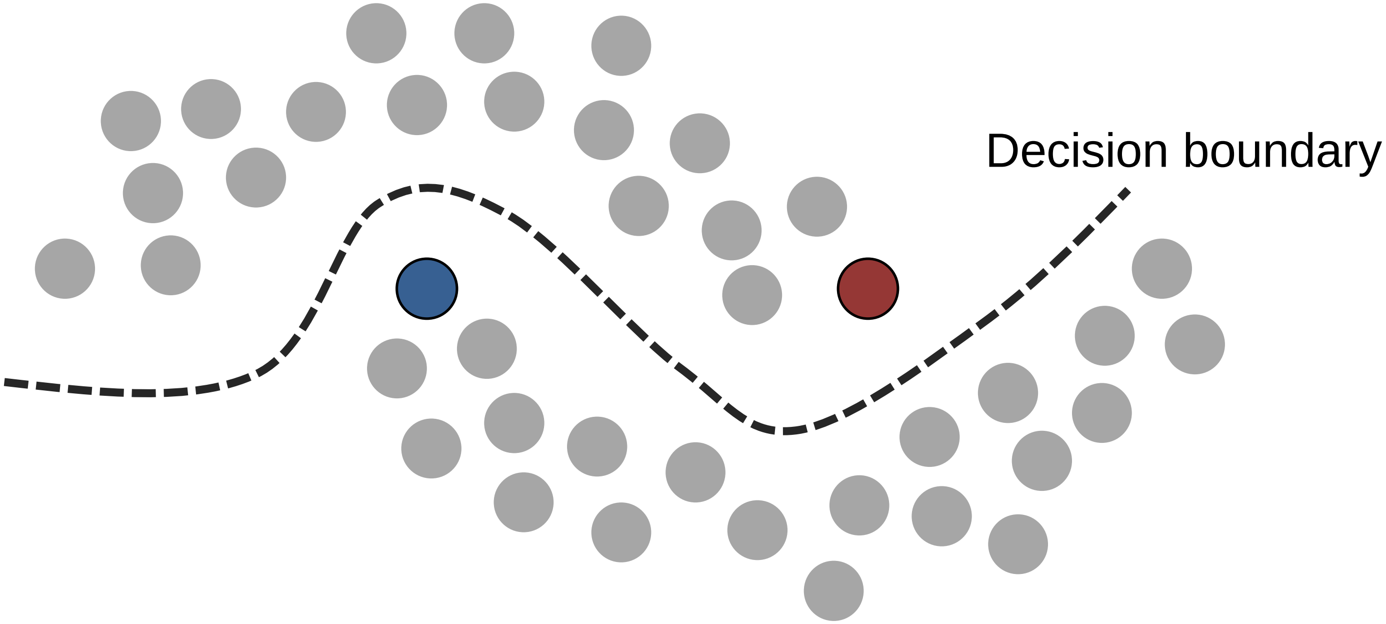


https://en.wikipedia.org/wiki/Semi-supervised_learning

Graph

Tasks

- ❖ Graph-level : graph classification
- ❖ Edge-level : edge classification and link prediction
- ❖ **Node-level** : node regression and **classification**
  - **Graph-based semi-supervised learning**

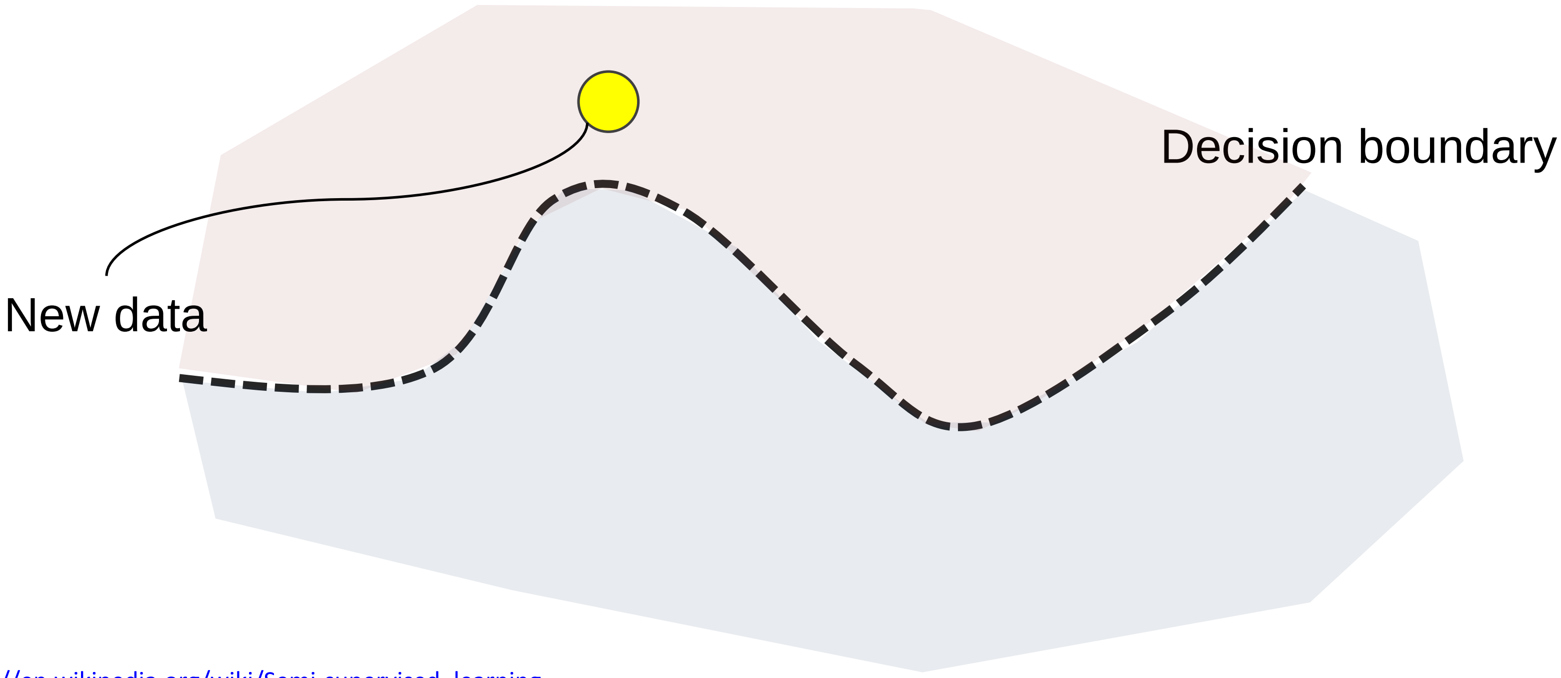


https://en.wikipedia.org/wiki/Semi-supervised_learning

Graph

Tasks

- ❖ Graph-level : graph classification
- ❖ Edge-level : edge classification and link prediction
- ❖ **Node-level** : node regression and **classification**
  - **Graph-based semi-supervised learning**



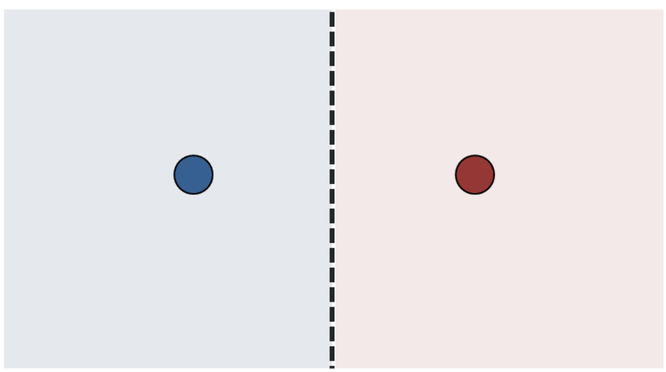
https://en.wikipedia.org/wiki/Semi-supervised_learning

Graph

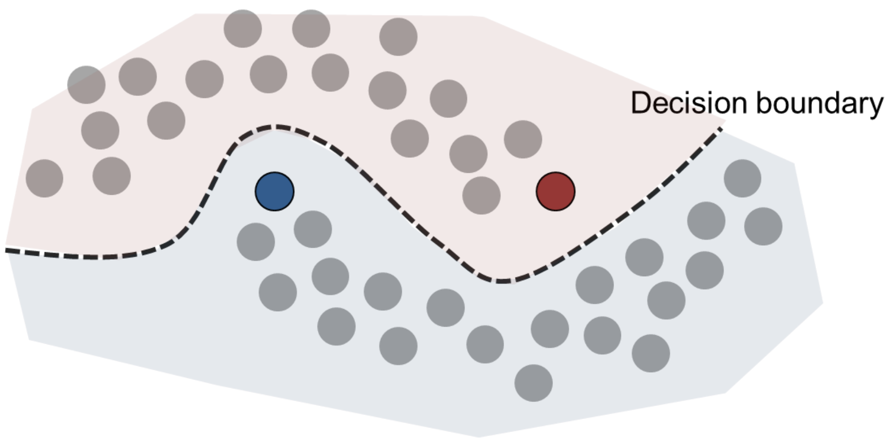
Tasks

- ❖ Graph-level : graph classification
- ❖ Edge-level : edge classification and link prediction
- ❖ **Node-level** : node regression and **classification**
  - **Graph-based semi-supervised learning**

Supervised



Semi-supervised



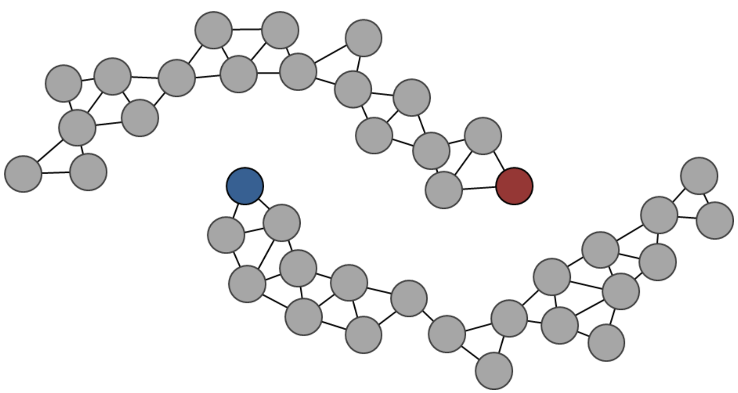
https://en.wikipedia.org/wiki/Semi-supervised_learning

Graph

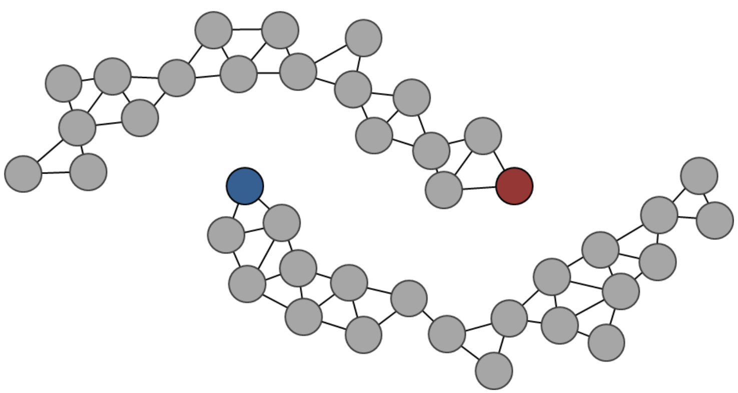
Tasks

- ❖ Graph-level : graph classification
- ❖ Edge-level : edge classification and link prediction
- ❖ **Node-level** : node regression and **classification**
  - **Graph-based semi-supervised learning**

Transductive



Inductive



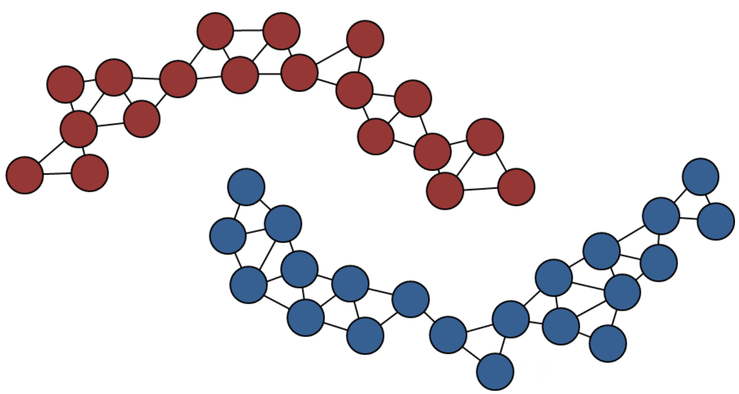
https://en.wikipedia.org/wiki/Semi-supervised_learning

Graph

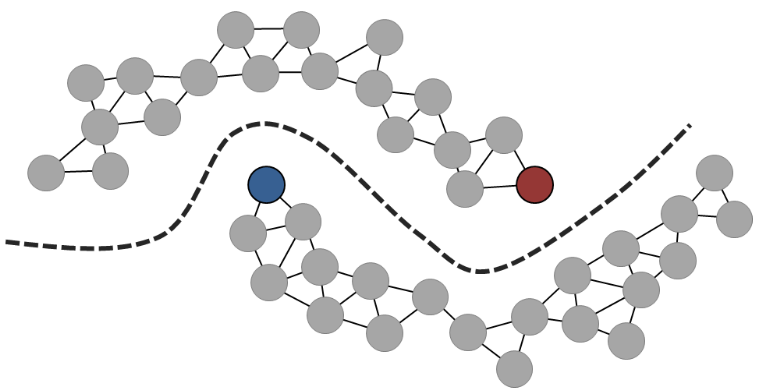
Tasks

- ❖ Graph-level : graph classification
- ❖ Edge-level : edge classification and link prediction
- ❖ **Node-level** : node regression and **classification**
  - **Graph-based semi-supervised learning**

Transductive



Inductive



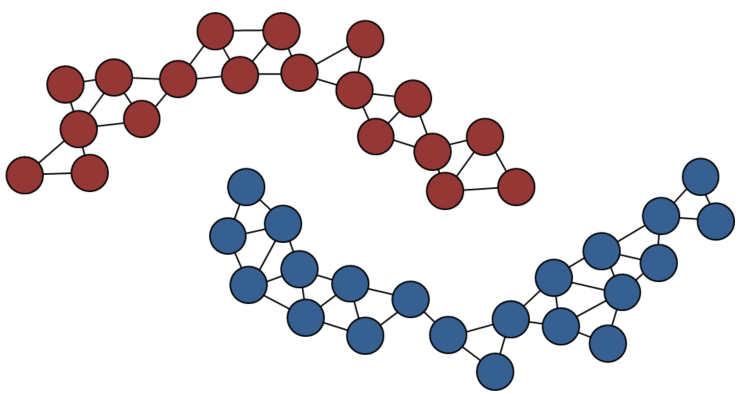
https://en.wikipedia.org/wiki/Semi-supervised_learning

Graph

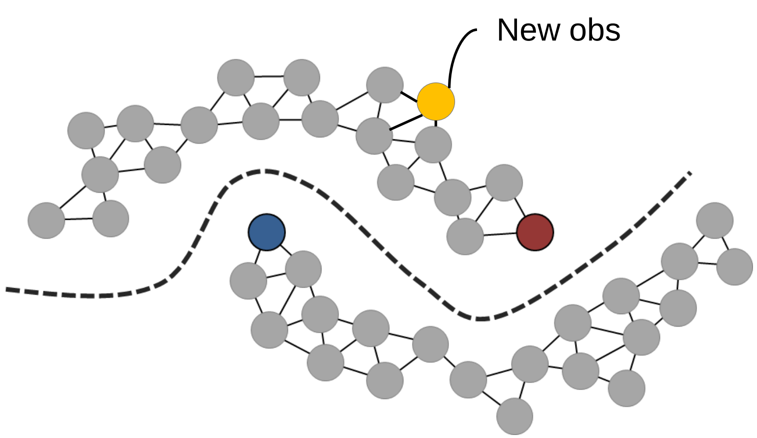
Tasks

- ❖ Graph-level : graph classification
- ❖ Edge-level : edge classification and link prediction
- ❖ **Node-level** : node regression and **classification**
  - **Graph-based semi-supervised learning**

Transductive



Inductive



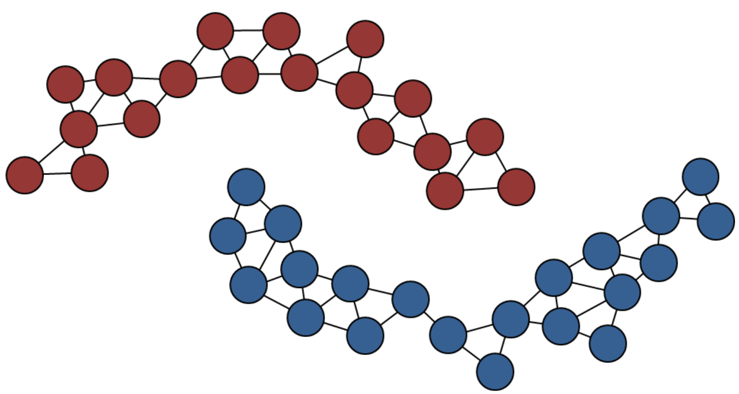
https://en.wikipedia.org/wiki/Semi-supervised_learning

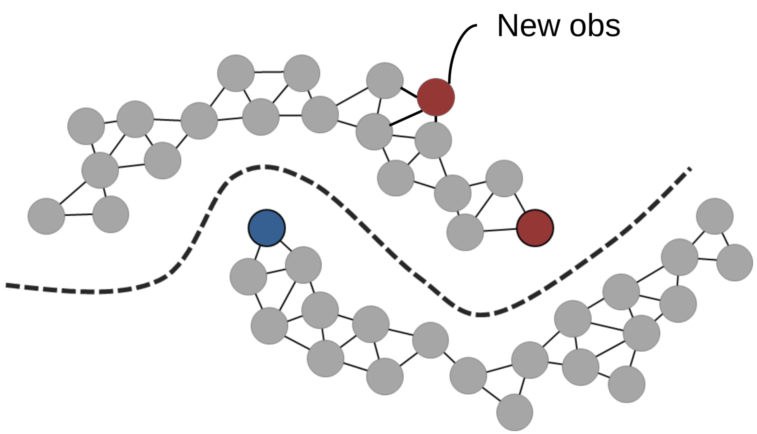
Graph

Tasks

- ❖ Graph-level : graph classification
- ❖ Edge-level : edge classification and link prediction
- ❖ **Node-level** : node regression and **classification**
  - **Graph-based semi-supervised learning**

Transductive	Inductive



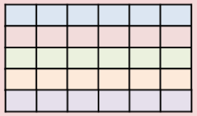


https://en.wikipedia.org/wiki/Semi-supervised_learning

Graph Convolutional Networks

그래프 데이터

Node − feature matrix



Degree matrix

2	0	0	0	0
0	4	0	0	0
0	0	2	0	0
0	0	0	3	0
0	0	0	0	3

Adjacency matrix

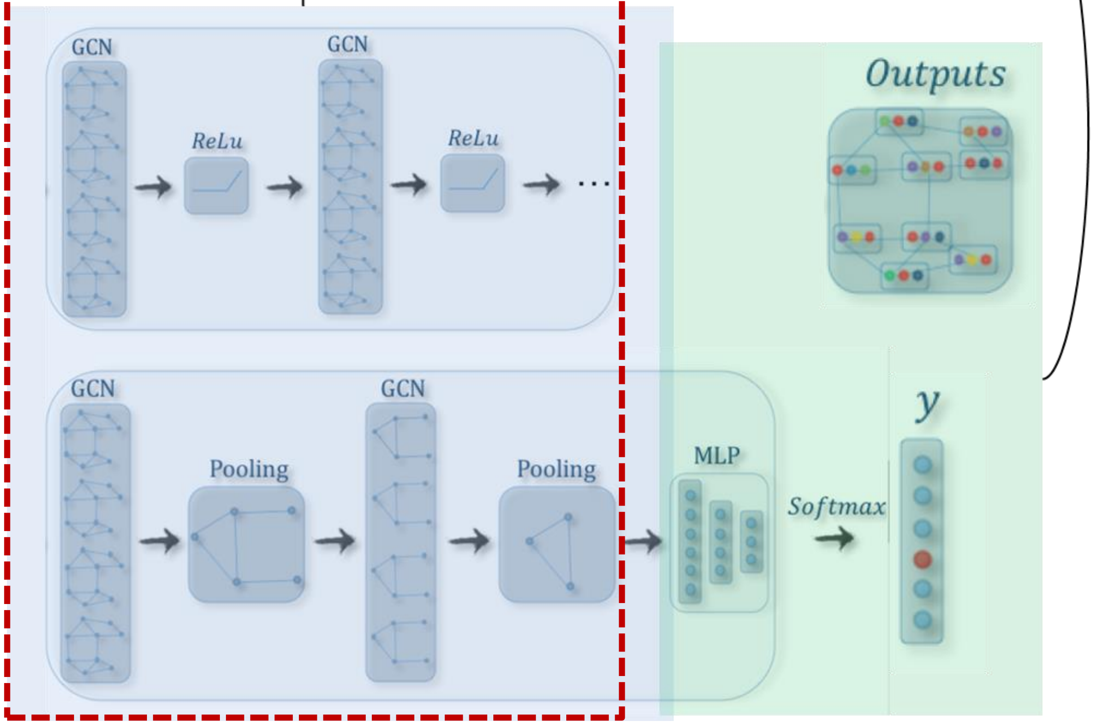
0	1	0	1	0
1	0	1	1	1
0	1	0	0	1
1	1	0	0	1
0	1	1	1	0

Laplacian matrix

2	-1	0	-1	0
-1	4	-1	-1	-1
0	-1	2	0	-1
-1	-1	0	3	-1
0	-1	-1	-1	3

Feature extraction
그래프 데이터 특징에 적합한
NN 구조

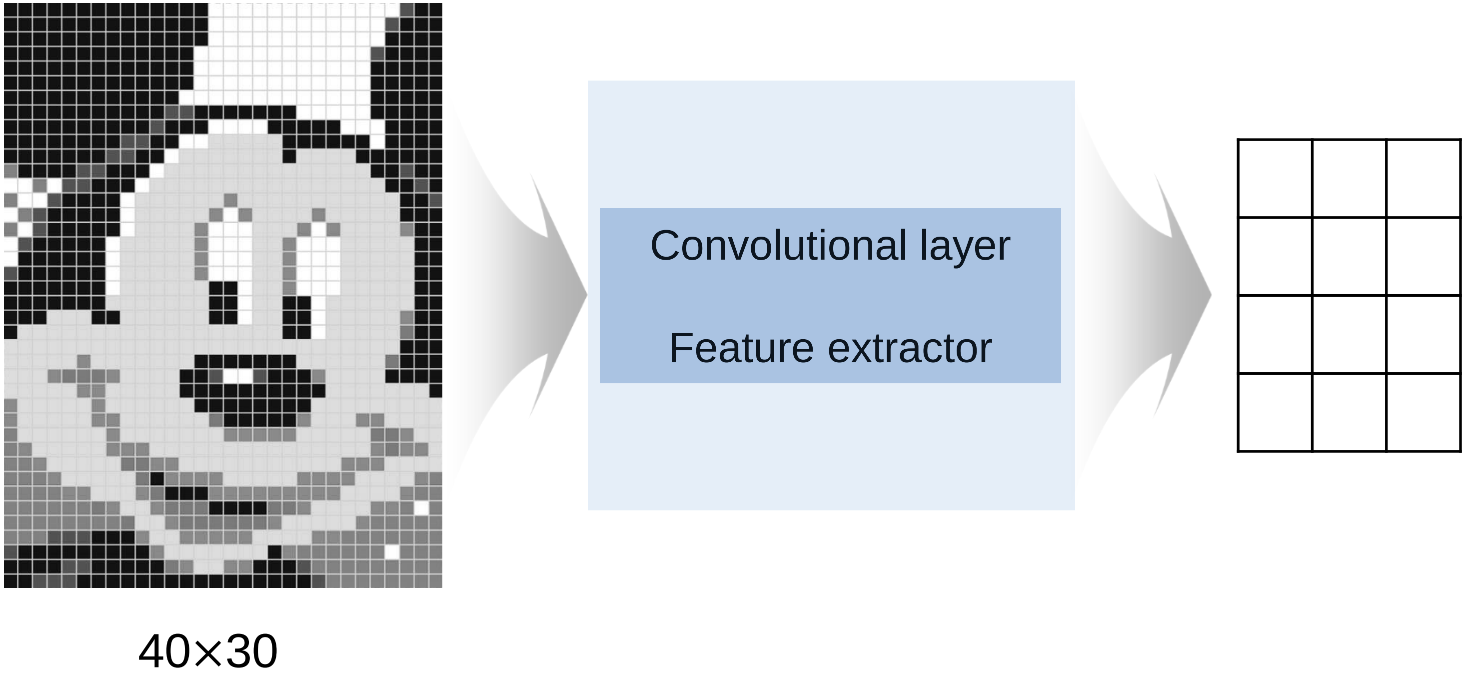
Task해결
Graph-level
Edge-level
Node-level



Graph Convolutional Networks

Convolutional layer

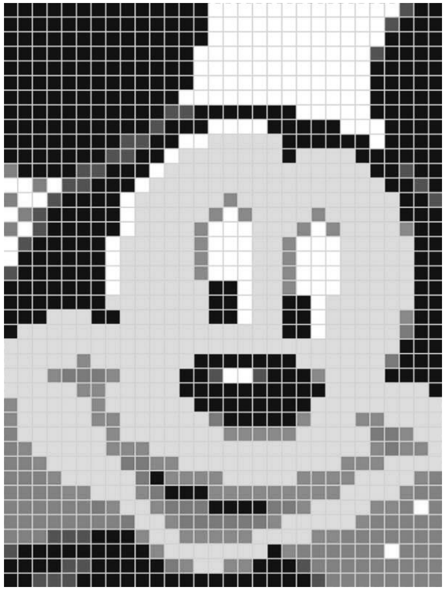
- Sparse connection
- Weight sharing
- Receptive field

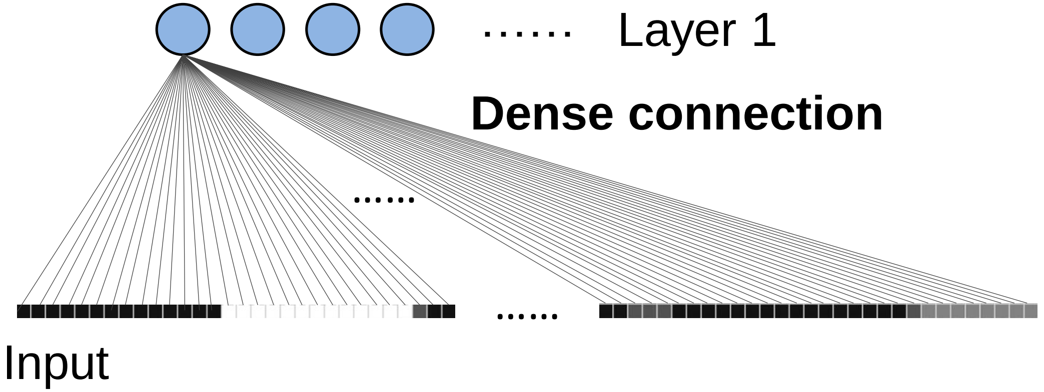


Graph Convolutional Networks

Convolutional layer

- **Sparse connection**
- Weight sharing
- Receptive field

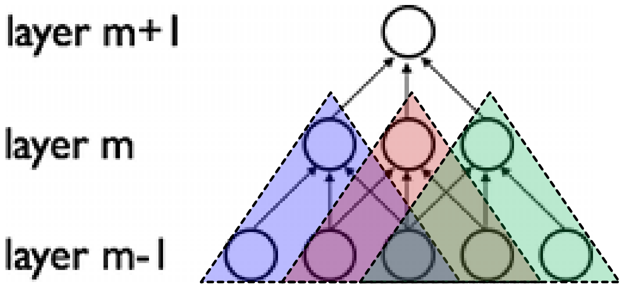


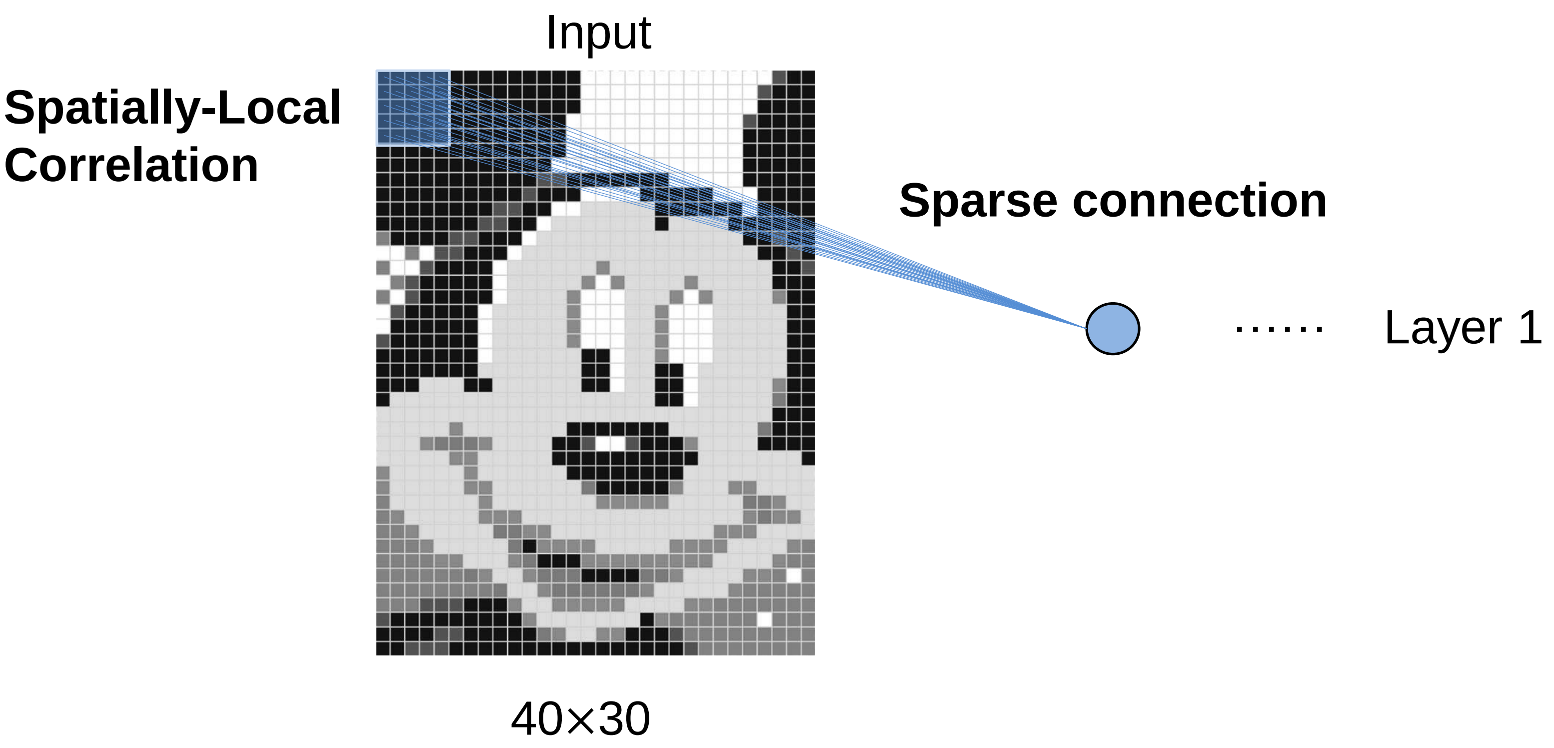


Graph Convolutional Networks

Convolutional layer

- **Sparse connection**
- Weight sharing
- Receptive field

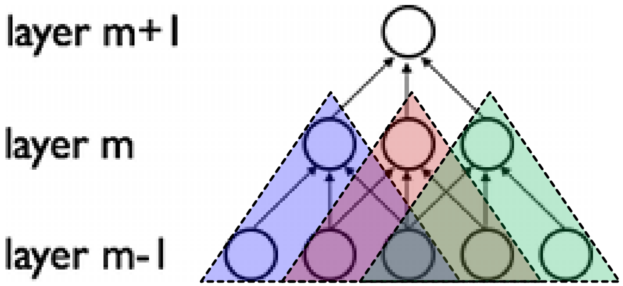




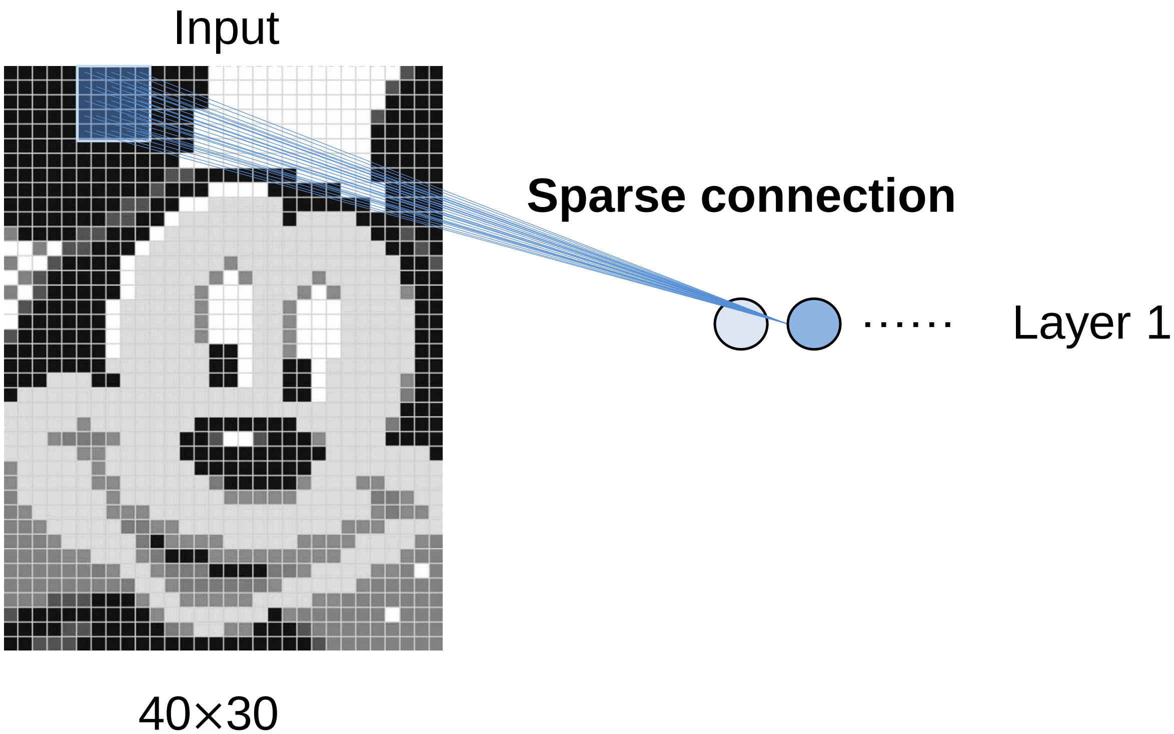
Graph Convolutional Networks

Convolutional layer

- ❖ **Sparse connection**
- ❖ Weight sharing
- ❖ Receptive field



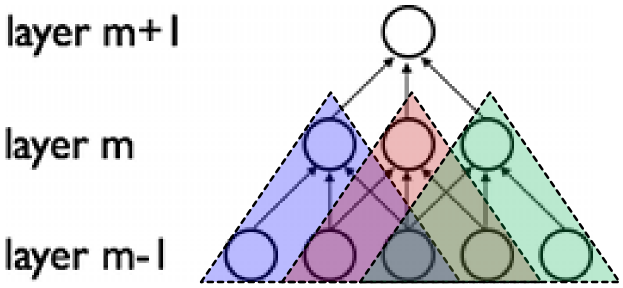
Spatially-Local Correlation



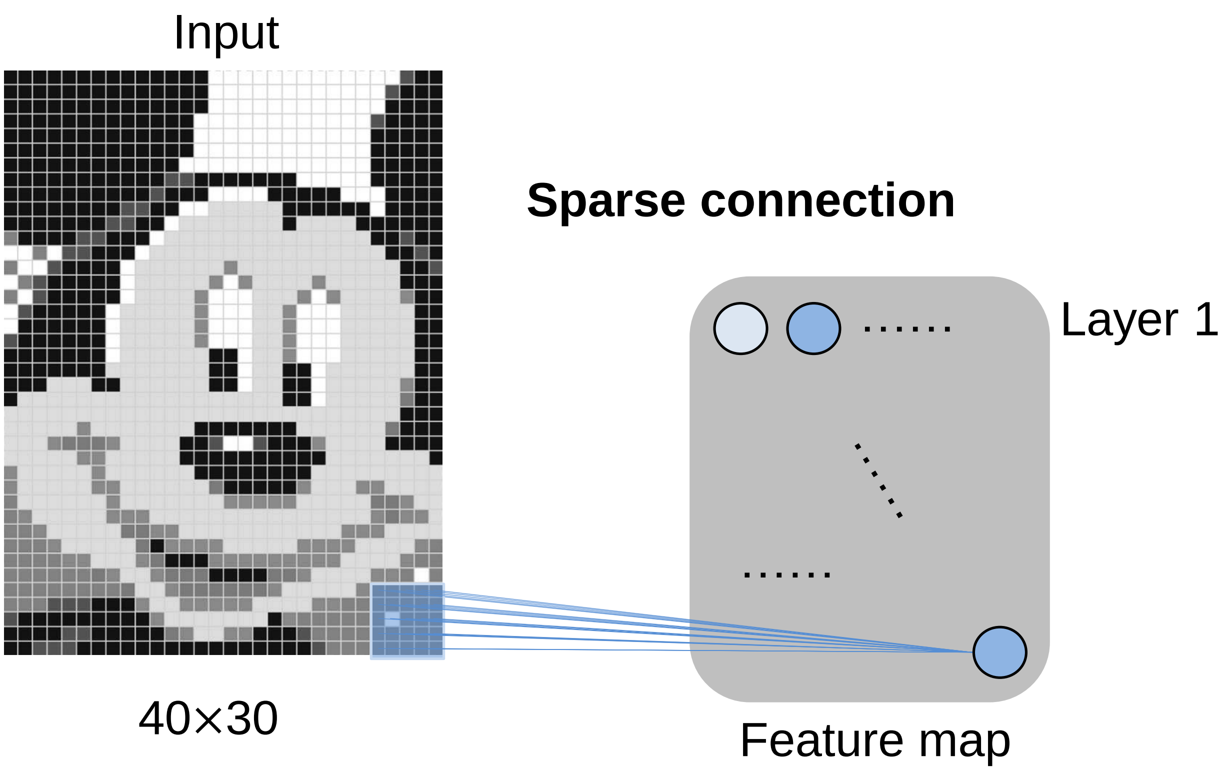
Graph Convolutional Networks

Convolutional layer

- ❖ **Sparse connection**
- ❖ Weight sharing
- ❖ Receptive field



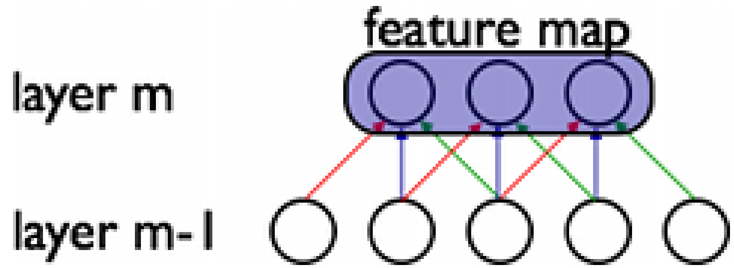
Spatially-Local Correlation



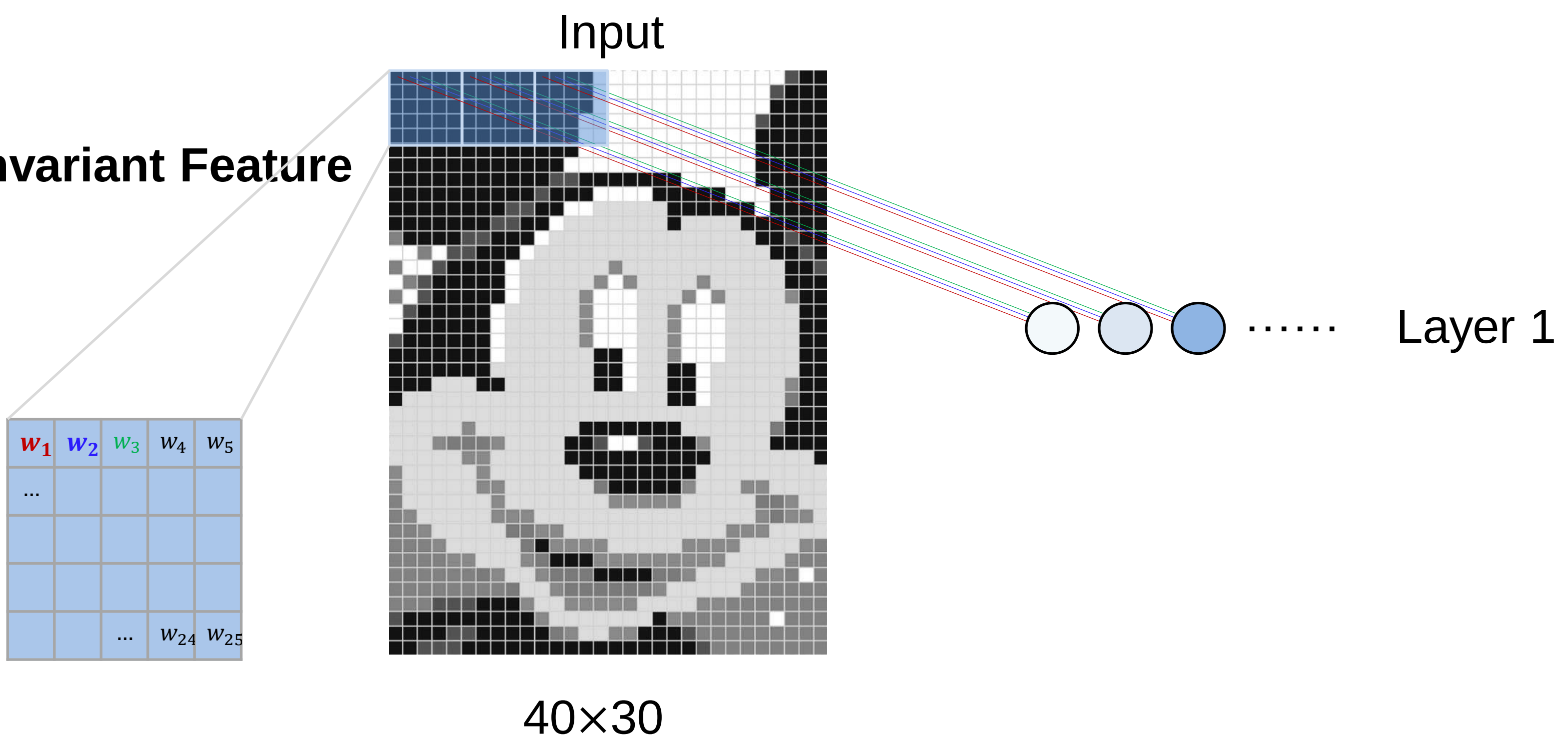
Graph Convolutional Networks

Convolutional layer

- ❖ Sparse connection
- ❖ **Weight sharing**
- ❖ Receptive field



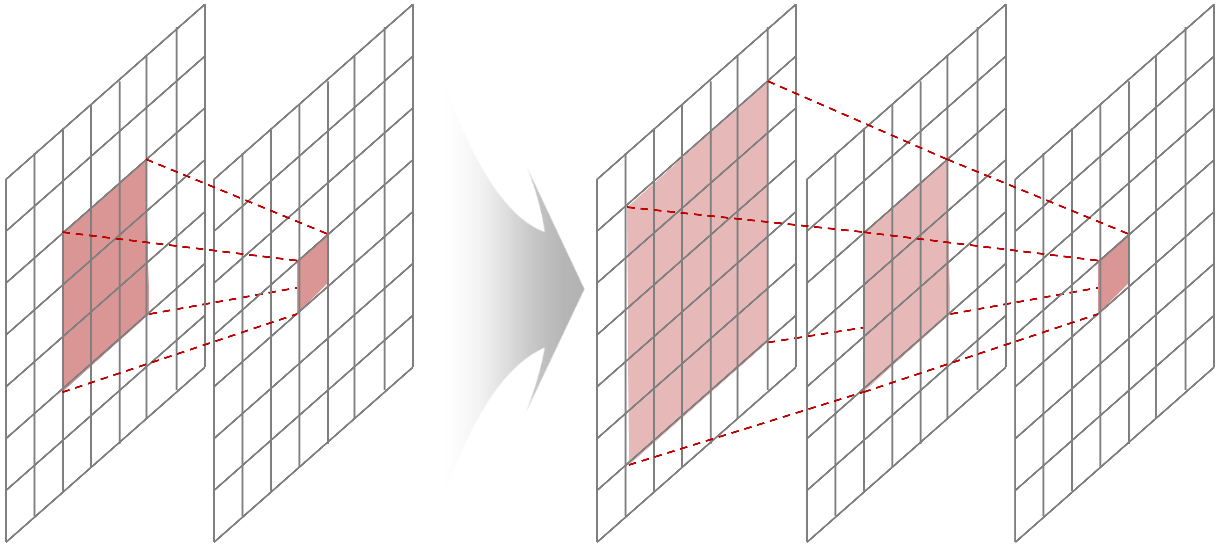
Invariant Feature



Graph Convolutional Networks

Convolutional layer

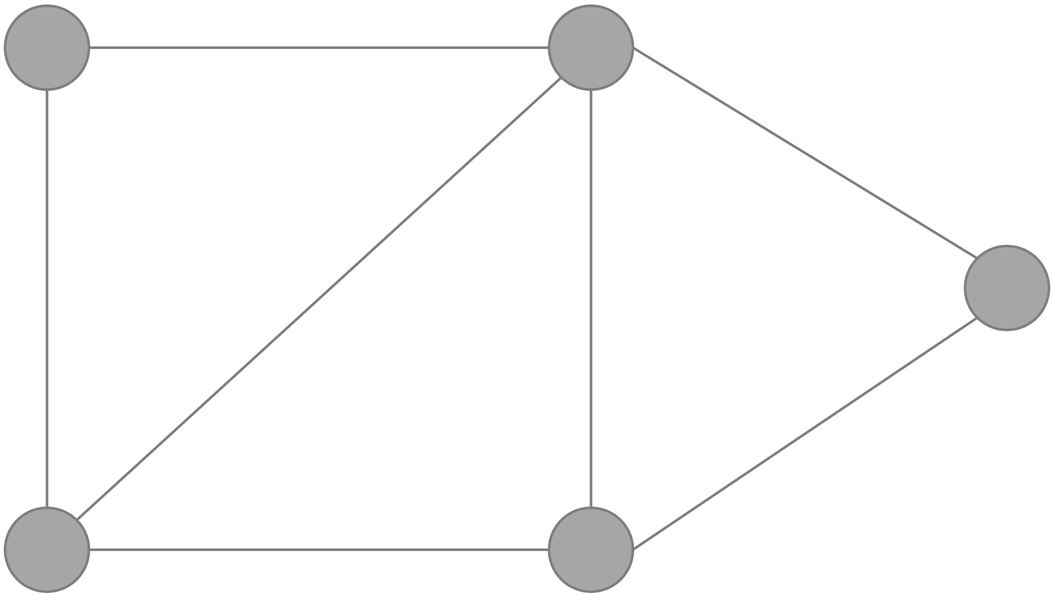
- ❖ Sparse connection
- ❖ Weight sharing
- ❖ **Receptive field**



Graph Convolutional Networks

Convolutional layer

- ❖ Sparse connection
- ❖ Weight sharing
- ❖ Receptive field



Semi-supervised classification with GCN

Graph-based semi-supervised learning (Transductive)

1489회 인용

Published as a conference paper at ICLR 2017

SEMI-SUPERVISED CLASSIFICATION WITH GRAPH CONVOLUTIONAL NETWORKS

Thomas N. Kipf
University of Amsterdam
T.N.Kipf@uva.nl

Max Welling
University of Amsterdam
Canadian Institute for Advanced Research (CIFAR)
M.Welling@uva.nl

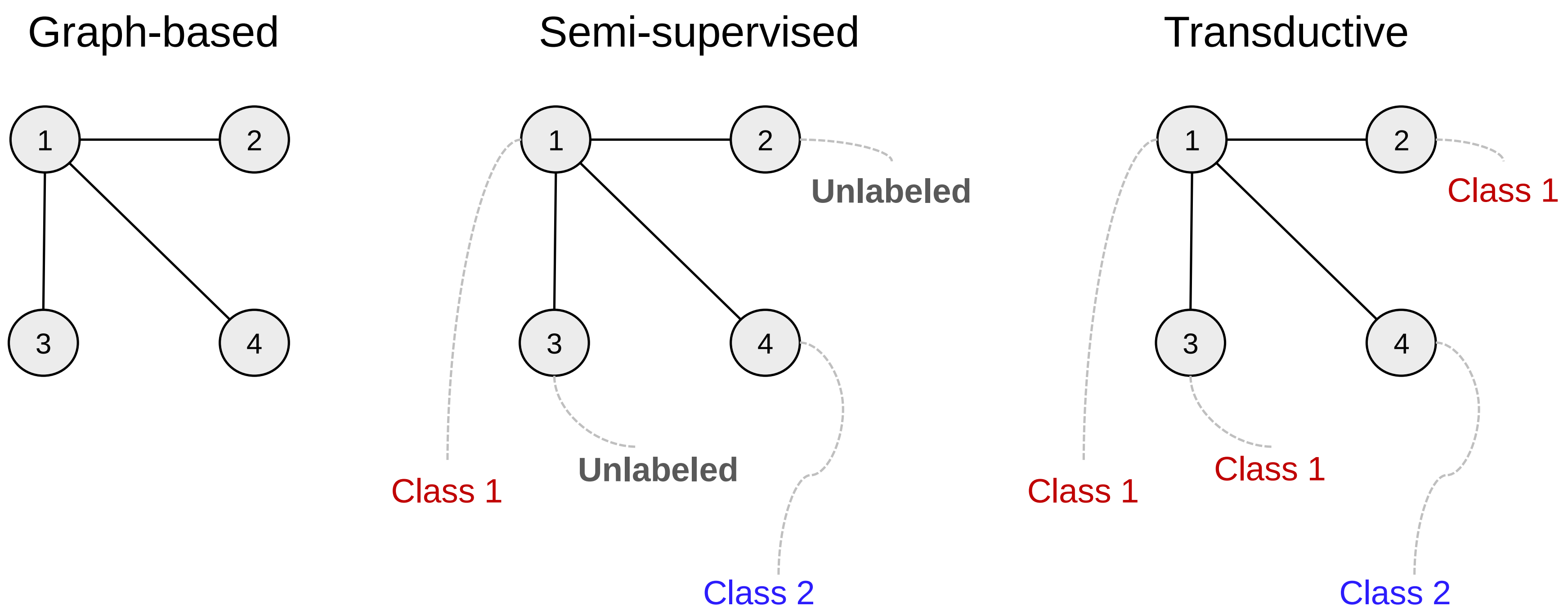
ABSTRACT

We present a scalable approach for semi-supervised learning on graph-structured data that is based on an efficient variant of convolutional neural networks which operate directly on graphs. We motivate the choice of our convolutional architecture via a localized first-order approximation of spectral graph convolutions. Our model scales linearly in the number of graph edges and learns hidden layer representations that encode both local graph structure and features of nodes. In a number of experiments on citation networks and on a knowledge graph dataset we demonstrate that our approach outperforms related methods by a significant margin.

Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.

Semi-supervised classification with GCN

Graph-based semi-supervised learning (Transductive)

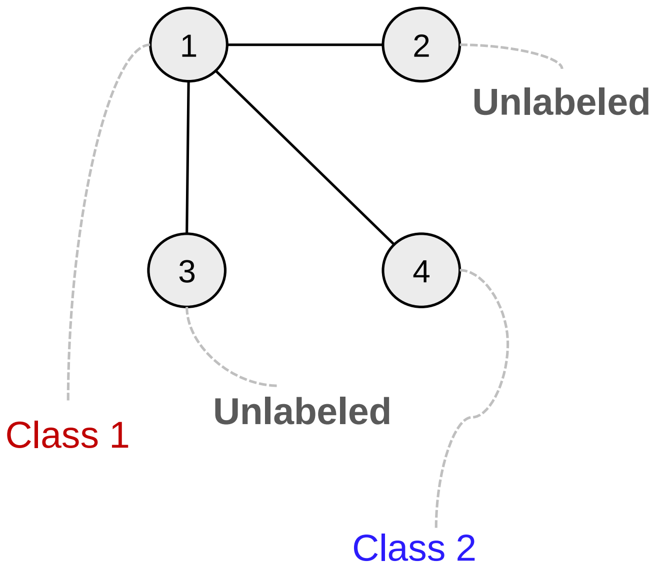


Semi-supervised classification with GCN

Graph-based semi-supervised learning (Transductive)

❖ 기존 문제 해결법 : graph-based regularization

❖ Graph Laplacian regularization term



$$Loss = Loss_0 + \lambda Loss_{reg},$$

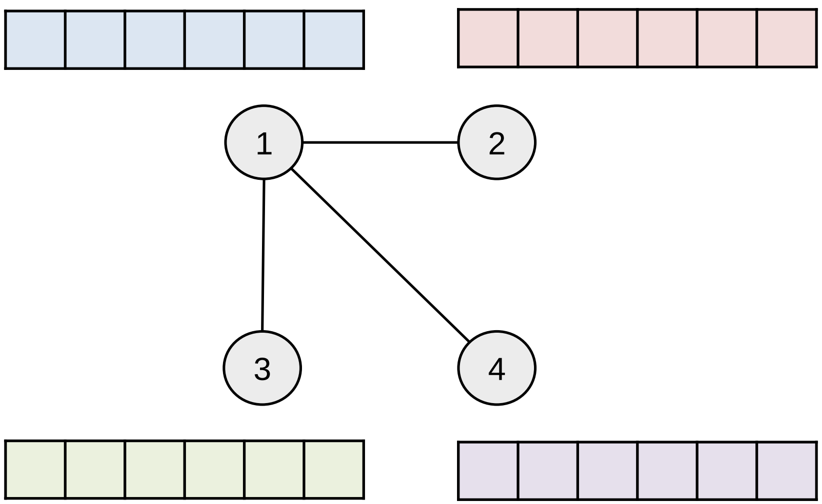
$$Loss_0 = supervised\ loss$$
$$with\ respect\ to\ the\ labeld\ part\ of\ the\ graph$$

$$Loss_{reg} = \sum_{i,j} A_{ij} \left\| f(X_i) - f(X_j) \right\|^2 = f(X)^T L f(X)$$

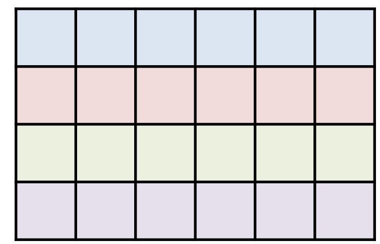
Semi-supervised classification with GCN

Graph-based semi-supervised learning (Transductive)

- ❖ 기존 문제 해결법 : graph-based regularization
- ❖ Graph Laplacian regularization term



$$Node - feature\ matrix$$
$$X \in R^{n \times F}$$



$$Adjacency\ matrix$$
$$A \in R^{n \times n}$$

0	1	1	1
1	0	0	0
1	0	0	0
1	0	0	0

Semi-supervised classification with GCN

Graph-based semi-supervised learning (Transductive)

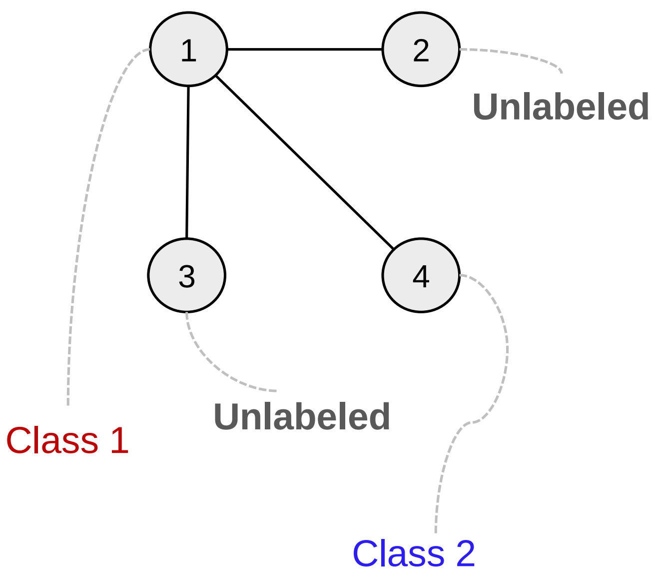
❖ 기존 문제 해결법 : graph-based regularization

❖ Graph Laplacian regularization term

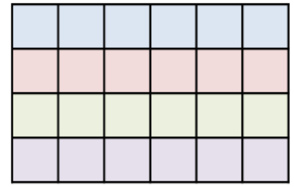
$$Loss = Loss_0 + \lambda Loss_{reg},$$

$Loss_0 = supervised\ loss$
$with\ respect\ to\ the\ labeld\ part\ of\ the\ graph$

$$Loss_{reg} = \sum_{i,j} A_{ij} \|f(X_i) - f(X_j)\|^2 = f(X)^T L f(X)$$



$Node - feature\ matrix$
$X \in R^{n \times F}$



Class 1
Unlabeled
Unlabeled
Class 2

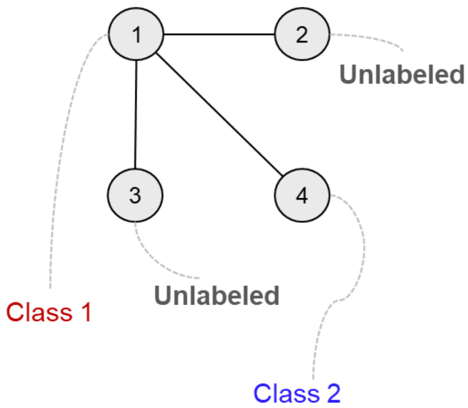
$Adjacency\ matrix$
$A \in R^{n \times n}$

0	1	1	1
1	0	0	0
1	0	0	0
1	0	0	0

Semi-supervised classification with GCN

Graph-based semi-supervised learning (Transductive)

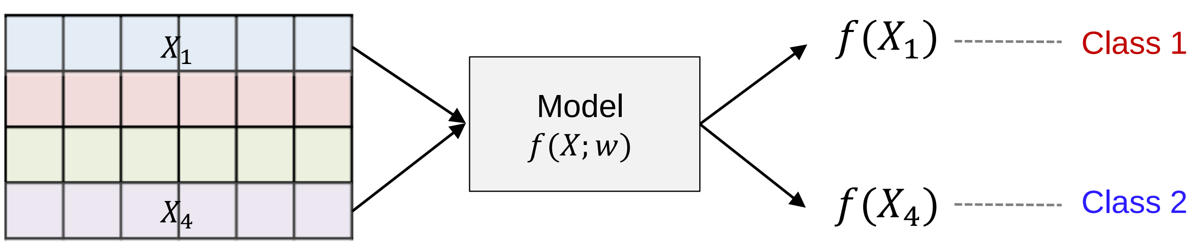
- ❖ 기존 문제 해결법 : graph-based regularization
- ❖ Graph Laplacian regularization term



$$Loss = Loss_0 + \lambda Loss_{reg},$$

$$Loss_0 = supervised\ loss$$
$$with\ respect\ to\ the\ labeld\ part\ of\ the\ graph$$

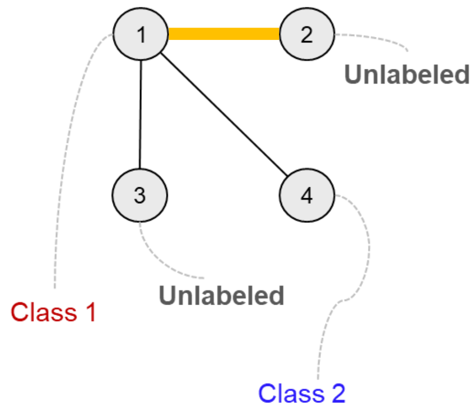
$$Loss_{reg} = \sum_{i,j} A_{ij} \| f(X_i) - f(X_j) \|^2 = f(X)^T L f(X)$$



Semi-supervised classification with GCN

Graph-based semi-supervised learning (Transductive)

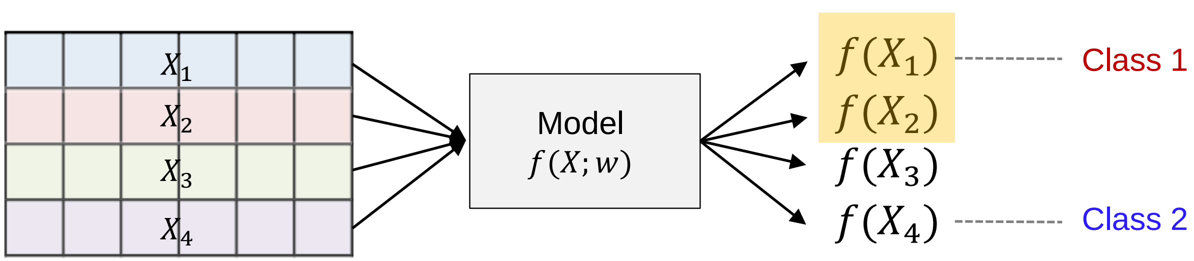
- ❖ 기존 문제 해결법 : graph-based regularization
- ❖ Graph Laplacian regularization term



$$Loss = Loss_0 + \lambda Loss_{reg},$$

$$Loss_0 = supervised\ loss$$
$$with\ respect\ to\ the\ labeld\ part\ of\ the\ graph$$

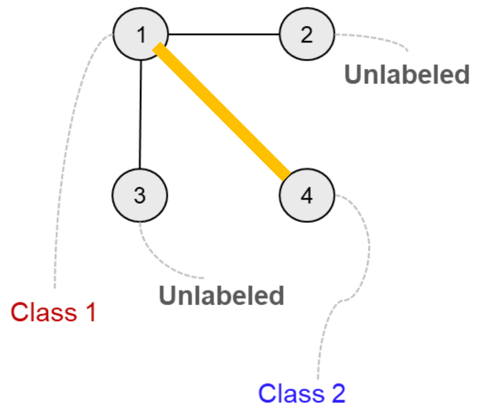
$$Loss_{reg} = \sum_{i,j} A_{ij} \| f(X_i) - f(X_j) \|^2 = f(X)^T L f(X)$$



Semi-supervised classification with GCN

Graph-based semi-supervised learning (Transductive)

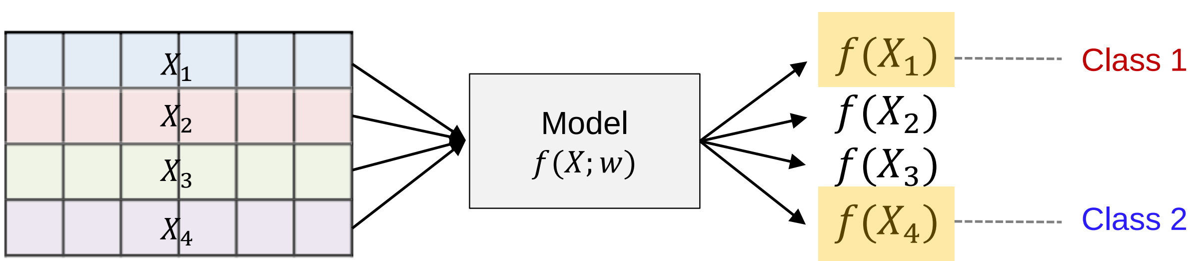
- ❖ 기존 문제 해결법 : graph-based regularization
- ❖ Graph Laplacian regularization term



$$Loss = Loss_0 + \lambda Loss_{reg},$$

$$Loss_0 = supervised\ loss$$
$$with\ respect\ to\ the\ labeld\ part\ of\ the\ graph$$

$$Loss_{reg} = \sum_{i,j} A_{ij} \| f(X_i) - f(X_j) \|^2 = f(X)^T L f(X)$$

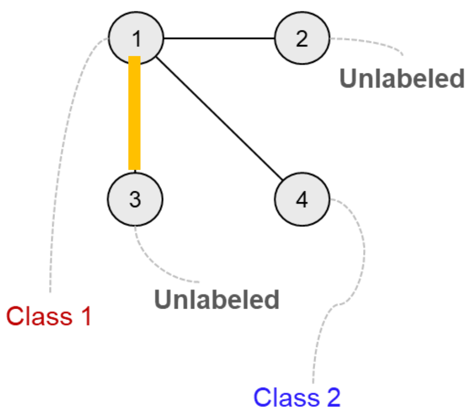


Semi-supervised classification with GCN

Graph-based semi-supervised learning (Transductive)

❖ 기존 문제 해결법 : graph-based regularization

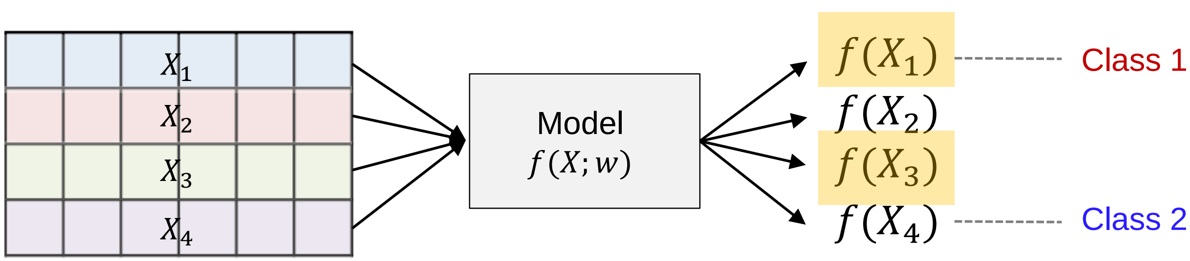
❖ Graph Laplacian regularization term



$$Loss = Loss_0 + \lambda Loss_{reg},$$

$$Loss_0 = supervised\ loss$$
$$with\ respect\ to\ the\ labeld\ part\ of\ the\ graph$$

$$Loss_{reg} = \sum_{i,j} A_{ij} \| f(X_i) - f(X_j) \|^2 = f(X)^T L f(X)$$

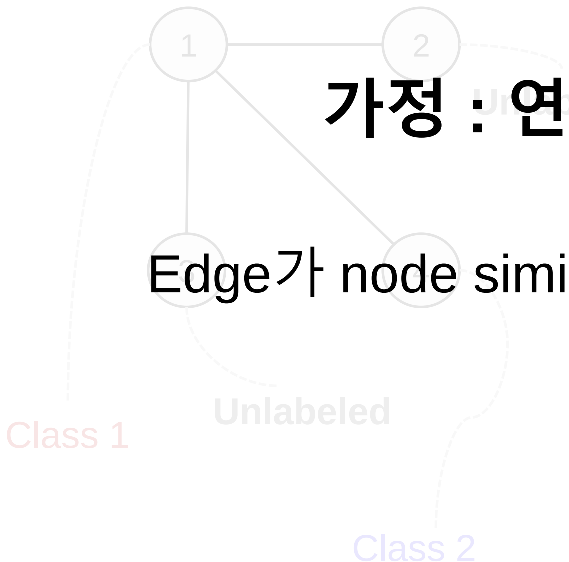


Semi-supervised classification with GCN

Graph-based semi-supervised learning (Transductive)

❖ 기존 문제 해결법 : graph-based regularization

❖ Graph Laplacian regularization term



가정 : 연결된 노드는 레이블이 같다

$$Loss = Loss_0 + \lambda Loss_{reg},$$

Edge가 node similarity를 의미하지 않는 경우에 제약 있음

$Loss_0$ = supervised loss with respect to the labeld part of the graph

$$Loss_{reg} = \sum_{i,j} A_{ij} \left\| f(X_i) - f(X_j) \right\|^2 = f(X)^T L f(X)$$

Semi-supervised classification with GCN

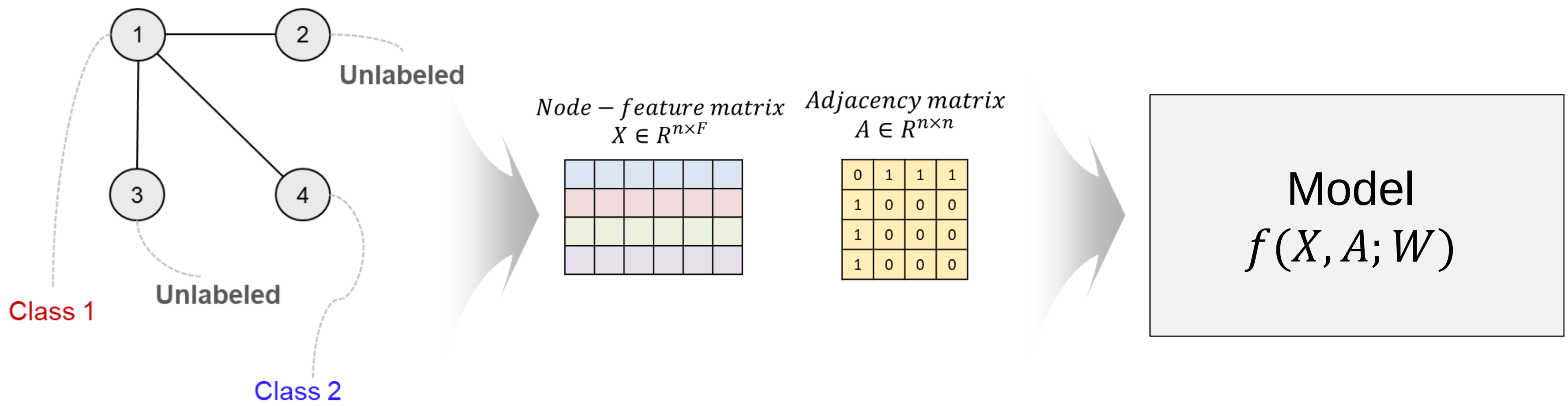
Graph-based semi-supervised learning (Transductive)

❖ 직접적으로 graph structure가 NN모델에 입력

$$Z = f(X, A) = softmax(\hat{A} ReLU(\hat{A} X W^0) W^1)$$

$$Renormalization\ trick:\ \hat{A} = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}, \tilde{A} = (A + I_N)$$

$$Cross\ entropy\ Loss = -\sum_{l \in Y_L} \sum_{f=1}^{F} Y_{lf} \ln Z_{lf}$$



Semi-supervised classification with GCN

Graph-based semi-supervised learning (Transductive)

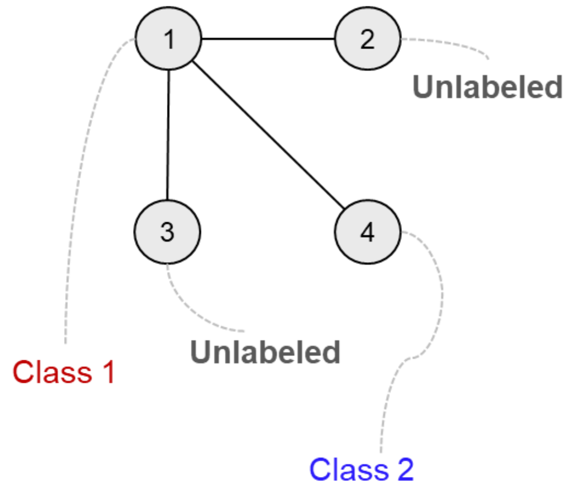
❖ 직접적으로 graph structure가 NN모델에 입력

$$Z = f(X, A) = softmax(\hat{A} ReLU(\hat{A} X W^0) W^1)$$

$$Renormalization\ trick:\ \hat{A} = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}, \tilde{A} = (A + I_N)$$

F : number of class
Y_L : all labeled examples

$$Cross\ entropy\ Loss = -\sum_{l \in Y_L} \sum_{f=1}^{F} Y_{lf} \ln Z_{lf}$$



Semi-supervised classification with GCN

Graph-based semi-supervised learning (Transductive)

❖ 직접적으로 graph structure가 NN모델에 입력

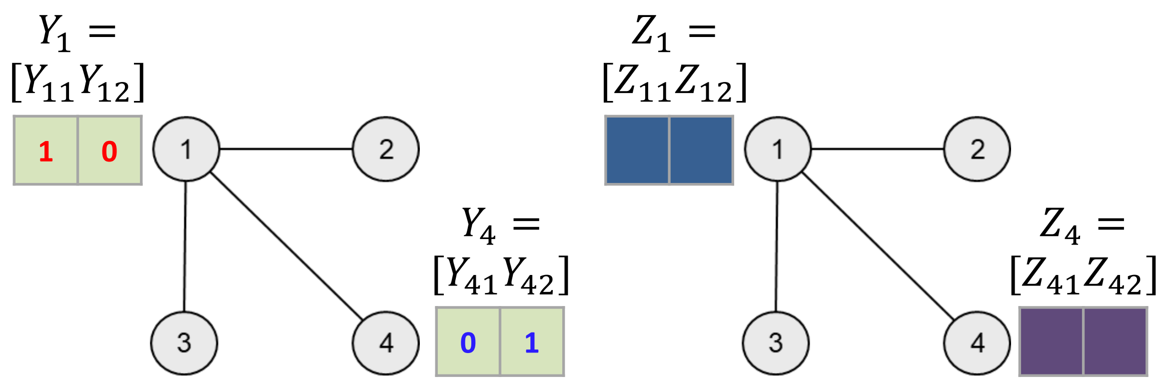
$$Z = f(X, A) = softmax(\hat{A} ReLU(\hat{A} X W^0) W^1)$$

$$Renormalization\ trick: \hat{A} = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}, \tilde{A} = (A + I_N)$$

F : number of class
Y_L : all labeled examples

$$Cross\ entropy\ Loss = -\sum_{l \in Y_L} \sum_{f=1}^{F} Y_{lf} \ln Z_{lf}$$

$$= -(\ln Z_{11} + \ln Z_{42})$$



Semi-supervised classification with GCN

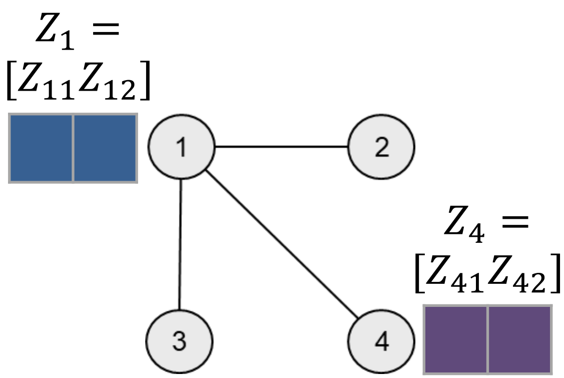
Graph-based semi-supervised learning (Transductive)

❖ 직접적으로 graph structure가 NN모델에 입력

$$Z = f(X, A) = softmax(\hat{A} ReLU(\hat{A} X W^0) W^1)$$

$$Renormalization\ trick: \hat{A} = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}, \tilde{A} = (A + I_N)$$

$$Cross\ entropy\ Loss = -\sum_{l \in Y_L} \sum_{f=1}^{F} Y_{lf} \ln Z_{lf}$$



Node Embedding

Semi-supervised classification with GCN

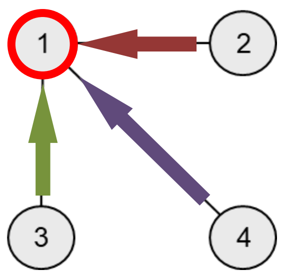
Graph-based semi-supervised learning (Transductive)

❖ 직접적으로 graph structure가 NN모델에 입력

$$Z = f(X, A) = softmax(\hat{A}ReLU(\hat{A}XW^{0})W^{1})$$

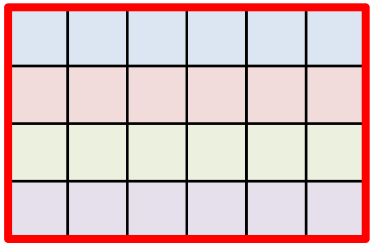
$$Renormalization\ trick: \hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}, \tilde{A} = (A + I_N)$$

$$Cross\ entropy\ Loss = -\sum_{l \in Y_L}\sum_{f=1}^{F} Y_{lf} \ln Z_{lf}$$



$$ReLU(\mathbf{AX}W^{0})$$

0	1	1	1
1	0	0	0
1	0	0	0
1	0	0	0



Semi-supervised classification with GCN

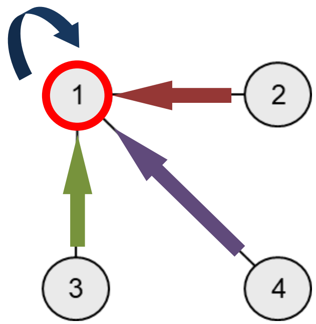
Graph-based semi-supervised learning (Transductive)

❖ 직접적으로 graph structure가 NN모델에 입력

$$Z = f(X, A) = softmax(\hat{A}ReLU(\hat{A}XW^0)W^1)$$

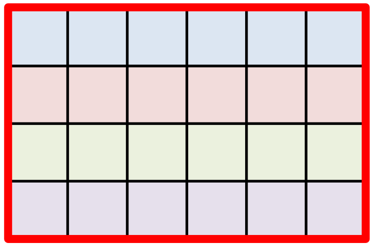
$$Renormalization\ trick: \hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}, \tilde{A} = (A + I_N)$$

$$Cross\ entropy\ Loss = -\sum_{l \in Y_L}\sum_{f=1}^{F} Y_{lf} ln Z_{lf}$$



$ReLU((A + I_N)XW^0)$

1	1	1	1
1	1	0	0
1	0	1	0
1	0	0	1



Semi-supervised classification with GCN

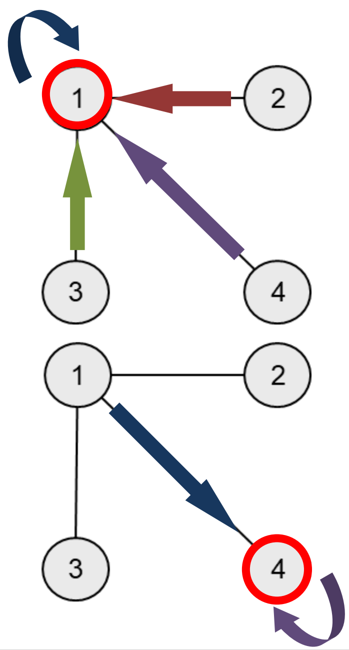
Graph-based semi-supervised learning (Transductive)

❖ 직접적으로 graph structure가 NN모델에 입력

$$Z = f(X, A) = softmax(\hat{A} ReLU(\hat{A} X W^{0}) W^{1})$$

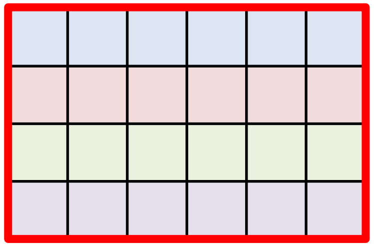
$$Renormalization\ trick: \hat{A} = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}, \tilde{A} = (A + I_N)$$

$$Cross\ entropy\ Loss = -\sum_{l \in Y_L} \sum_{f=1}^{F} Y_{lf} \ln Z_{lf}$$



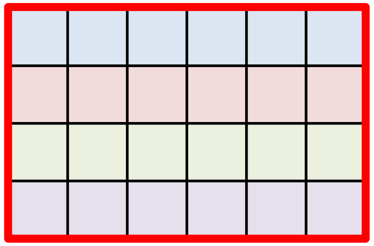
$ReLU((A + I_N) X W^{0})$

1	1	1	1
1	1	0	0
1	0	1	0
1	0	0	1



$ReLU((A + I_N) X W^{0})$

1	1	1	1
1	1	0	0
1	0	1	0
1	0	0	1



Semi-supervised classification with GCN

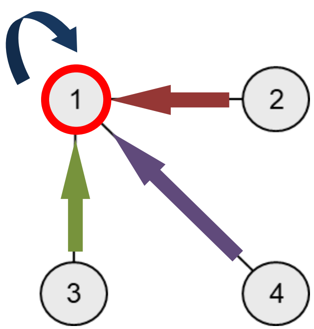
Graph-based semi-supervised learning (Transductive)

❖ 직접적으로 graph structure가 NN모델에 입력

$$Z = f(X, A) = softmax(\hat{A} ReLU(\hat{A} X W^0) W^1)$$

$$Renormalization\ trick:\ \hat{A} = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}, \tilde{A} = (A + I_N)$$

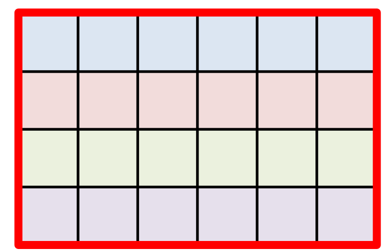
$$Cross\ entropy\ Loss = -\sum_{l \in Y_L} \sum_{f=1}^{F} Y_{lf} \ln Z_{lf}$$

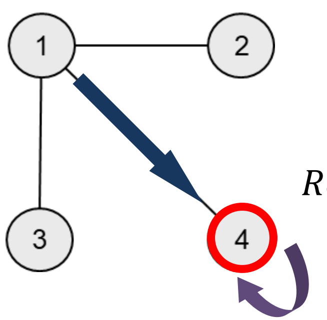


$$ReLU((\tilde{D}^{-\frac{1}{2}}(A + I_N)\tilde{D}^{-\frac{1}{2}})XW^0)$$

Average

¼	¼	¼	¼
½	½	0	0
½	0	½	0
½	0	0	½

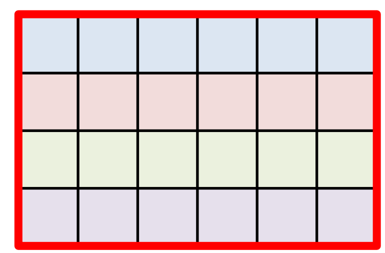




$$ReLU((\tilde{D}^{-\frac{1}{2}}(A + I_N)\tilde{D}^{-\frac{1}{2}})XW^0)$$

Average

¼	¼	¼	¼
½	½	0	0
½	0	½	0
½	0	0	½



Semi-supervised classification with GCN

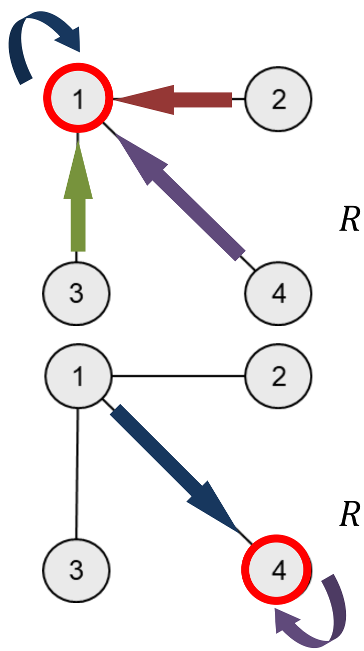
Graph-based semi-supervised learning (Transductive)

❖ 직접적으로 graph structure가 NN모델에 입력

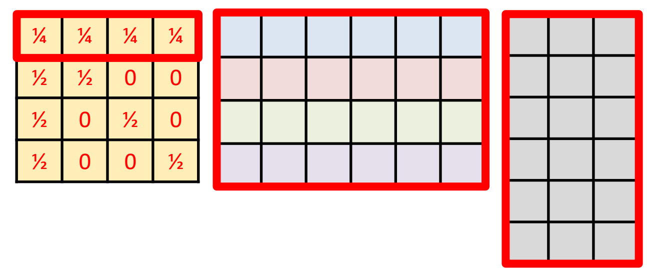
$$Z = f(X, A) = softmax(\hat{A} ReLU(\hat{A} X W^0) W^1)$$

$$Renormalization\ trick: \hat{A} = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}, \tilde{A} = (A + I_N)$$

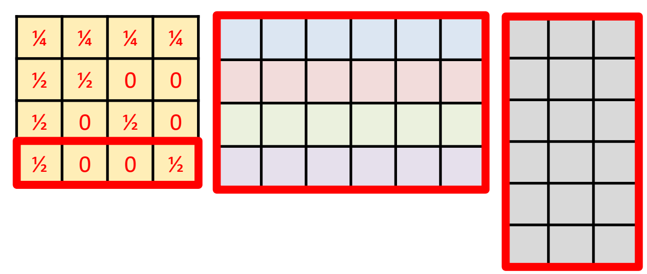
$$Cross\ entropy\ Loss = -\sum_{l \in Y_L} \sum_{f=1}^{F} Y_{lf} \ln Z_{lf}$$



$$ReLU((\tilde{D}^{-\frac{1}{2}}(A + I_N)\tilde{D}^{-\frac{1}{2}})XW^0)$$



$$ReLU((\tilde{D}^{-\frac{1}{2}}(A + I_N)\tilde{D}^{-\frac{1}{2}})XW^0)$$



Semi-supervised classification with GCN

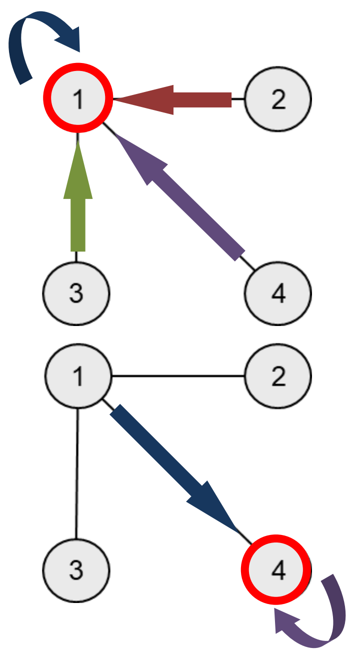
Graph-based semi-supervised learning (Transductive)

❖ 직접적으로 graph structure가 NN모델에 입력

$$Z = f(X, A) = softmax(\hat{A}ReLU(\hat{A}XW^0)W^1)$$

$$Renormalization\ trick:\ \hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}, \tilde{A} = (A + I_N)$$

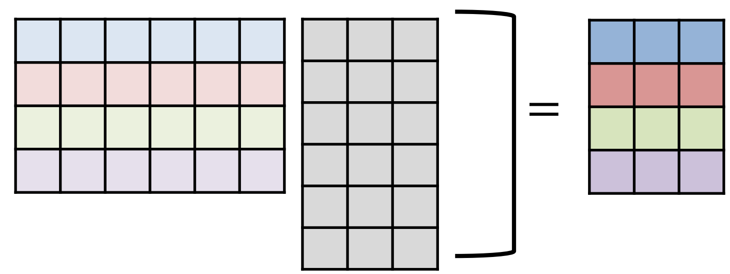
$$Cross\ entropy\ Loss = -\sum_{l \in Y_L}\sum_{f=1}^{F} Y_{lf} \ln Z_{lf}$$



$$\boldsymbol{ReLU}\left(\left(\boldsymbol{\tilde{D}}^{-\frac{1}{2}}(\boldsymbol{A} + \boldsymbol{I_N})\boldsymbol{\tilde{D}}^{-\frac{1}{2}}\right)\boldsymbol{XW^0}\right) = ReLU(\hat{A}XW^0) = H_1$$

$= \boldsymbol{ReLU}$

¼	¼	¼	¼
½	½	0	0
½	0	½	0
½	0	0	½



Semi-supervised classification with GCN

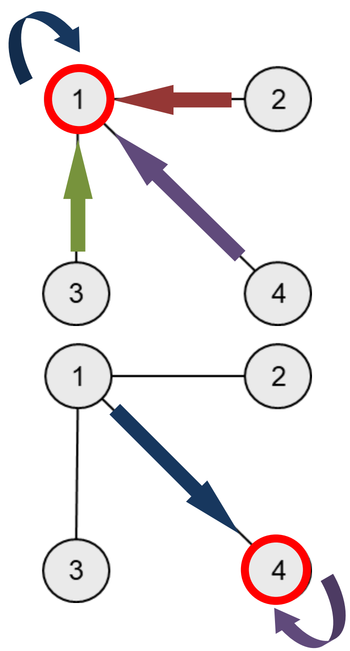
Graph-based semi-supervised learning (Transductive)

❖ 직접적으로 graph structure가 NN모델에 입력

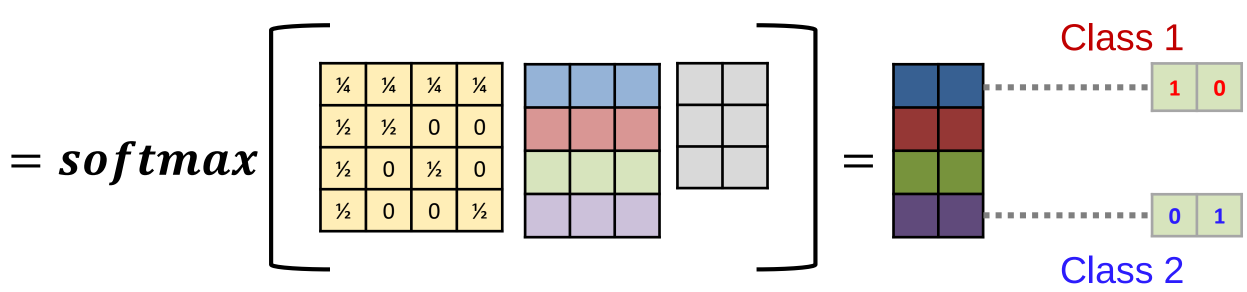
$$Z = f(X, A) = softmax(\hat{A} ReLU(\hat{A} X W^0) W^1)$$

$$Renormalization\ trick:\ \hat{A} = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}, \tilde{A} = (A + I_N)$$

$$Cross\ entropy\ Loss = -\sum_{l \in Y_L} \sum_{f=1}^{F} Y_{lf} \ln Z_{lf}$$



$$Z = f(X, A) = softmax(\hat{A} H_1 W^1)$$



Semi-supervised classification with GCN

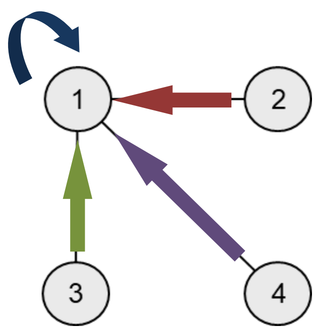
Graph-based semi-supervised learning (Transductive)

❖ 직접적으로 graph structure가 NN모델에 입력

$$Z = f(X, A) = softmax(\hat{A} ReLU(\hat{A} X W^0) W^1)$$

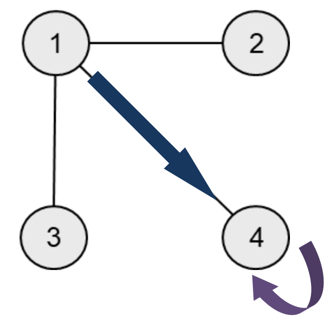
$$Renormalization\ trick: \hat{A} = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}, \tilde{A} = (A + I_N)$$

$$Cross\ entropy\ Loss = -\sum_{l \in Y_L} \sum_{f=1}^{F} Y_{lf} \ln Z_{lf}$$



Sparse connection

Weight sharing



Semi-supervised classification with GCN

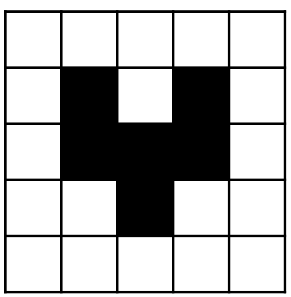
Graph-based semi-supervised learning (Transductive)

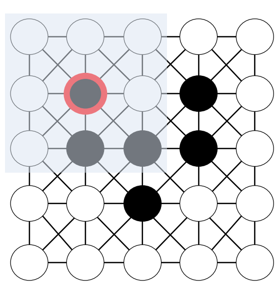
❖ 직접적으로 graph structure가 NN모델에 입력

$$Z = f(X, A) = softmax(\hat{A} ReLU(\hat{A} X W^0) W^1)$$

$$Renormalization\ trick: \hat{A} = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}, \tilde{A} = (A + I_N)$$

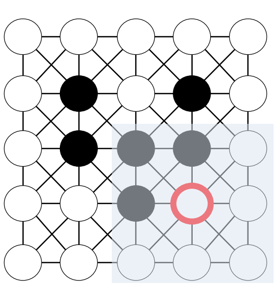
$$Cross\ entropy\ Loss = -\sum_{l \in Y_L} \sum_{f=1}^{F} Y_{lf} \ln Z_{lf}$$





Sparse connection

Weight sharing



Semi-supervised classification with GCN

Graph-based semi-supervised learning (Transductive)

❖ 직접적으로 graph structure가 NN모델에 입력

Layer 1

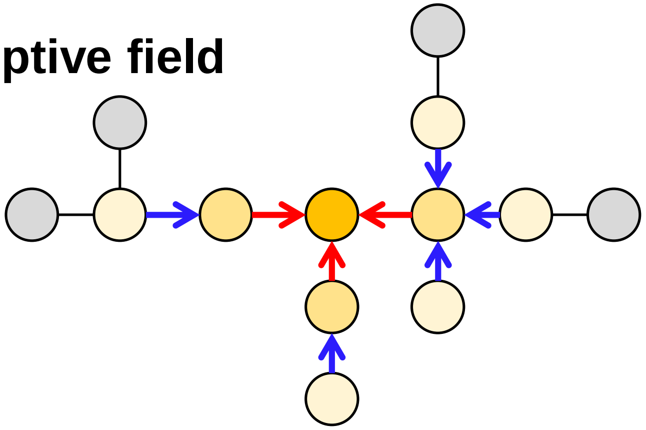
$$Z = f(X, A) = softmax(\hat{A}\,ReLU(\hat{A}XW^0)W^1)$$

Layer 2

$$Renormalization\ trick:\ \hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}},\ \tilde{A} = (A + I_N)$$

$$Cross\ entropy\ Loss = -\sum_{l \in Y_L}\sum_{f=1}^{F} Y_{lf} \ln Z_{lf}$$

Receptive field



Semi-supervised classification with GCN

Graph-based semi-supervised learning (Transductive)

Table 1: Dataset statistics, as reported in Yang et al. (2016).

Dataset	Type	Nodes	Edges	Classes	Features	Label rate
Citeseer	Citation network	3,327	4,732	6	3,703	0.036
Cora	Citation network	2,708	5,429	7	1,433	0.052
Pubmed	Citation network	19,717	44,338	3	500	0.003
NELL	Knowledge graph	65,755	266,144	210	5,414	0.001

Table 2: Summary of results in terms of classification accuracy (in percent).

Method	Citeseer	Cora	Pubmed	NELL
ManiReg [3]	60.1	59.5	70.7	21.8
SemiEmb [28]	59.6	59.0	71.1	26.7
LP [32]	45.3	68.0	63.0	26.5
DeepWalk [22]	43.2	67.2	65.3	58.1
ICA [18]	69.1	75.1	73.9	23.1
Planetoid* [29]	64.7 (26s)	75.7 (13s)	77.2 (25s)	61.9 (185s)
GCN (this paper)	**70.3** (7s)	**81.5** (4s)	**79.0** (38s)	**66.0** (48s)
GCN (rand. splits)	67.9 ± 0.5	80.1 ± 0.5	78.9 ± 0.7	58.4 ± 1.7

Inductive Representation Learning on Large Graphs

Graph-based semi-supervised learning (Inductive)

564회 인용

Inductive Representation Learning on Large Graphs

William L. Hamilton* wleif@stanford.edu
Rex Ying* rexying@stanford.edu
Jure Leskovec jure@cs.stanford.edu

Department of Computer Science
Stanford University
Stanford, CA, 94305

Abstract

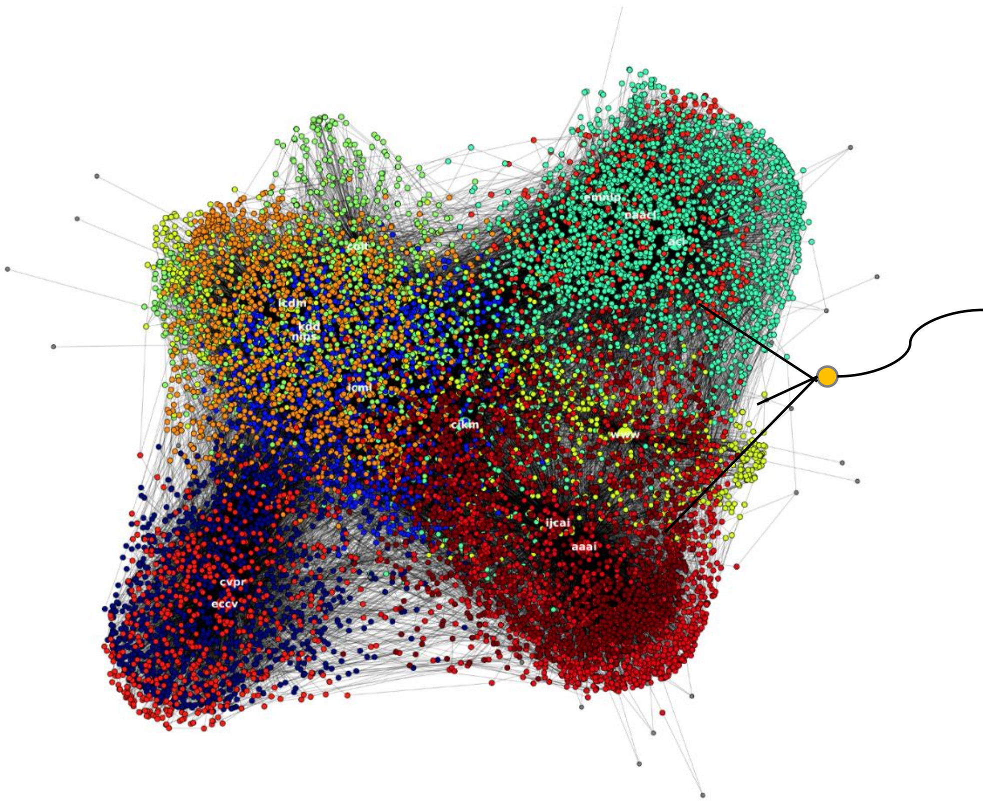
Low-dimensional embeddings of nodes in large graphs have proved extremely useful in a variety of prediction tasks, from content recommendation to identifying protein functions. However, most existing approaches require that all nodes in the graph are present during training of the embeddings; these previous approaches are inherently *transductive* and do not naturally generalize to unseen nodes. Here we present GraphSAGE, a general *inductive* framework that leverages node feature information (e.g., text attributes) to efficiently generate node embeddings for previously unseen data. Instead of training individual embeddings for each node, we learn a function that generates embeddings by sampling and aggregating features from a node's local neighborhood. Our algorithm outperforms strong baselines on three inductive node-classification benchmarks: we classify the category of unseen nodes in evolving information graphs based on citation and Reddit post data, and we show that our algorithm generalizes to completely unseen graphs using a multi-graph dataset of protein-protein interactions.

Hamilton, W., Ying, Z., & Leskovec, J. (2017). Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems* (pp. 1024-1034).

Inductive Representation Learning on Large Graphs

Graph-based semi-supervised learning (Inductive)

- ❖ Transductive 한계점 多 → Inductive
- ❖ Full batch (memory on Large Graphs), Unseen Node, Completely Unseen graph

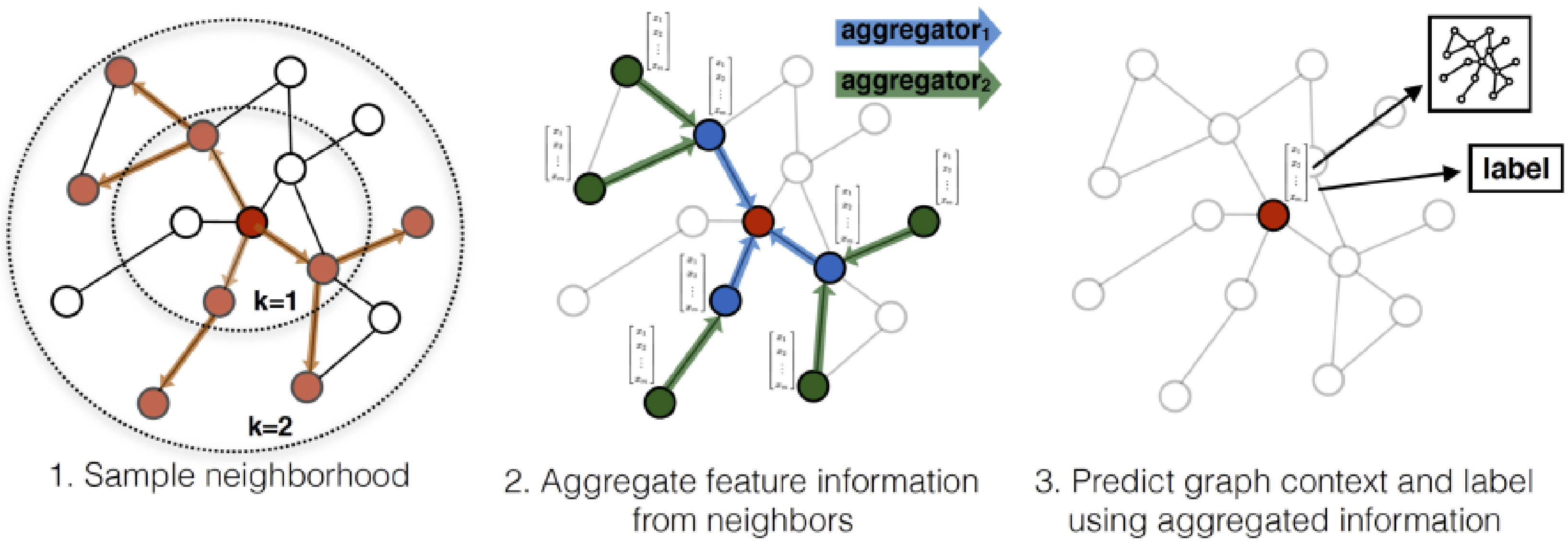


New papers
New posts on Reddit
New users
New Youtube videos

Inductive Representation Learning on Large Graphs

Graph-based semi-supervised learning (Inductive)

- ❖ GraphSAGE (SAmple and aggreGatE)
- ❖ Full batch → Mini batch
- ❖ Transductive → Inductive (Sampling)
- ❖ Average × → Aggregating (Mean / LSTM / Pooling aggregator)

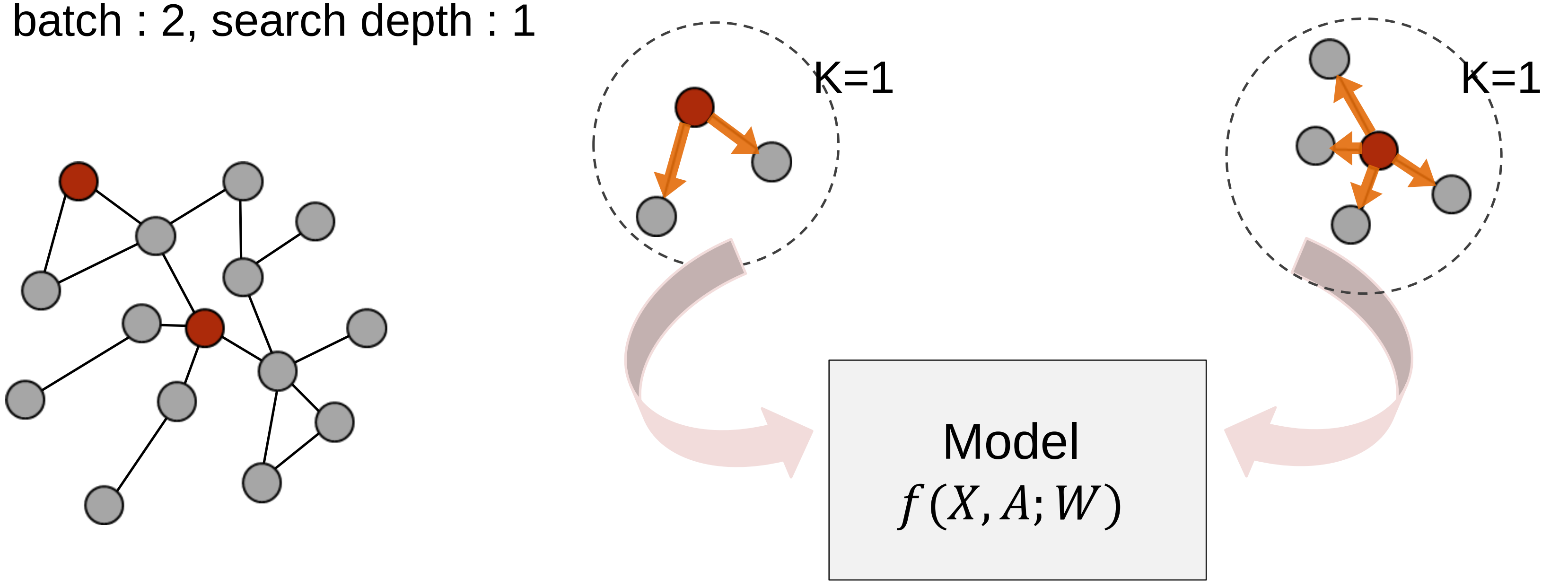


Inductive Representation Learning on Large Graphs

Graph-based semi-supervised learning (Inductive)

- ❖ GraphSAGE (SAmple and aggreGatE)
- ❖ Full batch → Mini batch
- ❖ Transductive → Inductive (Sampling)
- ❖ Average × → Aggregating (Mean / LSTM / Pooling aggregator)

Mini batch : 2, search depth : 1

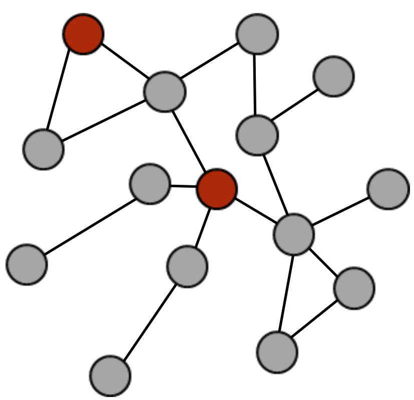


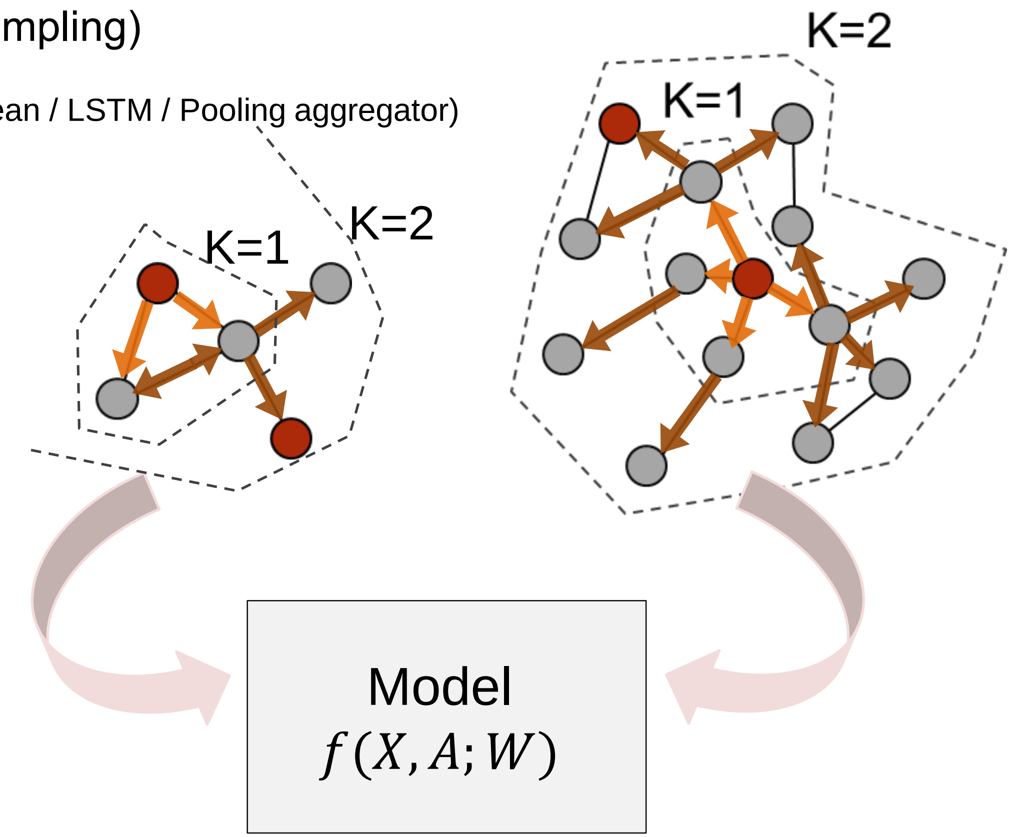
Inductive Representation Learning on Large Graphs

Graph-based semi-supervised learning (Inductive)

- GraphSAGE (SAmple and aggreGatE)
- Full batch → Mini batch
- Transductive → Inductive (Sampling)
- Average × → Aggregating (Mean / LSTM / Pooling aggregator)

Mini batch : 2, search depth : 2



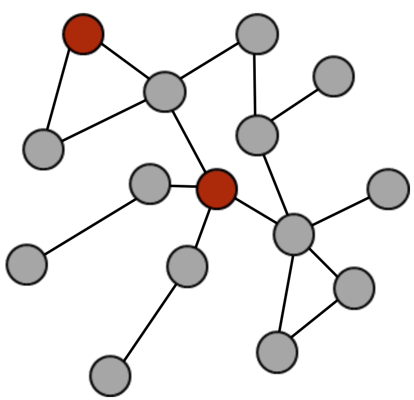


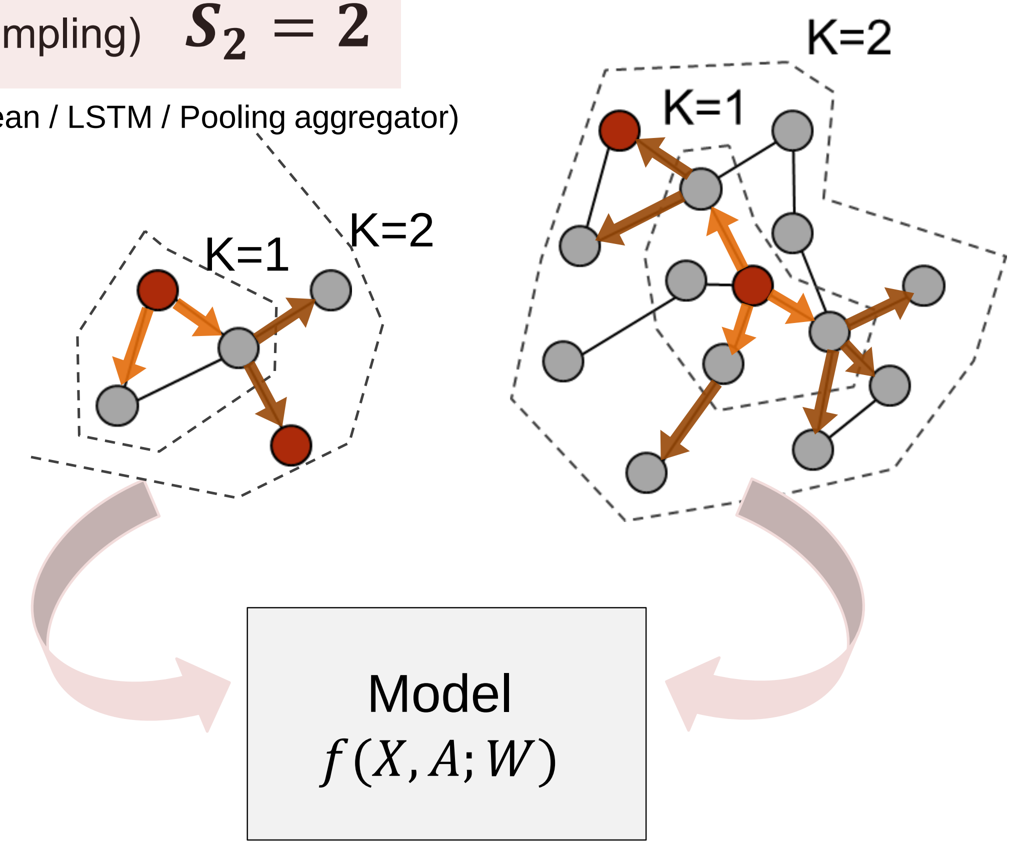
Inductive Representation Learning on Large Graphs

Graph-based semi-supervised learning (Inductive)

- ❖ GraphSAGE (SAmple and aggreGatE)
- ❖ Full batch → Mini batch $S_1 = 3$
- ❖ Transductive → Inductive (Sampling) $S_2 = 2$
- ❖ Average × → Aggregating (Mean / LSTM / Pooling aggregator)

Mini batch : 2, search depth : 2

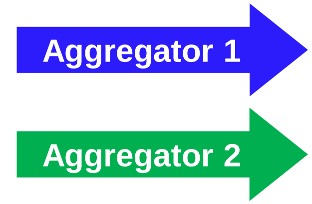




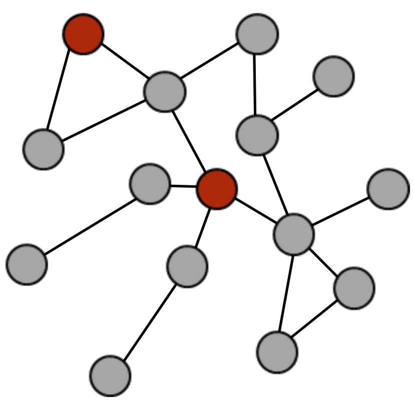
Inductive Representation Learning on Large Graphs

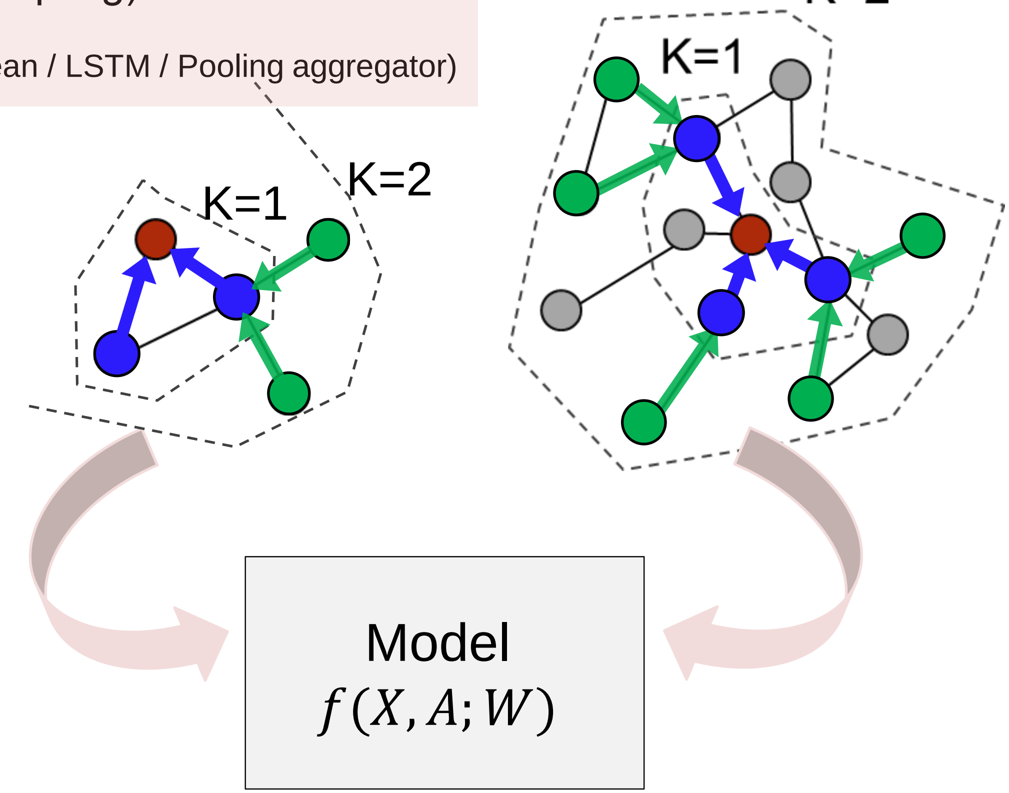
Graph-based semi-supervised learning (Inductive)

- ❖ GraphSAGE (SAmple and aggreGatE)
- ❖ Full batch → Mini batch
- ❖ Transductive → Inductive (Sampling)
- ❖ Average × → Aggregating (Mean / LSTM / Pooling aggregator)



Mini batch : 2, search depth : 2





Inductive Representation Learning on Large Graphs

Graph-based semi-supervised learning (Inductive)

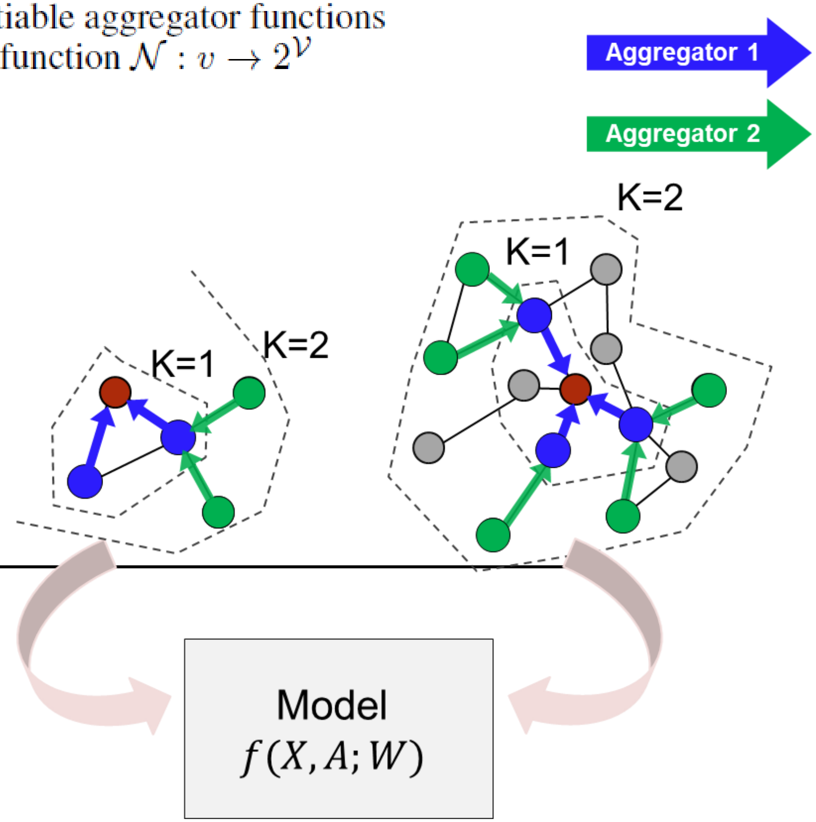
❖ GraphSAGE (SAmple and aggreGatE)

Algorithm 1: GraphSAGE embedding generation (i.e., forward propagation) algorithm

Input : Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$; input features $\{\mathbf{x}_v, \forall v \in \mathcal{V}\}$; depth K; weight matrices $\mathbf{W}^k, \forall k \in \{1, ..., K\}$; non-linearity σ; differentiable aggregator functions $\text{AGGREGATE}_k, \forall k \in \{1, ..., K\}$; neighborhood function $\mathcal{N} : v \rightarrow 2^{\mathcal{V}}$

Output: Vector representations $\mathbf{z}_v$ for all $v \in \mathcal{V}$

$\mathbf{h}_v^0 \leftarrow \mathbf{x}_v, \forall v \in \mathcal{V}$;
for $k = 1...K$ **do**
    for $v \in \mathcal{V}$ **do**
        $\mathbf{h}_{\mathcal{N}(v)}^k \leftarrow \text{AGGREGATE}_k(\{\mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v)\})$;
        $\mathbf{h}_v^k \leftarrow \sigma\left(\mathbf{W}^k \cdot \text{CONCAT}(\mathbf{h}_v^{k-1}, \mathbf{h}_{\mathcal{N}(v)}^k)\right)$
    end
    $\mathbf{h}_v^k \leftarrow \mathbf{h}_v^k / \|\mathbf{h}_v^k\|_2, \forall v \in \mathcal{V}$
end
$\mathbf{z}_v \leftarrow \mathbf{h}_v^K, \forall v \in \mathcal{V}$



Graph Attention Networks

Graph-based semi-supervised learning + Attention mechanism

398회 인용

Published as a conference paper at ICLR 2018

GRAPH ATTENTION NETWORKS

Petar Veličković*
Department of Computer Science and Technology
University of Cambridge
petar.velickovic@cst.cam.ac.uk

Guillem Cucurull*
Centre de Visió per Computador, UAB
gcucurull@gmail.com

Arantxa Casanova*
Centre de Visió per Computador, UAB
ar.casanova.8@gmail.com

Adriana Romero
Montréal Institute for Learning Algorithms
adriana.romero.soriano@umontreal.ca

Pietro Liò
Department of Computer Science and Technology
University of Cambridge
pietro.lio@cst.cam.ac.uk

Yoshua Bengio
Montréal Institute for Learning Algorithms
yoshua.umontreal@gmail.com

ABSTRACT

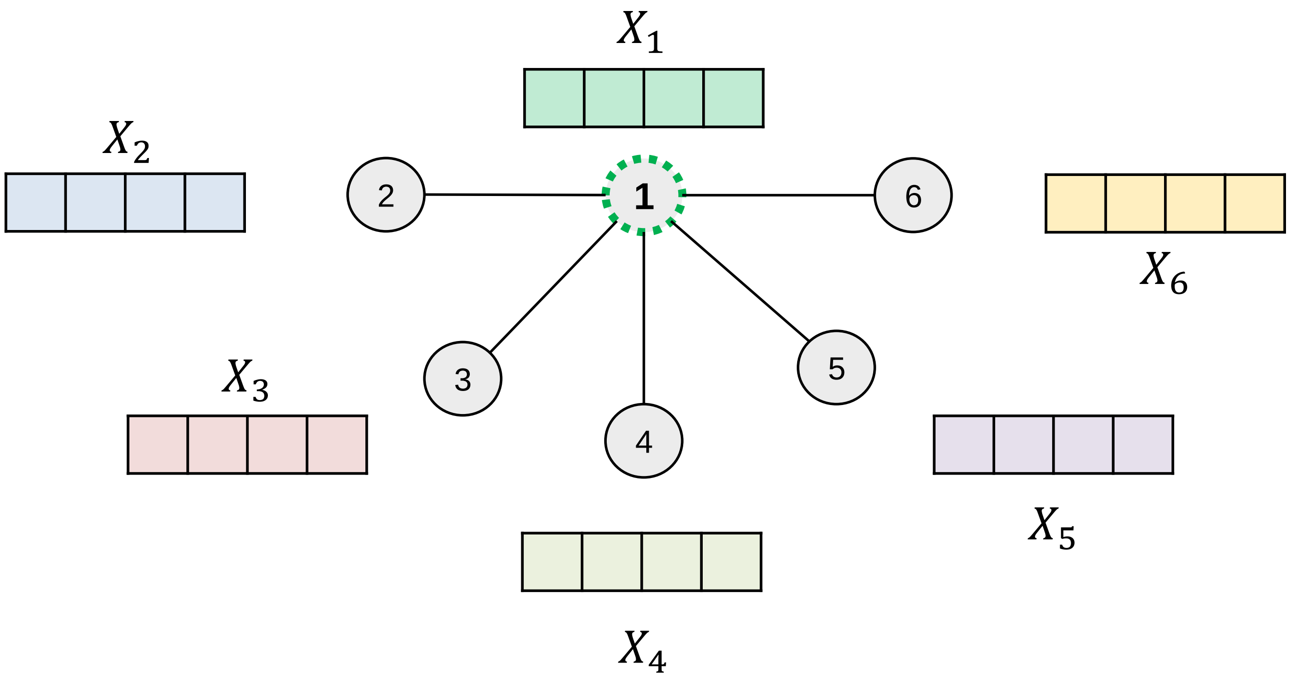
We present graph attention networks (GATs), novel neural network architectures that operate on graph-structured data, leveraging masked self-attentional layers to address the shortcomings of prior methods based on graph convolutions or their approximations. By stacking layers in which nodes are able to attend over their neighborhoods' features, we enable (implicitly) specifying different weights to different nodes in a neighborhood, without requiring any kind of costly matrix operation (such as inversion) or depending on knowing the graph structure upfront. In this way, we address several key challenges of spectral-based graph neural networks simultaneously, and make our model readily applicable to inductive as well as transductive problems. Our GAT models have achieved or matched state-of-the-art results across four established transductive and inductive graph benchmarks: the *Cora*, *Citeseer* and *Pubmed* citation network datasets, as well as a *protein-protein interaction* dataset (wherein test graphs remain unseen during training).

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2017). Graph attention networks. arXiv preprint arXiv:1710.10903.

Graph Attention Networks

Graph-based semi-supervised learning + Attention mechanism

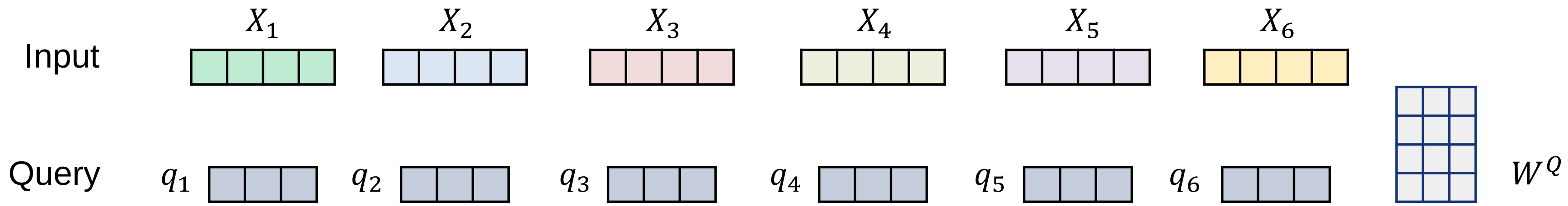
- ❖ Graph Attention Networks (GATs)
- ❖ Masked Self-attention + Multi-head



Graph Attention Networks

Graph-based semi-supervised learning + Attention mechanism

- ❖ Graph Attention Networks (GATs)
- ❖ Masked Self-attention + Multi-head

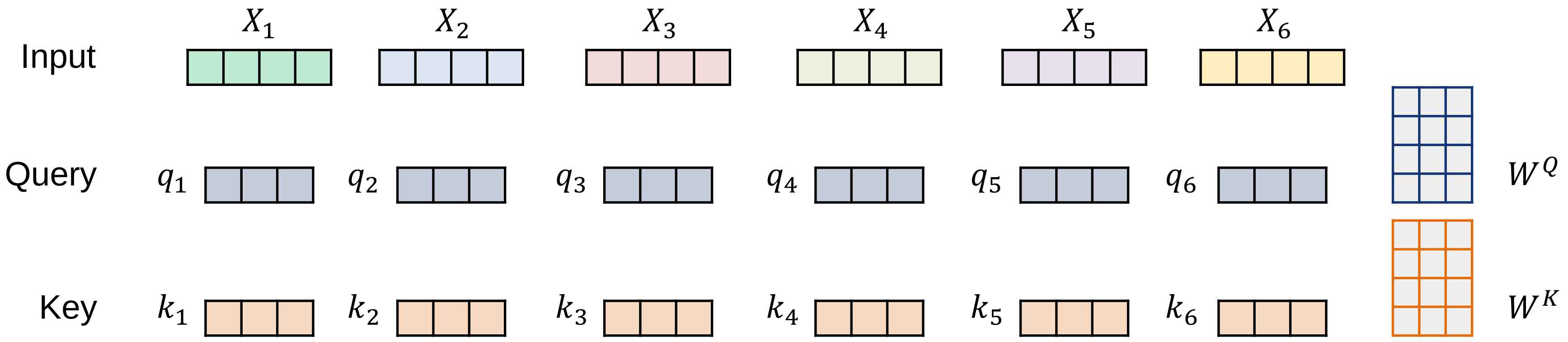


http://jalammar.github.io/illustrated-transformer/

Graph Attention Networks

Graph-based semi-supervised learning + Attention mechanism

- Graph Attention Networks (GATs)
- Masked Self-attention + Multi-head

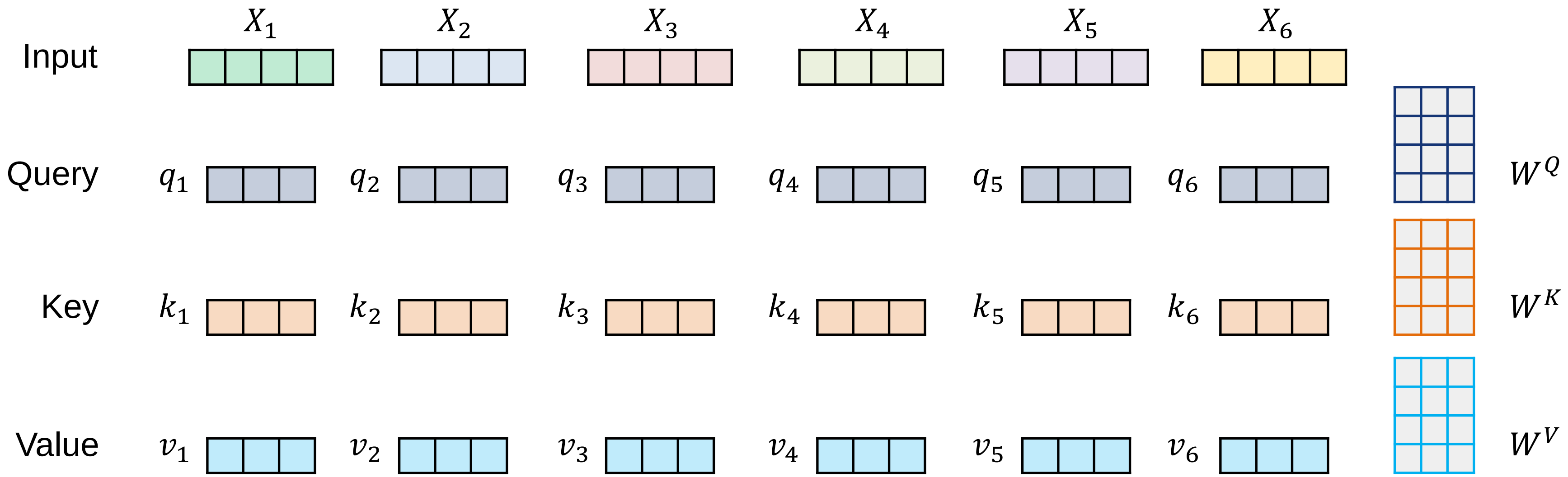


http://jalammar.github.io/illustrated-transformer/

Graph Attention Networks

Graph-based semi-supervised learning + Attention mechanism

- ❖ Graph Attention Networks (GATs)
- ❖ Masked Self-attention + Multi-head

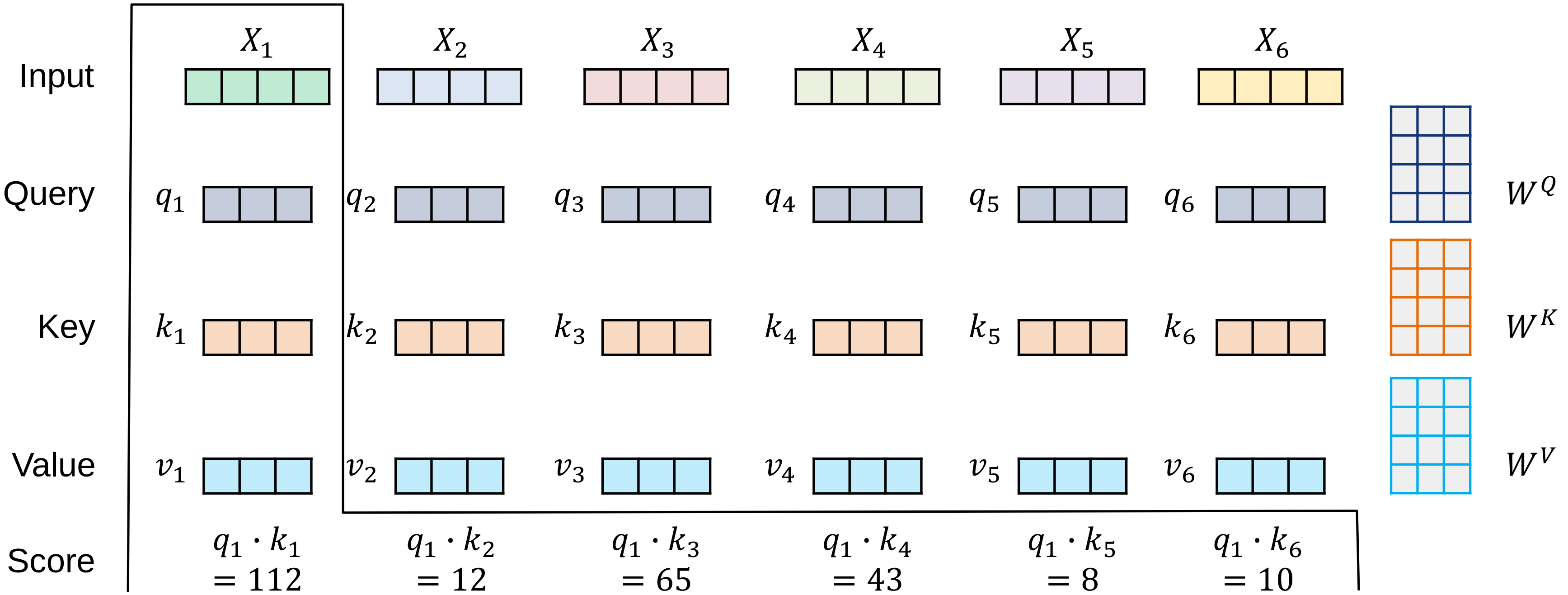


http://jalammar.github.io/illustrated-transformer/

Graph Attention Networks

Graph-based semi-supervised learning + Attention mechanism

- ❖ Graph Attention Networks (GATs)
- ❖ Masked Self-attention + Multi-head

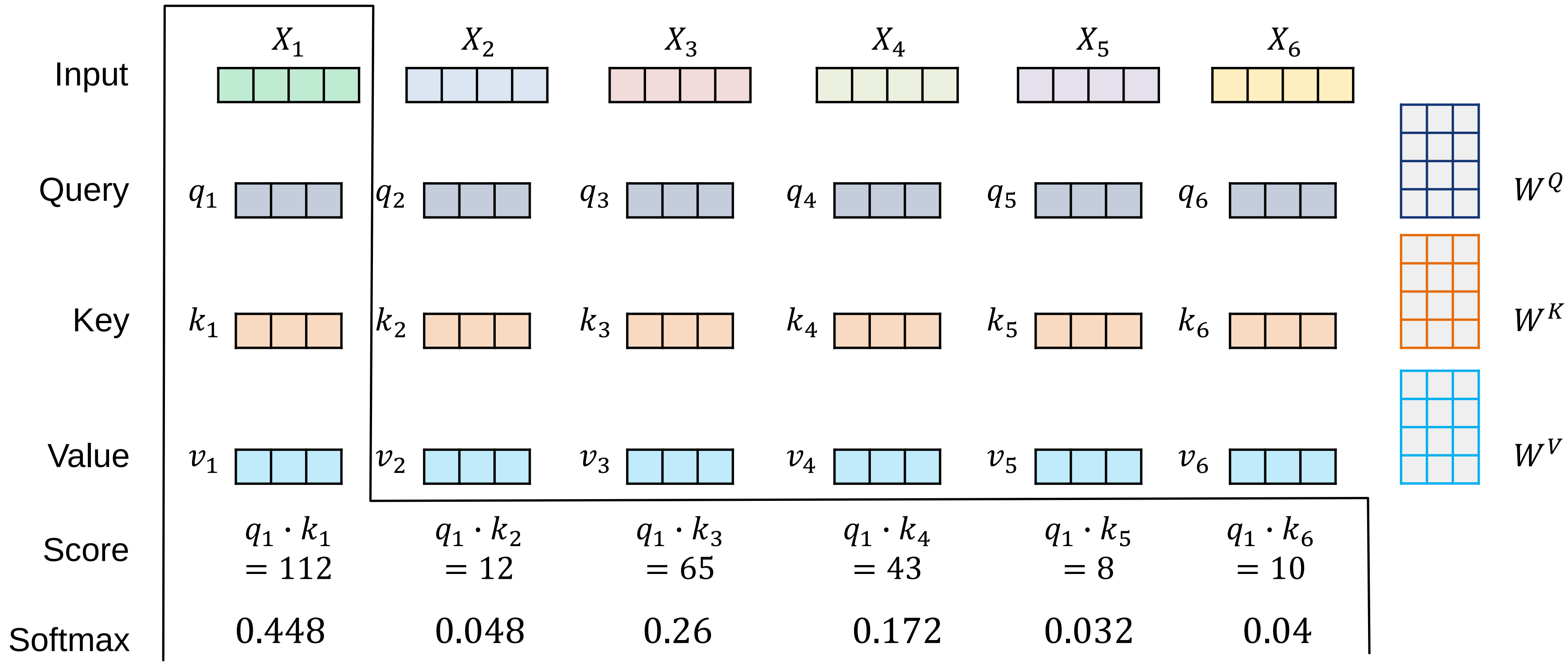


http://jalammar.github.io/illustrated-transformer/

Graph Attention Networks

Graph-based semi-supervised learning + Attention mechanism

- Graph Attention Networks (GATs)
- Masked Self-attention + Multi-head

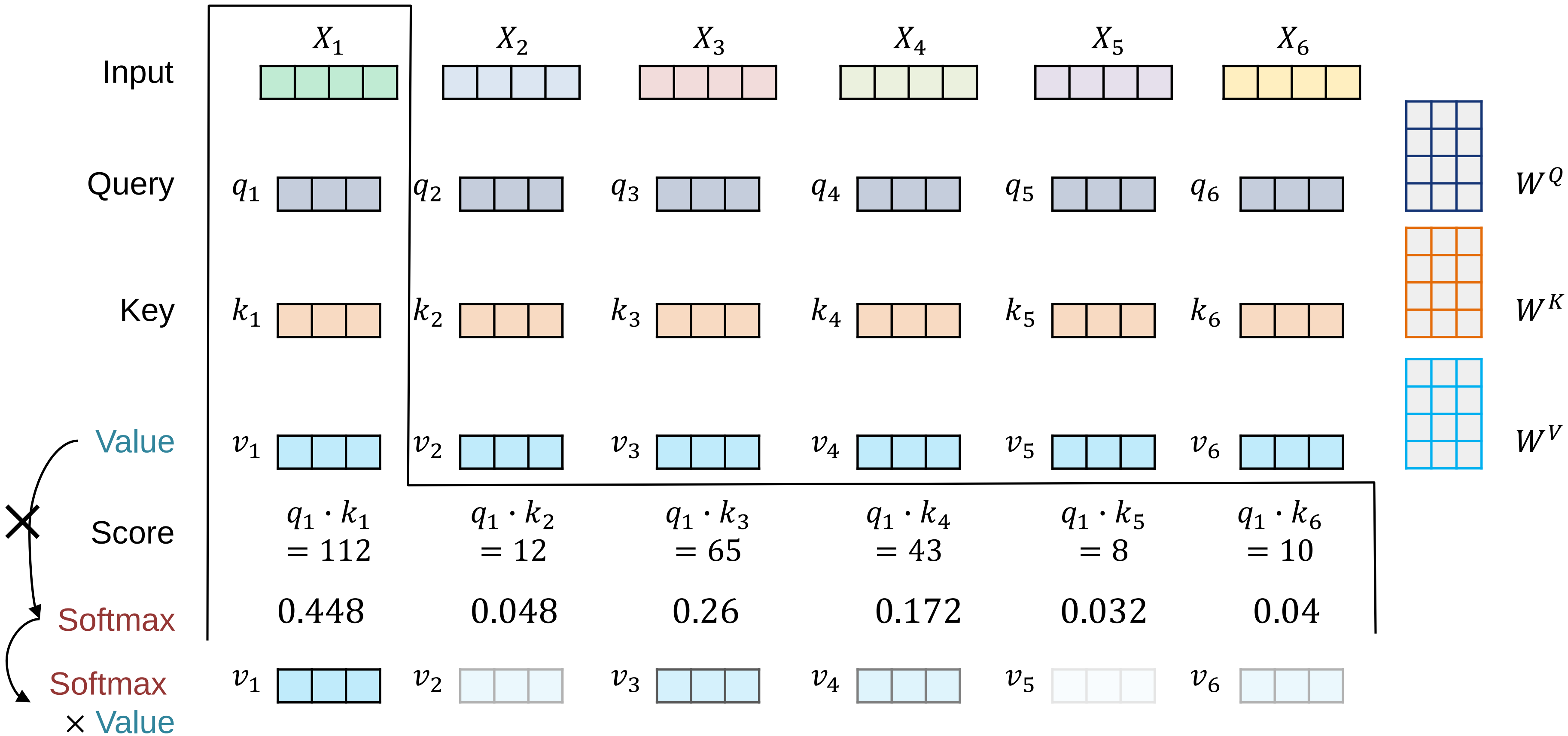


http://jalammar.github.io/illustrated-transformer/

Graph Attention Networks

Graph-based semi-supervised learning + Attention mechanism

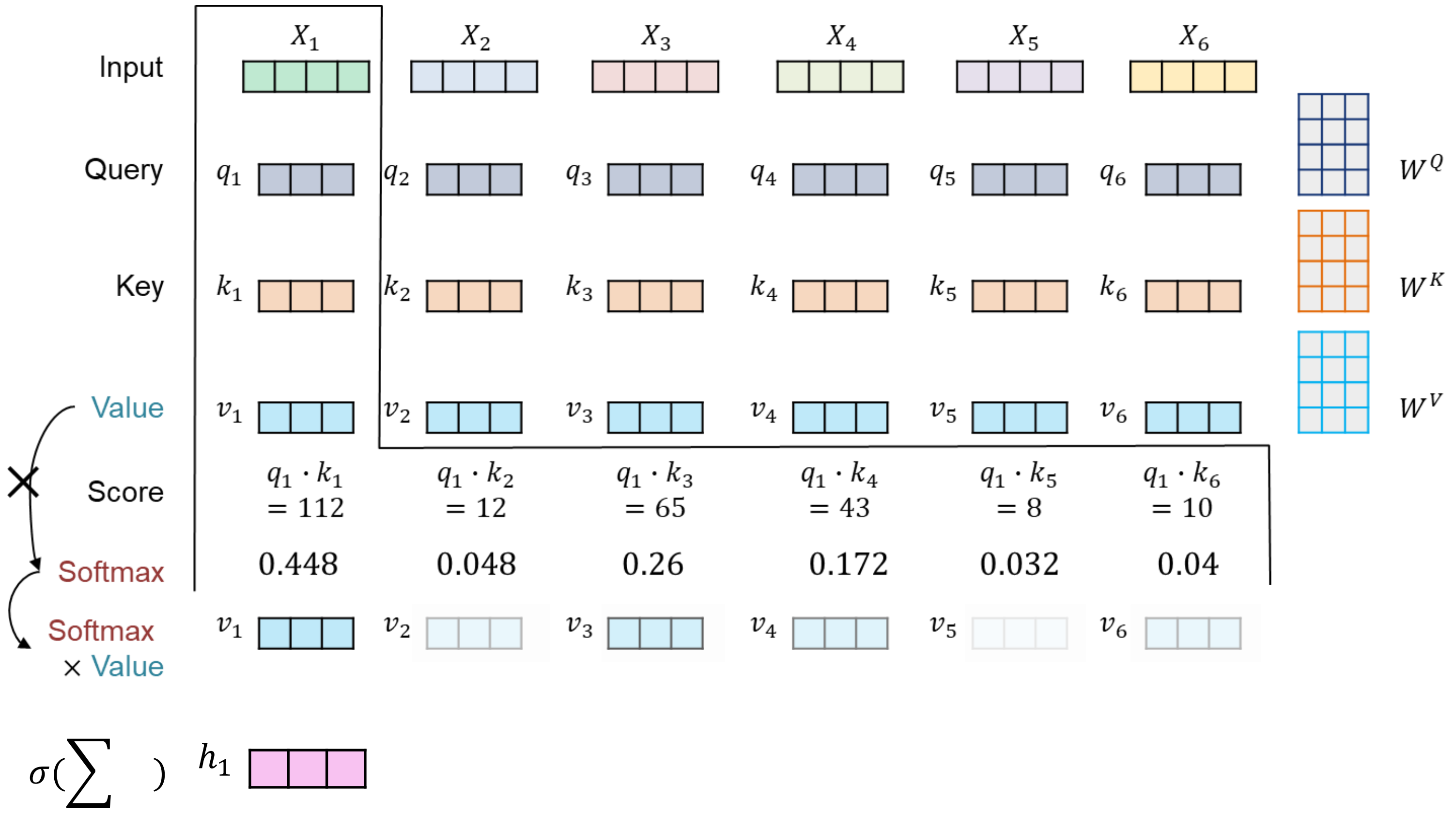
- ❖ Graph Attention Networks (GATs)
- ❖ Masked Self-attention + Multi-head



Graph Attention Networks

Graph-based semi-supervised learning + Attention mechanism

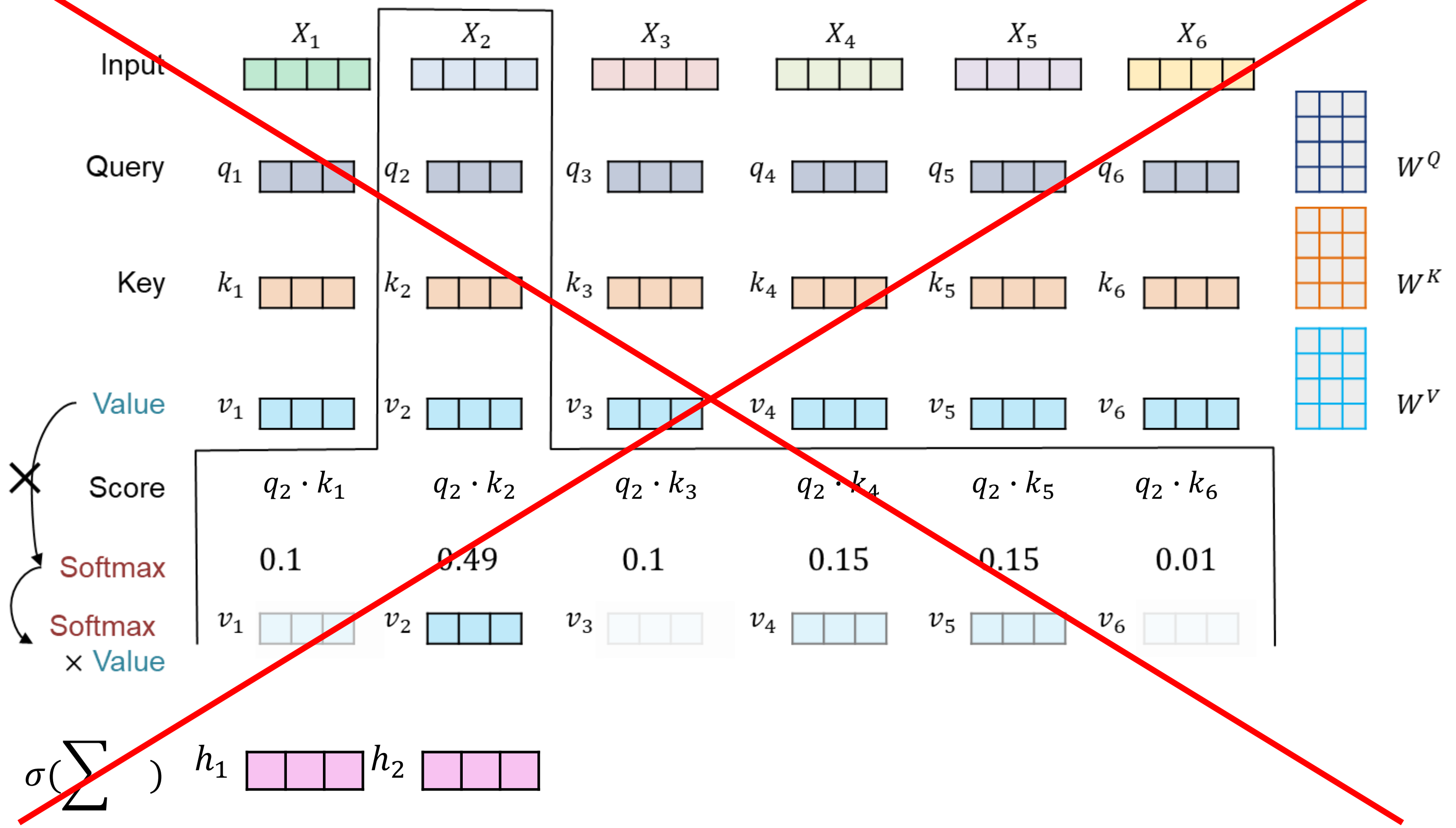
- ❖ Graph Attention Networks (GATs)
- ❖ Masked Self-attention + Multi-head



Graph Attention Networks

Graph-based semi-supervised learning + Attention mechanism

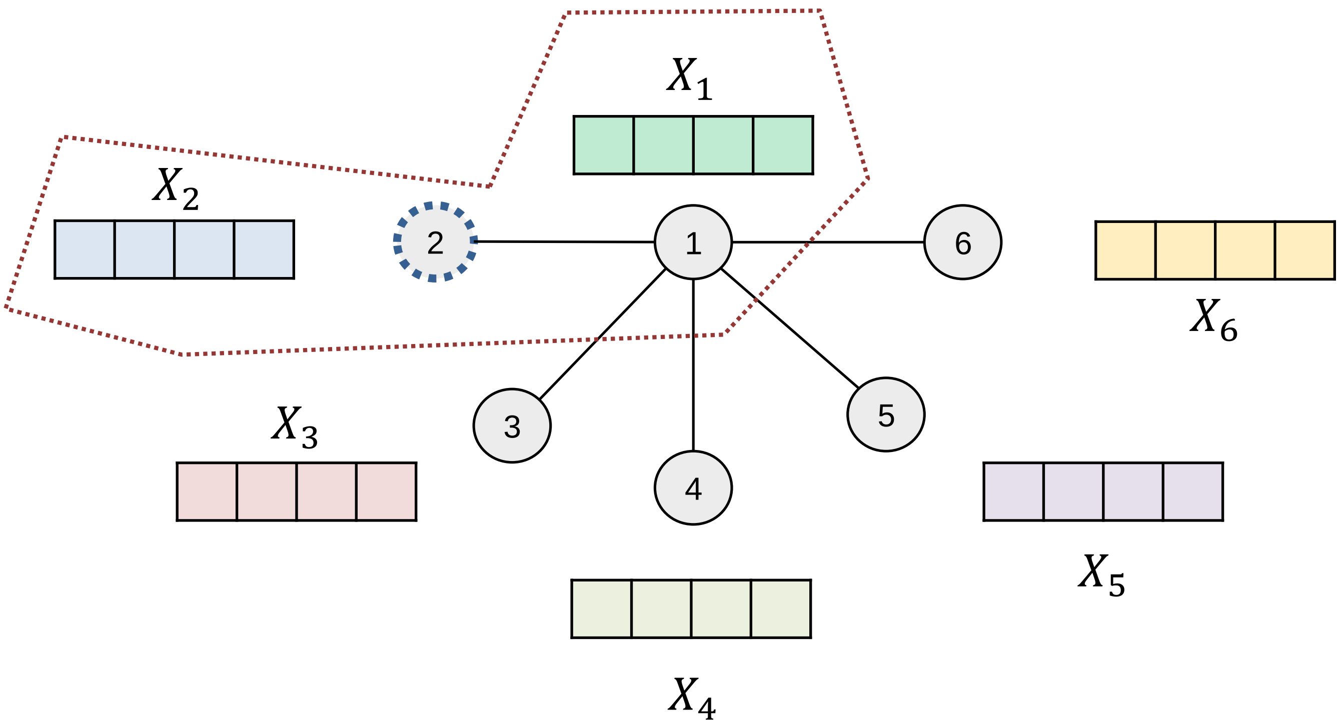
- ❖ Graph Attention Networks (GATs)
- ❖ Masked Self-attention + Multi-head



Graph Attention Networks

Graph-based semi-supervised learning + Attention mechanism

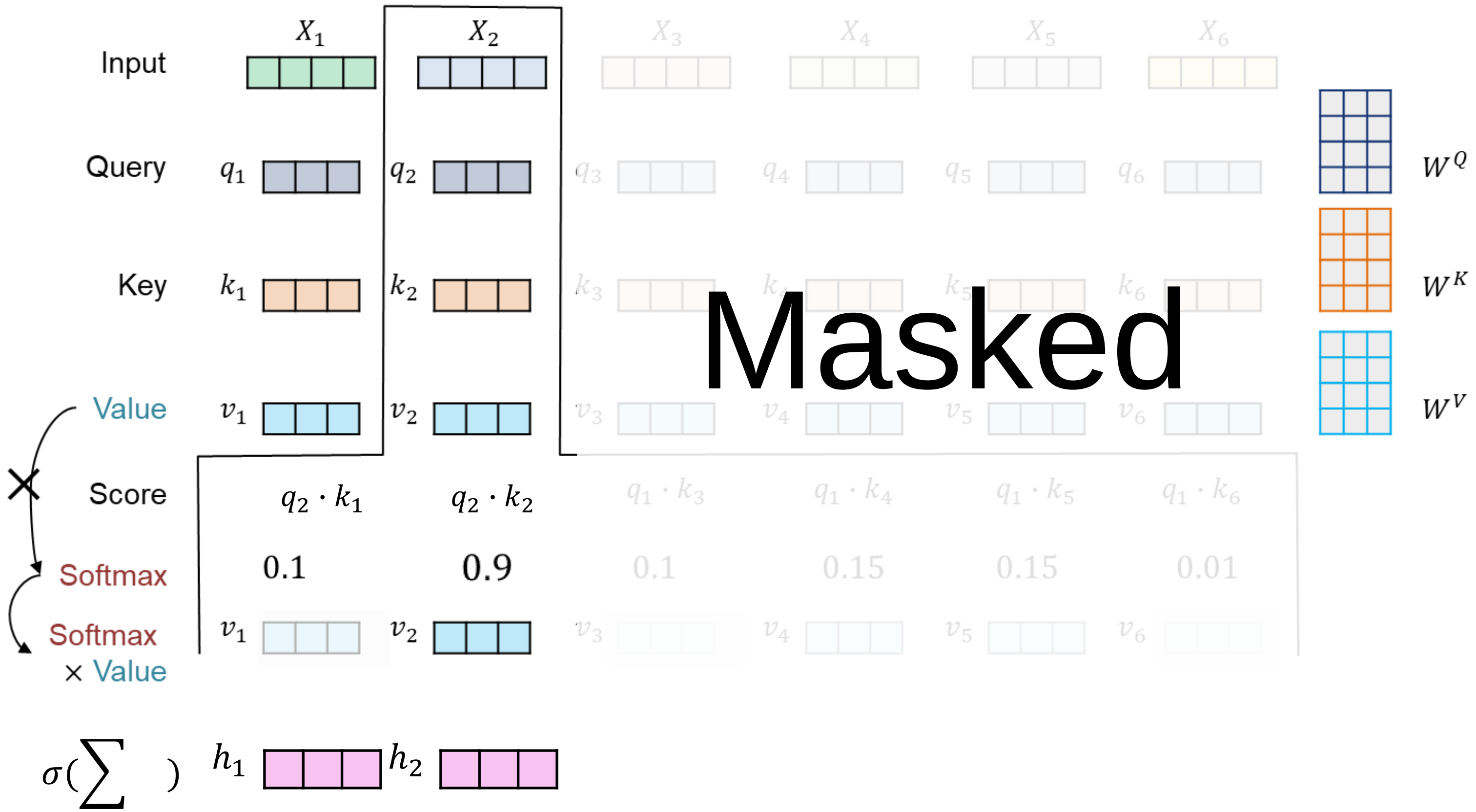
- ❖ Graph Attention Networks (GATs)
- ❖ **Masked** Self-attention + Multi-head



Graph Attention Networks

Graph-based semi-supervised learning + Attention mechanism

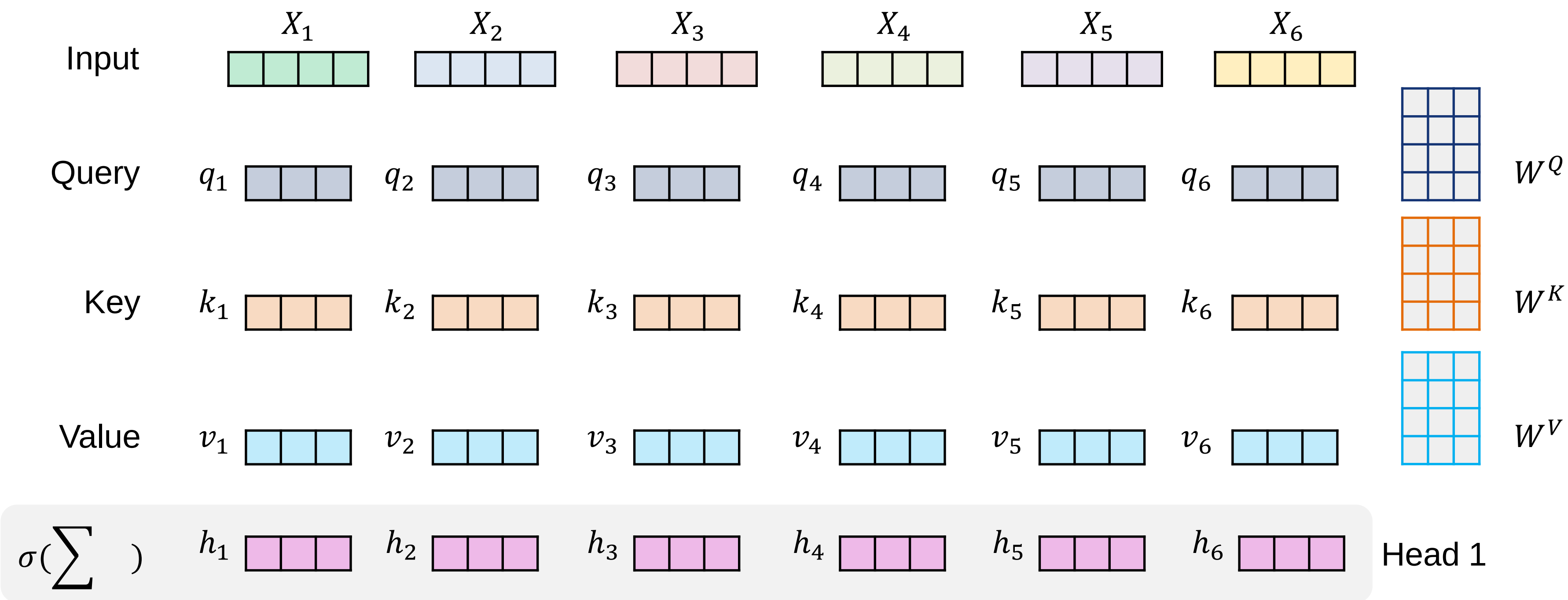
- ❖ Graph Attention Networks (GATs)
- ❖ **Masked** Self-attention + Multi-head



Graph Attention Networks

Graph-based semi-supervised learning + Attention mechanism

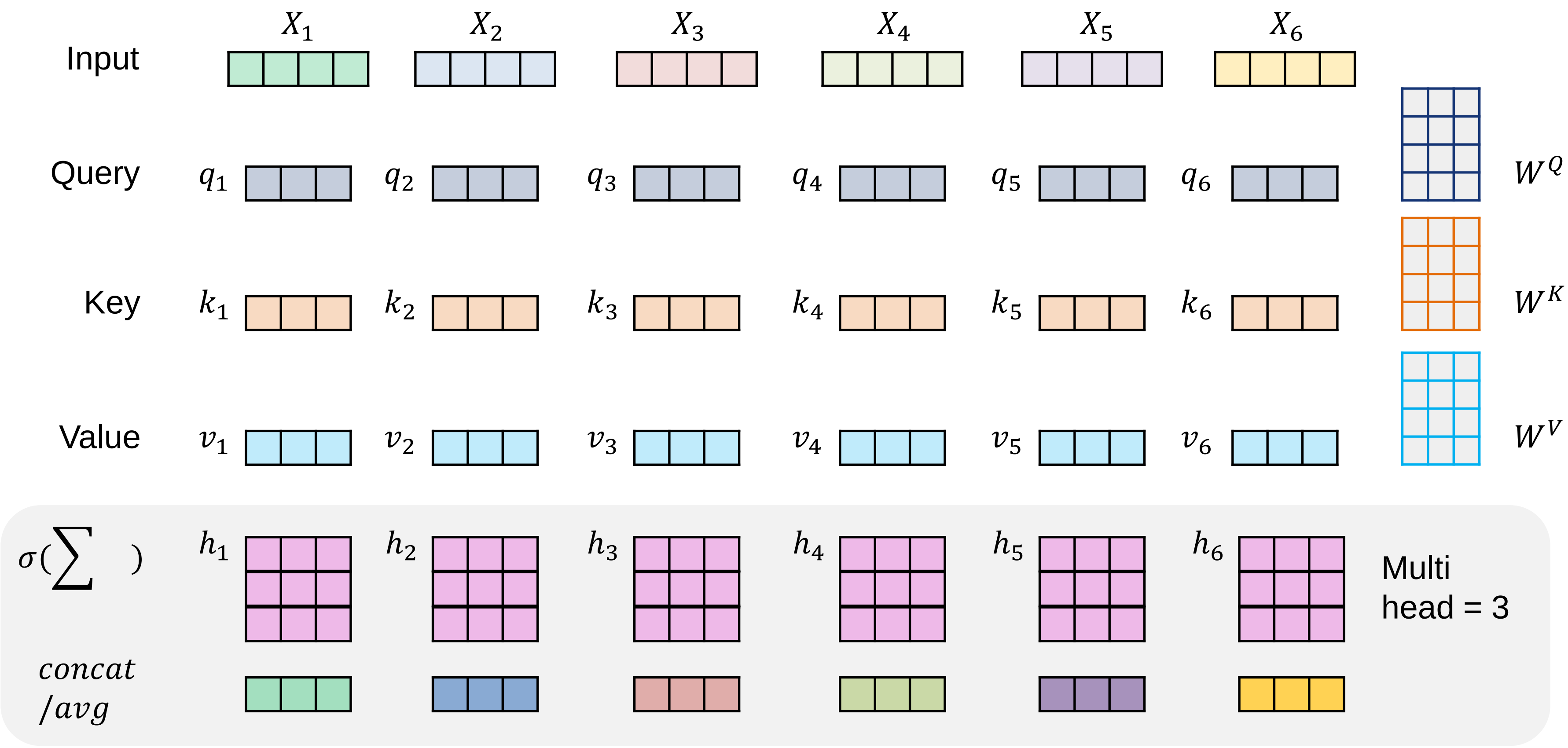
- Graph Attention Networks (GATs)
- **Masked** Self-attention + Multi-head



Graph Attention Networks

Graph-based semi-supervised learning + Attention mechanism

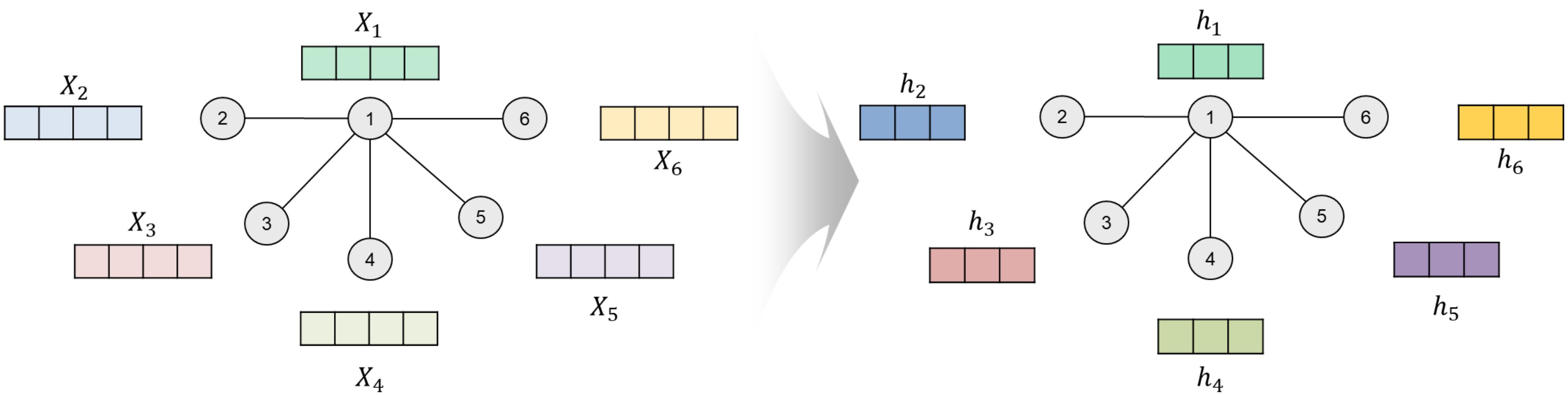
- Graph Attention Networks (GATs)
- Masked Self-attention **+ Multi-head**



Graph Attention Networks

Graph-based semi-supervised learning + Attention mechanism

- ❖ Graph Attention Networks (GATs)
- ❖ Masked Self-attention + Multi-head



Graph Attention Networks

Graph-based semi-supervised learning + Attention mechanism

❖ Evaluation

Table 1: Summary of the datasets used in our experiments.

	Cora	**Citeseer**	**Pubmed**	**PPI**
Task	Transductive	Transductive	Transductive	Inductive
# Nodes	2708 (1 graph)	3327 (1 graph)	19717 (1 graph)	56944 (24 graphs)
# Edges	5429	4732	44338	818716
# Features/Node	1433	3703	500	50
# Classes	7	6	3	121 (multilabel)
# Training Nodes	140	120	60	44906 (20 graphs)
# Validation Nodes	500	500	500	6514 (2 graphs)
# Test Nodes	1000	1000	1000	5524 (2 graphs)

Graph Attention Networks

Graph-based semi-supervised learning + Attention mechanism

- Evaluation

Table 2: Summary of results in terms of classification accuracies, for Cora, Citeseer and Pubmed. GCN-64* corresponds to the best GCN result computing 64 hidden features (using ReLU or ELU).

Transductive

Method	**Cora**	**Citeseer**	**Pubmed**
MLP	55.1%	46.5%	71.4%
ManiReg (Belkin et al., 2006)	59.5%	60.1%	70.7%
SemiEmb (Weston et al., 2012)	59.0%	59.6%	71.7%
LP (Zhu et al., 2003)	68.0%	45.3%	63.0%
DeepWalk (Perozzi et al., 2014)	67.2%	43.2%	65.3%
ICA (Lu & Getoor, 2003)	75.1%	69.1%	73.9%
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**%
MoNet (Monti et al., 2016)	$81.7 \pm 0.5\%$	—	$78.8 \pm 0.3\%$
GCN-64*	$81.4 \pm 0.5\%$	$70.9 \pm 0.5\%$	$\mathbf{79.0} \pm 0.3\%$
GAT (ours)	$\mathbf{83.0} \pm 0.7\%$	$\mathbf{72.5} \pm 0.7\%$	$\mathbf{79.0} \pm 0.3\%$

Graph Attention Networks

Graph-based semi-supervised learning + Attention mechanism

❖ Evaluation

Table 3: Summary of results in terms of micro-averaged F_1 scores, for the PPI dataset. GraphSAGE* corresponds to the best GraphSAGE result we were able to obtain by just modifying its architecture. Const-GAT corresponds to a model with the same architecture as GAT, but with a constant attention mechanism (assigning same importance to each neighbor; GCN-like inductive operator).

Inductive

Method	**PPI**
Random	0.396
MLP	0.422
GraphSAGE-GCN (Hamilton et al., 2017)	0.500
GraphSAGE-mean (Hamilton et al., 2017)	0.598
GraphSAGE-LSTM (Hamilton et al., 2017)	0.612
GraphSAGE-pool (Hamilton et al., 2017)	0.600
GraphSAGE*	0.768
Const-GAT (ours)	0.934 ± 0.006
GAT (ours)	$\mathbf{0.973} \pm 0.002$


Q
&
A

Appendix. Inductive Representation Learning on Large Graphs

❖ GraphSAGE (SAmple and aggreGatE)

Algorithm 2: GraphSAGE minibatch forward propagation algorithm

Input : Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$;
input features $\{\mathbf{x}_v, \forall v \in \mathcal{B}\}$;
depth K; weight matrices $\mathbf{W}^k, \forall k \in \{1, ..., K\}$;
non-linearity σ;
differentiable aggregator functions $\text{AGGREGATE}_k, \forall k \in \{1, ..., K\}$;
neighborhood sampling functions, $\mathcal{N}_k : v \rightarrow 2^{\mathcal{V}}, \forall k \in \{1, ..., K\}$

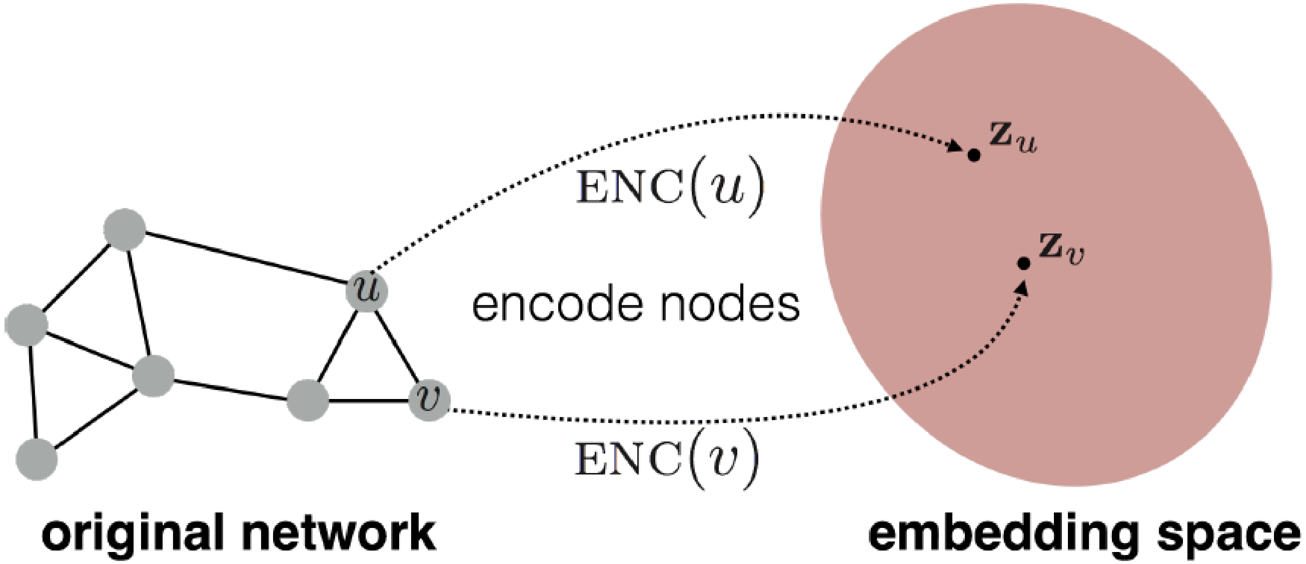
Output : Vector representations $\mathbf{z}_v$ for all $v \in \mathcal{B}$

$\mathcal{B}^K \leftarrow \mathcal{B}$;
for $k = K...1$ **do**
  $\mathcal{B}^{k-1} \leftarrow \mathcal{B}^k$;
  for $u \in \mathcal{B}^k$ **do**
    $\mathcal{B}^{k-1} \leftarrow \mathcal{B}^{k-1} \cup \mathcal{N}_k(u)$;
  end
end
$\mathbf{h}_u^0 \leftarrow \mathbf{x}_v, \forall v \in \mathcal{B}^0$;
for $k = 1...K$ **do**
  for $u \in \mathcal{B}^k$ **do**
    $\mathbf{h}_{\mathcal{N}(u)}^k \leftarrow \text{AGGREGATE}_k(\{\mathbf{h}_{u'}^{k-1}, \forall u' \in \mathcal{N}_k(u)\})$;
    $\mathbf{h}_u^k \leftarrow \sigma\left(\mathbf{W}^k \cdot \text{CONCAT}(\mathbf{h}_u^{k-1}, \mathbf{h}_{\mathcal{N}(u)}^k)\right)$;
    $\mathbf{h}_u^k \leftarrow \mathbf{h}_u^k / \|\mathbf{h}_u^k\|_2$;
  end
end
$\mathbf{z}_u \leftarrow \mathbf{h}_u^K, \forall u \in \mathcal{B}$

Appendix. Inductive Representation Learning on Large Graphs

❖ GraphSAGE (SAmple and aggreGatE)

Goal is to encode nodes so that similarity in the embedding space (e.g., dot product) approximates similarity in the original network.



http://snap.stanford.edu/proj/embeddings-www/

Appendix. Inductive Representation Learning on Large Graphs

❖ GraphSAGE (SAmple and aggreGatE)

3.2 Learning the parameters of GraphSAGE

In order to learn useful, predictive representations in a fully unsupervised setting, we apply a graph-based loss function to the output representations, $\mathbf{z}_u, \forall u \in \mathcal{V}$, and tune the weight matrices, $\mathbf{W}^k, \forall k \in \{1, ..., K\}$, and parameters of the aggregator functions via stochastic gradient descent. The graph-based loss function encourages nearby nodes to have similar representations, while enforcing that the representations of disparate nodes are highly distinct:

$$J_{\mathcal{G}}(\mathbf{z}_u) = -\log\left(\sigma(\mathbf{z}_u^\top \mathbf{z}_v)\right) - Q \cdot \mathbb{E}_{v_n \sim P_n(v)} \log\left(\sigma(-\mathbf{z}_u^\top \mathbf{z}_{v_n})\right), \quad (1)$$

where v is a node that co-occurs near u on fixed-length random walk, σ is the sigmoid function, P_n is a negative sampling distribution, and Q defines the number of negative samples. Importantly, unlike previous embedding approaches, the representations $\mathbf{z}_u$ that we feed into this loss function are generated from the features contained within a node's local neighborhood, rather than training a unique embedding for each node (via an embedding look-up).

This unsupervised setting emulates situations where node features are provided to downstream machine learning applications, as a service or in a static repository. In cases where representations are to be used only on a specific downstream task, the unsupervised loss (Equation 1) can simply be replaced, or augmented, by a task-specific objective (e.g., cross-entropy loss).

Appendix. Graph Attention Networks

❖ Experimental Setup

3.3 EXPERIMENTAL SETUP

Transductive learning For the transductive learning tasks, we apply a two-layer GAT model. Its architectural hyperparameters have been optimized on the Cora dataset and are then reused for Citeseer. The first layer consists of $K = 8$ attention heads computing $F' = 8$ features each (for a total of 64 features), followed by an exponential linear unit (ELU) (Clevert et al., 2016) nonlinearity. The second layer is used for classification: a single attention head that computes C features (where C is the number of classes), followed by a softmax activation. For coping with the small training set sizes, regularization is liberally applied within the model. During training, we apply L_2 regularization with $\lambda = 0.0005$. Furthermore, dropout (Srivastava et al., 2014) with $p = 0.6$ is applied to both layers' inputs, as well as *to the normalized attention coefficients* (critically, this means that at each training iteration, each node is exposed to a stochastically sampled neighborhood). Similarly as observed by Monti et al. (2016), we found that Pubmed's training set size (60 examples) required slight changes to the GAT architecture: we have applied $K = 8$ output attention heads (instead of one), and strengthened the L_2 regularization to $\lambda = 0.001$. Otherwise, the architecture matches the one used for Cora and Citeseer.

Appendix. Graph Attention Networks

❖ Experimental Setup

Inductive learning For the inductive learning task, we apply a three-layer GAT model. Both of the first two layers consist of $K = 4$ attention heads computing $F' = 256$ features (for a total of 1024 features), followed by an ELU nonlinearity. The final layer is used for (multi-label) classification: $K = 6$ attention heads computing 121 features each, that are averaged and followed by a logistic sigmoid activation. The training sets for this task are sufficiently large and we found no need to apply L_2 regularization or dropout—we have, however, successfully employed skip connections (He et al., 2016) across the intermediate attentional layer. We utilize a batch size of 2 graphs during training. To strictly evaluate the benefits of applying an attention mechanism in this setting (i.e. comparing with a near GCN-equivalent model), we also provide the results when a *constant attention mechanism*, $a(x, y) = 1$, is used, with the same architecture—this will assign the same weight to every neighbor.

Both models are initialized using Glorot initialization (Glorot & Bengio, 2010) and trained to minimize cross-entropy on the training nodes using the Adam SGD optimizer (Kingma & Ba, 2014) with an initial learning rate of 0.01 for Pubmed, and 0.005 for all other datasets. In both cases we use an early stopping strategy on both the cross-entropy loss and accuracy (transductive) or micro-F_1 (inductive) score on the validation nodes, with a patience of 100 epochs[1].