

# A Practical Introduction to (Explainable) Graph Learning

Yong-Min Shin

School of Mathematics and Computing (Computational Science and Engineering)

Yonsei University

2025.04.02



수학계산학부(계산과학공학)

School of Mathematics and Computing  
(Computational Science and Engineering)



이화여자대학교  
EWha WOMANS UNIVERSITY

## Part 1: A practical introduction to graphs and graph neural networks

1. Understanding of **graphs** as a **general data type**
2. Understanding of the general framework of **graph neural networks (GNNs)**

## Part 2: Towards explainable graph learning with attention

1. Understanding the basic concepts of **explainable AI**
2. **Answer to the question: Can we understand graph attention networks using attention?**



**Part 1: A practical introduction to graphs and graph neural networks**

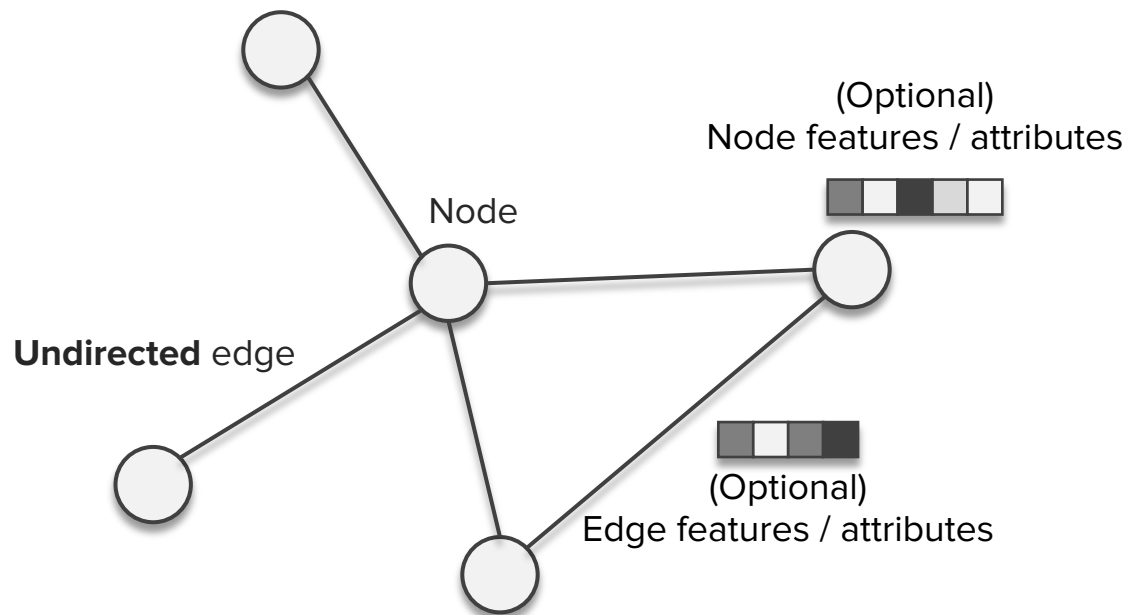
## **Understanding of graphs as a general data type**

\*This part is heavily influenced by one of my academic heros, Petar Veličković. These are some materials from his public materials that I have referred to:

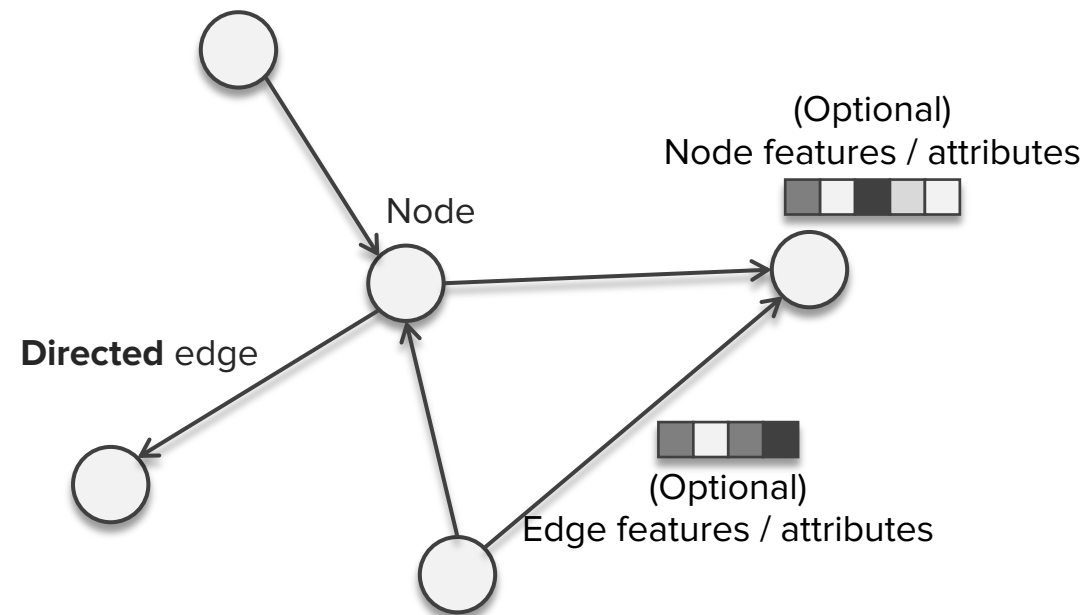
- (Slide) Everything is Connected: Graph Neural Networks from the Ground Up (2021)
- (Blog) Graph & Geometric ML in 2024: Where We Are and What's Next (Part II – Applications)

# Graphs as an abstract datatype

Graphs are an abstract type of data where nodes (entities) are **connected** by edges (connections)



Undirected graph



Directed graph

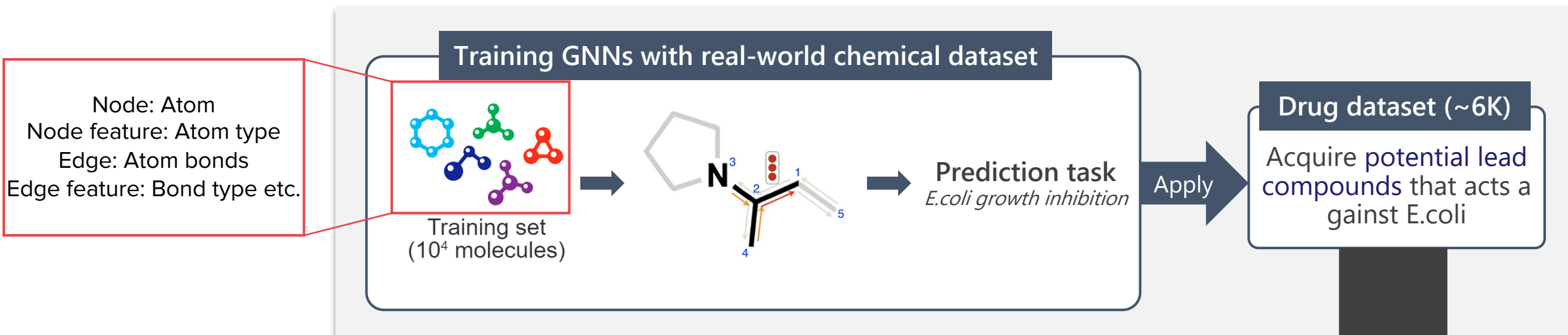
...But honestly, looking at this does not result in a **practical** understanding of graphs.

Therefore, we will look at **various applications** in the field of **graph machine learning** before moving our discussion further.



# Area 1) Biology & Chemistry Research

## Example 1: The discovery of Halicin, GNN-guided antibiotic discovery



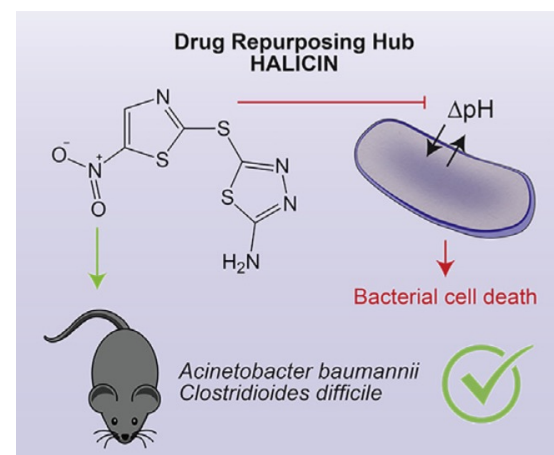
Article

### A Deep Learning Approach to Antibiotic Discovery

Jonathan M. Stokes,<sup>1,2,3</sup> Kevin Yang,<sup>3,4,10</sup> Kyle Swanson,<sup>3,4,10</sup> Wengong Jin,<sup>3,4</sup> Andres Cubillos-Ruiz,<sup>1,2,5</sup> Nina M. Donghia,<sup>1,2</sup> Craig R. MacNair,<sup>6</sup> Shawn French,<sup>6</sup> Lindsey A. Carfrae,<sup>6</sup> Zohar Bloom-Ackermann,<sup>2,7</sup> Victoria M. Tran,<sup>8</sup> Anush Chippipkoppa-Pepe,<sup>3,7</sup> Ahmed H. Badran,<sup>1</sup> Ian W. Andrews,<sup>1,2,8</sup> Emma J. Chory,<sup>1,2</sup> George M. Church,<sup>3,7,9</sup> Eric D. Brown,<sup>10</sup> Tommi S. Jaakkola,<sup>3,4</sup> Regina Barzilay,<sup>3,4,9,\*</sup> and James J. Collins<sup>1,2,5,8,9,11,\*</sup>

<sup>1</sup>Department of Biological Engineering, Synthetic Biology Center, Institute for Medical Engineering and Science, Massachusetts Institute of Technology, Cambridge, MA 02139, USA  
<sup>2</sup>Broad Institute of MIT and Harvard, Cambridge, MA 02142, USA  
<sup>3</sup>Machine Learning for Pharmaceutical Discovery and Synthesis Consortium, Massachusetts Institute of Technology, Cambridge, MA 02139, USA  
<sup>4</sup>Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139, USA  
<sup>5</sup>Wyss Institute for Biologically Inspired Engineering, Harvard University, Boston, MA 02115, USA  
<sup>6</sup>Department of Biochemistry and Biomedical Sciences, Michael G. DeGroote Institute for Infectious Disease Research, McMaster University, Hamilton, ON L8N 3Z5, Canada  
<sup>7</sup>Department of Genetics, Harvard Medical School, Boston, MA 02115, USA  
<sup>8</sup>Harvard-MIT Program in Health Sciences and Technology, Cambridge, MA 02139, USA  
<sup>9</sup>Abdul Latif Jameel Clinic for Machine Learning in Health, Massachusetts Institute of Technology, Cambridge, MA 02139, USA  
<sup>10</sup>These authors contributed equally  
<sup>11</sup>Lead Contact

\*Correspondence: regina@csail.mit.edu (R.B.), jimjc@mit.edu (J.J.C.)  
<https://doi.org/10.1016/j.cell.2020.01.021>



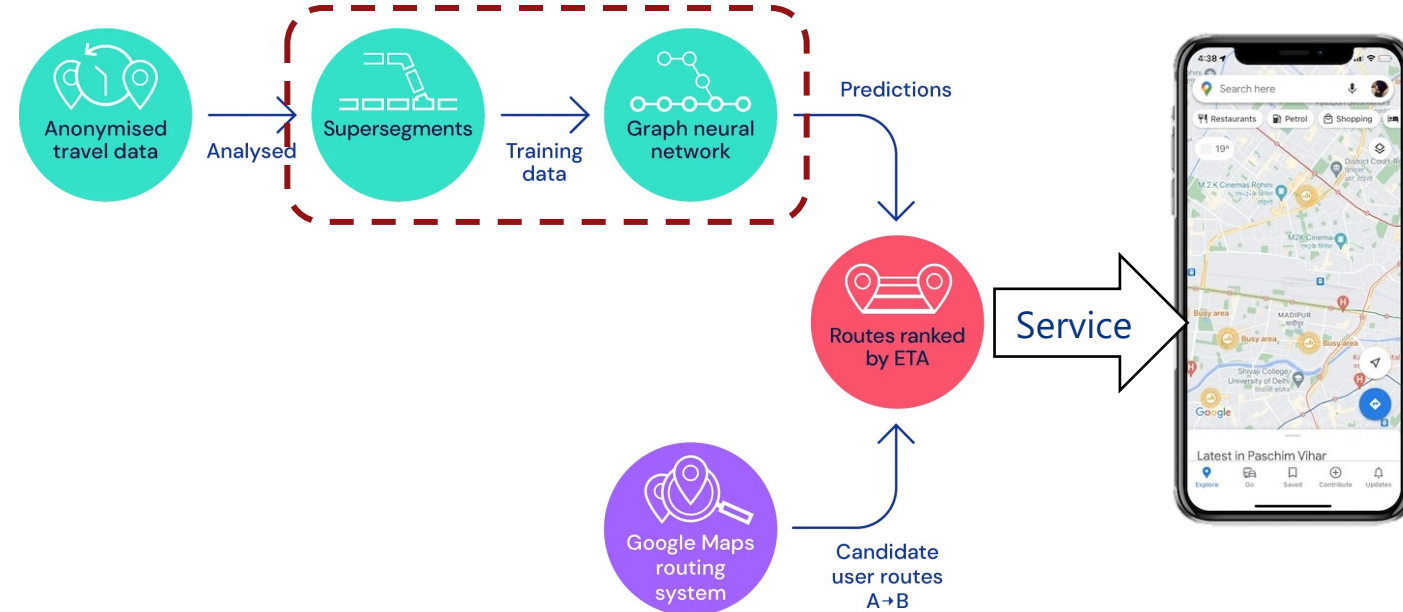
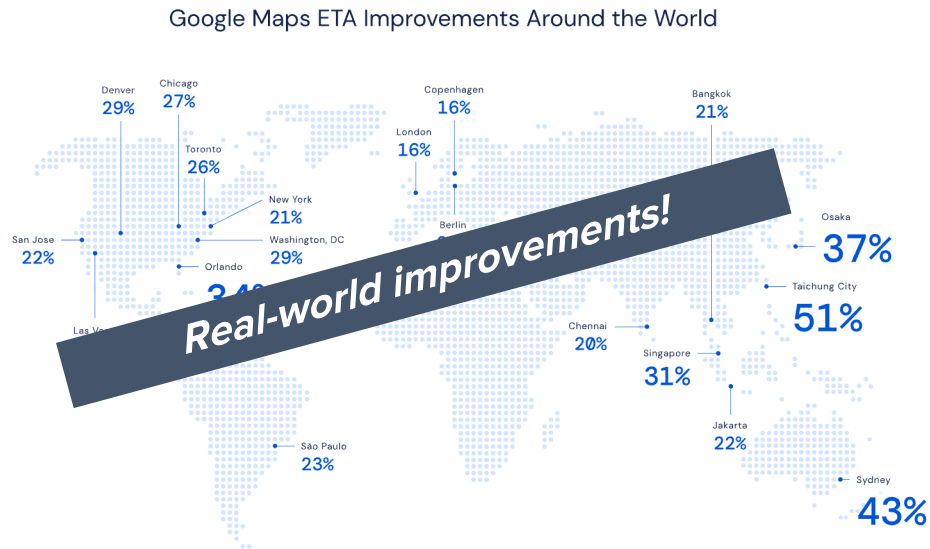
Further screening  
& Empirical testing

Stokes, Jonathan M., et al. "A deep learning approach to antibiotic discovery." Cell 180.4 (2020): 688-702.

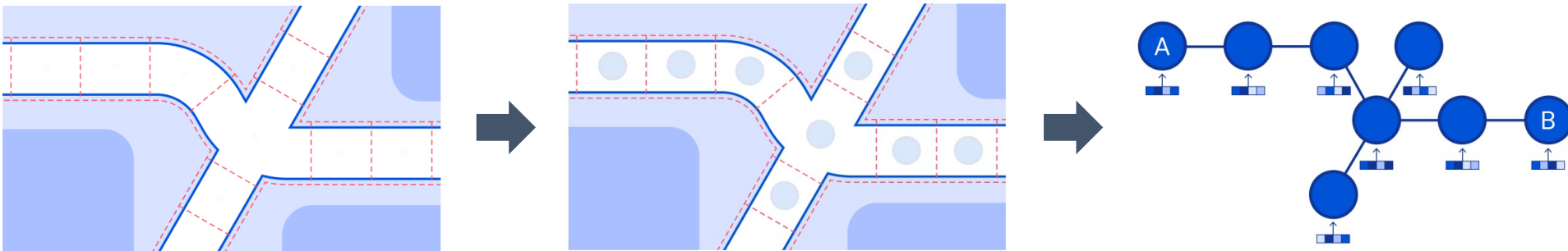
Yang, Kevin, et al. "Analyzing learned molecular representations for property prediction." Journal of chemical information and modeling 59.8 (2019): 3370-3388.

# Area 2) ETA prediction

## Example 2: DeepMind's improvement of Google map's ETA (Estimated Time of Arrival) prediction



Unlike chemical datasets, constructing a graph is less straightforward. In these cases, **how to construct the graph** is also a crucial contribution.

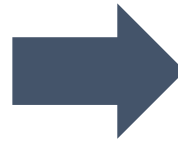
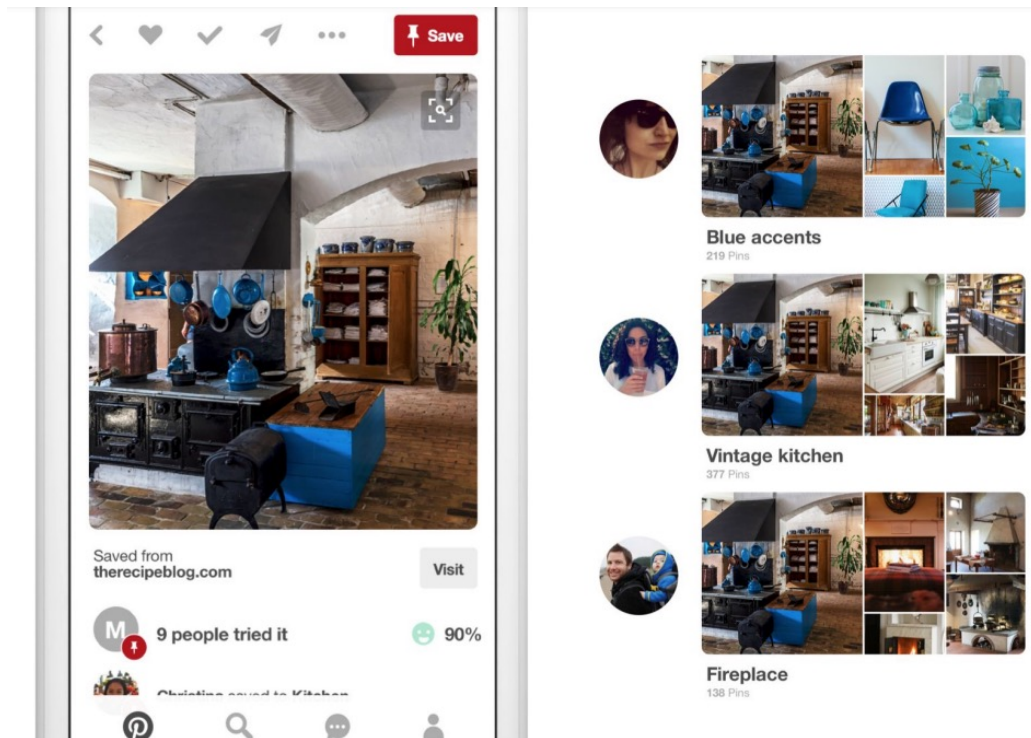


# Area 3) Recommender systems

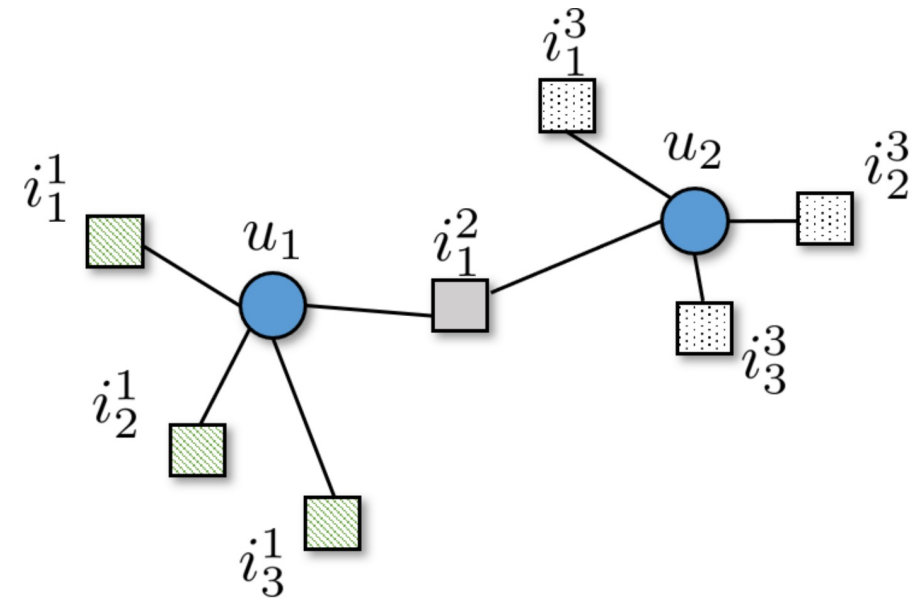
7

## Example 3: Pinterest (social platform)

### Image & User interaction in Pinterest



### User-item interaction graph



# Area 3) Recommender systems

## Example 4: Other industry usecases

Applied Research Track

CIKM '20, October 19–23, 2020, Virtual Event, Ireland

### P-Companion: A Principled Framework for Diversified Complementary Product Recommendation

Junheng Hao<sup>1,\*</sup>, Tong Zhao<sup>2</sup>, Jin  
University of California, Los Angeles  
[jhaoyz, weijun, tcs@ucla.edu, haojunheng@ucla.edu]

#### ABSTRACT

Complementary product recommendation (CPR), aiming at providing product suggestions that are often bought together to satisfy joint demand, forms a pivotal component of e-commerce service, however, existing methods are far from optimal. Given one product, how to recommend its complementary products of different types is the key problem we tackle in this work. We first conduct an analysis to correct the inaccurate assumptions adopted by existing work to show that co-purchased products are not always complementary and further propose a new strategy to generate clean distant supervision labels for CPR modeling. Moreover, to bridge in the gap from existing work that CPR does not only need relevance modeling but also requires diversity to fulfill the whole purchase demand, we develop a deep learning framework, P-Companion, to explicitly model both relevance and diversity. More specifically, given one product with its product type, P-Companion first uses an encoder-

Applied Data Science Track Paper

KDD '20, August 23–27, 2020, Virtual Event, USA

### ConSTGAT: Contextual Spatial-Temporal Graph Attention Network for Travel Time Estimation at Baidu Maps

Xiaomin Fang, Jizhou Huang\*, Fan Wang, Lingli  
Baidu Inc, China  
{fangxiaomin01, huangjizhou01, wang.fan, lingli}@baidu.com

#### ABSTRACT

The task of travel time estimation (TTE) is to estimate the travel time for a given route and departure time. TTE is an important problem in intelligent transportation systems such as navigation, ride-hailing services. This task is challenging because of many essential aspects, such as traffic prediction and contextual information. First, the accuracy of traffic prediction is strongly correlated with the traffic speed of the road segments in a route. Existing work mainly adopts spatial-temporal graph neural networks to improve the accuracy of traffic prediction, where spatial and temporal information is used separately. However, one drawback is that the spatial and temporal correlations are not fully exploited to obtain better accuracy. Second, contextual information of a route, i.e., the connections of adjacent road segments in the route, is an essential factor that impacts the driving speed. Previous work mainly uses sequential encoding models to address this issue. However, it is difficult to scale up sequential models to large-scale real-world

### AliGraph: A Comprehensive Graph Neural Network Platform

Rong Zhu, Kun Zhao, Hongxia Yang\*, Weilin Wu, Yanchi Chen, Zhibo Gong  
{red.zr, kun.zhao, yang.yhx, weilin.wu, yanchi.chen, zhibo.gong}@alibaba-inc.com

#### ABSTRACT

An increasing number of machine learning tasks are dealing with large graph datasets, which capture rich information and relationship among potentially billions of elements. Graph Neural Network (GNN) becomes an effective way to address the graph learning problem by converting the graph data into a low dimensional space while keeping both the structural and property information to the maximum extent and constructing a neural network for training and referencing. However, it is challenging to provide an efficient graph storage and computation capabilities to facilitate GNN training and enable development of new GNN algorithms. In this paper, we present a comprehensive graph neural network system, namely AliGraph, which consists of distributed graph storage, optimized sampling operators and runtime to efficiently support not only existing popular GNNs but also a series of in-house developed ones for

### Graph Neural Networks for Friend Ranking in Large-scale Social Platforms

Aravind Sankar, Chao Zhang, Yizhou Sun, Yizhou Sun, Yizhou Sun  
University of Illinois at Chicago, Champaign

#### ABSTRACT

Graph Neural Networks (GNNs) have achieved significant advances in graph learning tasks such as node classification, link prediction, and community detection. In this paper, we study the application of GNNs to friend ranking in large-scale social platforms. We first investigate the unique challenges of friend ranking, where a significant fraction of users are inactive and have limited structural and engagement information. Moreover, users interact with different functionalities, communicate with diverse groups, and have multifaceted interaction patterns.

We study the application of GNNs for friend suggestion, providing the first investigation of GNN design for this task, to our knowledge. To leverage the rich knowledge of in-platform actions, we formulate friend suggestion as *multi-faceted friend ranking* with

### Knowing your FATE: Friendship, Action and Temporal Explanations for User Engagement Prediction on Social Apps

Xianfeng Zhou<sup>1</sup>, Neil Shah<sup>2</sup>, Xiaolin Shi<sup>3</sup>, Prasenjit Mitra<sup>1</sup>, Suhang Wang<sup>1\*</sup>  
<sup>1</sup>Pennsylvania State University, <sup>2</sup>Snap Inc., <sup>3</sup>University of Illinois at Chicago

#### ABSTRACT

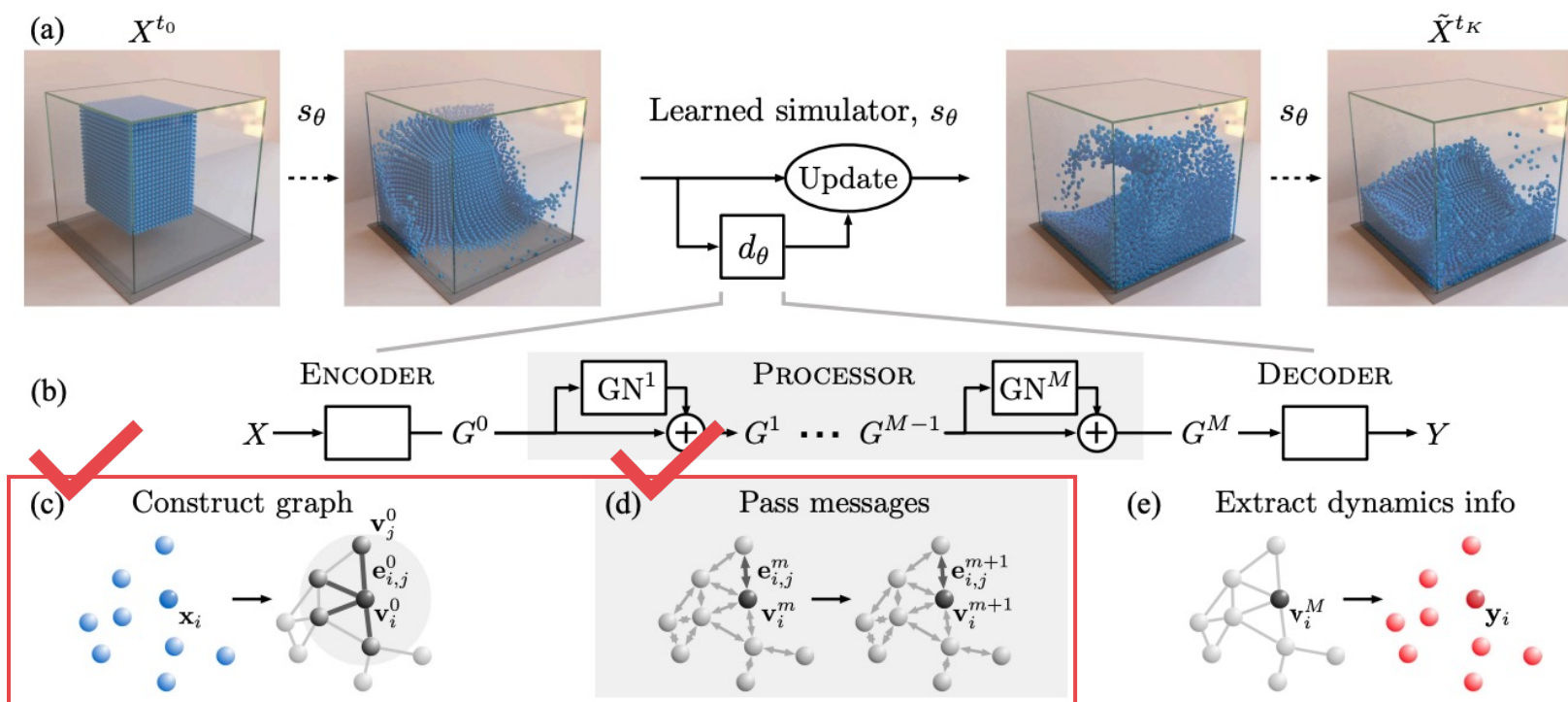
With the rapid growth of mobile social network applications (Apps) in recent years, user engagement has become increasingly important to provide useful insights for future App design and development. While several promising neural modeling approaches were recently pioneered for accurate user engagement prediction, their black-box designs are unfortunately limited in model explainability. In this paper, we study a novel problem of explainable user engagement prediction for social network Apps. First, we propose a flexible definition of user engagement for various business scenarios, based on future metric expectations. Next, we design an end-to-end neural framework, FATE, which incorporates three key factors that we identify to influence user engagement, namely friendships, user actions, and temporal dynamics to achieve explainable engagement predictions. FATE is based on a tensor-based graph neural network (GNN), LSTM and a mixture attention mechanism, which allows for (a) predictive explanations

existing users using different metrics, such as churn rate prediction [38] and lifespan analysis [39]. Others model user engagement with macroscopic features (e.g., demographic information) [1] and historical statistic features (e.g., user activities) [19]. Recently, Liu et al. [20] propose using dynamic action graphs, where nodes are in-App actions, and edges are transitions between actions, to predict future activity using a neural model.

Despite some success, existing methods generally suffer from the following: (1) They fail to model friendship dependencies or ignore user-user interactions when modeling user engagement. As users are connected in social Apps, their engagement affects each other [32]. For example, active users may keep posting new contents, which attract his/her friends and elevate their engagement. Thus, it is essential to capture friendship dependencies and user interactions when modeling user engagement. (2) Engagement objectives may differ across Apps and even across features. For example, an

arXiv:2006.15151v1 [cs.LG] 15 Jun 2020

## Example 5: Simulation of complex physical systems

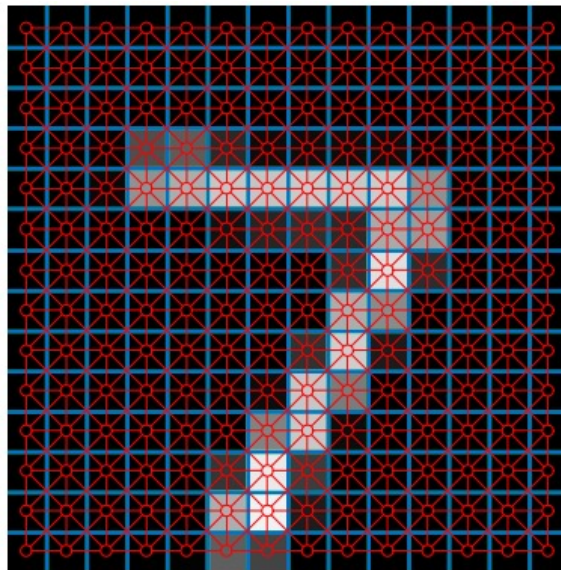


Similar to the ETA prediction task, how to construct the edges between particles will highly impact the rest of the learning process.

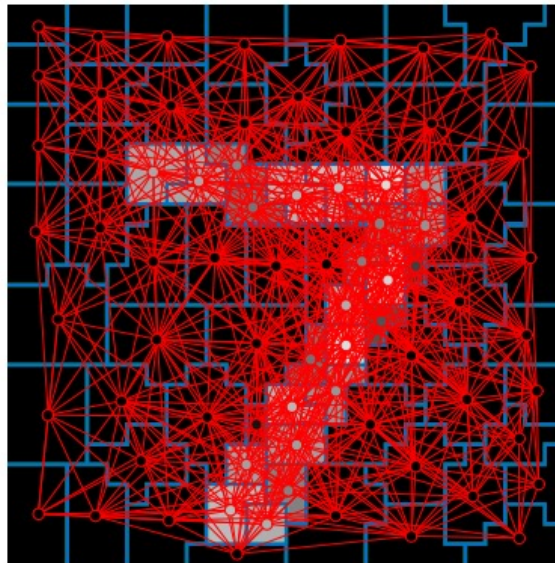


# Area 5) Images are actually grid-like graphs

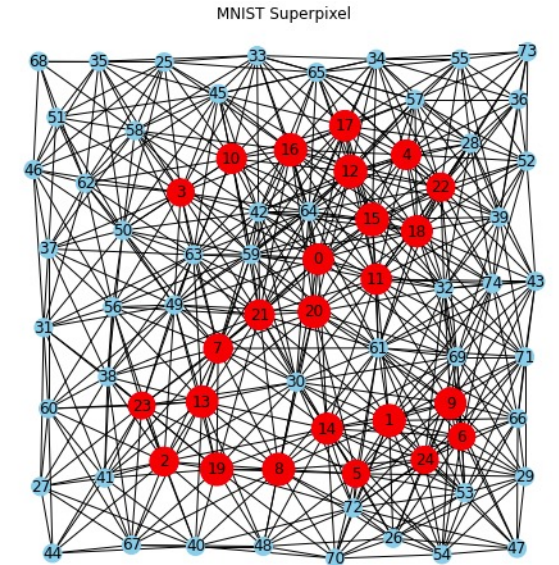
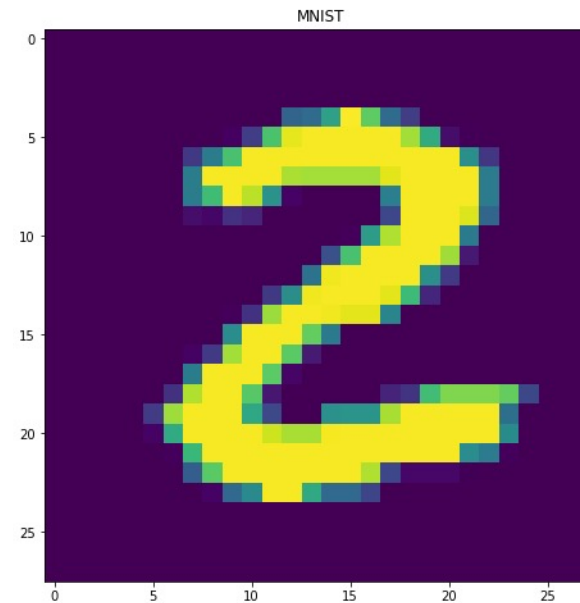
## Example 6: MNIST and MNIST-sp



Regular grid



Superpixels



MNIST-sp is quite commonly used as a benchmark dataset in the graph domain.

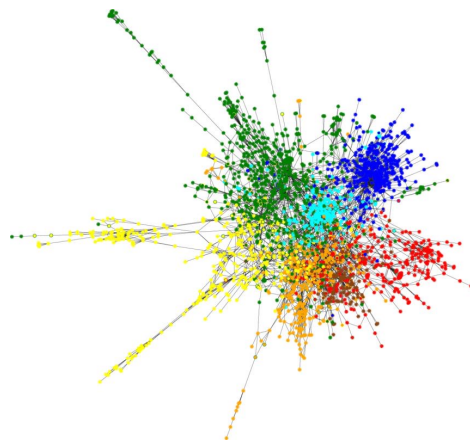
# In academia: Benchmark datasets in the literature

## Social



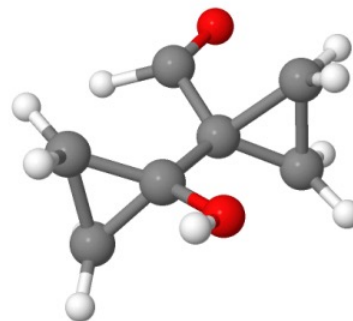
Node: People / Account  
Edge: Connection  
Node feature: Metadata

## Citation / Web



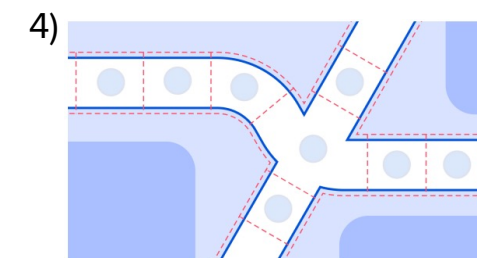
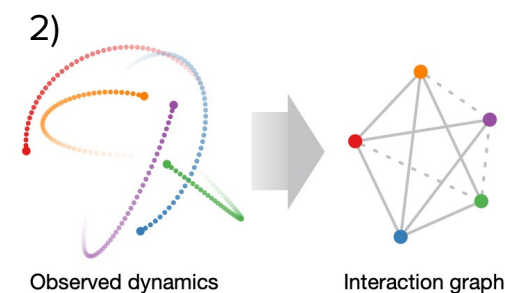
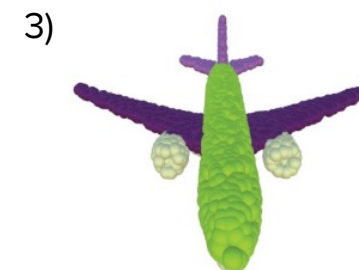
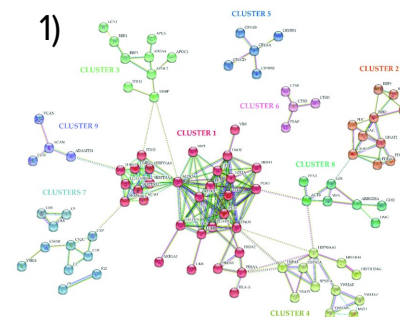
Node: Paper  
Edge: Citation  
Node feature: Abstract

## Molecules



Node: Atom  
Edge: Bond  
Node feature: Atom type  
Edge feature: Bond type

## Biology / Simulation / etc.



### Select benchmark datasets

- Reddit
- Ego-Facebook
- Github

- \*Planetoid dataset  
(Cora/Citeseer/Pubmed)
- Coauthor
- WebKB  
(Texas/Cornell/etc.)

- QM9
- Zinc
- MUTAG

- 1) **\*\*PPI** (protein-protein interaction)
- 2) Physical simulation (Kipf et al., 2018)
- 3) 3D point cloud (Wang et al., 2019)
- 4) Road network (Derrow-Pinion et al., 2021)

...and so much more

Yang et al., Revisiting Semi-Supervised Learning with Graph Embeddings, ICML 2016

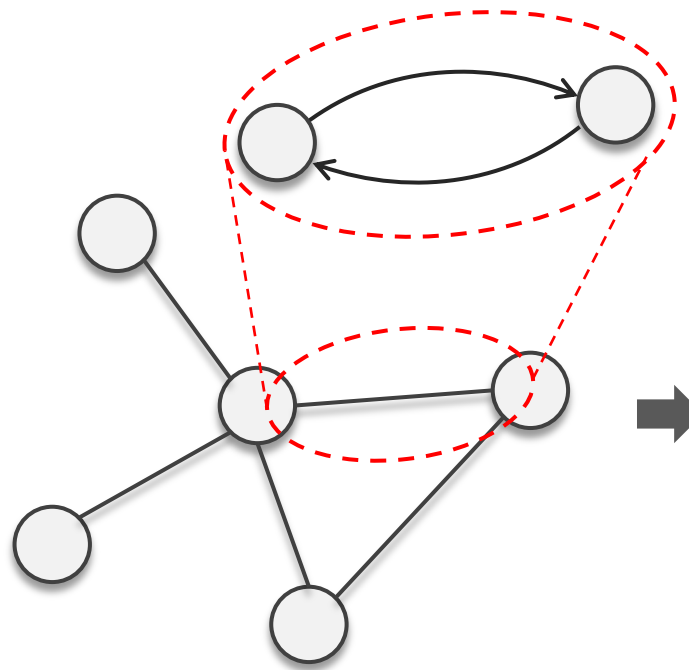
Kipf et al., Neural Relational Inference for Interacting Systems, ICML 2018

Wang et al., Dynamic Graph CNN for Learning on Point Clouds, ACM Transactions on Graphics 2019

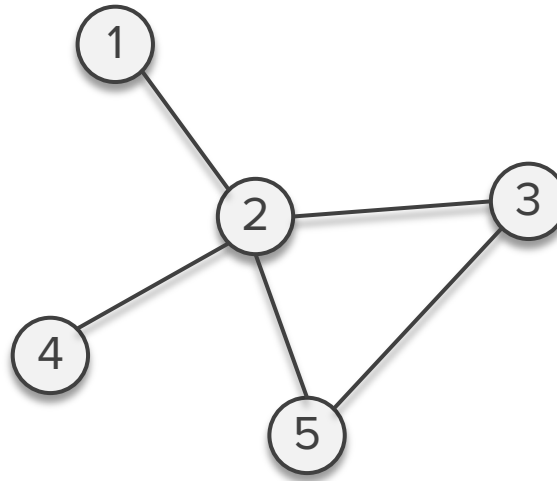
Derrow-Pinion et al., ETA Prediction with Graph Neural Networks in Google Maps, CIKM 2021

\*\*Image source: [https://www.researchgate.net/publication/324457787\\_iTRAQ\\_Quantitative\\_Proteomic\\_Analysis\\_of\\_Vitreous\\_from\\_Patients\\_with\\_Retinal\\_Detachment/figures?lo=1](https://www.researchgate.net/publication/324457787_iTRAQ_Quantitative_Proteomic_Analysis_of_Vitreous_from_Patients_with_Retinal_Detachment/figures?lo=1)

# Representing the graph as a adjacency matrix



\*We treat undirected edges as two directed edges going in both directions



**Undirected graph**

## Assign arbitrary node ordering

- **Graphs with canonical node ordering is not common**
- Related research topic: Positional encoding in graphs (Maskey et al., NeurIPS 2022)



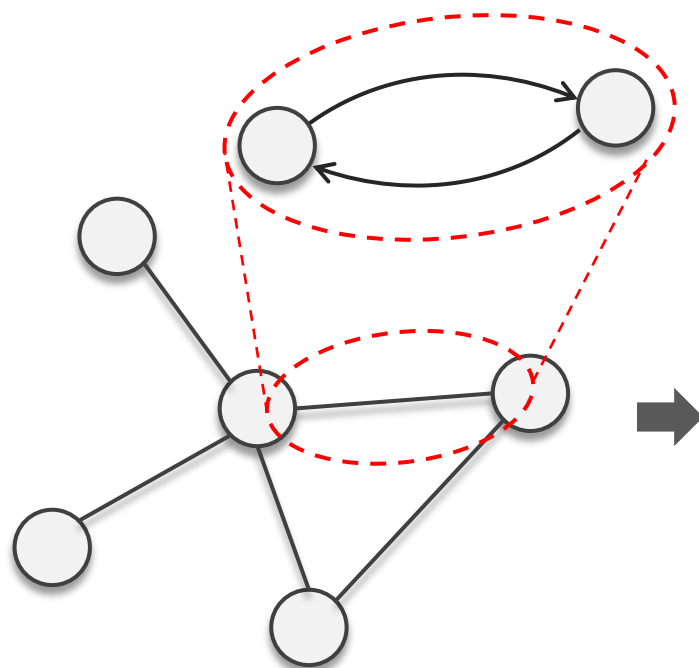
	1	2	3	4	5
1	0	1	0	0	0
2	1	0	1	1	1
3	0	1	0	0	1
4	0	1	0	0	0
5	0	1	0	1	0

## Adjacency matrix

- Represent edge by assigning 1 for (i, j)-th element if node i and j are connected
- For weighted graphs: Assign a real number
- For graphs with multiple edges: Assign integers
- For directed graphs: Asymmetric matrix

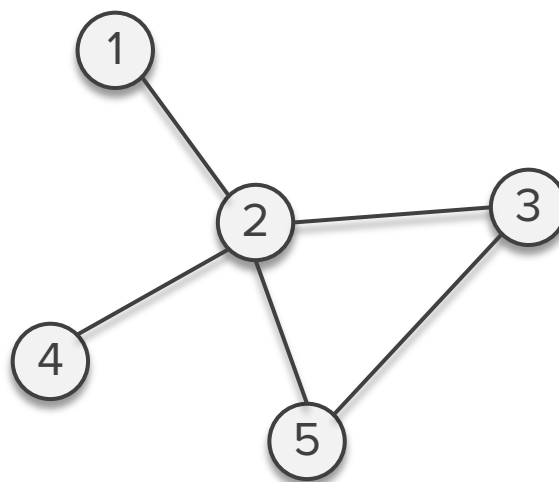


# Representing the graph as a adjacency matrix



**Undirected graph**

\*We treat undirected edges as two directed edges going in both directions



**Assign arbitrary node ordering**

- **Graphs with canonical node ordering is not common**
- Related research topic: Positional encoding in graphs (Maskey et al., NeurIPS 2022)



(1, 2), (2, 1), (2, 3), (3, 2), ...

**Edge list**

- Represent graph by listing all edges
- Notice that for undirected edges, (i, j) and (j, i) both appear
- More **memory efficient** than (dense) adjacency matrix

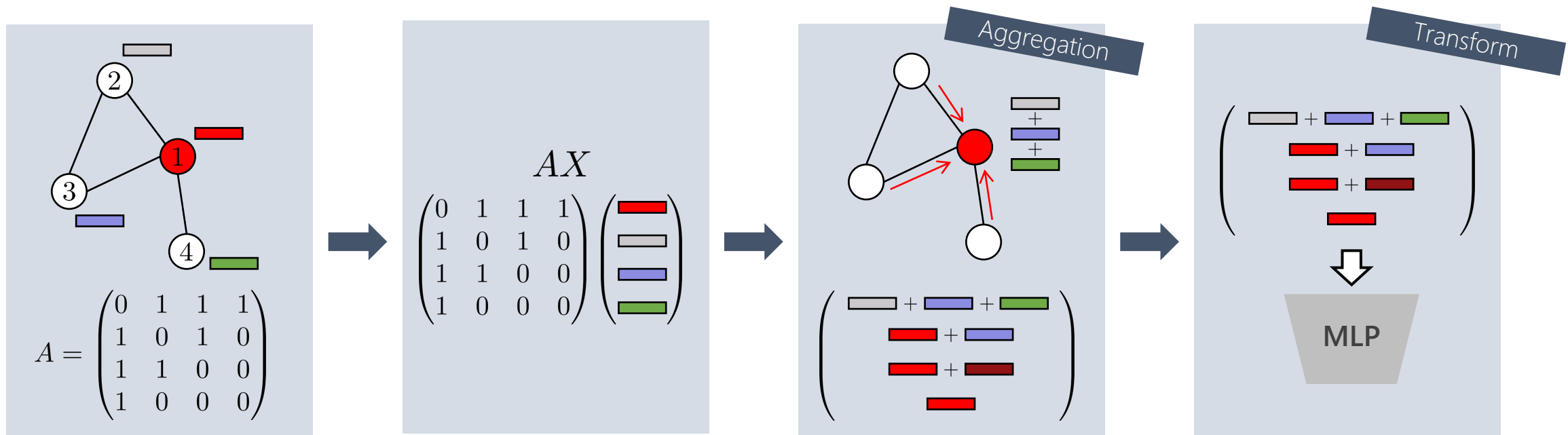
**Part 1: A practical introduction to graphs and graph neural networks**

# **Understanding of the general framework of graph neural networks (GNNs)**

# A simple, popular, and straightforward GNN

GCN (Graph Convolutional Network): Kipf & Welling, ICLR 2017

We are now ready to understand the basic principles of GNN, by looking at the most popular architecture.



Notice that, this whole procedure can be neatly expressed as:  $\sigma(AX\Theta)$

Non-linear activation function  $\sigma(\cdot)$

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

Node feature matrix  $X \in \mathbb{R}^{n \times d}$

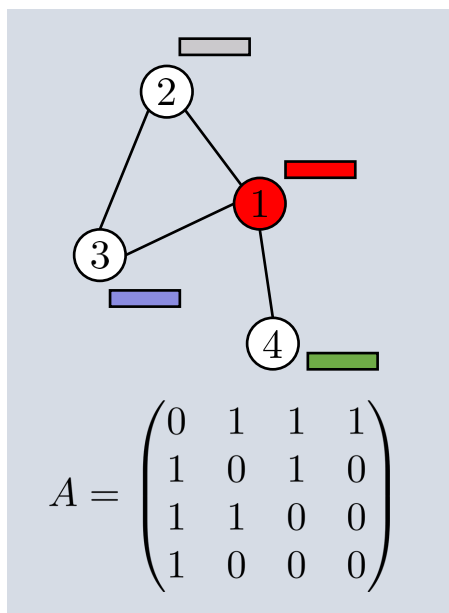
Learnable matrix  $\Theta \in \mathbb{R}^{d \times d'}$

n: # of nodes

d: node feature dimensions

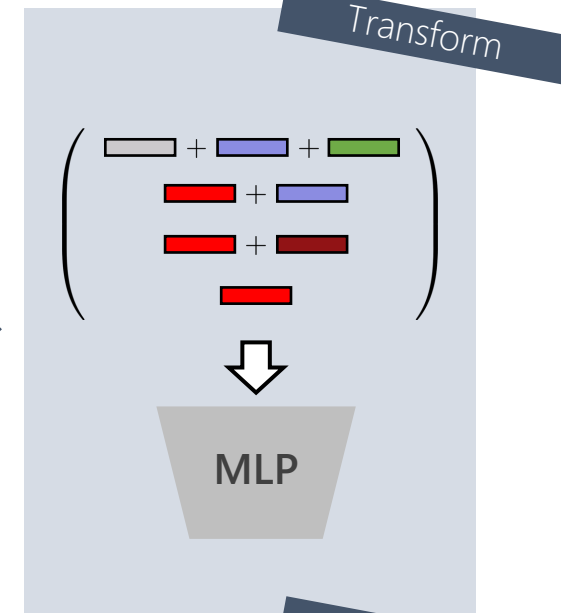
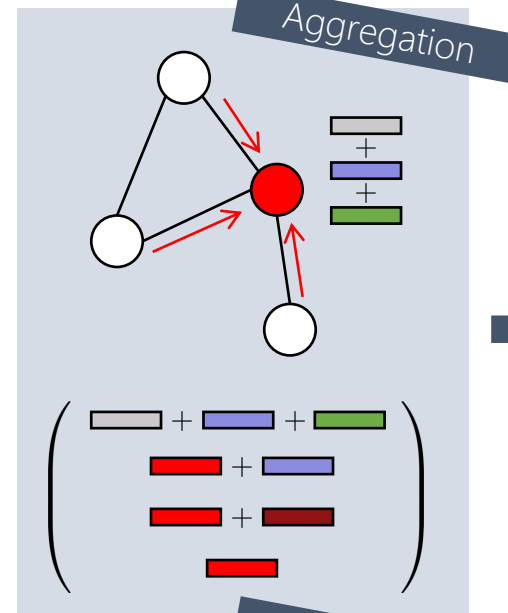
d': dimension for the next layer

# A deeper look into the node ordering problem

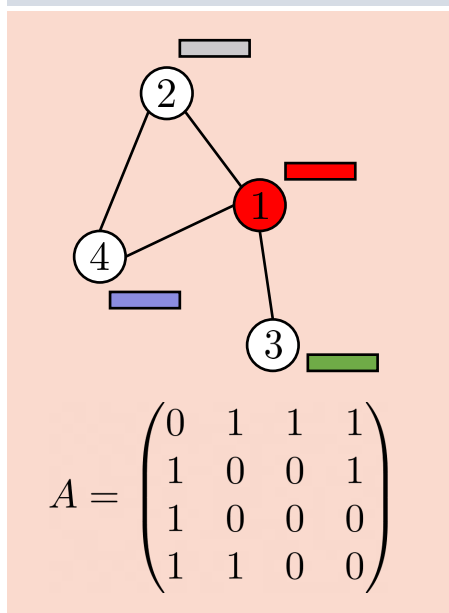


Matrix multiplication  $AX$ :

$$\begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \text{red} \\ \text{grey} \\ \text{blue} \\ \text{green} \end{pmatrix}$$

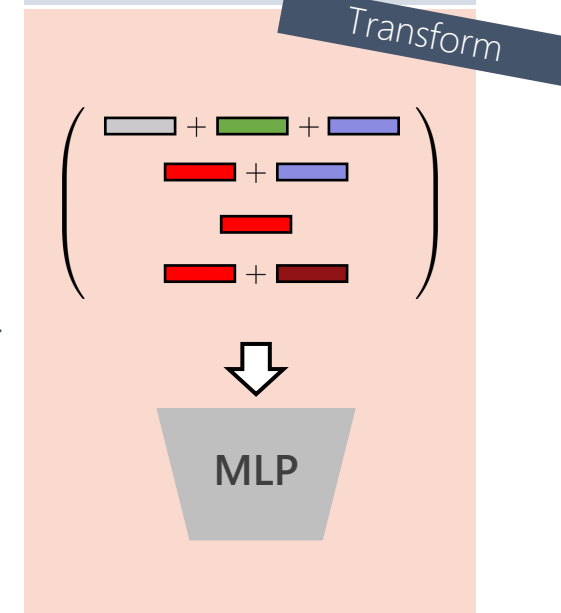
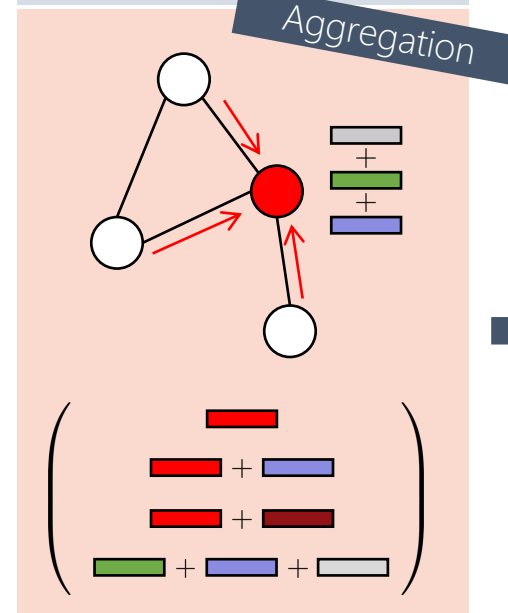


Alternate universe?  
(Different node ordering)

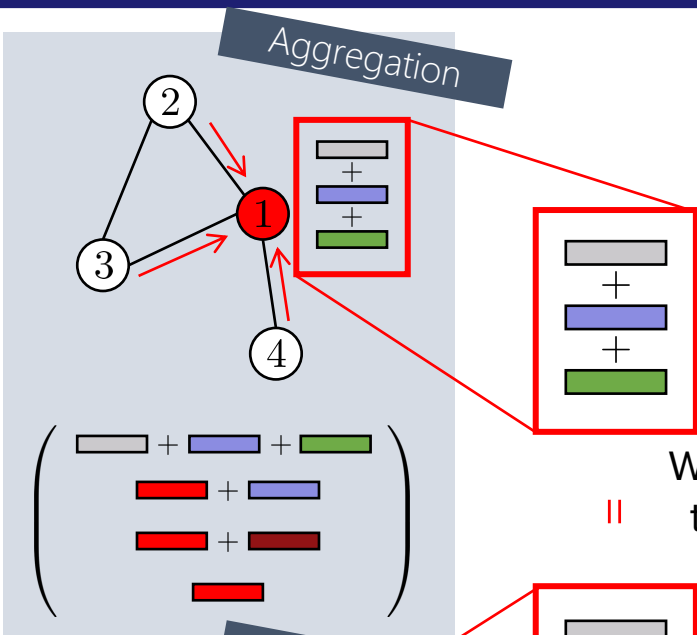


Matrix multiplication  $AX$ :

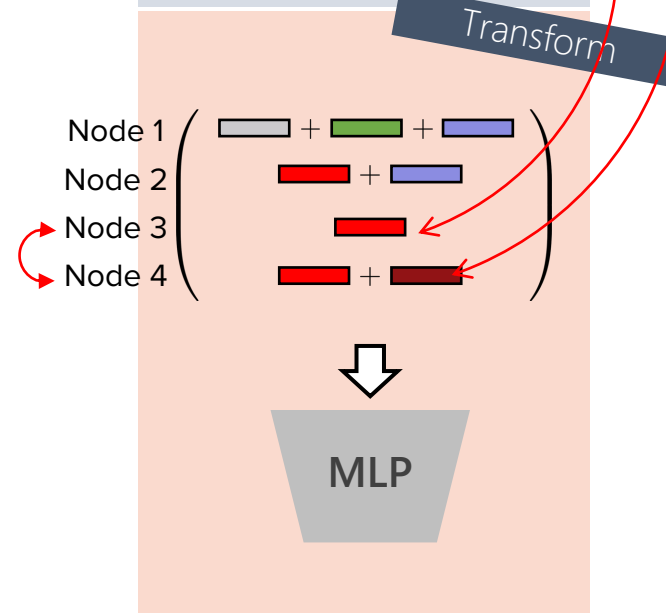
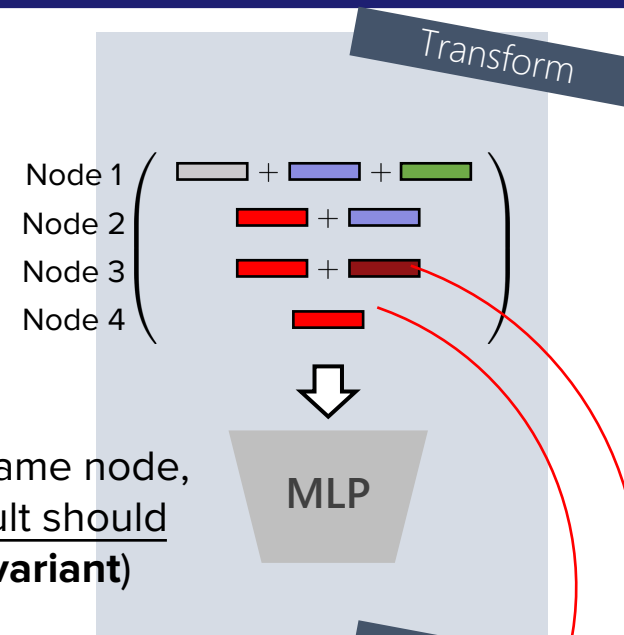
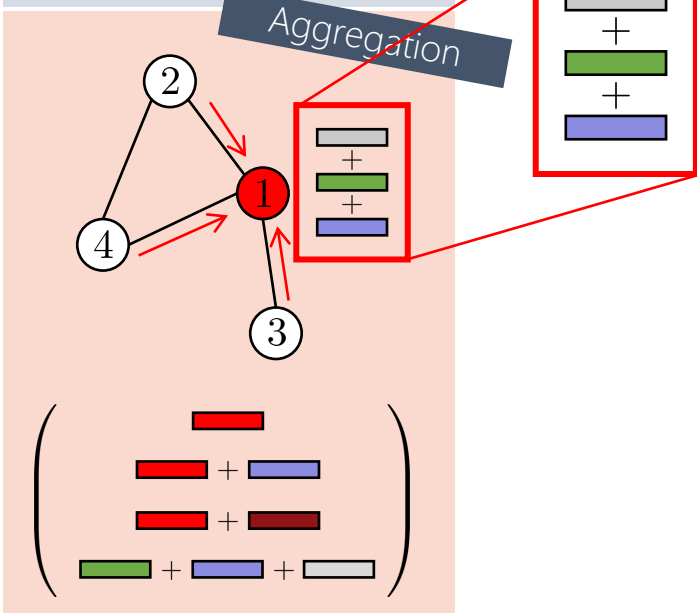
$$\begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \text{red} \\ \text{grey} \\ \text{green} \\ \text{blue} \end{pmatrix}$$



# A deeper look into the node ordering problem



We are still computing for the same node, therefore the aggregation result should not change (**permutation invariant**)



As we have changed the node order, this should also be reflected in the embedding matrix (**permutation equivariant**)

# Practical design choices of GCN

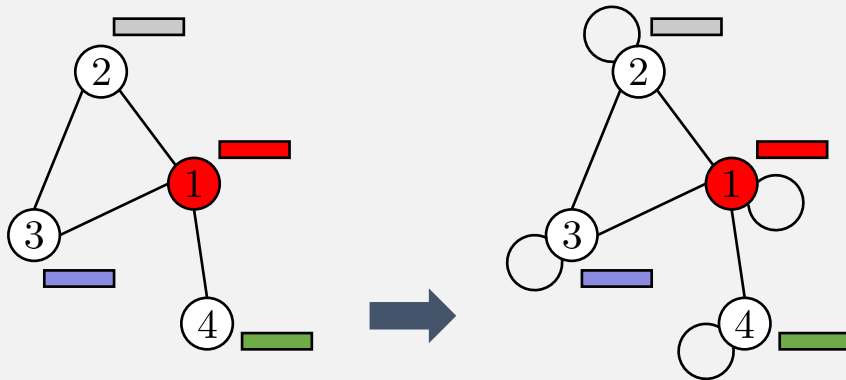
Of course, we can get creative with the graph structure to solve some practical issues

Problem 1: The information of the neighbor nodes are aggregated but not the node itself.

Problem 2: The adjacency matrix is not normalized, and the scale of the feature vectors may explode for repeated layers.

## Resolution to problem 1

Add **self-loops** to each node

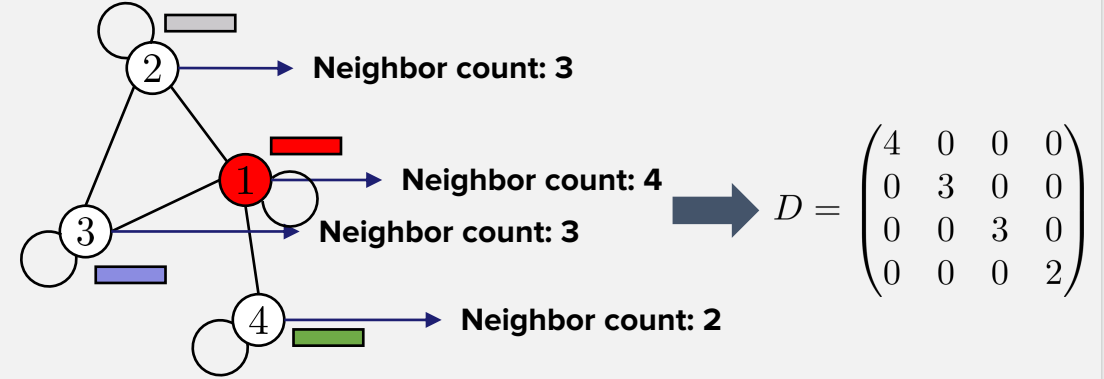


$$A = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

$$\hat{A} = A + I = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

## Resolution to problem 2

**Normalization** of  $\hat{A}$ :



$$D = \begin{pmatrix} 4 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 2 \end{pmatrix}$$

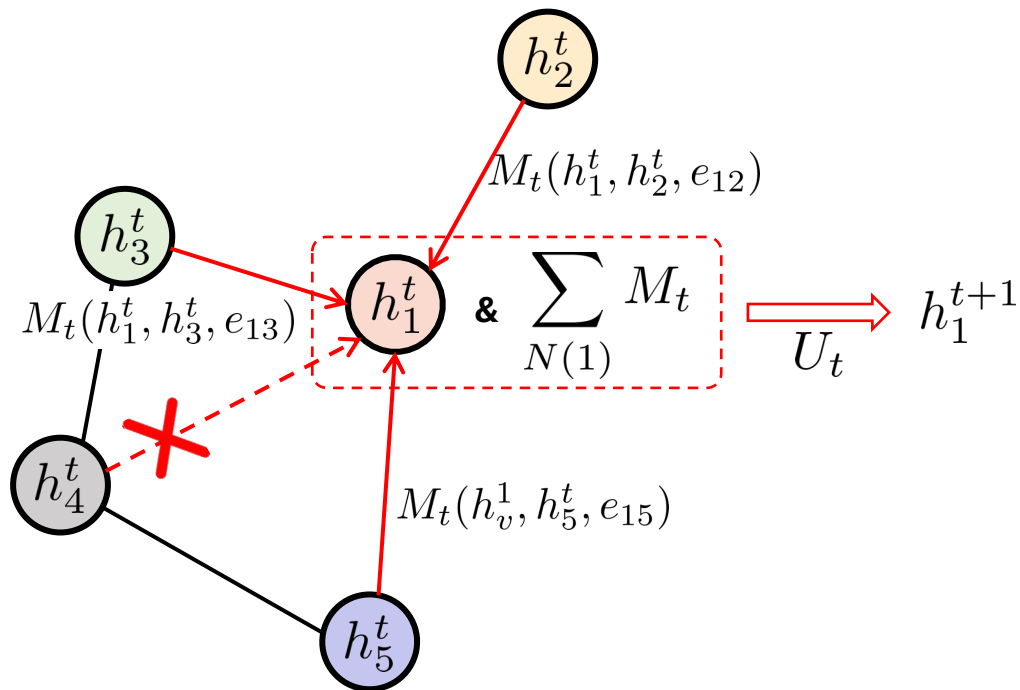
$$\tilde{A} = \hat{D}^{-1/2} \hat{A} \hat{D}^{-1/2} = \begin{pmatrix} \frac{1}{4} & \frac{1}{\sqrt{12}} & \frac{1}{\sqrt{12}} & \frac{1}{\sqrt{8}} \\ \frac{1}{\sqrt{12}} & \frac{1}{3} & \frac{1}{3} & 0 \\ \frac{1}{\sqrt{12}} & \frac{1}{3} & \frac{1}{3} & 0 \\ \frac{1}{\sqrt{8}} & 0 & 0 & \frac{1}{2} \end{pmatrix}$$

Final layer of GCN:  $\sigma(\tilde{A}X\Theta)$

# Abstraction: A general message-passing layer of GNNs

## 1. Message passing phase (Aggregation)

$$m_v^{t+1} = \sum_{w \in N(v)} M_t(h_v^t, h_w^t, e_{vw})$$

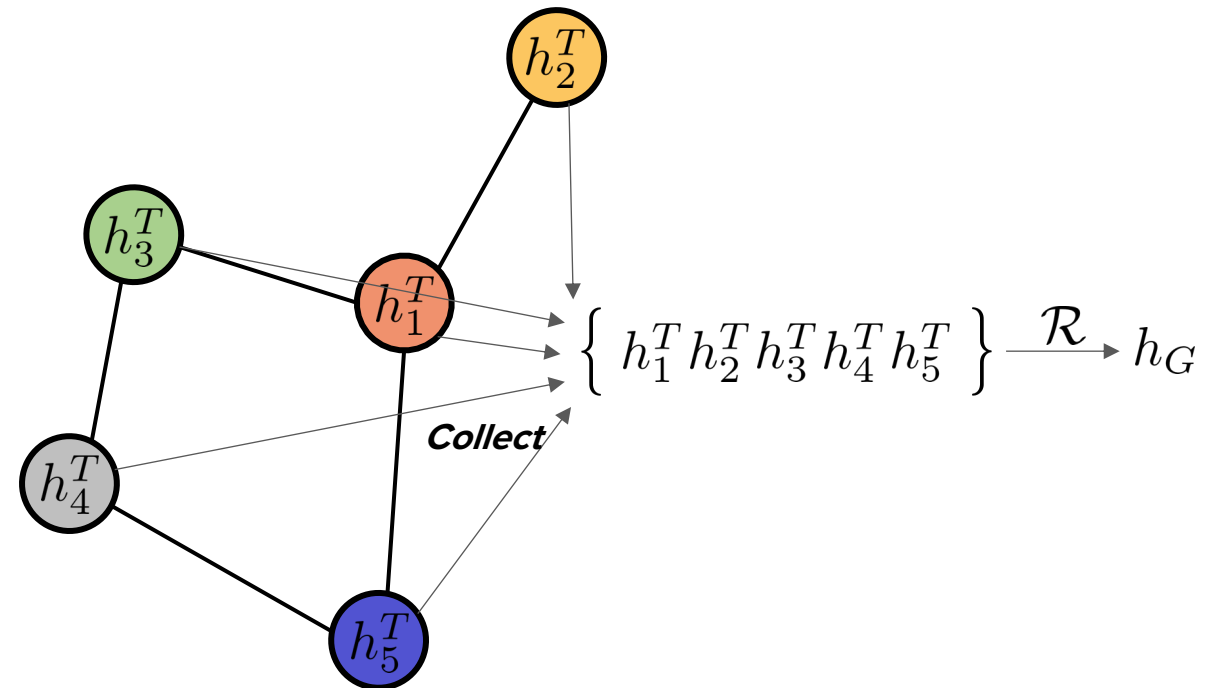


## 2. Update phase (Transformation)

$$h_v^{t+1} = U_t(h_v^t, m_v^{t+1})$$

## 3. Readout phase (Only for graph-level tasks)

$$h_G = \mathcal{R}(h_1^T, \dots, h_V^T)$$



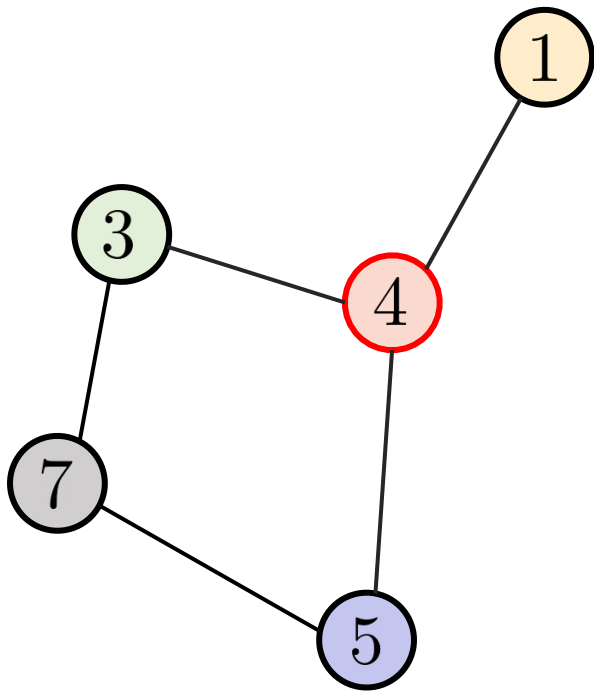
\*Usually, we cite these papers for the term “message-passing”

[First formal introduction of the concept] Gilmer et al., “Neural Message Passing for Quantum Chemistry”, ICML 2017

[Comprehensive discussion & abstraction] Bronstein et al., Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges, arXiv 2021

# Abstraction: A general message-passing layer of GNNs

## GNN layer (Message-passing neural networks)



$$\mathbf{h}_u = \phi \left( \mathbf{x}_u, \bigoplus_{v \in \mathcal{N}_u} \psi(\mathbf{x}_u, \mathbf{x}_v) \right)$$



This operation must be permutation invariant to ensure the same result for different node orderings!  
**Summation / Average / Max pooling** etc.

So if we re-describe GCN for node 4, it would be...

$$\mathcal{N}_u = \{1, 3, 5\} \cup \{4\} \quad \psi(\mathbf{x}_u, \mathbf{x}_1) = \frac{1}{\sqrt{2 \times 4}} \mathbf{x}_1 \quad \phi = \text{MLP}$$

\*Usually, we cite these papers for the term “message-passing”

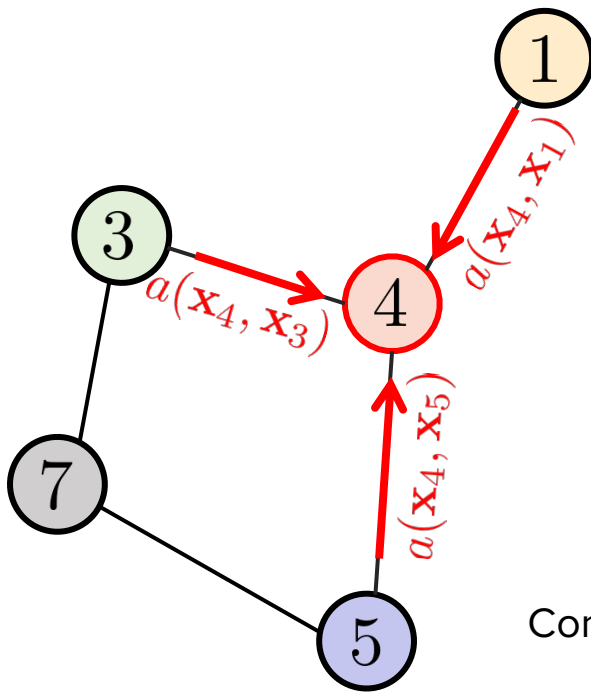
[First formal introduction of the concept] Gilmer et al., “Neural Message Passing for Quantum Chemistry”, ICML 2017

[Comprehensive discussion & abstraction] Bronstein et al., Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges, arXiv 2021



# Abstraction: A general message-passing layer of GNNs

Example: GAT (Veličković et al., ICLR 2018)



$$\mathbf{h}_u = \phi \left( \mathbf{x}_u, \bigoplus_{v \in \mathcal{N}_u} \psi(\mathbf{x}_u, \mathbf{x}_v) \right)$$

The **model decides** the strength of the ‘propagation’

$$\mathcal{N}_u = \{1, 3, 5\} \cup \{4\} \quad \psi(\mathbf{x}_u, \mathbf{x}_1) = a(\mathbf{x}_u, \mathbf{x}_1) \mathbf{x}_1 \quad \phi = \text{MLP}$$

Compare this part with GCN, the role of attention will be much more clear:

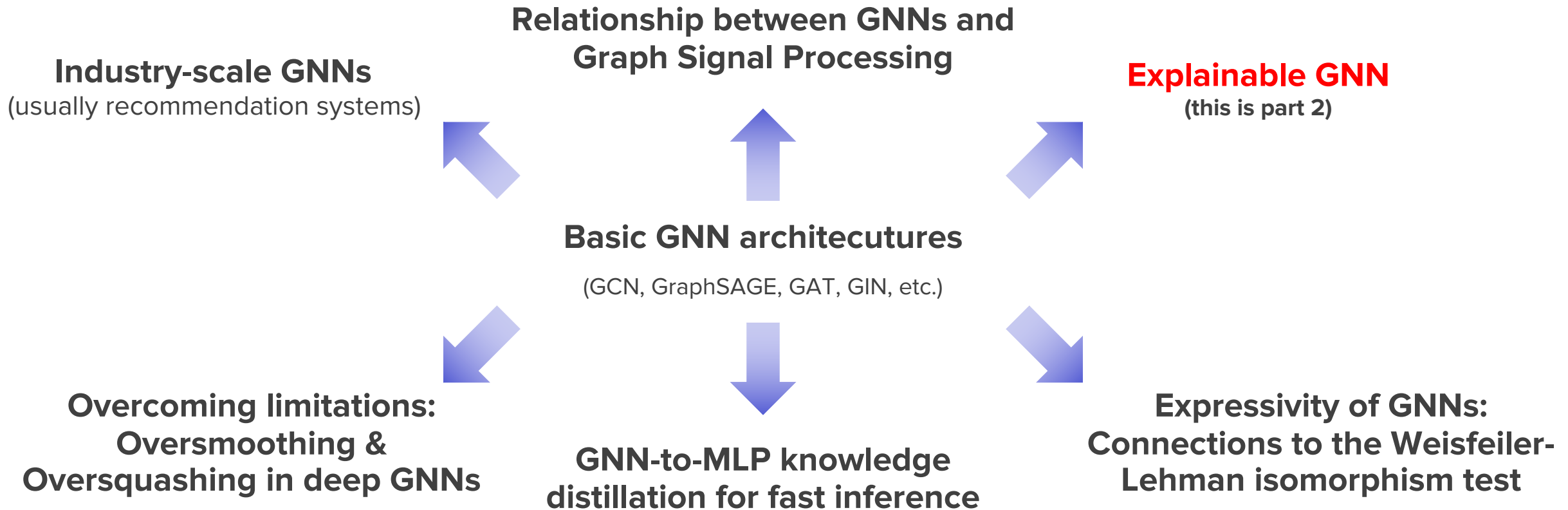
$$\psi(\mathbf{x}_u, \mathbf{x}_1) = \frac{1}{\sqrt{2 \times 4}} \mathbf{x}_1$$

The **node degree** decides the strength of the ‘propagation’

# There are a lot of fun & fundamental topics in the GNN literature

22

To name a few...



## PyTorch Geometric ([link](#))


### PyG Documentation

 **PyG** (*PyTorch Geometric*) is a library built upon  **PyTorch** to easily write and train Graph Neural Networks (GNNs) for a wide range of applications related to structured data.




It consists of various methods for deep learning on graphs and other irregular structures, also known as **geometric deep learning**, from a variety of published papers. In addition, it consists of easy-to-use mini-batch loaders for operating on many small and single giant graphs, **multi GPU-support**, **torch.compile** support, **DataPipe** support, a large number of common benchmark datasets (based on simple interfaces to create your own), the **GraphGym** experiment manager, and helpful transforms, both for learning on arbitrary graphs as well as on 3D meshes or point clouds.

- Jure Leskovec (Stanford/KumoAI/Snapchat)
- Faster library updates
- (Seems like) A larger community

## Deep Graph Library ([link](#))



The banner for the Deep Graph Library features a blue background with a network graph pattern. It includes the text "DEEP GRAPH LIBRARY" and "Easy Deep Learning on Graphs". Below this, there are two buttons: "Install" (green) and "GitHub" (white). Below the banner, there are three columns of text describing the library's features.

Framework Agnostic	Efficient And Scalable	Diverse Ecosystem
Build your models with PyTorch, TensorFlow or Apache MXNet.	Fast and memory-efficient message passing primitives for training Graph Neural Networks. Scale to giant graphs via multi-GPU acceleration and distributed training infrastructure.	DGL empowers a variety of domain-specific projects including <b>DGL-KE</b> for learning large-scale knowledge graph embeddings, <b>DGL-LifeSci</b> for bioinformatics and cheminformatics, and many others.
  		

- Slower library updates
- Variable framework support
- Can be tricky to install older versions



- Additional library: NetworkX ([link](#)) – Library for **graphs in general**
  - Not a library for ML/DL
  - Often used in junction with PyG/DGL

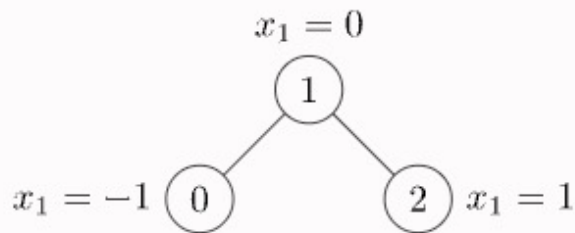
# Wait, just one more slide on the library for graph learning

## A very small PyG example

```
import torch
from torch_geometric.data import Data

edge_index = torch.tensor([[0, 1, 1, 2],
                           [1, 0, 2, 1]], dtype=torch.long)
x = torch.tensor([[ -1], [0], [1]], dtype=torch.float)

data = Data(x=x, edge_index=edge_index)
>>> Data(edge_index=[2, 4], x=[3, 1])
```



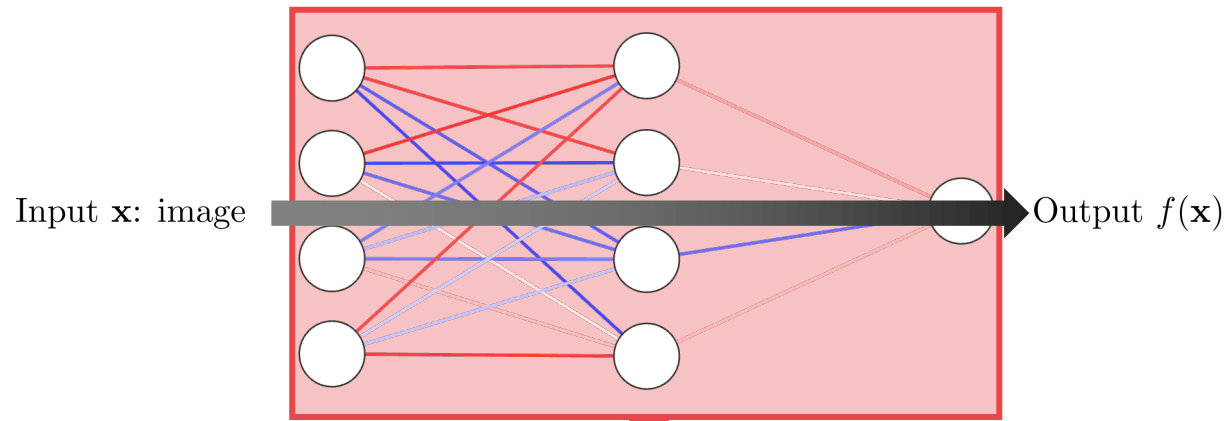
- You *at minimum* need to define `data.edge_index`
- Node features are usually represented as `data.x`
- Don't forget to include both directions for undirected graphs
- Most graph processing/manipulation tools are in `torch_geometric.utils`. Or just transform into a `networkx` object!

**Part 2: Towards explainable graph learning with attention**

## **Understanding the basic concepts of explainable AI**

# Why explainable AI?

Neural networks have complex structure with a lot of parameters.



- Large number of parameters
  - High-nonlinearity
  - Complex inner structure
- has made them very *\*hard to interpret and understand*, making it a **black box**.

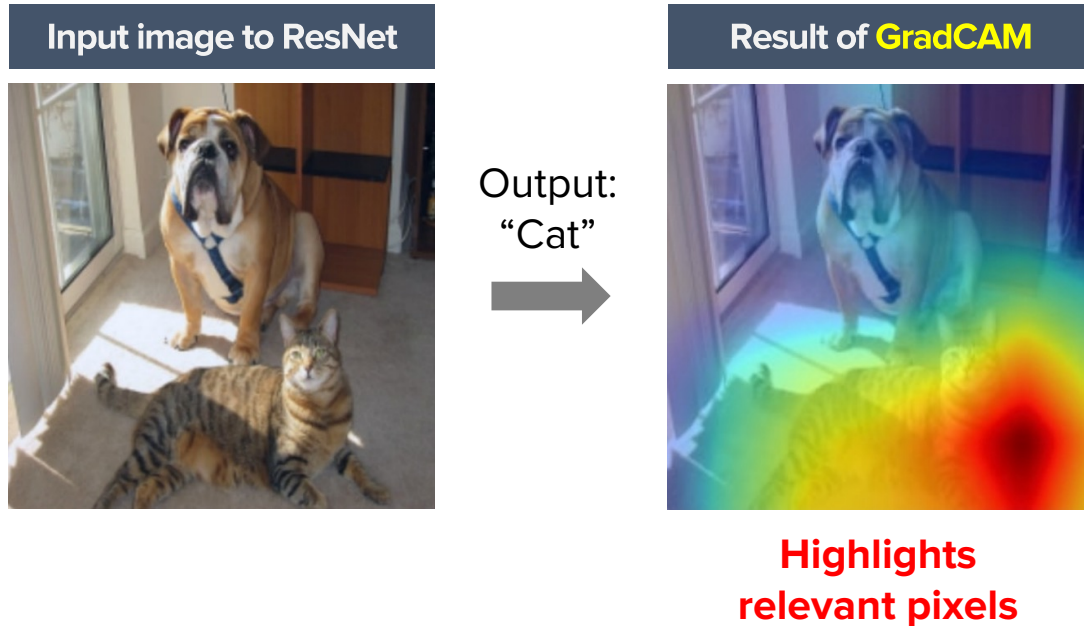
Main question of Explainable AI:

**Why** has a neural network model made its prediction?



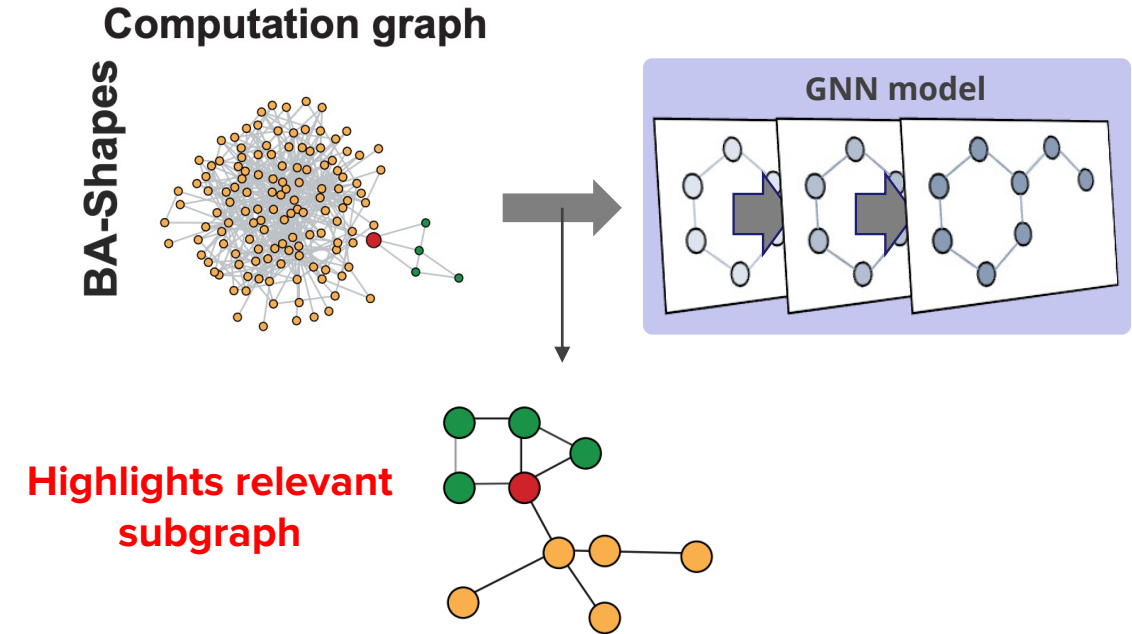
# Attribution maps: The most popular type of explanation

**Attribution maps** are one of the most popular ways, especially in CV and NLP.



Example: DTD [1], LRP [2], LIME [3], GradCAM [4], ...

Similar approaches are also popular in **GNN explanations**, too.



Example: GNNExplainer [5], PGExplainer [6], ...

- [1] Montavon et al., "Explaining nonlinear classification decisions with deep Taylor decomposition", Pattern Recognit. 65: 211-22 (2017)
- [2] Bach et al., "On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation", PLOS ONE 10(7): e0130140.
- [3] Riberiro et al., "'Why Should I Trust You?': Explaining the Predictions of Any Classifier", KDD 2016
- [4] Selvaraju et al., "Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization", ICCV 2017
- [5] Ying et al., "GNNExplainer: Generating Explanations for Graph Neural Networks", NeurIPS 2019
- [6] Luo et al., "Parameterized explainer for graph neural network", NeurIPS 2020

**Part 2: Towards explainable graph learning with attention**

**Can we understand graph attention networks using attention?**



# What is attention?

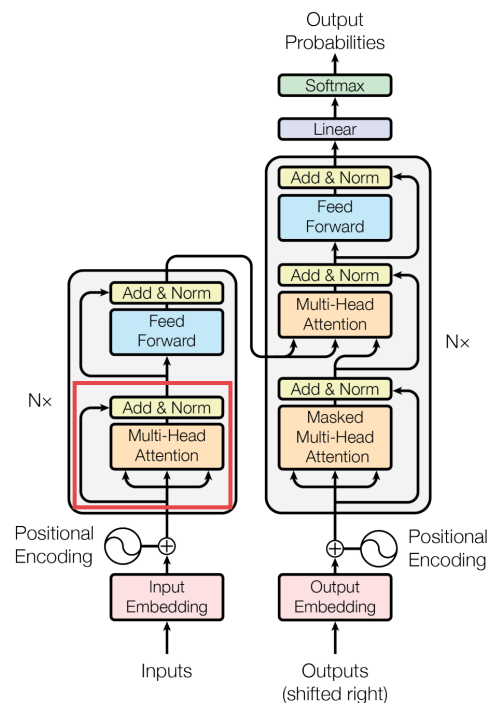
A weighted sum operation where the **weights are determined by the model**

The FBI is chasing a criminal on the run .  
 The FBI is chasing a criminal on the run .  
 The FBI is chasing a criminal on the run .  
 The FBI is chasing a criminal on the run .  
 The FBI is chasing a criminal on the run .  
 The FBI is chasing a criminal on the run .  
 The FBI is chasing a criminal on the run .  
 The FBI is chasing a criminal on the run .

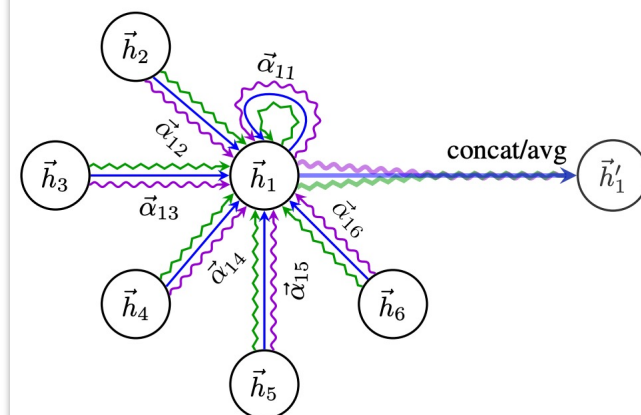
Figure 1: Illustration of our model while reading the sentence *The FBI is chasing a criminal on the run*. Color *red* represents the current word being fixated, *blue* represents memories. Shading indicates the degree of memory activation.

**Long Short-Term Memory Networks  
(LSTMN)**

(Cheng et al., EMNLP 2016)



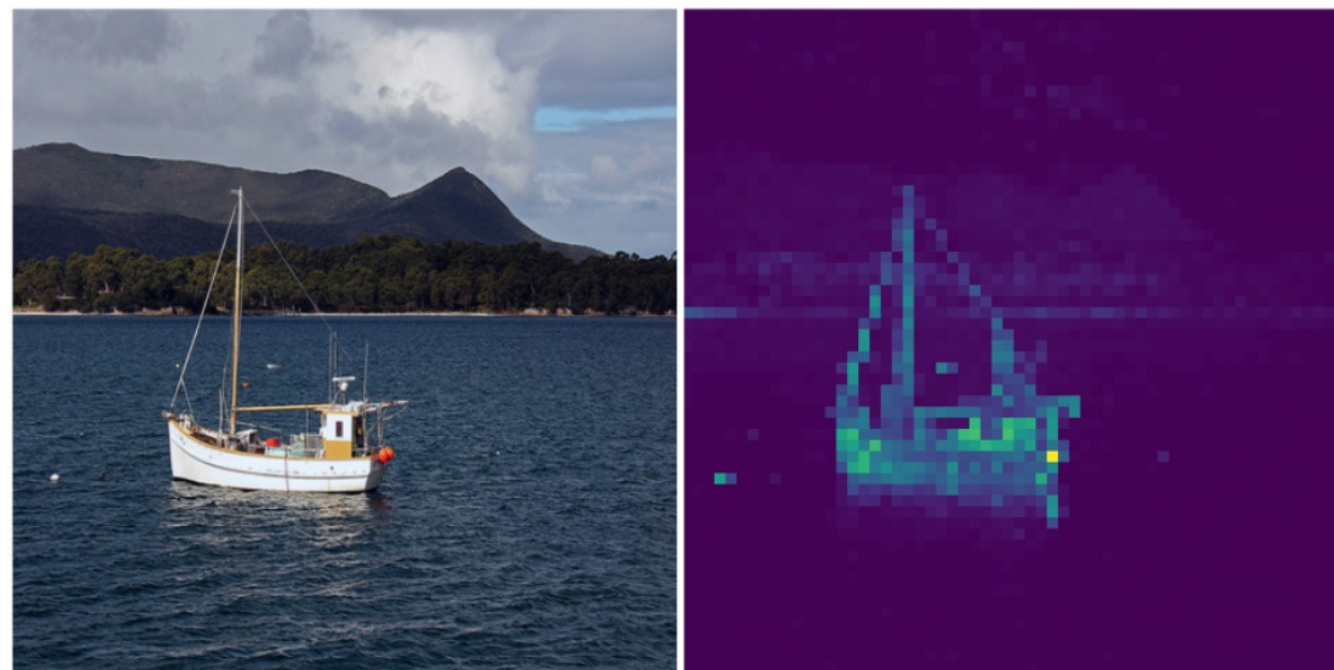
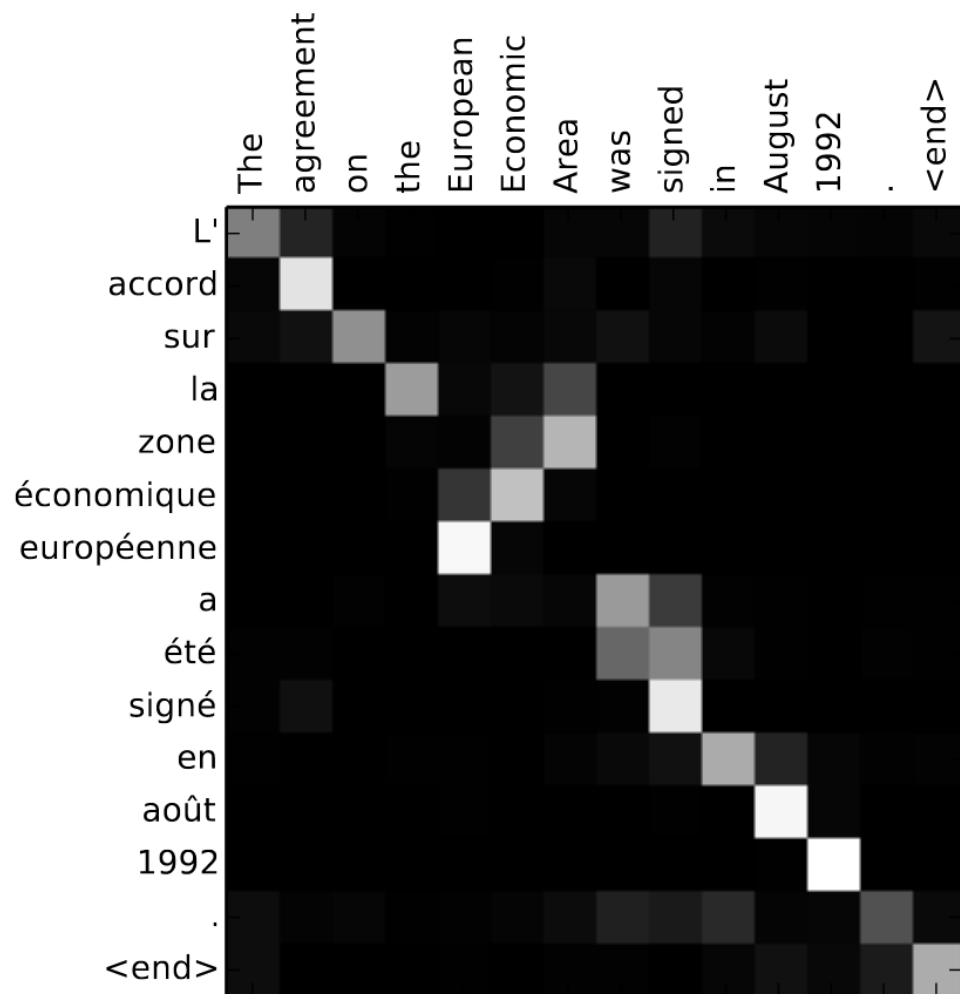
**Transformer networks**  
(Vaswani et al., NeurIPS 2017)



**Graph attention networks**  
(Velickovic et al., ICLR 2018)

# Can we interpret attention = attribution?

When we think of the role of attention, we can naturally interpret as 'where the model looks'  
 ...which is essentially **attribution maps**! (at least intuitively)



# Can we just say attention = attribution?

**Attention is heavily studied as an important candidate for model explanation**

**Is attention explanation?**

**Attention is not Explanation**

**Sarthak Jain**

Northeastern University  
jain.sar@husky.neu.edu

**Byron C. Wallace**

**Attention is not not Explanation**

**Sarah Wiegrefe\***

**Yuval Pinter\***

School of Interactive Computing  
Georgia Institute of Technology  
uyp@gatech.edu

**Is Attention Explanation? An Introduction to the Debate**

**Adrien Bibal, Rémi Cardon, David Alfter, Rodrigo Wilkens, Xiaoou Wang,**

**Thomas François\* and Patrick Watrin\***

CENTAL, IL&C, University of Louvain, Belgium

{adrien.bibal, remi.cardon,  
thomas.francois,patrick.watrin}

**How Much Does Attention Actually Attend?**

**Questioning the Importance of Attention in Pretrained Transformers**

**Michael Hassid<sup>♡</sup> Hao Peng<sup>◇</sup> Daniel Rotem<sup>♡</sup> Jungo Kasai<sup>★</sup> Ivan Montero<sup>★\*</sup>  
Noah A. Smith<sup>◇</sup> Roy Schwartz<sup>♡</sup>**

<sup>♡</sup>School of Computer Science & Engineering, Hebrew University of Jerusalem

<sup>◇</sup>Allen Institute for Artificial Intelligence <sup>★</sup>Apple, Inc.

<sup>★</sup>Paul G. Allen School of Computer Science & Engineering, University of Washington  
{michael.hassid,daniel.rotem,roy.schwartz1}@email.huji.ac.il  
haop@allenai.org {jksai,nasmith}@cs.washington.edu ivamon@apple.com

**How to generate better  
attention heatmaps in transformers?**

**Quantifying Attention Flow in Transformers**

**Samira Abnar**

ILLC, University of Amsterdam  
s.abnar@uva.nl

**Willem Zuidema**

ILLC, University of Amsterdam

**Transformer Interpretability Beyond Attention Visualization**

Hila Chefer<sup>1</sup> Shir Gur<sup>1</sup> Lior Wolf<sup>1,2</sup>

<sup>1</sup>The School of Computer Science, Tel Aviv University

<sup>2</sup>Facebook AI Research (FAIR)

**Generic Attention-model Explainability for Interpreting  
Bi-Modal and Encoder-Decoder Transformers**

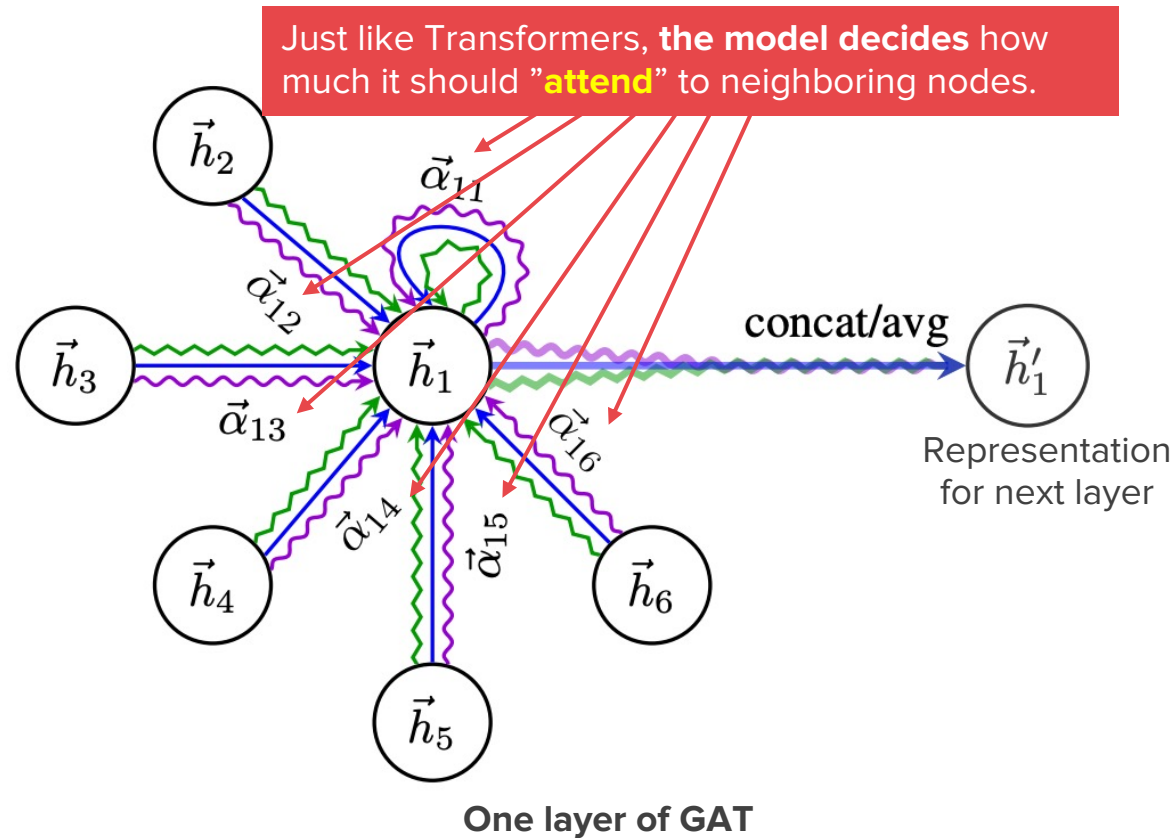
Hila Chefer<sup>1</sup> Shir Gur<sup>1</sup> Lior Wolf<sup>1,2</sup>

<sup>1</sup>The School of Computer Science, Tel Aviv University

<sup>2</sup>Facebook AI Research (FAIR)

# So graph attention networks (GATs) are explainable?

Well, the majority of the literature seems to overlook GATs as a valid candidate of ‘inherently explainable model’

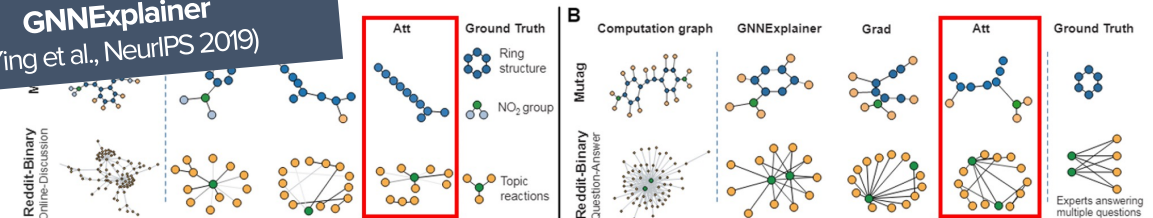


**GNN-XAI evaluation**  
(Sanchez-Lengeling et al., NeurIPS 2020)

	Node-level tasks											
	GraphSAGE	GraphNets	GAT	GCN	BA-Community	MPNN	GraphNets	GAT	GCN	MPNN	GraphNets	GAT
GradInput	0.27	0.27	0.27	0.38	0.38	0.38	0.38	0.38	0.62	0.62	0.62	0.62
SmoothGrad(GT)	0.58	0.64	0.39	0.52	0.51	0.5	0.5	0.5	0.65	0.71	0.66	0.67
GradCAM-last	0.79	0.84	0.86	0.8	0.7	0.67	0.68	0.61	0.7	0.77	0.81	0.7
GradCAM-all	0.67	0.78	0.65	0.76	0.67	0.71	0.73	0.57	0.68	0.7	0.67	0.68
IG	--	--	--	0.81	0.75	0.72	0.62	0.62	--	--	--	--
Attention Weights	--	--	--	0.5	--	--	--	0.5	--	--	--	0.49

“...have several blocks and attention heads, so for each component we take their average to combine them to a scalar value assigned to each edge.”

**GNNExplainer**  
(Ying et al., NeurIPS 2019)



“...it is **not obvious** which attention weights need to be used for edge importance, ... Each edge’s importance is thus **computed as the average attention weight across all layers.**”

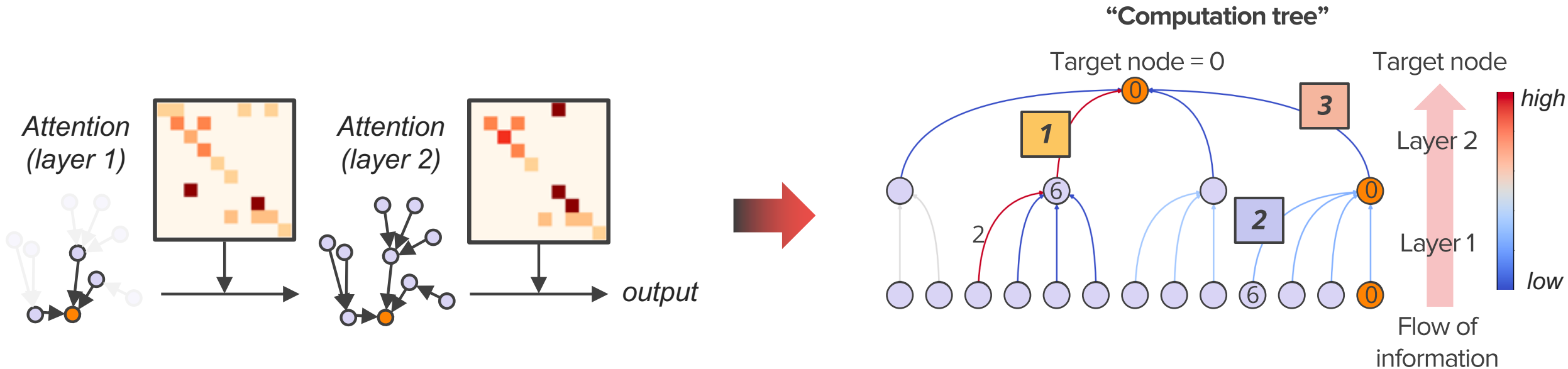
**PGExplainer**  
(Luo et al., NeurIPS 2020)

	Explanation AUC					
	0.750	0.905	0.612	0.717	0.783	
PGExplainer	0.739	0.824	0.667	0.674	0.765	
Improve	4.1%	13.0%	4.1%	3.7%	24.7%	11.5%
	Inference Time (ms)					
	650.60	696.61	690.13	713.40	934.72	409.98
PGExplainer	10.92	24.07	6.36	6.72	80.13	9.68
Speed-up	59x	29x	108x	106x	12x	42x

“Each edge’s importance is obtained by **averaging its attention weights across all attention layers.**”

# GATs are explainable... with a little bit of extra effort

We just need to consider the ‘flow’ of information better within the GAT model



Proposed calculation (Shin et al., AAAI 2025)

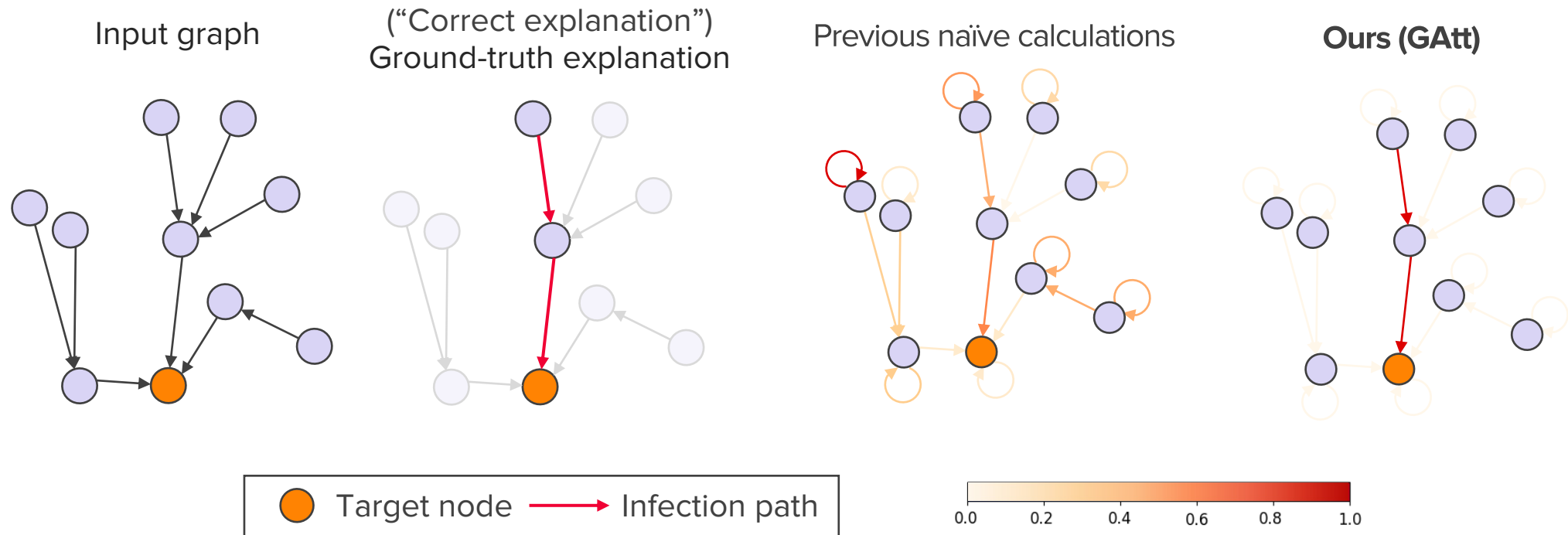
$$\phi_{6,0}^0 = 1 + 3 \times 2$$

“Importance of edge (6, 0) when the target node is 0”

1. Add all occurrences in the computation tree
2. Multiply other attention weights along the flow of information

# And we can immediately get better attribution maps

## Case study: Infection dataset (Faber et al., KDD 2021)



\*Note: We do not change the GAT model. Remember, our contribution is how to calculate attribution maps after the training is complete



## Part 1: A practical introduction to graphs and graph neural networks

1. Understanding of **graphs** as a **general data type**
  - Nodes & Edges (“connections”)
  - A lot of things can be represented as a graph, including images!
2. Understanding of the general framework of **graph neural networks (GNNs)**
  - Message-passing = GNN (Unless it’s a graph transformer)
  - Aggregation + Transformation

## Part 2: Towards explainable graph learning with attention

1. Understanding the basic concepts of **explainable AI**
  - Attribution maps = “Important parts of the input”
  - A lot of GNN explanations are also attribution maps
2. **Answer to the question: Can we understand graph attention networks using attention?**
  - (Shin et al., AAAI 2025) Conclusion: YES, but with a little bit of effort
  - BTW, this conclusion is applicable to other GNNs with self-attention

**Thank you!**

Please feel free to ask any questions :)

*[jordan7186.github.io](https://jordan7186.github.io)*

*[jordan3414@yonsei.ac.kr](mailto:jordan3414@yonsei.ac.kr)*