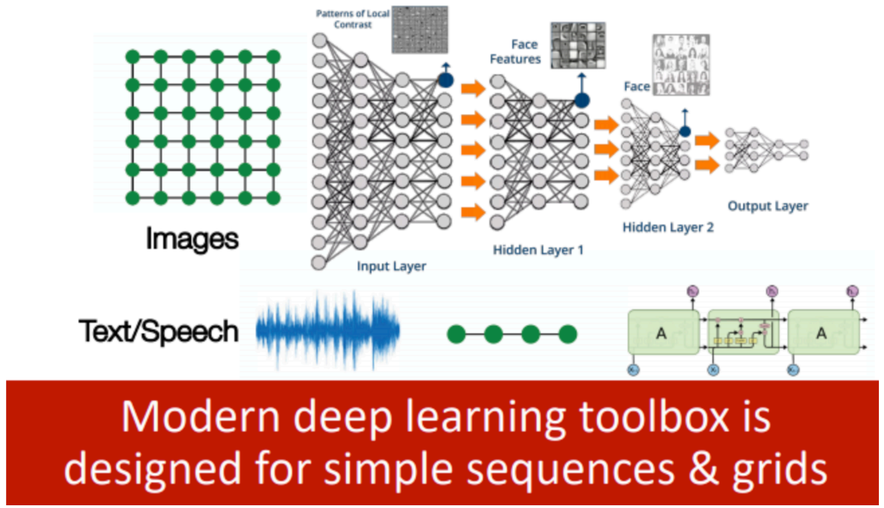
Graph Attention Network

Jaeyoung Cheong

0. Review

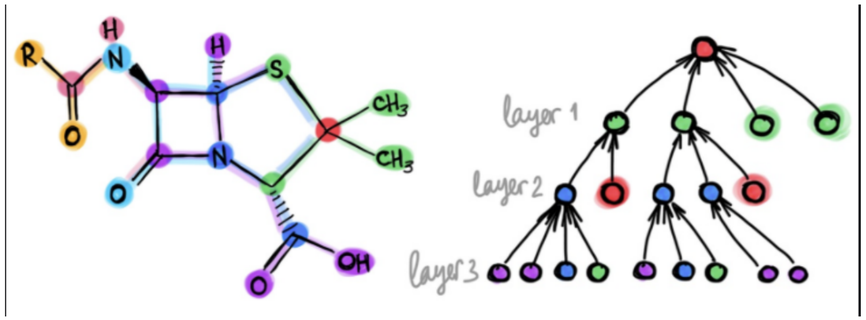
□ 왜 GNN이 요구되는가?

⇒ 기존의 딥러닝 or 머신러닝 기법은 유클리드 공간 위에서 표현된 데이터를 처리하는 데 용이



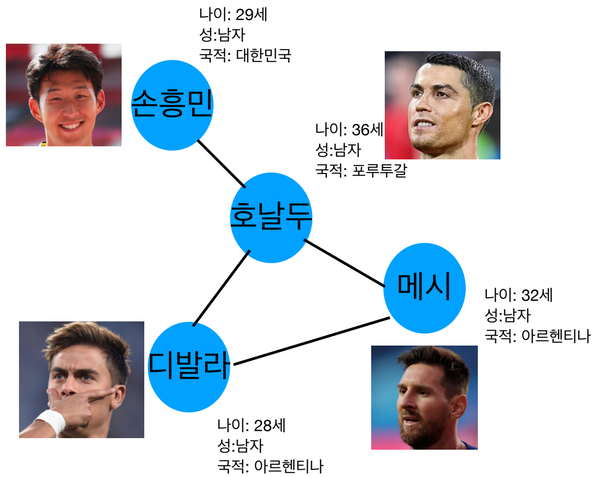
⇒ 하지만 이들은 비유클리드 공간 위에서 표현된 데이터(e.g. 그래프)를 처리하는 데 부적합

⇒ 그래프 데이터를 처리하는 데 특화된 Graph Neural Network를 이용해보자!



0. Review

A review of GCN



	Feature1	Feature2	Feature3	
son	0	1	0	대한민국
Ronaldo	0	1	1	포르투갈
Messi	1	1	1	
Dybala	1	1	1	둘다 아르헨티나

국적을 나타내는 Features라고 해보자

N(node) x F(features)의 **Node Feature Matrix** X_{ij}

	son	Ronaldo	Messi	Dybala
son	0	1	0	0
Ronaldo	1	0	1	1
Messi	0	1	0	1
Dybala	0	1	1	0

N x N(node)의 **Adjacent Matrix** A_{ij}

	son	Ronaldo	Messi	Dybala
son	1	0	0	0
Ronaldo	0	3	0	0
Messi	0	0	2	0
Dybala	0	0	0	2

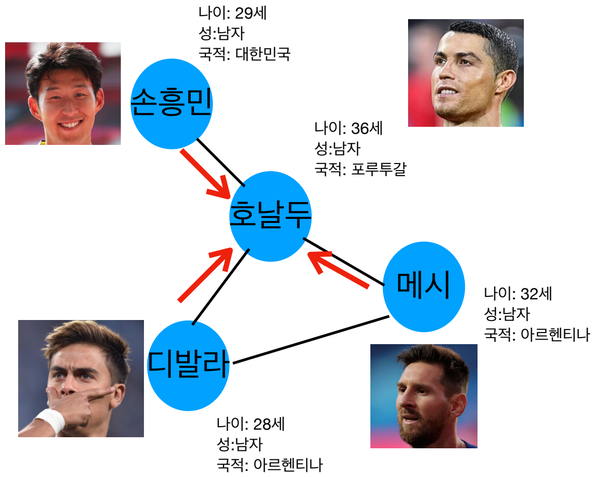
D Degree matrix

참고: [#30.Lec] Basic of Graph Convolution Network - 딥러닝 홀로서기

0. Review

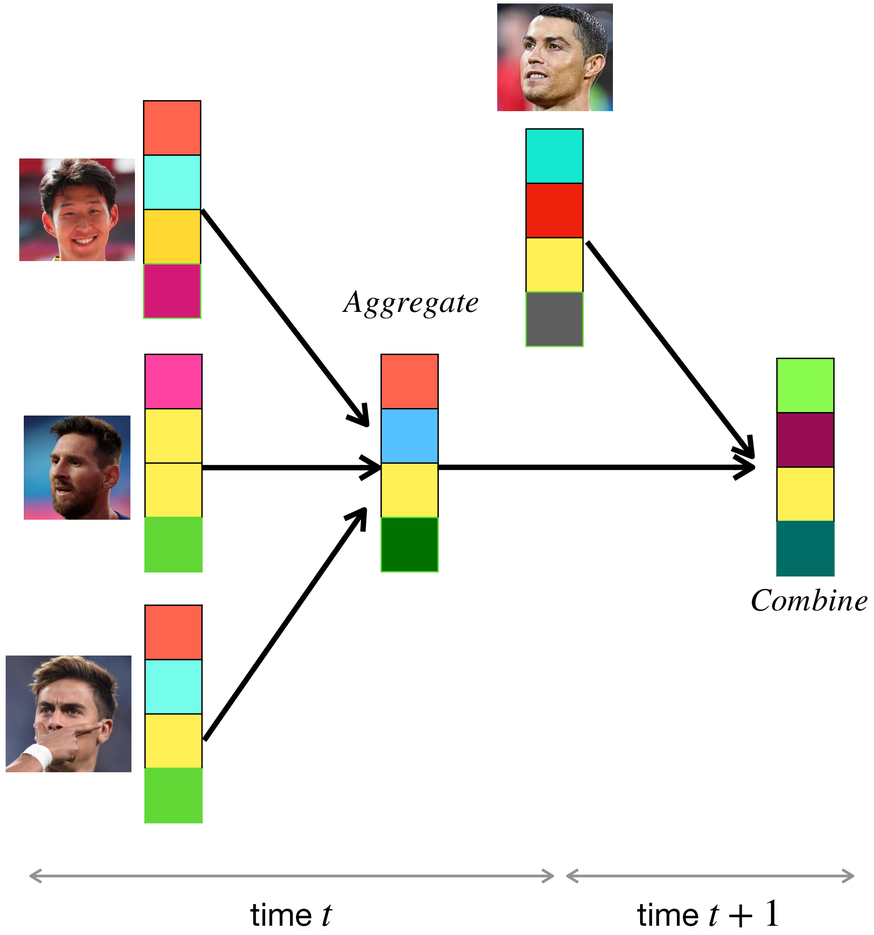
A summary of terms

Aggregate (Message Passing) → **Combine (Update)**



Target Node: C.Ronaldo
Neighbors: Son, Dybala, Messi

Aggregate: Neighbors'info ⇒ Target node at time t



0. Review

A summary of terms

Name	Variant	Aggregator	Updater
Spectral Methods	ChebNet	$\mathbf{N}_k = \mathbf{T}_k(\tilde{\mathbf{L}})\mathbf{X}$	$\mathbf{H} = \sum_{k=0}^{K} \mathbf{N}_k \mathbf{\Theta}_k$
	1^{st}-order model	$\mathbf{N}_0 = \mathbf{X}$ $\mathbf{N}_1 = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\mathbf{X}$	$\mathbf{H} = \mathbf{N}_0\mathbf{\Theta}_0 + \mathbf{N}_1\mathbf{\Theta}_1$
	Single parameter	$\mathbf{N} = (\mathbf{I}_N + \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}})\mathbf{X}$	$\mathbf{H} = \mathbf{N}\mathbf{\Theta}$
	GCN	$\mathbf{N} = \tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{X}$	$\mathbf{H} = \mathbf{N}\mathbf{\Theta}$
Non-spectral Methods	Neural FPs	$\mathbf{h}^t_{\mathcal{N}_v} = \mathbf{h}^{t-1}_v + \sum_{k=1}^{\mathcal{N}_v} \mathbf{h}^{t-1}_k$	$\mathbf{h}^t_v = \sigma(\mathbf{h}^t_{\mathcal{N}_v}\mathbf{W}^{\mathcal{N}_v}_L)$
	DCNN	Node classification: $\mathbf{N} = \mathbf{P}^*\mathbf{X}$ Graph classification: $\mathbf{N} = 1^T_N\mathbf{P}^*\mathbf{X}/N$	$\mathbf{H} = f\left(\mathbf{W}^c \odot \mathbf{N}\right)$
	GraphSAGE	$\mathbf{h}^t_{\mathcal{N}_v} = \text{AGGREGATE}_t\left(\{\mathbf{h}^{t-1}_u, \forall u \in \mathcal{N}_v\}\right)$	$\mathbf{h}^t_v = \sigma\left(\mathbf{W}^t \cdot [\mathbf{h}^{t-1}_v \| \mathbf{h}^t_{\mathcal{N}_v}]\right)$
Graph Attention Networks	GAT	$\alpha_{vk} = \frac{\exp\left(\text{LeakyReLU}\left(\mathbf{a}^T[\mathbf{W}\mathbf{h}_v \| \mathbf{W}\mathbf{h}_k]\right)\right)}{\sum_{j\in\mathcal{N}_v}\exp(\text{LeakyReLU}(\mathbf{a}^T[\mathbf{W}\mathbf{h}_v \| \mathbf{W}\mathbf{h}_j]))}$ $\mathbf{h}^t_{\mathcal{N}_v} = \sigma\left(\sum_{k\in\mathcal{N}_v}\alpha_{vk}\mathbf{W}\mathbf{h}_k\right)$ Multi-head concatenation: $\mathbf{h}^t_{\mathcal{N}_v} = \Big\|_{m=1}^{M}\sigma\left(\sum_{k\in\mathcal{N}_v}\alpha^m_{vk}\mathbf{W}^m\mathbf{h}_k\right)$ Multi-head average: $\mathbf{h}^t_{\mathcal{N}_v} = \sigma\left(\frac{1}{M}\sum_{m=1}^{M}\sum_{k\in\mathcal{N}_v}\alpha^m_{vk}\mathbf{W}^m\mathbf{h}_k\right)$	$\mathbf{h}^t_v = \mathbf{h}^t_{\mathcal{N}_v}$

0. Overview

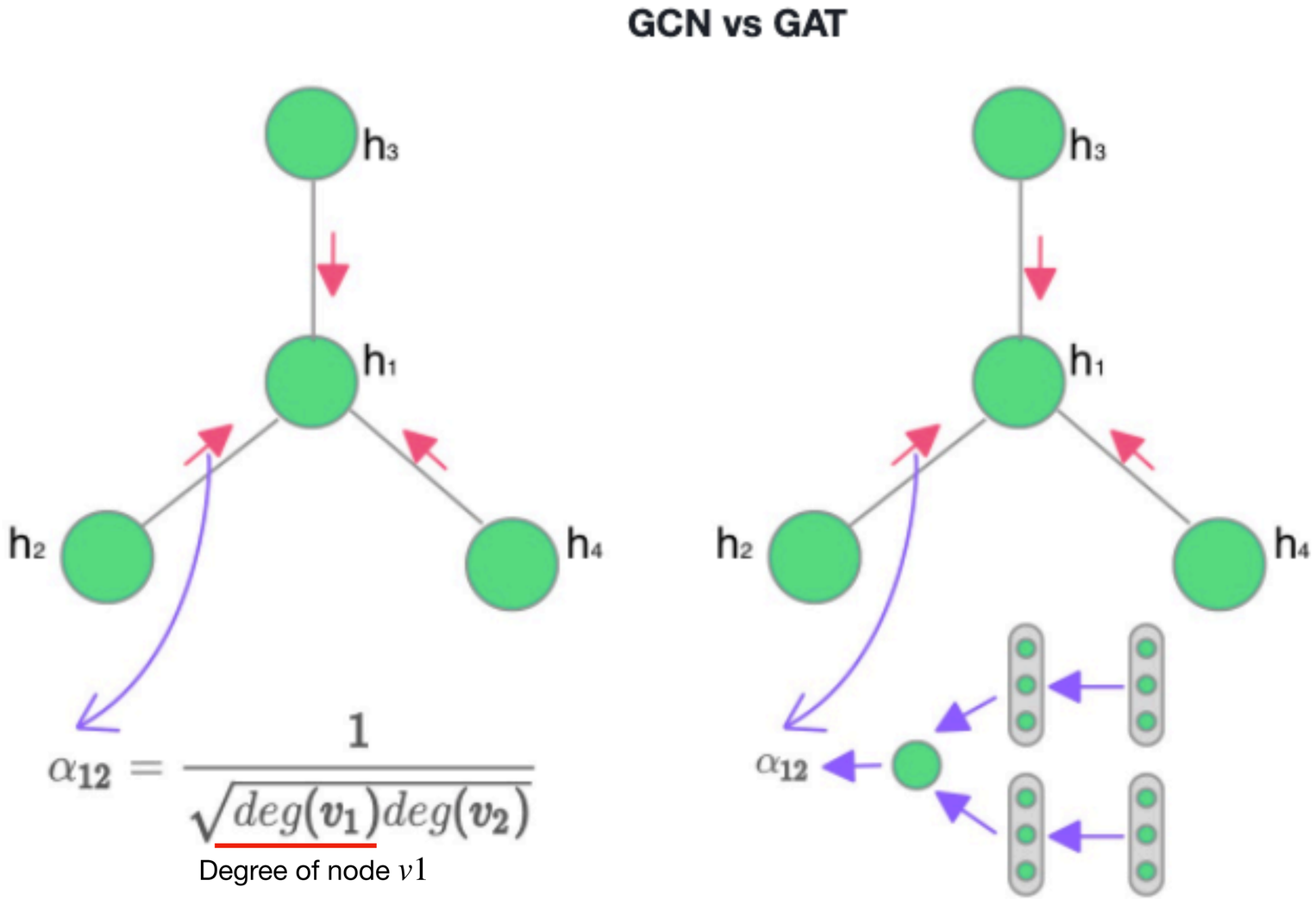


그림: https://dsgiitr.com/blogs/gat/

1. Introduction

□ Motivations

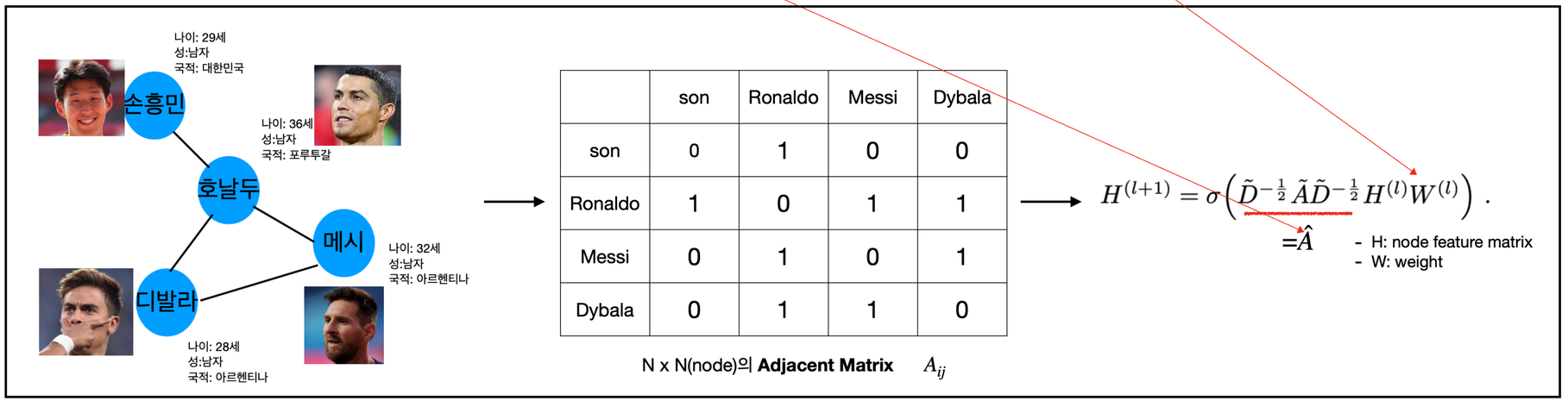
- leveraging masked self-attentional layers to address the shortcomings of prior methods based on GCN or their approximations.

□ Limits of earlier works

- GCN의 경우 learned filters는 그래프 구조에 의존 ⇒ 특정 구조에서 학습된 모델은 다양한 구조를 가진 그래프에 직접적으로 적용될 수 없음

- 일부 earlier approaches는 different sized neighborhoods에서 작동하는 operator를 정의하는 데 어려움이 있음

Sharing a trainable weight



	son	Ronaldo	Messi	Dybala
son	0	1	0	0
Ronaldo	1	0	1	1
Messi	0	1	0	1
Dybala	0	1	1	0

N x N(node)의 **Adjacent Matrix** A_{ij}

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

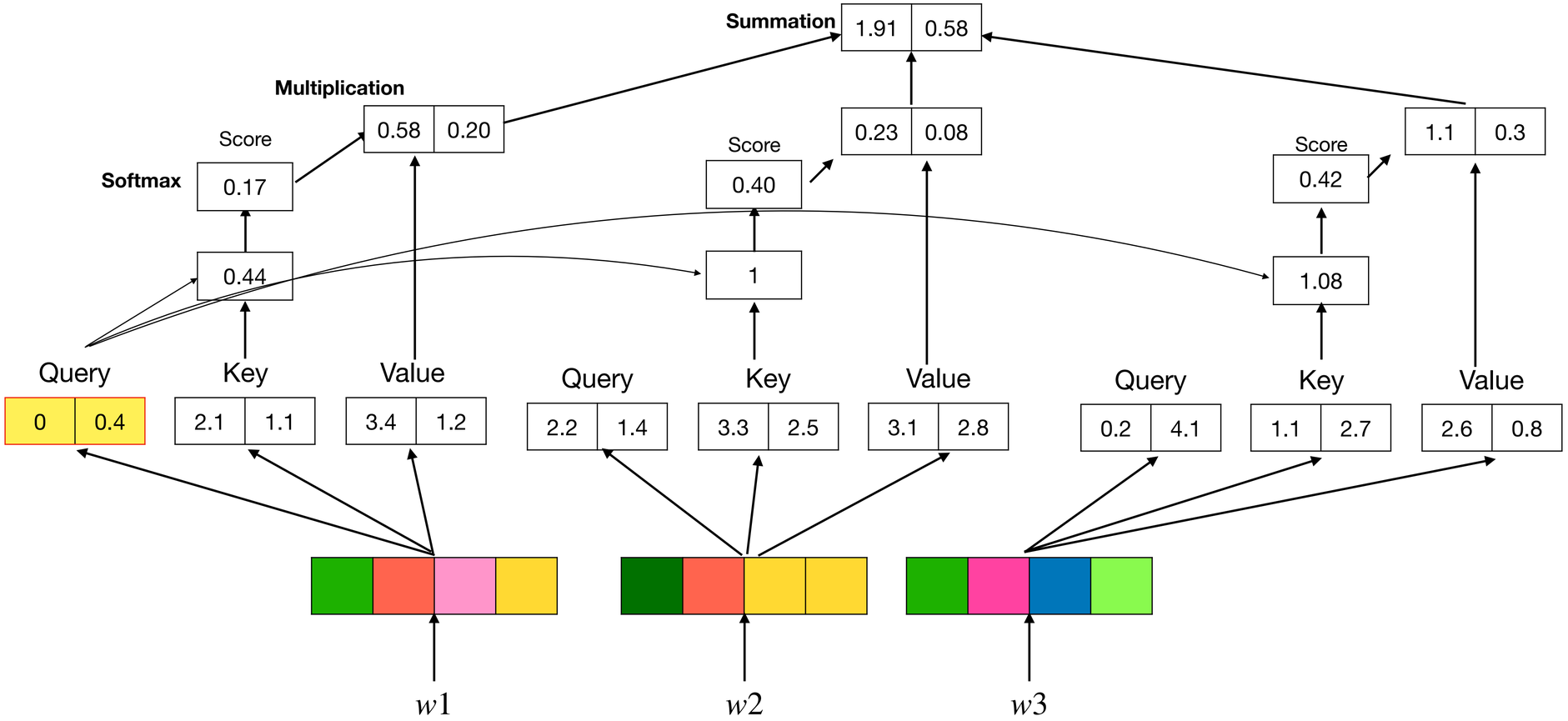
- H: node feature matrix
- W: weight

1. Introduction

□ Self-Attention

- 그래프 구조 데이터에 대한 노드 분류를 수행하기 위해, 어텐션 기반 아키텍처 제안
  → 임의의 노드의 이웃 노드들에 대해 집중attention하면서 각 노드의 hidden representations를 계산한다.

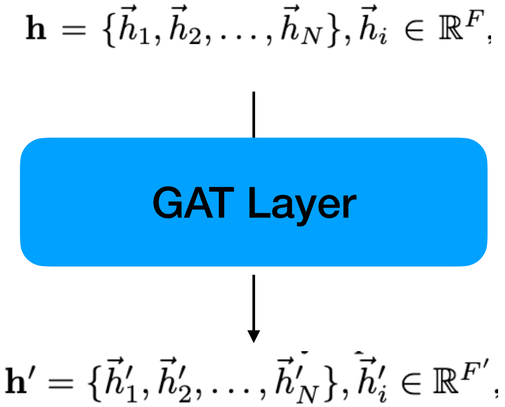
이점1: 연산적으로 효율적이다. ∵ 노드 이웃 쌍들에 대해 병렬적parallelizable으로 연산
이점2: 셀프 어텐션은 다양한 차수를 가진 그래프 노드에 적용될 수 있다
⇒ 각 edge에 공통된 weight가 아닌 **가중치를 반영한 trainable weight**를 할당할 수 있



2. GAT Architecture

Node Features 집합 $\mathbf{h} = \{\vec{h}_1, \vec{h}_2, \ldots, \vec{h}_N\}, \vec{h}_i \in \mathbb{R}^F,$ N: 노드의 개수, F: feature의 개수

New Node Features 집합 $\mathbf{h}' = \{\vec{h}'_1, \vec{h}'_2, \ldots, \vec{h}'_N\}, \vec{h}'_i \in \mathbb{R}^{F'},$

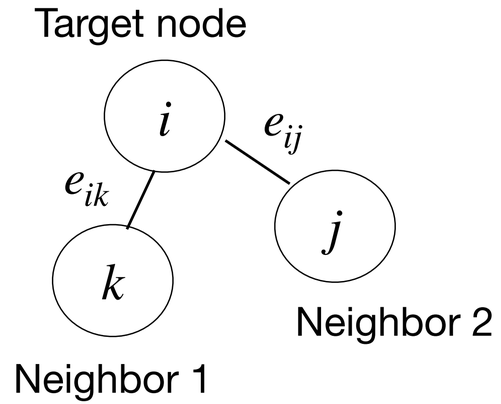


Self Attention 계수

$e_{ij} = a(\mathbf{W}\vec{h}_i, \mathbf{W}\vec{h}_j)$ a shared attentional mechanism $a : \mathbb{R}^{F'} \times \mathbb{R}^{F'} \rightarrow \mathbb{R}$

→ node i에 대한 node j의 중요도 의미

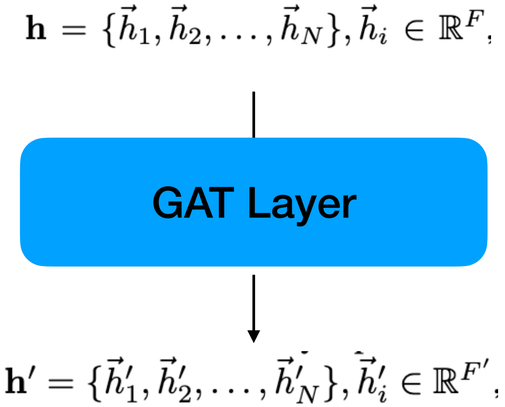
→ node j는 node i의 이웃neighbor



2. GAT Architecture

Node Features 집합 $\mathbf{h} = \{\vec{h}_1, \vec{h}_2, \ldots, \vec{h}_N\}, \vec{h}_i \in \mathbb{R}^F,$ N: 노드의 개수, F: feature의 개수

New Node Features 집합 $\mathbf{h}' = \{\vec{h}'_1, \vec{h}'_2, \ldots, \vec{h}'_N\}, \vec{h}'_i \in \mathbb{R}^{F'},$

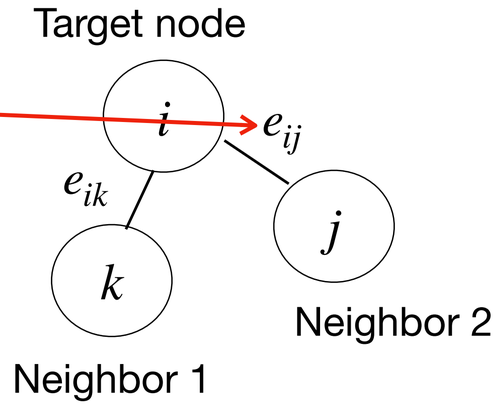


Self Attention 계수

$e_{ij} = a(\mathbf{W}\vec{h}_i, \mathbf{W}\vec{h}_j)$ a shared attentional mechanism $a : \mathbb{R}^{F'} \times \mathbb{R}^{F'} \rightarrow \mathbb{R}$

→ node i에 대한 node j의 중요도 의미

→ node j는 node i의 이웃neighbor

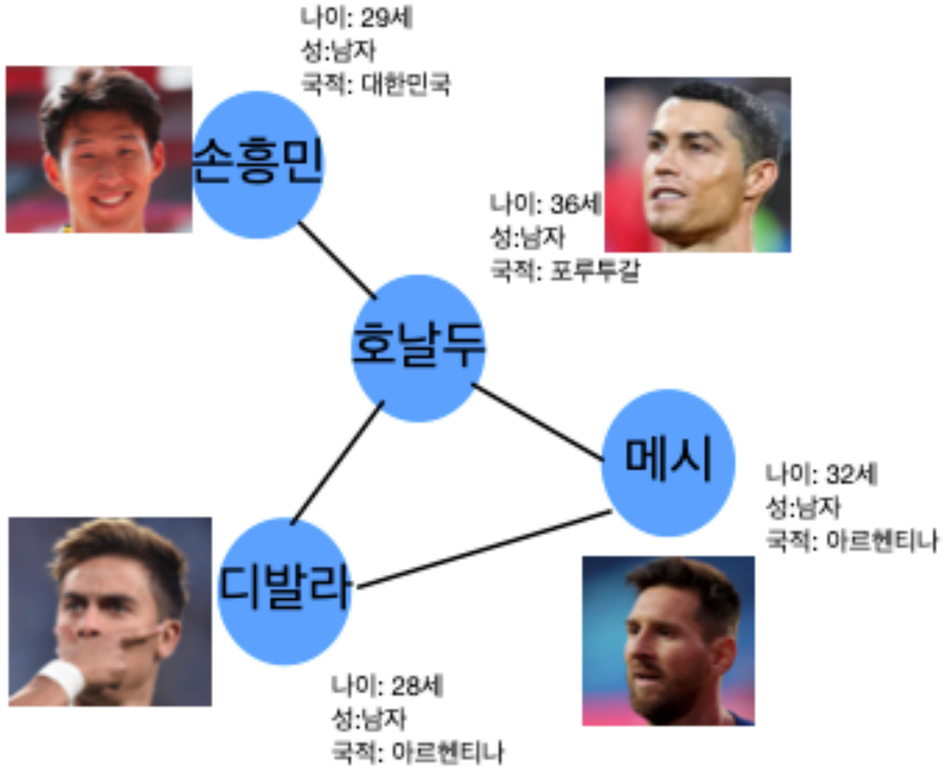


2. GAT Architecture

Attention Mechanism

$\mathbf{h} = \{\vec{h}_1, \vec{h}_2, \ldots, \vec{h}_N\}, \vec{h}_i \in \mathbb{R}^F$, N: 노드의 개수, F: feature의 개수

$\mathbf{h}' = \{\vec{h}'_1, \vec{h}'_2, \ldots, \vec{h}'_N\}, \vec{h}'_i \in \mathbb{R}^{F'}$,



1	2	3	4
손흥민	호날두	디발라	메시

2. GAT Architecture

Attention Mechanism

[1] 입력 노드 피처: $h_1, h_2, \ldots, h_M$

[2] 셀프 어텐션

① **어텐션 스코어** $e_{ij} = a(Wh_i, Wh_j)$

⇒ 타깃 노드 i에 대한 이웃 노드 j의 중요성

⇒ a는 Learnable weight. 이것과 $concat(Wh_i, Wh_j)$를 내적 해줌

② **정규화된 어텐션 스코어** $\alpha_{ij} = softmax_j(e_ij) = \frac{exp(e_{ij})}{\sum_j exp(e_{ik})}$

⇒ 노드 간의 비교를 쉽게 하기 위해 계수를 정규화해줌

③ **최종 어텐션 스코어**

- $S = concat(Wh_i, Wh_j)$
- $Z = LeakyReLU(nn.Linear(S))$
- $\alpha_{ij} = softmax(Z)$

[3] 업데이트

- **싱글 헤드 어텐션** $h_i' = \sigma(\sum_j \alpha_{ij} W h_j)$
- **안정화**

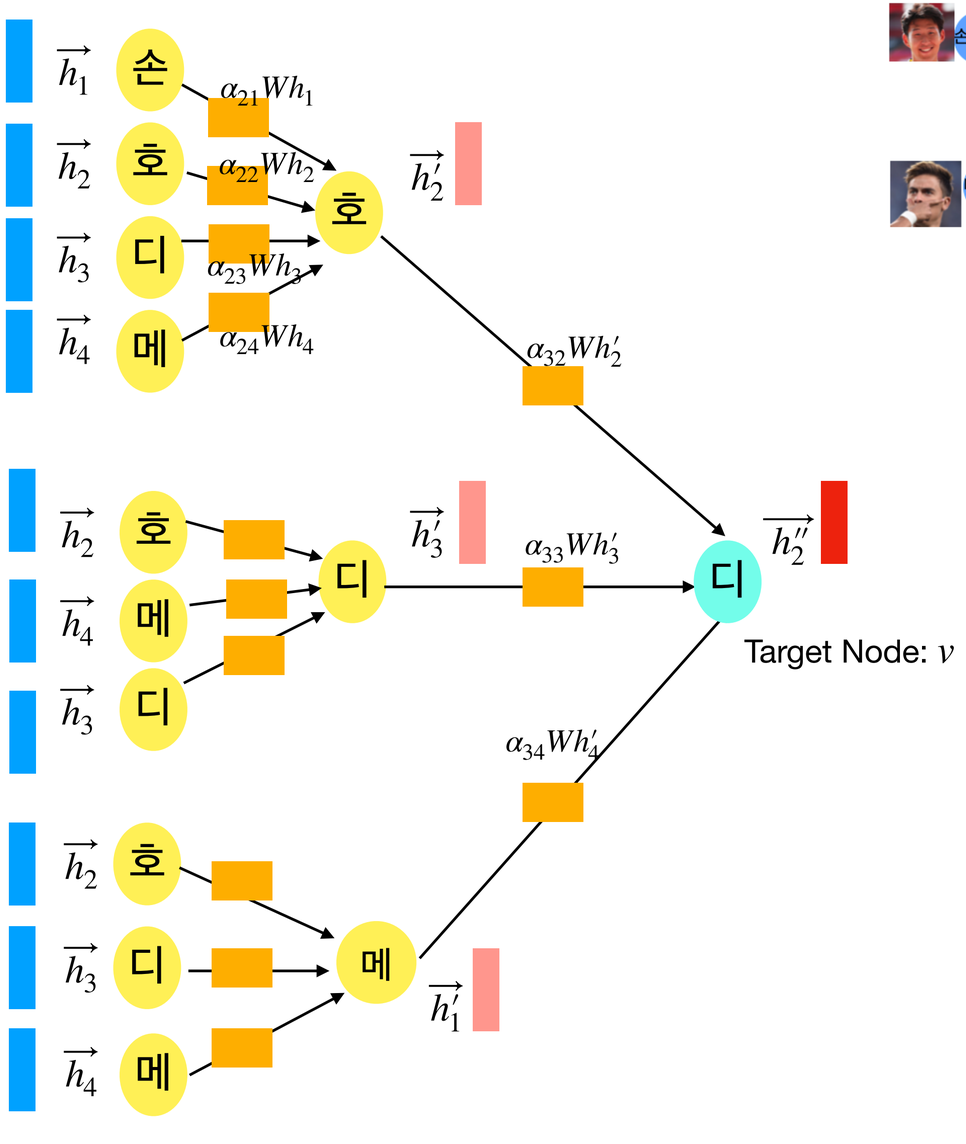
① **멀티 헤드 어텐션:** $h_i' = ||_{k=1}^{K} \sigma(\sum_j \alpha_{ij}^k W^k h_j)$

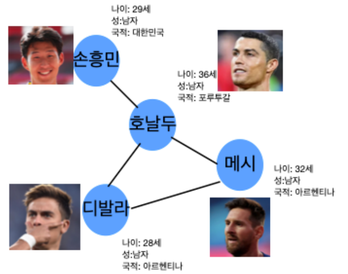
② **평균화:** $h_i = \sigma(\frac{1}{K} \sum_k^K \sum_j \alpha_{ij}^k W^k h_j)$

⇒ 셀프-어텐션의 학습 과정을 안정화시키기

⇒ 저자는 ②를 취하는 것이 더 적절하다고 주장

$h' = W\vec{h} =$ F' W F $*$ $\vec{h}$ F $=$ F'



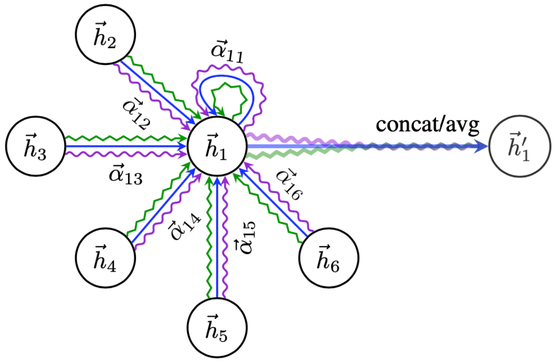


2. GAT Architecture

Multi-head attention

- 셀프 어텐션의 학습 과정을 안정적으로 만드는 데 멀티헤드 어텐션이 효과적
- K 개의 독립적인 어텐션 헤드가 연접concatenated된다

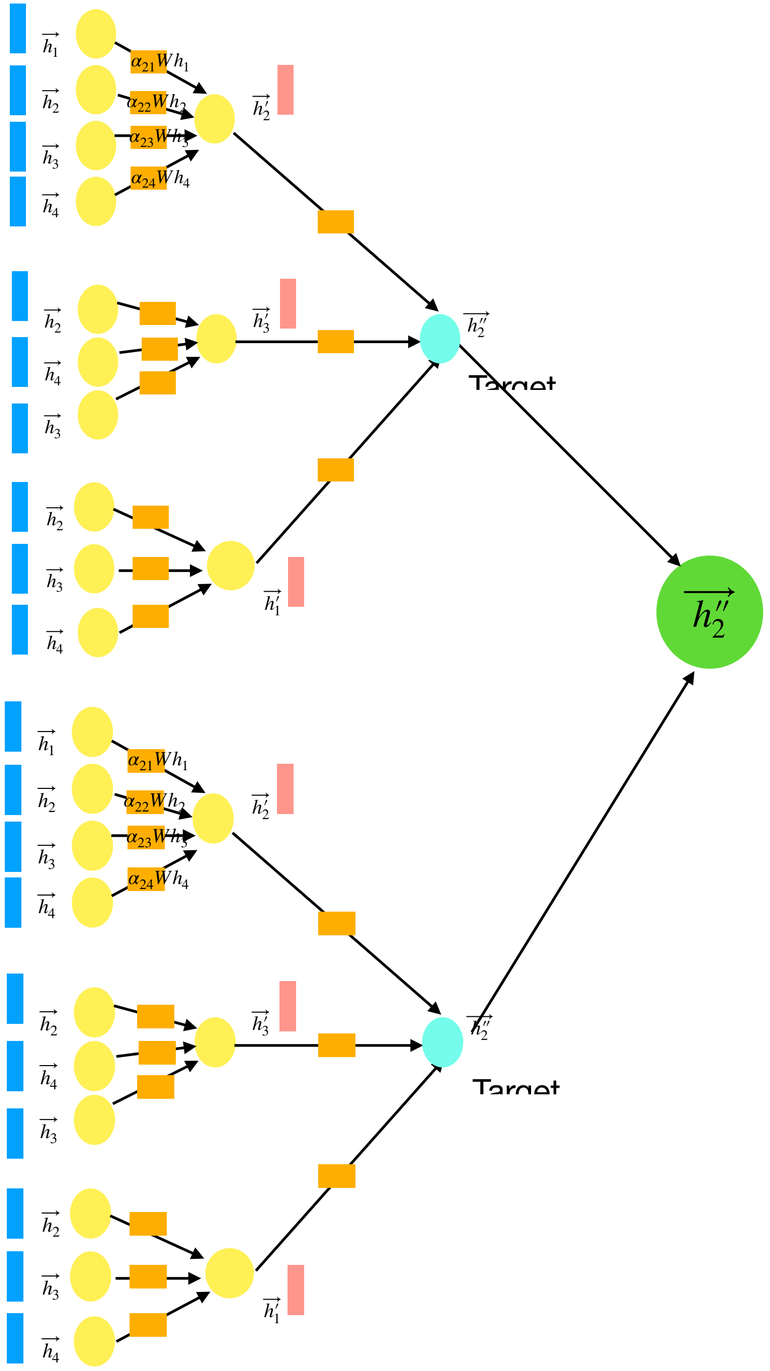
$$\vec{h}'_i = \Big\Vert_{k=1}^{K} \sigma\left(\sum_{j\in\mathcal{N}_i} \alpha_{ij}^k \mathbf{W}^k \vec{h}_j\right)$$



- 위 같이 연결망의 마지막 레이어에서 멀티헤드 어텐션을 수행하는 경우, 연접 concatenation은 합리적인 선택이 아님

- **평균화averaging을 이용하고 nonlinearity를 그 뒤에 수행하는 것이 더 합리적**

$$\vec{h}'_i = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in\mathcal{N}_i} \alpha_{ij}^k \mathbf{W}^k \vec{h}_j\right)$$



3. Related Works

이전 GNN 모델과 GAT의 비교

□ 연산적 효율성

- No eigendecompositions or similar costly matrix operations are required

□ 해석성 증가

- learned attentional weight의 분석은 해석성interpretability을 증가시켜준다.

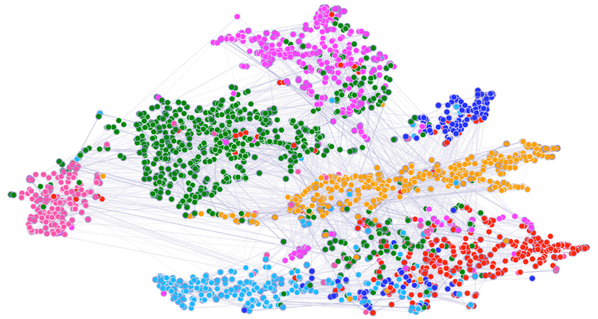


Figure 2: A t-SNE plot of the computed feature representations of a pre-trained GAT model's first hidden layer on the Cora dataset. Node colors denote classes. Edge thickness indicates aggregated normalized attention coefficients between nodes i and j, across all eight attention heads ($\sum_{k=1}^{K} \alpha_{ij}^{k} + \alpha_{ji}^{k}$).

□ 그래프 구조에 의존하지 않음

- GCN은 hidden state vector를 구하기 위해 그래프 구조의 정보를 담은 인접 행렬을 사용했지만, GAT는 이것에 의존하지 않는다.

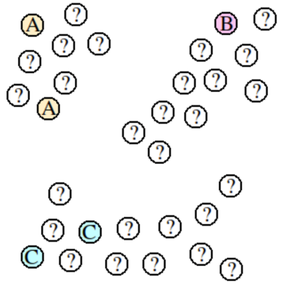
4. Evaluation

- Datasets

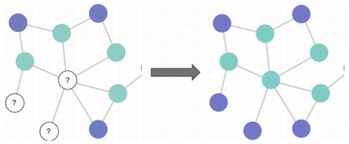
Table 1: Summary of the datasets used in our experiments.

	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)

- **Transductive Learning** [7],[8]

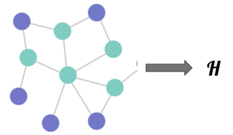


- 일부는 레이블이 없는 학습 셋이 주어짐
- 일종의 준지도학습이라고 간주될 수 있음

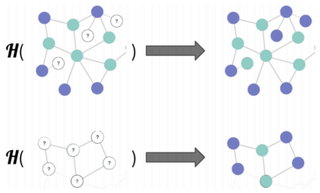


- **Inductive Learning**

- 레이블된 훈련 셋으로 모델 학습 (지도 학습과 유사)
- 그 모델로 새로운 unseen 그래프를 예측



(a) A model $\mathcal{H}$ is learned over some graph



(b) The model is then by applied to new nodes and edges

- Results: Table2: accuracy, Table3: the micro averaged F1 score [6] 참고

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 ± 0.5%	—	78.8 ± 0.3%
GCN-64*	81.4 ± 0.5%	70.9 ± 0.5%	**79.0** ± 0.3%
GAT (ours)	**83.0** ± 0.7%	**72.5** ± 0.7%	**79.0** ± 0.3%

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)	**0.973** ± 0.002

References

[1] Veličković, Petar, et al. "Graph attention networks." arXiv preprint arXiv:1710.10903 (2017).
[2] https://dos-tacos.github.io/translation/transductive-learning/
[3] https://chioni.github.io/posts/gat/
[4] Prefered AI, ‘S6 Presenter: Graph Attention Networks by Zhang Ce’, https://www.youtube.com/watch?v=6hbWpbi0Z24
[5] https://newsight.tistory.com/313
[6] https://unlimitedpower.tistory.com/entry/IR-%EB%A7%88%EC%9D%B4%ED%81%AC%EB%A1%9C-%ED%8F%89%EA%B7%A0Micro-average-%EB%A7%A4%ED%81%AC%EB%A1%9C-%ED%8F%89%EA%B7%A0Macro-average-%EC%9D%B4%EB%9E%80-%EB%AC%B4%EC%97%87%EC%9D%B8%EA%B0%80
[7] https://en.wikipedia.org/wiki/Transduction_(machine_learning)
[8] https://komputervision.wordpress.com/2020/03/24/inductive-learning-vs-transductive-learning/
[9] https://dsgiitr.com/blogs/gat/