Graph Embedding for Social Network Analysis

Jiani Zhang & Irwin King

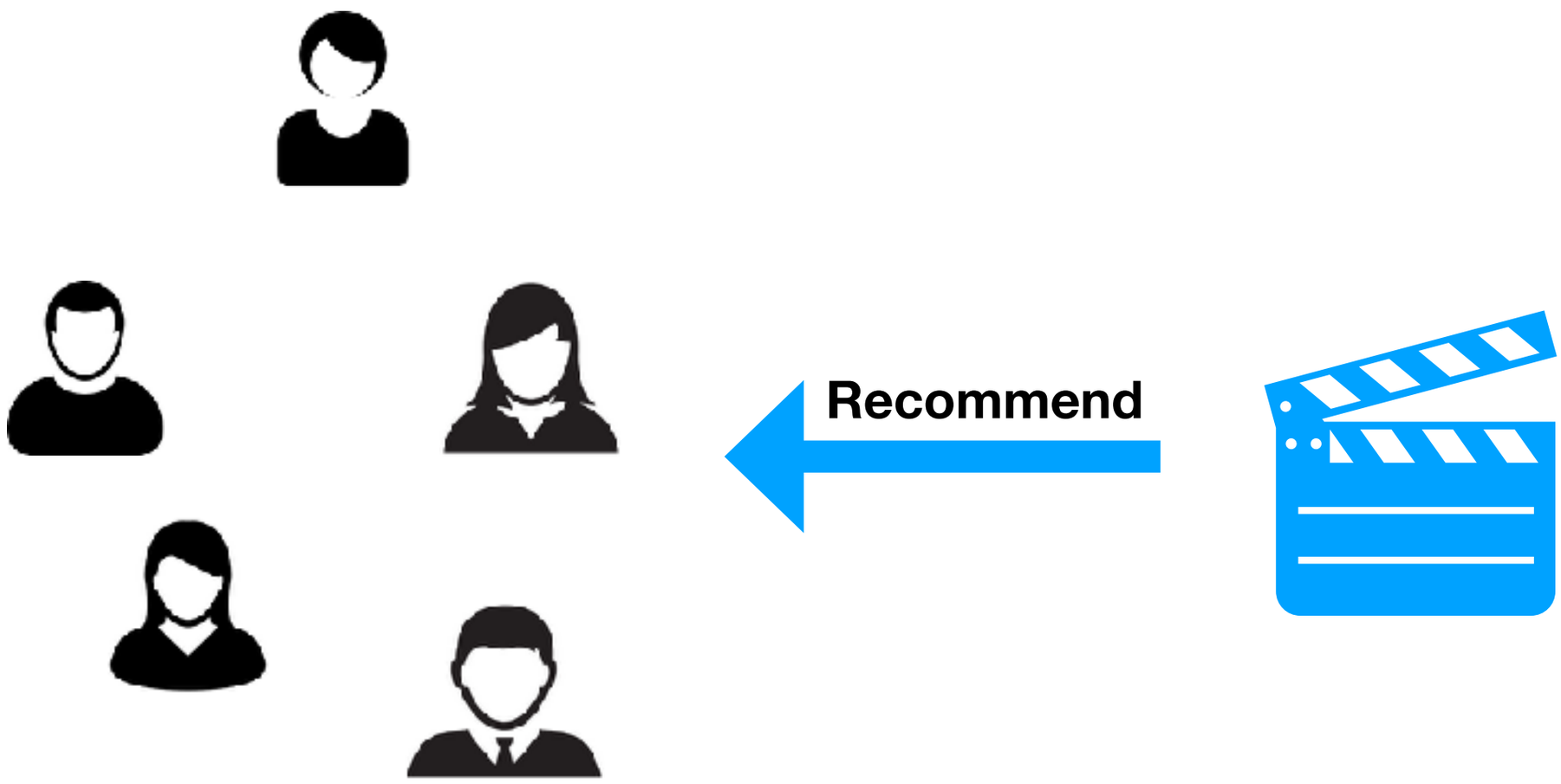
Director, Centre for eLearning Innovation and Technology (ELITE), CUHK
Director, Shenzhen Rich Media and Big Data Analytics and Application Key Lab, SZRI, CUHK
PI, The Knowledge and Education Exchange Platform (KEEP), VeriGuide System, CUHK
Former Associate Dean (Education), Faculty of Engineering, CUHK

Department of Computer Science and Engineering
The Chinese University of Hong Kong

king@cse.cuhk.edu.hk
http://www.cse.cuhk.edu.hk/~king

©2018-19 Irwin King. All rights reserved.

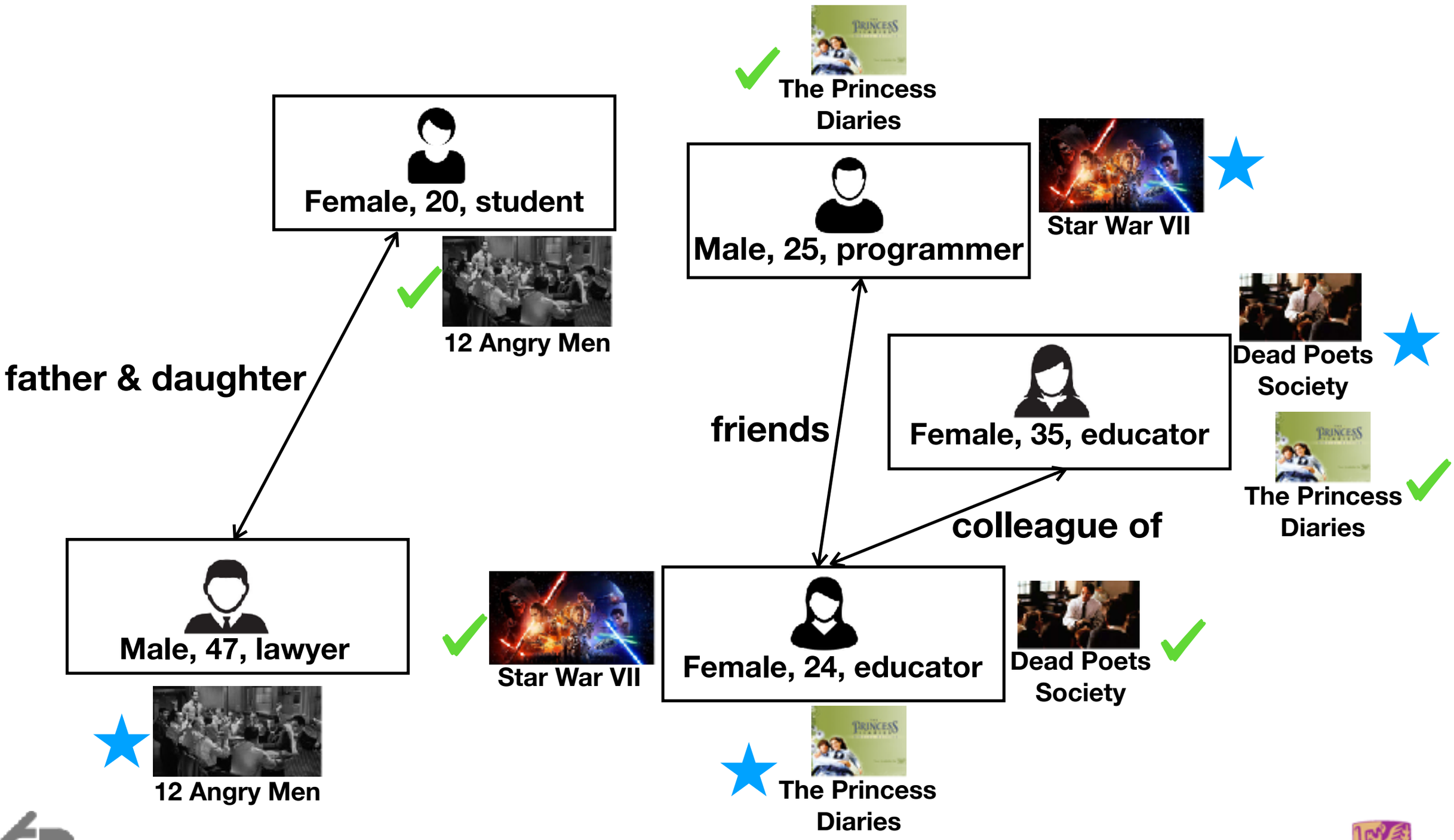
Movie Recommendation



Movie Recommendation

	 12 Angry Men	 Star War VII	 Star War VIII	 Dead Poets Society	 The Princess Diaries
Female, 20, student					★
Male, 25, programmer		★			
Female, 35, educator				★	
Female, 24, educator					★
Male, 47, lawyer	★				

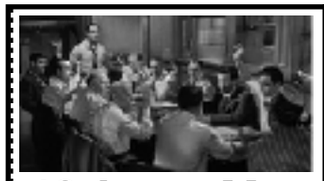
With Social Network



Movie Recommendation

	 12 Angry Men	 Star War VII	 Star War VIII	 Dead Poets Society	 The Princess Diaries
Female, 20, student	✓				★
Male, 24, programmer		★			✓
Female, 35, educator				★	✓
Female, 24, educator		✓		✓	★
Male, 47, lawyer	★				

Movie Recommendation

	 12 Angry Men	 Star War VII	 Star War VIII	 Dead Poets Society	 The Princess Diaries
Female, 20, student	✓				★
Male, 24, programmer		★	✓		✓
Female, 35, educator				★	✓
Female, 24, educator		✓		✓	★
Male, 47, lawyer	★				

Main Contents

- Backgrounds
- Graph Embedding Techniques
  - Taxonomy by Learning Tasks
  - Shallow Graph Embedding Methods
  - Deep Graph Embedding Methods
- Evaluation on Tasks
- Future Directions





Social Networks

- Social Networks are intrinsically in the form of **graphs**
  - **Nodes** (individuals) and ties
  - **Edges** or **Links** (relationships or interactions)



Applications of Graph Embedding

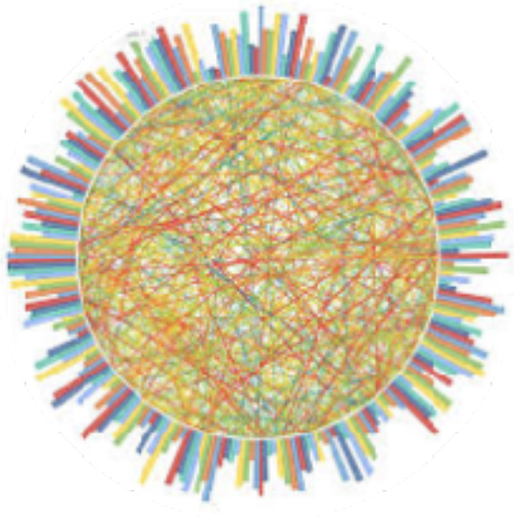
Node Classification



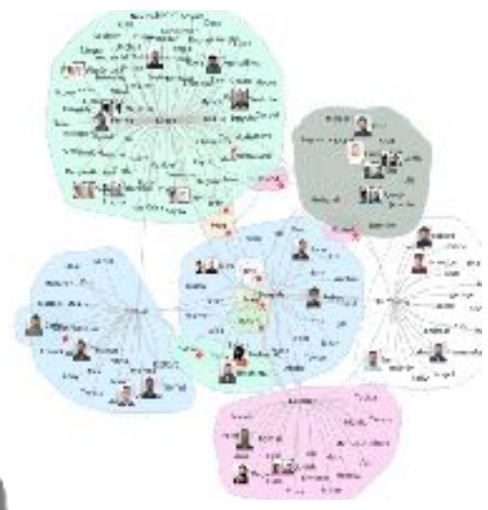
Recommendation



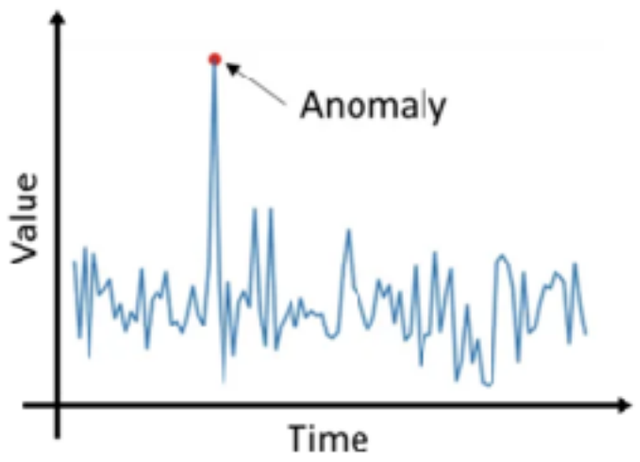
Visualisation



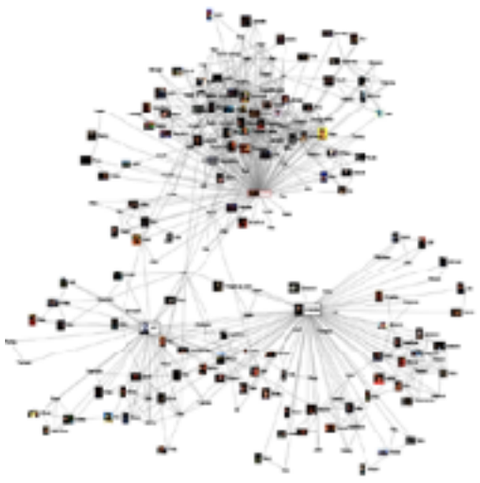
Community Detection



Anomaly Detection



Link Prediction



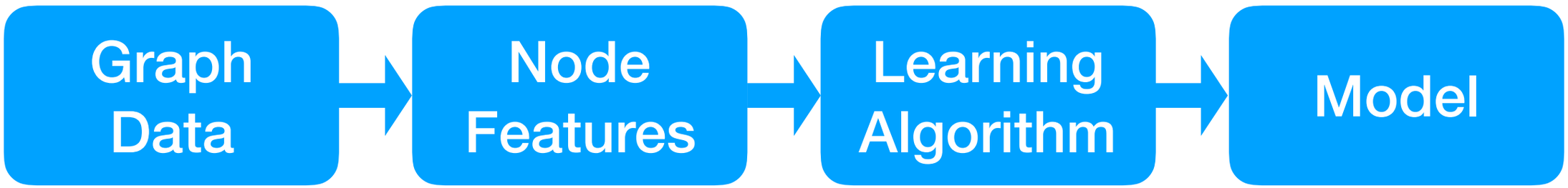
Feature Engineering

- Traditional social network analysis (SNA) measures structural properties of networks
  - **Node**: degree, PageRank score, betweenness, closeness, eigenvector, …
  - **Pairs**: #common neighbors, ...
  - **Groups**: cluster assignments, cliques, cores, clans, ...





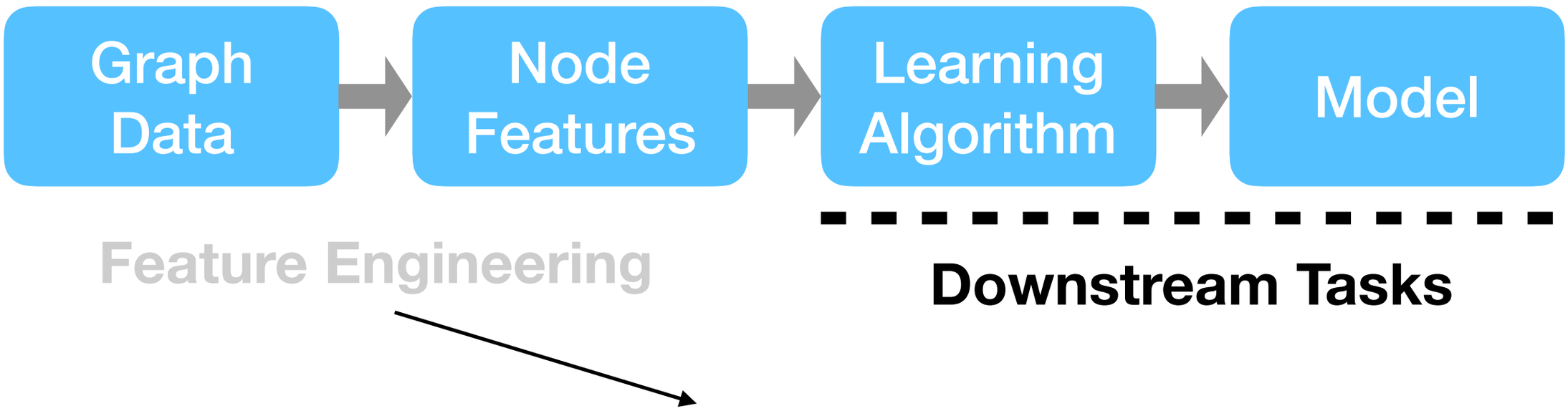
Graph Learning Lifecycle



Feature Engineering → **Feature Learning**

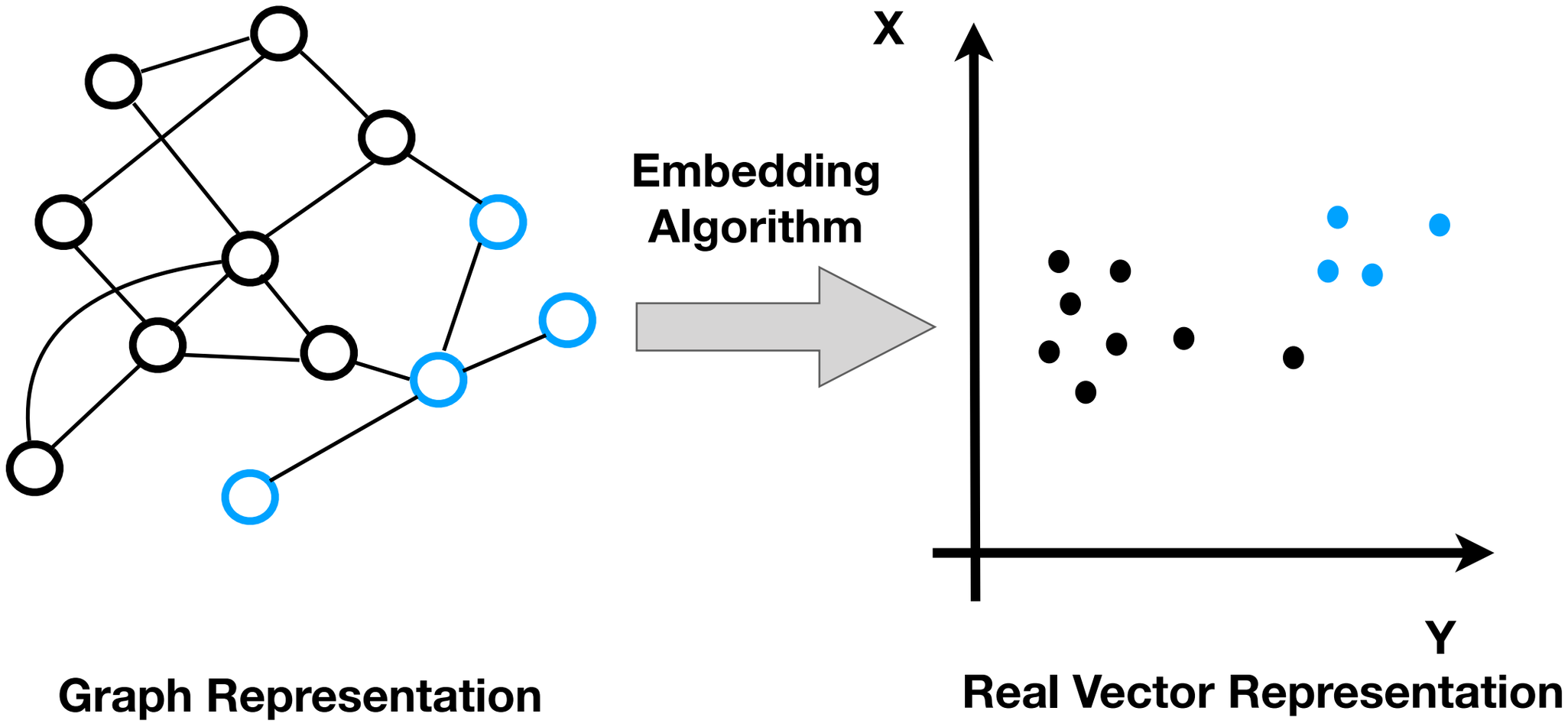
Downstream Tasks

Graph Learning Lifecycle



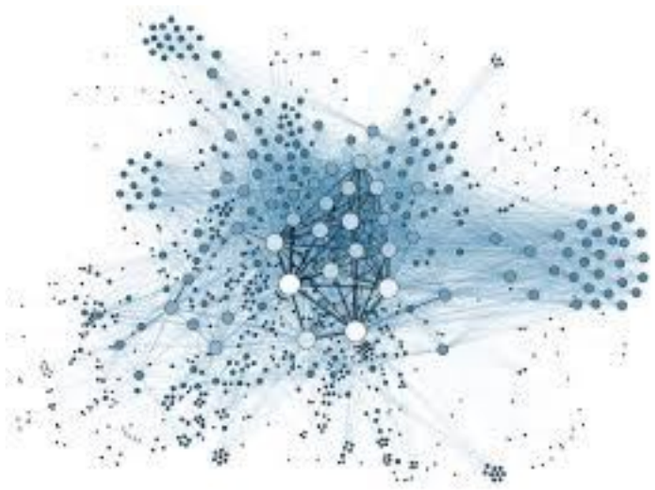
Feature Learning via graph embedding

What is Graph Embedding?



Feature Learning via Graph Embedding

Low Dimension Representation



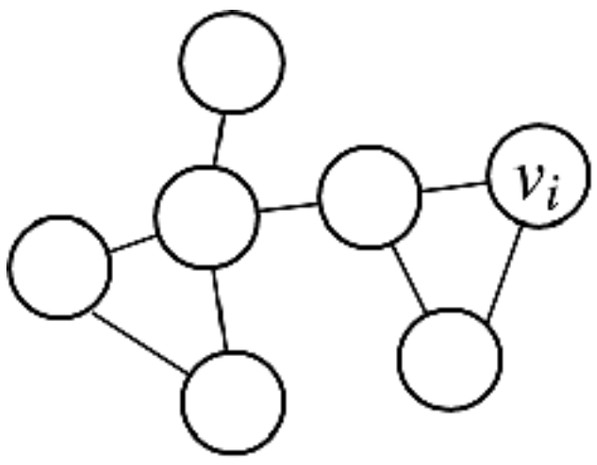
Feature Learning

Vector Representation

Task Specific Learning

Tasks

- Anomaly Detection
- Attribute Prediction
- Clustering
- Link Prediction
- ⋮



$V \to \mathbb{R}^d, d \leqslant N$

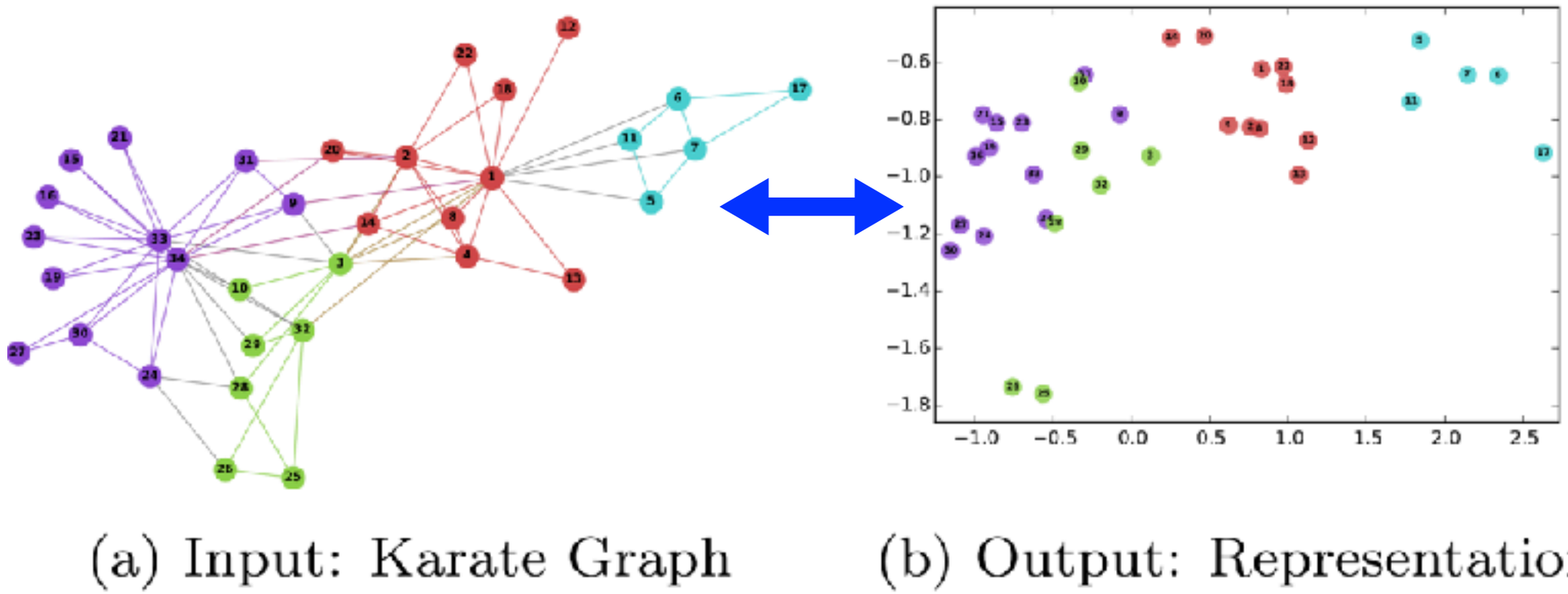
$f : v_i \to \mathbb{R}^d$

$\mathbf{e}_i = [x_1^i, x_2^i, \ldots, x_d^i]$

Node Vector / Node Embedding / Node Representation

A Visual Example

- Zachary's Karate Club network:



(a) Input: Karate Graph (b) Output: Representation

[Perozzi et al. KDD 2014]

Intuition: Embedding of nodes to lower-dimension, so that "**similar**" nodes in the graph have embeddings that are **close** together.

Graph Embedding

Shallow Embedding Methods

Matrix Factorization Based Methods

- [Belkin et al. NIPS 2002]
- [Ahmed et al. WWW 2013]
- [Cao et al. KDD 2015]
- [Ou et al. KDD 2016]

Random Walk Based Methods

- [Perozzi et al. KDD 2014]
- [Grover et al. KDD 2016]

Deep Embedding Methods

Graph Autoencoder

- [Cao et al. AAAI 2016]
- [Wang et al. KDD 2016]

Graph Convolutional Network

- [Bruna et al., ICLR 2014]
- [Duvenaud et al., NIPS 2015]
- [Kipf & Welling, ICLR 2017]

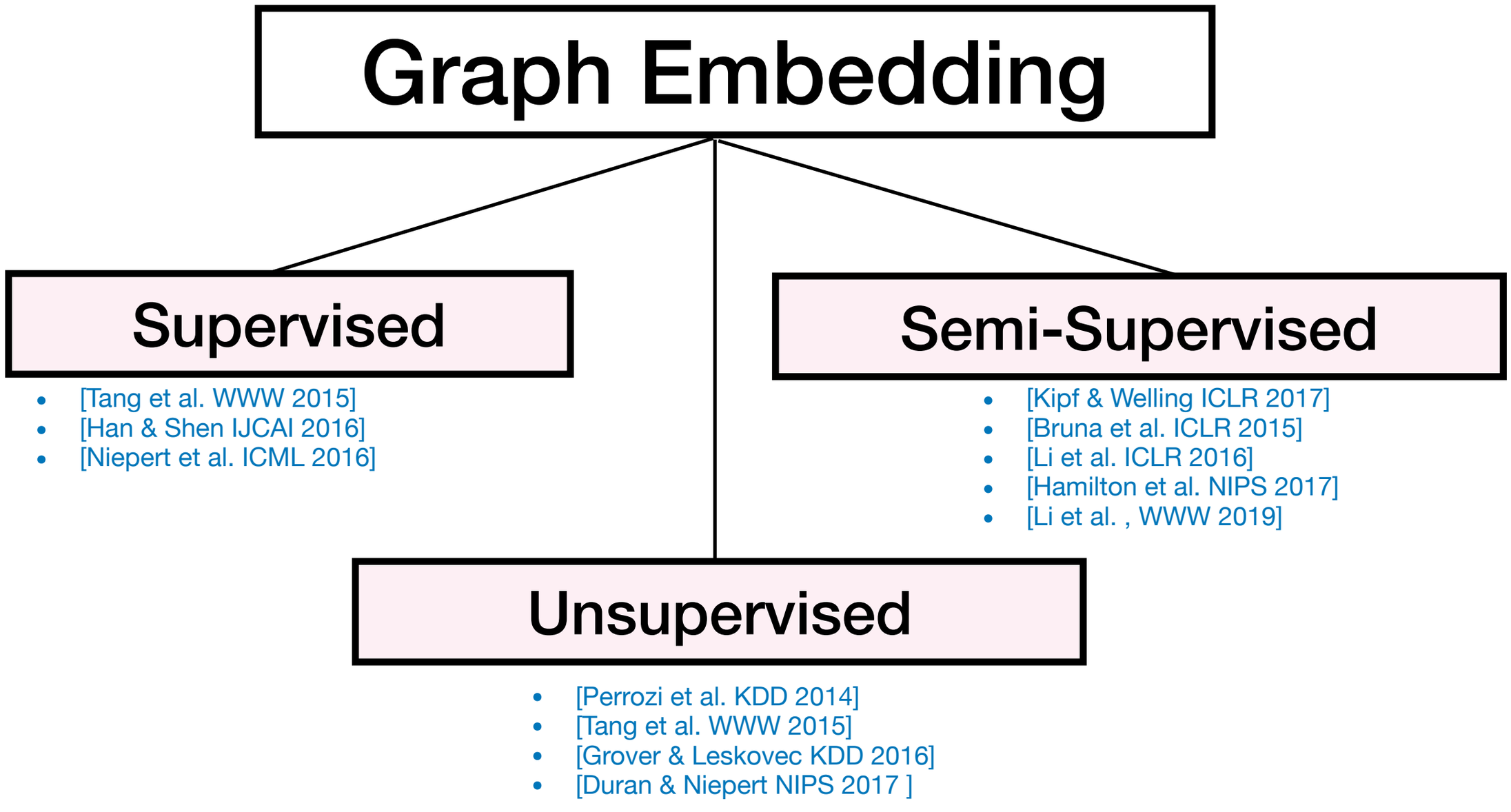
Graph-Gated Recurrent Unit (Graph-GRU)

- [Li et al. ICLR 2018]
- [Yan et al. AAAI'2018]
- [Wu et al. IJCAI'2019]
- [Guo et al. AAAI 2019]
- Our work [Zhang et al. IJCAI 2019]
- Our work [Zhang et al. UAI 2018]



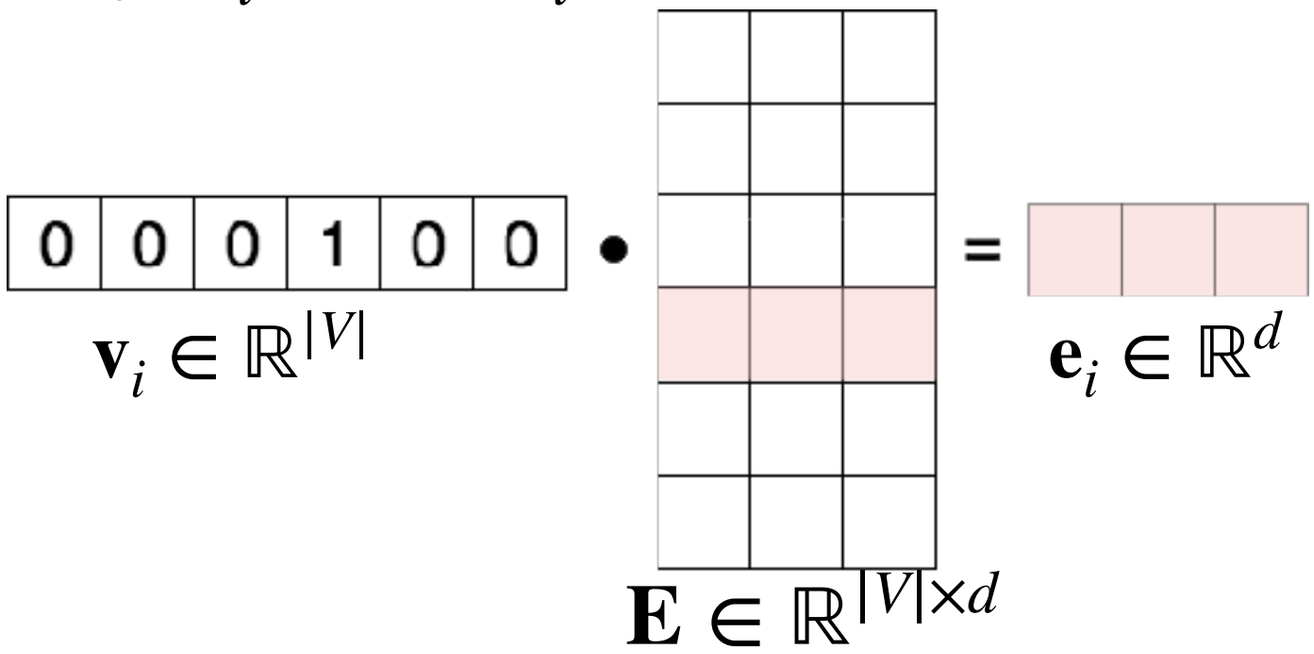


Graph Embedding Taxonomy by Learning Paradigms



Shallow Embedding Methods

Maps nodes to vector embeddings with an "**embedding lookup**" table: $f(\mathbf{v}_i, \mathbf{E}) = \mathbf{e}_i$

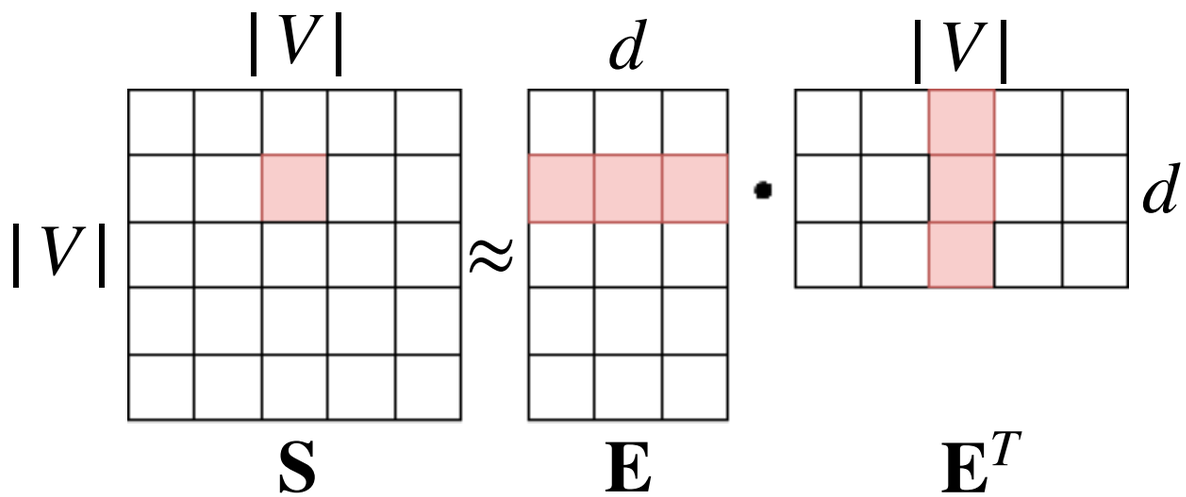


1. **Matrix Factorization-based Methods**

2. **Random Walk-based Methods**

Matrix Factorization-Based

- **Dimensionality Reduction**
- **Encoder:** $f(\mathbf{v}_i) = \mathbf{v}_i\mathbf{E}$
- **Decoder:** pairwise similarity

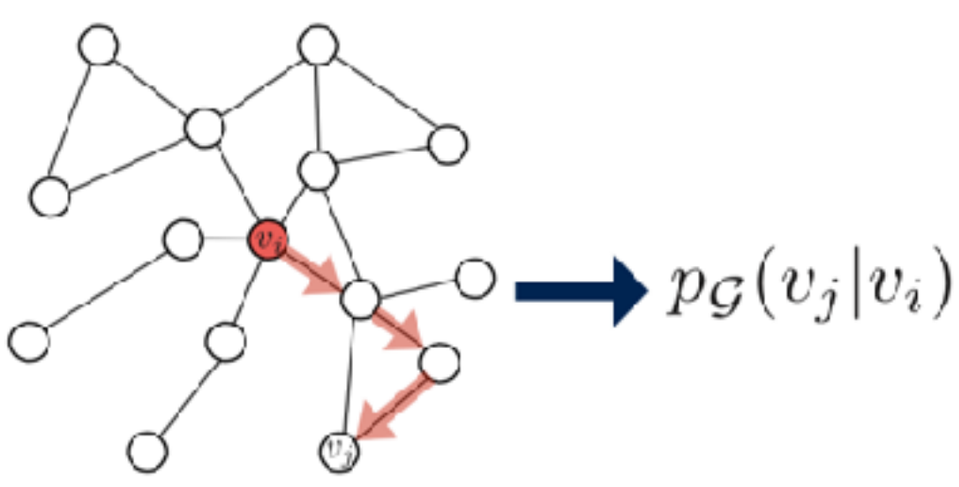


- **Loss:** $L \approx ||\mathbf{E}\mathbf{E}^T - \mathbf{S}||^2$
- $\mathbf{S}$ is a matrix of user-defined pairwise similarity measurement
- Laplacian Eigenmaps [Belkin et al. NIPS 2002]; Graph Factorization [Ahmed et al. WWW 2013]; GraRep [Cao et al. KDD 2015]; HOPE [Ou et al. KDD 2016]

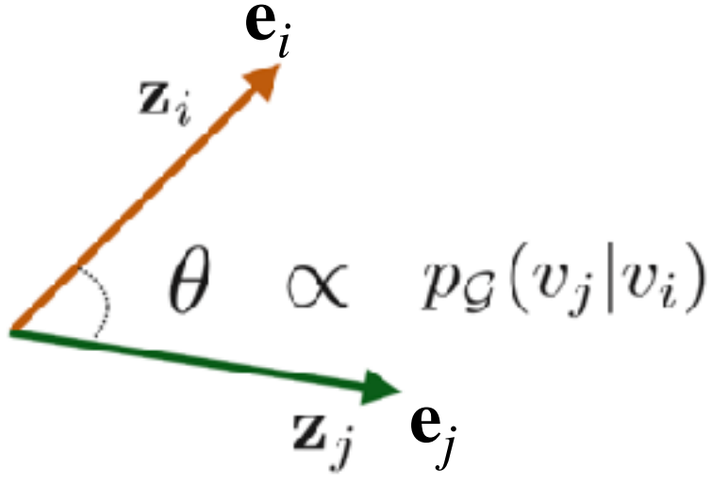
Random Walk-Based

- Nodes **co-occur** on **short** random walks over the graph

—> Nodes would have **similar** embeddings



1. Run random walks to obtain co-occurrence statistics.



2. Optimize embeddings based on co-occurrence statistics.

[Hamilton et al. IEEE 2018]

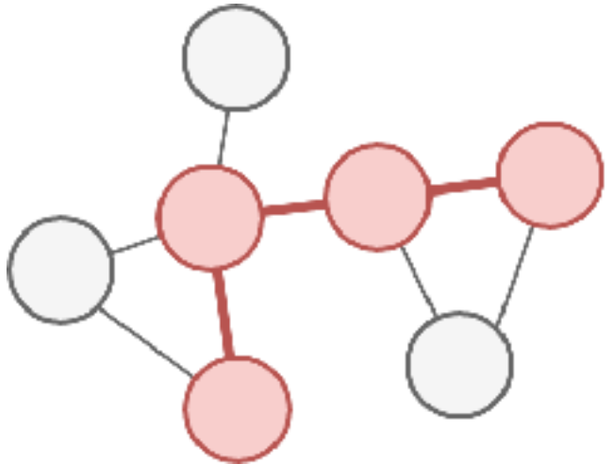




Random Walk-Based (Cont.)

- Deepwalk [Perozzi et al. KDD 2014] & Node2Vec [Grover et al. KDD 2016]

$$\mathbf{Dec}(\mathbf{e}_i, \mathbf{e}_j) \triangleq \frac{\mathbf{exp}(\mathbf{e}_i^T \mathbf{e}_j)}{\sum_{v_k \in V} \mathbf{exp}(\mathbf{e}_i^T \mathbf{e}_k)}$$



- Cross-entropy Loss:

$$L = \sum_{(v_i, v_j) \in D} -\mathbf{log}(\mathbf{Dec}(\mathbf{e}_i, \mathbf{e}_j))$$

Drawbacks of Shallow Embedding Methods

1. **No parameters sharing**
   - Simply an embedding lookup based on arbitrary node id
2. **Fail to leverage node attributes**
   - E.g., user profiles on a social network
3. **Cannot generate embeddings for previously unseen nodes**
   - The Cold Start problem





Deep Embedding Methods

- Use more **complex** encoders
  - Often based on **deep neural networks**
  - Depend more generally on the **structure** and **attributes** of the graph

1. **Graph Autoencoder**
2. **Graph Convolutional Network (GCN)**
3. **Graph-Gated Recurrent Unit (Graph-GRU)**





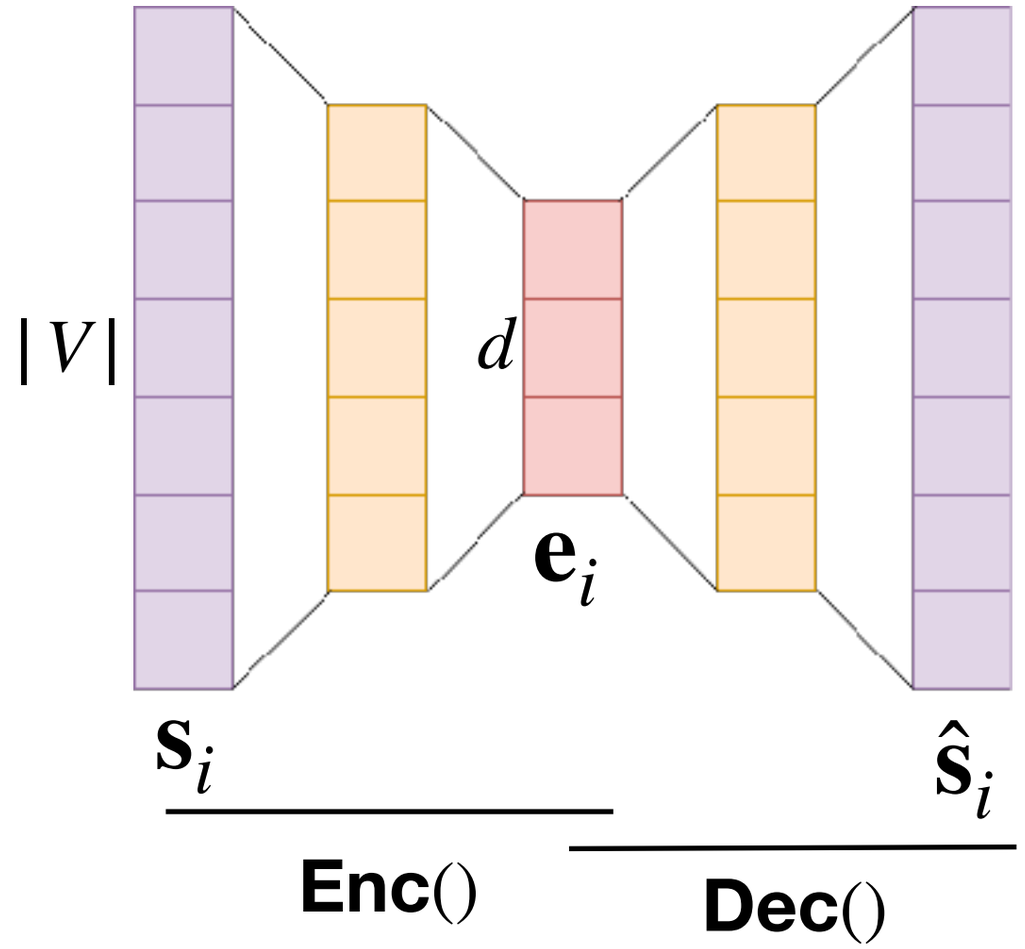
Graph Autoencoder

- To compress information about a node's local neighbourhood

$$\mathbf{Dec}(\mathbf{Enc}(\mathbf{s}_i)) = \mathbf{Dec}(\mathbf{e}_i) \approx \mathbf{s}_i,$$

$$L = \sum_{v_i \in V} ||\mathbf{Dec}(\mathbf{e}_i) - \mathbf{s}_i||^2$$

- $\mathbf{s}_i \in \mathbb{R}^{|V|}$
- $\mathbf{e}_i \in \mathbb{R}^d, d \ll |V|$



Graph Autoencoder

- Deep Neural Graph Representations [Cao et al. AAAI 2016]:
  - $\mathbf{s}_i \triangleq$ the pointwise mutual information of two nodes co-occurring on random walks

- Structural Deep Network Embeddings [Wang et al. KDD 2016]:
  - $\mathbf{s}_i \triangleq \mathbf{A}_i$ the adjacency vector of v_i





Graph Autoencoder

- **Advantages**:
  - Incorporate **structural** information as a form of **regularization**
- **Disadvantages**:
  - Difficult to deal with **large scale** graphs
    - The **input dimension** to the autoencoder is **fixed** at |V|
  - Cannot cope with **unseen nodes**
    - The **structure and size** of the autoencoder is **fixed**





Graph Convolutional Network (GCN)

- Spatial Approach:
  - Aggregate information from **Local Neighborhood**

    + Parameter Sharing

1. Define **neighborhood**
   - All neighbors 
   - Fixed size uniform sampling 
   - Random walk sampling 
2. Design graph **aggregator** $\mathbf{y}_i = r_\theta(\mathbf{x}_i, \{\mathbf{z}_i\})$

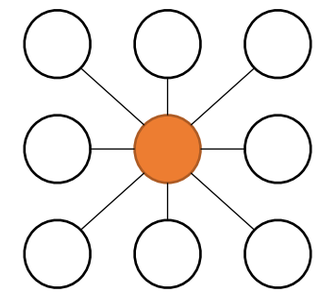
Graph Aggregator Properties

Patch --> Regular Grid

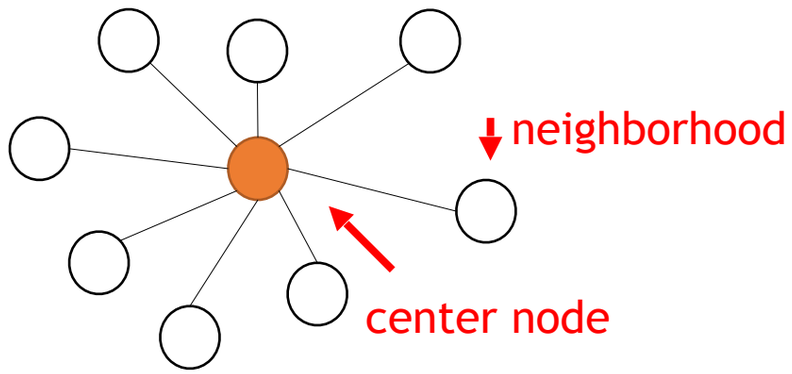
1. Permutation-sensitive
2. Fixed Size

Set --> Irregular Grid

1. **Permutation-invariant**
2. **Dynamic Resize**

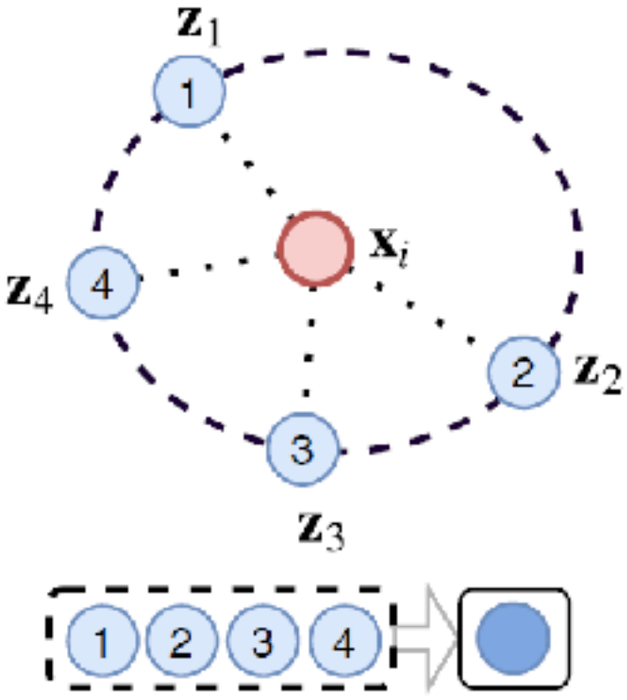


Convolution



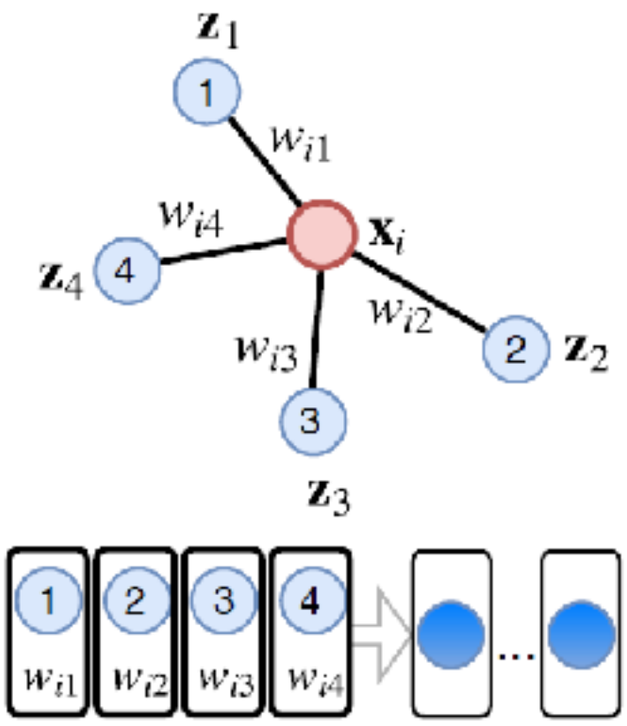
Graph Convolution

Graph Aggregators



Pooling-based

$$\mathbf{y}_i = \phi_o(\mathbf{x}_i \oplus \mathbf{pool}_{j \in \mathcal{N}_i}(\phi_v(\mathbf{z}_j)))$$

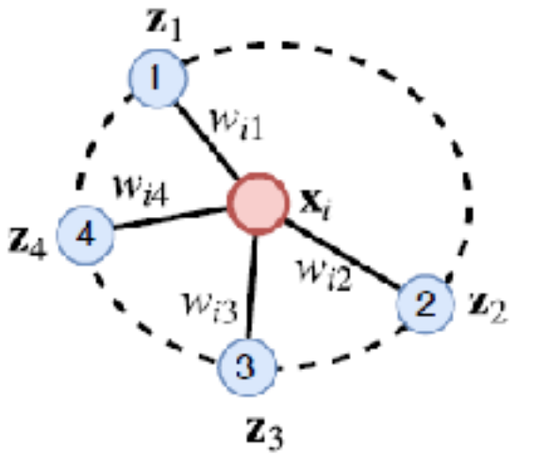


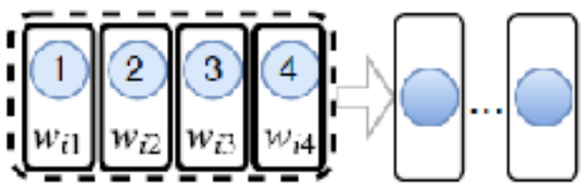
Pairwise-sum

$$\mathbf{y}_i = \phi_o(\mathbf{x}_i \oplus \Big\|_{k=1}^{K} \sum_{j \in \mathcal{N}_i} w_{i,j}^{(k)} \phi_v^{(k)}(\mathbf{z}_j)),$$

$$w_{i,j}^{(k)} = \phi_w^{(k)}(\mathbf{x}_i, \mathbf{z}_j).$$

Graph Aggregators





Attention-based Aggregator

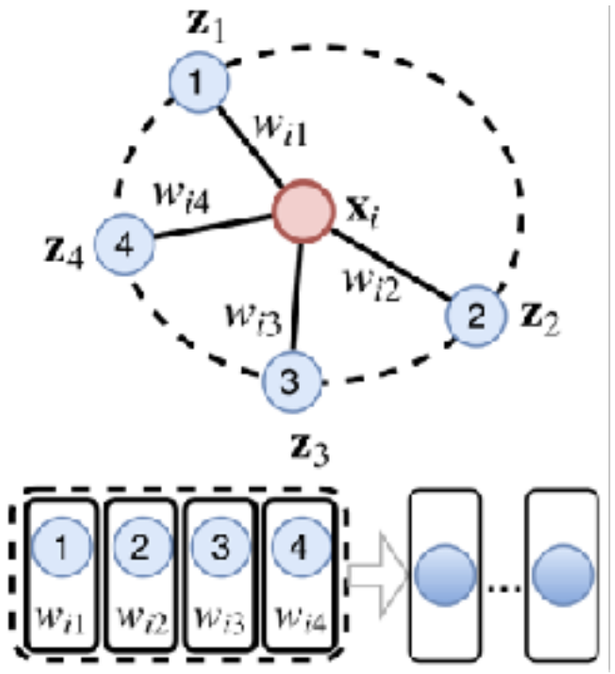
$$\mathbf{y}_i = \mathrm{FC}_{\theta_o}(\mathbf{x}_i \oplus \Big\|_{k=1}^{K} \sum_{j \in \mathcal{N}_i} w_{i,j}^{(k)} \mathrm{FC}_{\theta_v^{(k)}}^{h}(\mathbf{z}_j)),$$

$$w_{i,j}^{(k)} = \frac{\exp(\phi_w^{(k)}(\mathbf{x}_i, \mathbf{z}_j))}{\sum_{l=1}^{|\mathcal{N}_i|} \exp(\phi_w^{(k)}(\mathbf{x}_i, \mathbf{z}_l))},$$

$$\phi_w^{(k)}(\mathbf{x}, \mathbf{z}) = \langle \mathrm{FC}_{\theta_{xa}^{(k)}}(\mathbf{x}), \mathrm{FC}_{\theta_{za}^{(k)}}(\mathbf{z}) \rangle.$$

- Traditional multi-head attention mechanism
  - Equally consumes all attention heads
  - Loses the opportunity to benefit from some attention heads which are inherently more important than others.
- We propose the Gated Attention Networks (GaAN) [Zhang et al. UAI 2018].
  - Use a small convolutional subnetwork to compute a soft gate at each attention head to control its importance.

Graph Aggregators (Cont.)

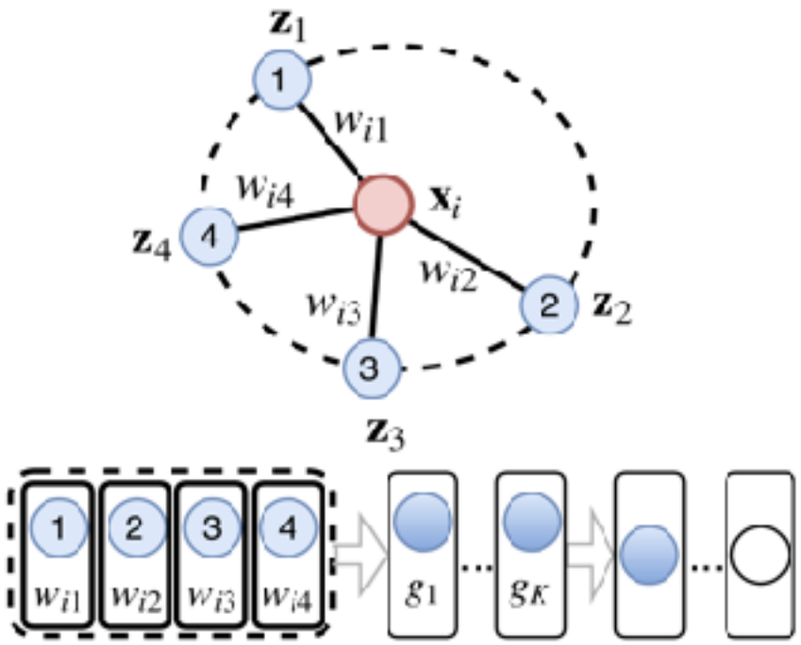


Attention-based Aggregator

$$\mathbf{y}_i = \mathrm{FC}_{\theta_o}(\mathbf{x}_i \oplus \Big\|_{k=1}^{K} \sum_{j \in \mathcal{N}_i} w_{i,j}^{(k)} \mathrm{FC}_{\theta_v^{(k)}}^{h}(\mathbf{z}_j)),$$

$$w_{i,j}^{(k)} = \frac{\exp(\phi_w^{(k)}(\mathbf{x}_i, \mathbf{z}_j))}{\sum_{l=1}^{|\mathcal{N}_i|} \exp(\phi_w^{(k)}(\mathbf{x}_i, \mathbf{z}_l))},$$

$$\phi_w^{(k)}(\mathbf{x}, \mathbf{z}) = \langle \mathrm{FC}_{\theta_{xa}^{(k)}}(\mathbf{x}), \mathrm{FC}_{\theta_{za}^{(k)}}(\mathbf{z}) \rangle.$$



Gated Attention Aggregator (GaAN)

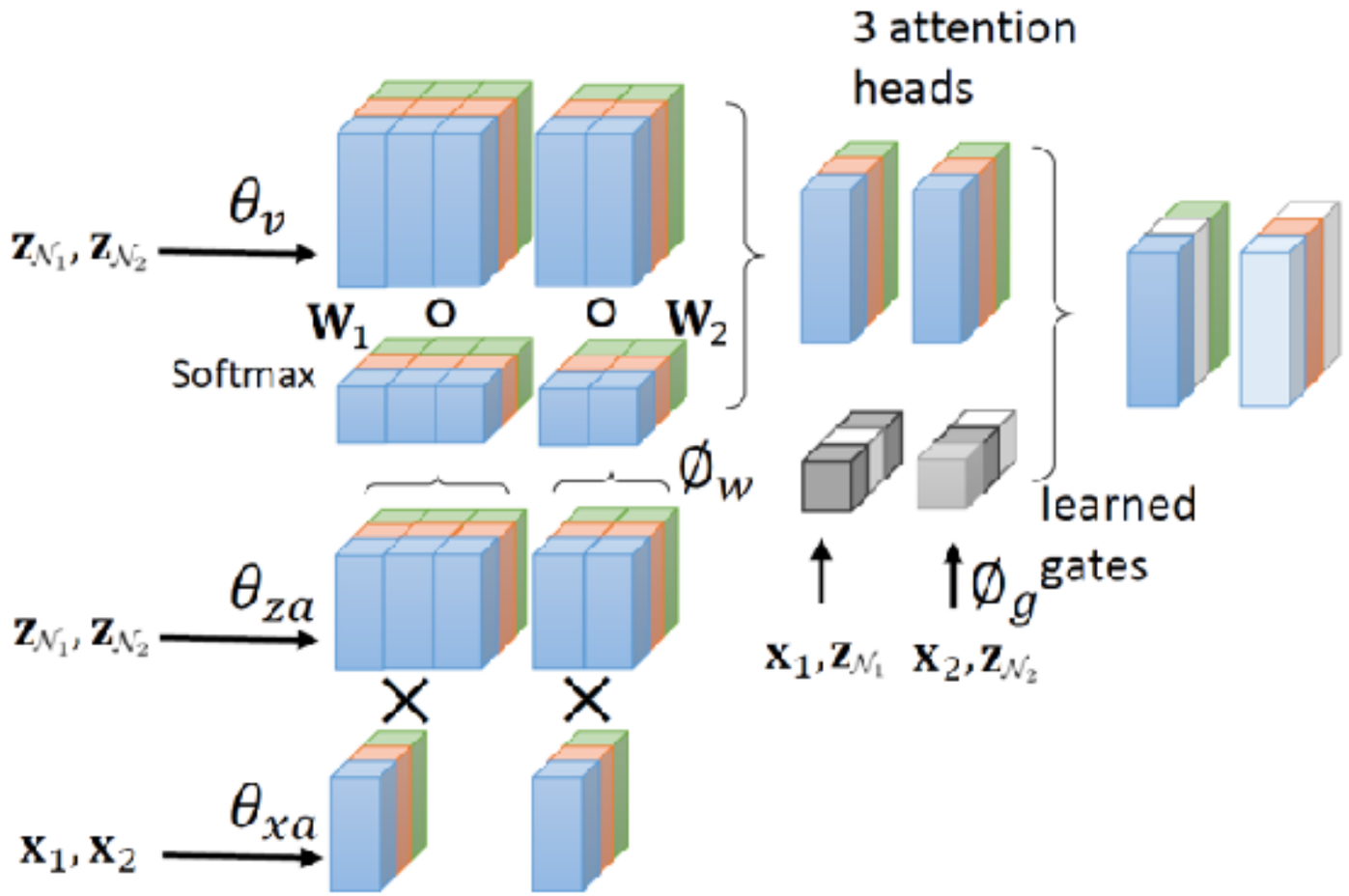
Igate

$$\mathbf{y}_i = \mathrm{FC}_{\theta_o}(\mathbf{x}_i \oplus \Big\|_{k=1}^{K} (g_i^{(k)} \sum_{j \in \mathcal{N}_i} w_{i,j}^{(k)} \mathrm{FC}_{\theta_v^{(k)}}^{h}(\mathbf{z}_j))),$$

$$\mathbf{g}_i = [g_i^{(1)}, ..., g_i^{(K)}] = \psi_g(\mathbf{x}_i, \mathbf{z}_{\mathcal{N}_i}),$$

[Zhang et al. UAI 2018]

GaAN: Gated Attention Networks



- Illustration of a three-head gated attention aggregator with two center nodes in a mini-batch.
- Different colors indicate different attention heads.

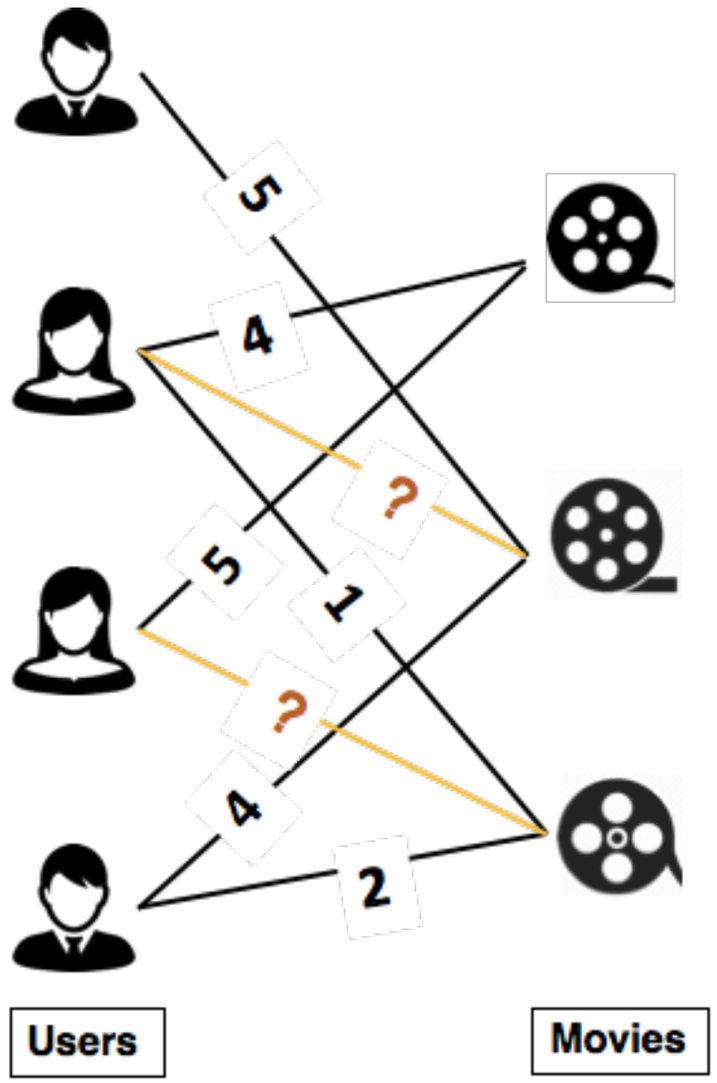
Graph Neural Networks for Recommendation

- Formalize the user-item interaction data as a **bipartite graph**

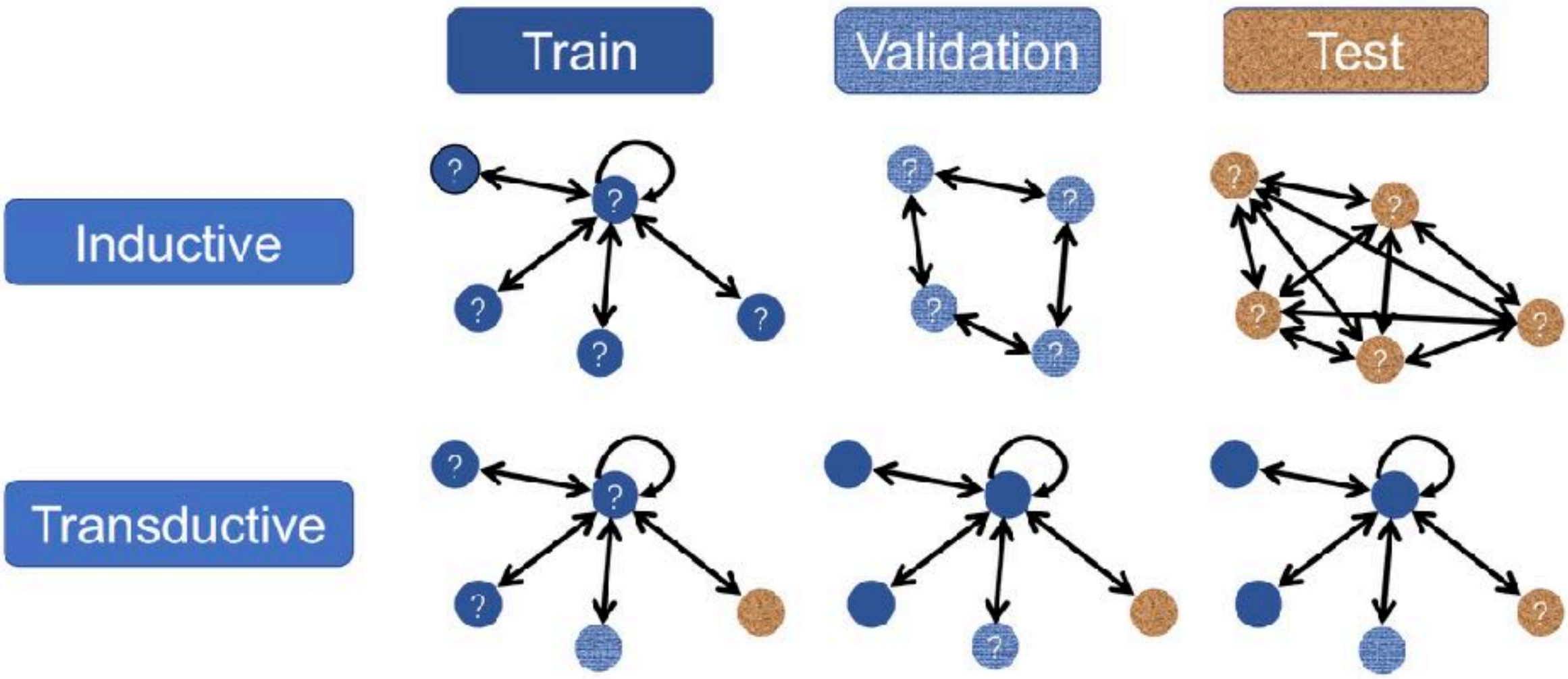
- **Rating prediction**
  - Predict some missing ratings given the existing rating pairs

- **Transductive** rating prediction

- **Cold start** scenario



Transductive v.s. Inductive Learning



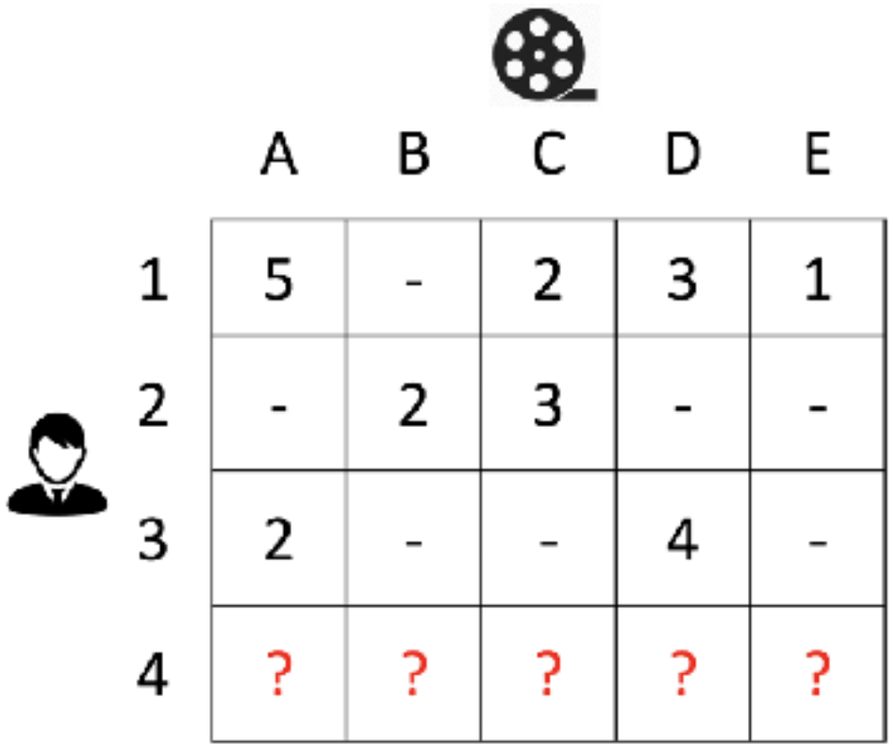
Mufei Li, Amazon AI Lab Shanghai

Transductive Rating Prediction

	A	B	C	D	E
1	5	-	2	3	1
2	?	2	3	-	?
3	2	?	?	4	-
4	-	-	2	?	-

- Train on observed ratings
- Predict the missing ratings , i.e., `?`
- All the testing users and items are **observed** in training data
- **Matrix completion** problem
  - Matrix Factorization (MF)

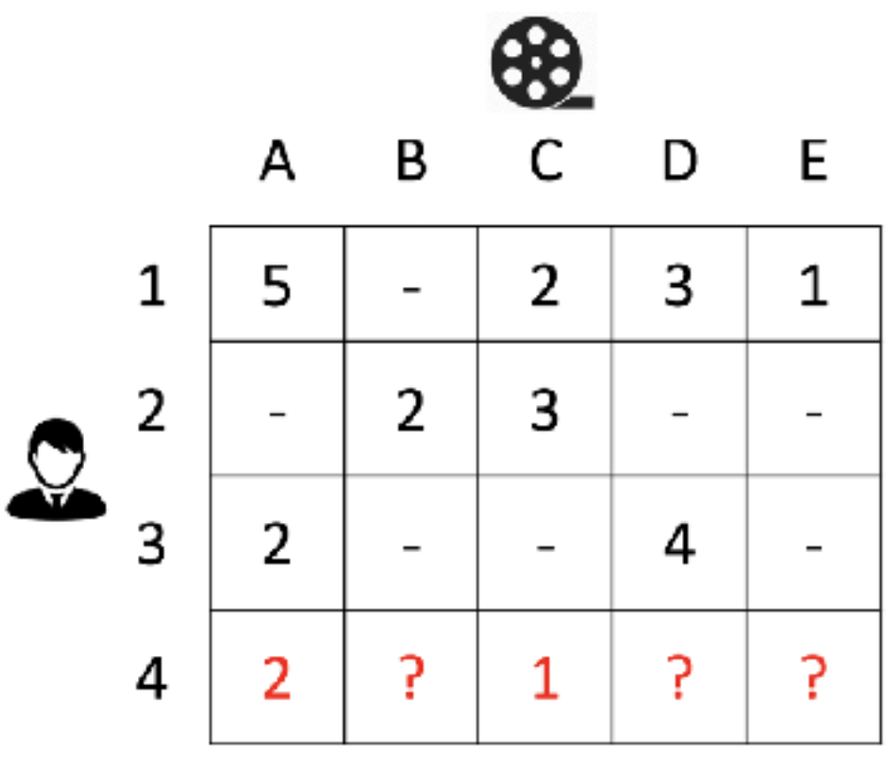
Cold Start Scenario 1: New Users/Items



	A	B	C	D	E
1	5	-	2	3	1
2	-	2	3	-	-
3	2	-	-	4	-
4	?	?	?	?	?

- Train on observed ratings
- Predict how **a new user** will rate the movie
- Content-based recommendation
  - Collaborative Deep Learning for Recommender Systems
    [Wang et al. KDD 2015]
  - To encode the user features, e.g., profiles

Cold Start Scenario 2: Ask-to-rate



- Train on the observed ratings
- Test time:
  - Ask **a new user** to **rate several movies**
  - Predict how a user will rate other movies
- **Inductive** rating prediction

Stacked and Reconstructed Graph Convolutional Networks

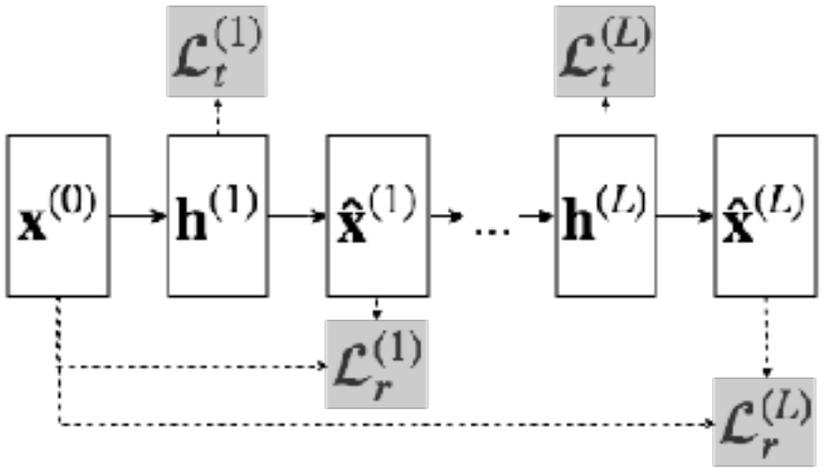
[Zhang et al. IJCAI 2019]

- **STAR-GCN**:
  - Can solve the **cold start problem** (**inductive**) in recommender systems
    - Embed nodes to low-dimensional vectors -> Scalable to large graphs
    - Multi-block structure: Mask and reconstruct node embeddings (BERT [Jacob et al. 2019] for Graph)
- **SOTA results** on both **transductive** and **inductive** (ask-to-rate) rating prediction task
  - Achieves the **best state-of-the-art** results





STAR-GCN



- A multi-block graph encoder-decoder

$\mathbf{x}^{(0)} \rightarrow \mathbf{h}^{(1)} \rightarrow \hat{\mathbf{x}}^{(1)} \rightarrow \mathbf{h}^{(2)} \rightarrow \hat{\mathbf{x}}^{(2)} \rightarrow ... \rightarrow \hat{\mathbf{x}}^{(L)}$

- Go beyond BERT
  - BERT has a single block

$\mathbf{x}^{(0)} \rightarrow \mathbf{h}^{(1)} \rightarrow \hat{\mathbf{x}}^{(1)}$

- Encoder ($\mathbf{x}^{(l-1)} \rightarrow \mathbf{h}^{(l)}$)
  - Generate node representations by encoding semantic graph structures + input features
- Decoder ($\mathbf{h}^{(l)} \rightarrow \hat{\mathbf{x}}^{(l)}$)
  - Recover masked input node embeddings
- Any variant of graph convolutional network (GCN) can be an encoder or decoder
- Loss
  - A rating prediction loss $\mathcal{L}_t$
  - A node reconstruction loss $\mathcal{L}_r$





Experiments: Dataset

Table 1: Statistics of the datasets. 'D_U' and 'D_V' are the input feature dimension of users and items, respectively.

	D_U	D_V	#U	#V	$\mathcal{R}$	#R
Flixster	3K	3K	2,341	2,956	0.5,1,..,5	26,173
Douban	3K	-	2,999	3,000	1,..,5	136,891
ML-100K	23	320	943	1,682	1,..,5	100K
ML-1M	23	320	6,040	3,706	1,..,5	1M
ML-10M	-	321	69,878	10,677	0.5,1,..,5	10M





Experiments: Transductive Rating Prediction

	Flixster	Douban	ML-100K	ML-1M	ML-10M
BiasMF [Koren *et al.*, 2009]	-	-	0.917	0.845	0.803
NNMF [Dziugaite and Roy, 2015]	-	-	0.907	0.843	-
I-AUTOREC [Sedhain *et al.*, 2015]	-	-	-	0.831	0.782
GRALS [Rao *et al.*, 2015]	1.245	0.833	0.945	-	-
CF-NADE [Zheng *et al.*, 2016]	-	-	-	**0.829**	0.771
Factorized EAE [Hartford *et al.*, 2018]	-	-	0.910	0.860	-
sRMGCNN [Monti *et al.*, 2017]	0.926	0.801	0.929	-	-
GC-MC [Berg *et al.*, 2017]	0.917	0.734	0.910	0.832	0.777
STAR-GCN	**0.879**±0.0030	**0.727**±0.0006	**0.895**±0.0009	0.832±0.0016	**0.770**±0.0001

RMSE: the lower the score, the better the performance

- **STAR-GCN** architecture achieves the **best state-of-the-art** results on **four out of five** datasets





Experiment: Inductive Rating Prediction (Ask-to-rate)

- **Goal**: to predict ratings where either users or items are not seen in the training phase
- Experiment Setting:
  - Keep 20% of user (or item) nodes as the testing nodes & remove them from the training graph

Datasets	Models	Items 20%			Users 20%		
		50%	30%	10%	50%	30%	10%
Douban	DropoutNet	-	-	-	0.797±0.002	0.797±0.003	0.797±0.001
	CDL	-	-	-	0.781±0.006	0.781±0.001	0.781±0.001
	STAR-GCN(- rec.)	0.734±0.001	0.746±0.001	0.777±0.002	0.731±0.000	0.738±0.000	0.753±0.001
	STAR-GCN(- rec., + fea.)	-	-	-	0.731±0.002	0.737±0.000	0.753±0.001
	STAR-GCN	**0.725**±0.001	**0.734**±0.001	**0.764**±0.000	**0.725**±0.001	**0.731**±0.001	0.747±0.001
	STAR-GCN(+ fea.)	-	-	-	**0.725**±0.002	**0.731**±0.000	**0.746**±0.000
ML-100K	DropoutNet	1.223±0.065	1.167±0.031	1.144±0.024	1.015±0.002	1.022±0.006	1.023±0.003
	CDL	1.083±0.009	1.082±0.007	1.082±0.007	1.011±0.005	1.013±0.006	1.015±0.004
	STAR-GCN(- rec.)	0.932±0.001	0.943±0.001	0.976±0.003	0.919±0.002	0.933±0.001	0.949±0.001
	STAR-GCN(- rec., + fea.)	0.928±0.002	0.941±0.002	0.977±0.004	0.916±0.005	0.931±0.004	0.951±0.005
	STAR-GCN	0.919±0.001	**0.926**±0.000	**0.954**±0.001	**0.907**±0.004	**0.917**±0.005	0.937±0.005
	STAR-GCN(+ fea.)	**0.918**±0.002	**0.926**±0.002	0.956±0.000	**0.907**±0.002	**0.917**±0.001	**0.936**±0.004
ML-1M	DropoutNet	1.169±0.120	1.134±0.034	1.256±0.128	1.002±0.001	1.005±0.005	1.003±0.001
	CDL	1.068±0.009	1.069±0.009	1.068±0.009	0.974±0.000	0.975±0.000	0.974±0.000
	STAR-GCN(- rec.)	0.862±0.001	0.872±0.004	0.903±0.004	0.859±0.002	0.868±0.001	0.891±0.001
	STAR-GCN(- rec., + fea.)	0.861±0.002	0.867±0.002	0.910±0.006	0.859±0.001	0.869±0.001	0.893±0.001
	STAR-GCN	**0.844**±0.000	**0.850**±0.000	**0.876**±0.004	**0.848**±0.001	**0.858**±0.001	**0.882**±0.000
	STAR-GCN(+ fea.)	**0.844**±0.001	0.851±0.001	**0.876**±0.002	0.849±0.001	**0.858**±0.000	0.883±0.001

RMSE: the lower the score, the better the performance

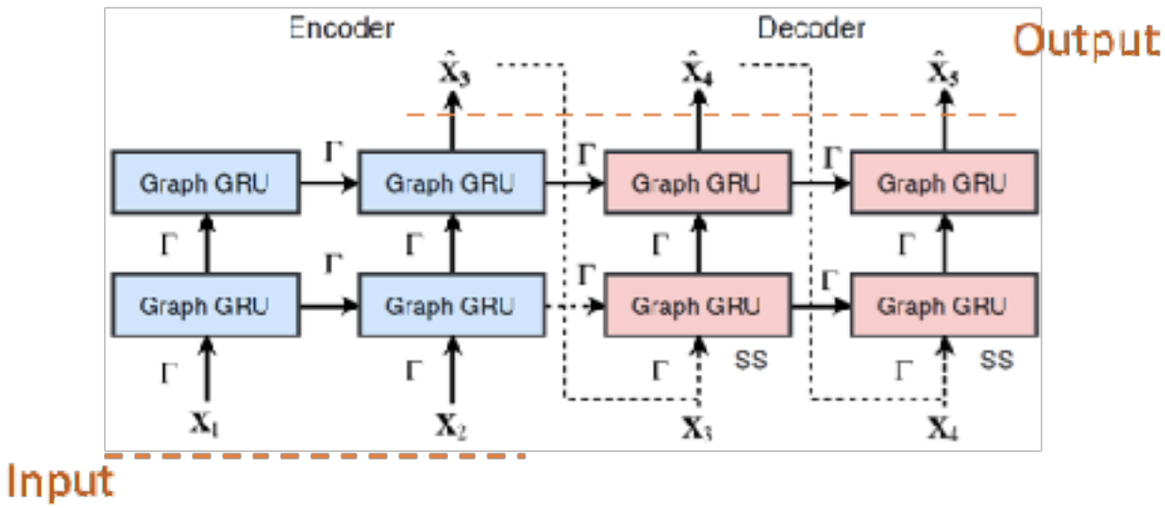
- STAR-GCN produces **significantly better** results than baselines

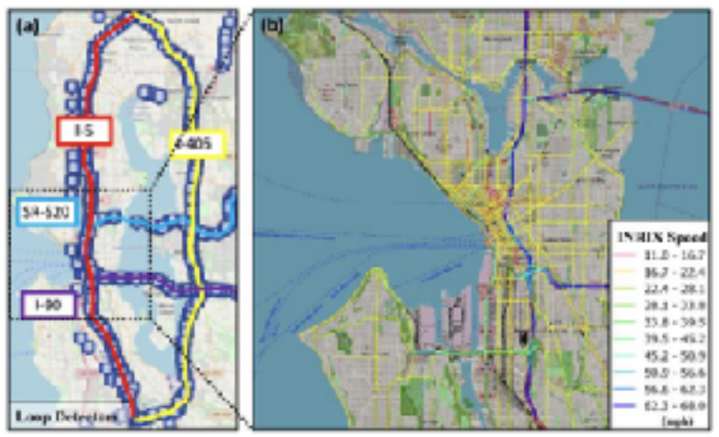




Graph Neural Networks for Spatiotemporal Graphs

- Spatial Dependency + Temporal Dependency
- Applications: Traffic Speed Forecasting Problem
- Predict the future K steps of traffic speeds in the sensor network $\hat{\mathbf{X}}_{J+1}, \hat{\mathbf{X}}_{J+2}, \ldots, \hat{\mathbf{X}}_{J+K}$, based on the previous J steps of observed traffic speeds $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_J$.





Graph Gated Recurrent Units (Graph GRU)

- A Simple GRU

$$\begin{aligned}
\mathbf{U}_t &= \sigma(\mathbf{W}_{ux}\mathbf{X}_t + \mathbf{W}_{uh}\mathbf{H}_{t-1}),\\
\mathbf{R}_t &= \sigma(\mathbf{W}_{rx}\mathbf{X}_t + \mathbf{W}_{rh}\mathbf{H}_{t-1}),\\
\mathbf{H}'_t &= h(\mathbf{W}_{hx}\mathbf{X}_t + \mathbf{W}_{hh}(\mathbf{R}_t \circ \mathbf{H}_{t-1})),\\
\mathbf{H}_t &= (1 - \mathbf{U}_t) \circ \mathbf{H}'_t + \mathbf{U}_t \circ \mathbf{H}_{t-1}.
\end{aligned}$$

- A Graph GRU

$$\begin{aligned}
\mathbf{U}_t &= \sigma(\Gamma_{\Theta_{xu}}(\mathbf{X}_t, \mathbf{X}_t; \mathcal{G}) + \Gamma_{\Theta_{hu}}(\mathbf{X}_t \oplus \mathbf{H}_{t-1}, \mathbf{H}_{t-1}; \mathcal{G})),\\
\mathbf{R}_t &= \sigma(\Gamma_{\Theta_{xr}}(\mathbf{X}_t, \mathbf{X}_t; \mathcal{G}) + \Gamma_{\Theta_{hr}}(\mathbf{X}_t \oplus \mathbf{H}_{t-1}, \mathbf{H}_{t-1}; \mathcal{G})),\\
\mathbf{H}'_t &= h(\Gamma_{\Theta_{xh}}(\mathbf{X}_t, \mathbf{X}_t; \mathcal{G}) + \mathbf{R}_t \circ \Gamma_{\Theta_{hh}}(\mathbf{X}_t \oplus \mathbf{H}_{t-1}, \mathbf{H}_{t-1}; \mathcal{G})),\\
\mathbf{H}_t &= (1 - \mathbf{U}_t) \circ \mathbf{H}'_t + \mathbf{U}_t \circ \mathbf{H}_{t-1}.
\end{aligned}$$

[Zhang et al. UAI 2018]

- $\Gamma_\Theta(\mathbf{X}, \mathbf{Z}; \mathcal{G})$ as applying the aggregator for all nodes in $\mathcal{G}$
- $\mathbf{X}_t$: input features
- $\mathbf{H}_t$: hidden states of the nodes
- $\mathbf{U}_t$: the update gate
- $\mathbf{R}_t$: the reset gate

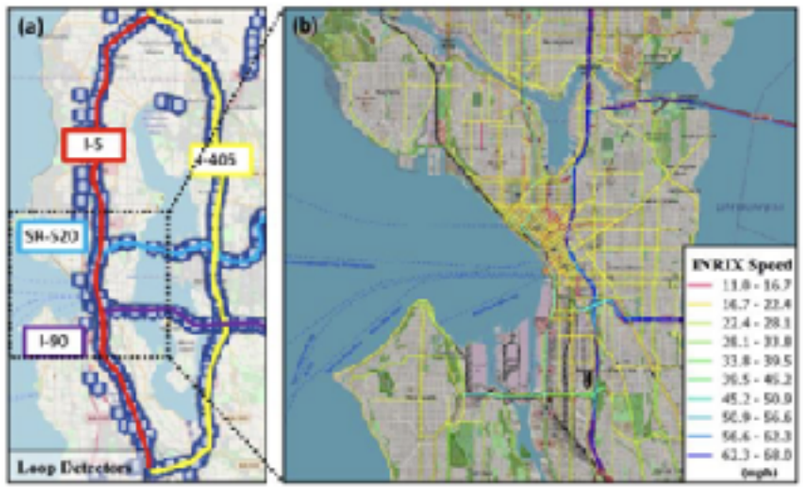




Graph Neural Networks for Traffic Speed Forecasting

- Data

Data	#Nodes	#Edges	#Timestamps
METR-LA	207	1,515	34,272



- **Nodes**: sensors measuring traffic speed
- **Edges**: the proximity between sensor pairs measured by road network distance
- The sensor speeds are recorded every five minutes
- A sequence of length 12 is used as the input to predict the future traffic speed in one hour (12 steps).

Experiments

[Zhang et al. UAI 2018]

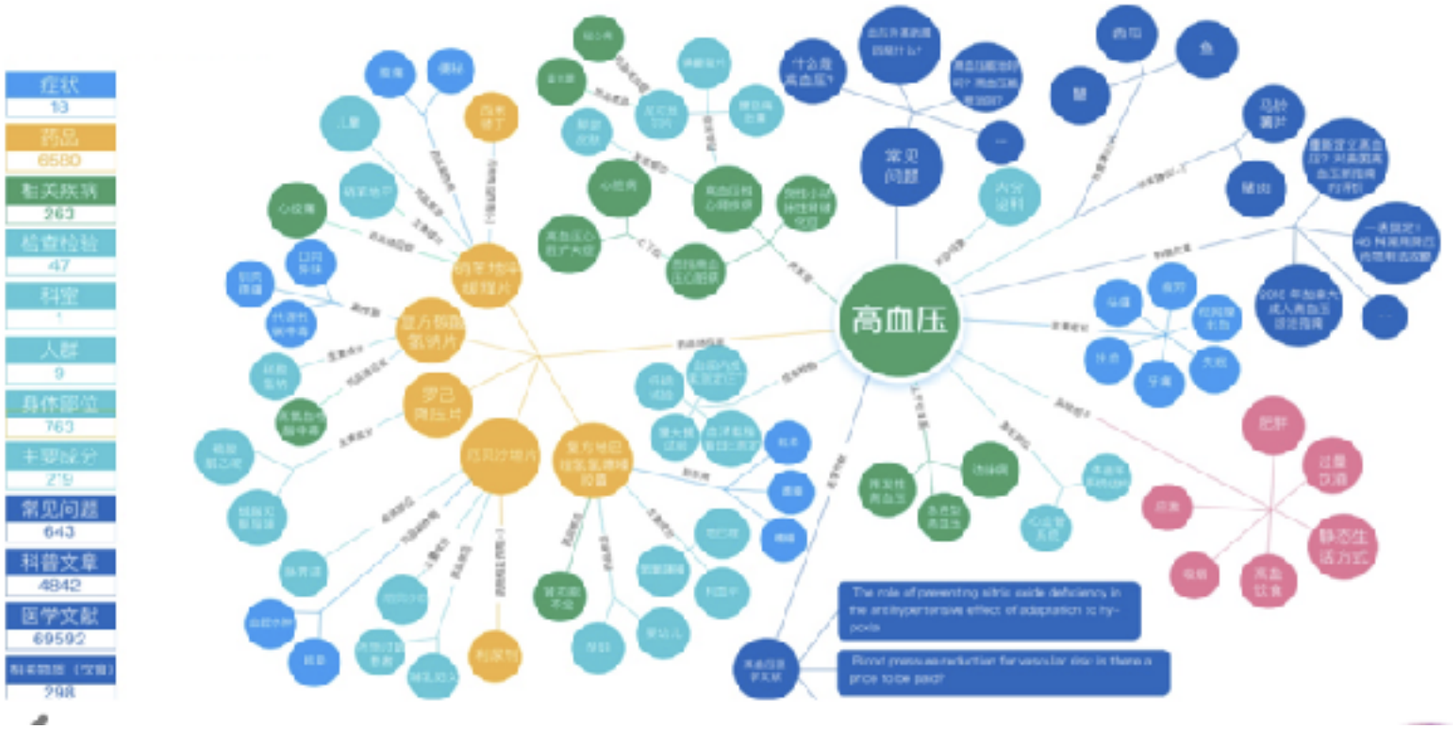
Models / T	15 min			30 min			60 min			Average		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
FC-LSTM (Li et al., 2018)	3.44	6.30	9.6%	3.77	7.23	10.9%	4.37	8.69	13.2%	3.86	7.41	11.2%
GCRNN (Li et al., 2018)	2.80	5.51	7.5%	3.24	6.74	9.0%	3.81	8.16	10.9%	3.28	6.80	9.13%
DCRNN† (Li et al., 2018)	2.77	5.38	7.3%	3.15	6.45	8.8%	**3.60**	**7.60**	**10.5%**	3.17	6.48	8.87%
Avg Pool	2.79	5.42	7.26%	3.20	6.52	8.84%	3.69	7.69	10.73%	3.22	6.54	8.94%
Max Pool	2.77	5.36	7.21%	3.18	6.45	8.78%	3.69	7.73	10.80%	3.21	6.51	8.93%
Pairwise + Sigmoid	2.76	5.36	7.14%	3.18	6.46	8.72%	3.70	7.73	10.77%	3.22	6.52	8.88%
Pairwise + Tanh	2.76	5.34	7.14%	3.18	6.46	8.73%	3.70	7.73	10.73%	3.21	6.51	8.87%
Attention-only	2.74	5.33	7.09%	3.16	6.45	8.69%	3.67	7.61	10.77%	3.19	6.49	8.85%
GaAN	**2.71**	**5.24**	**6.99%**	**3.12**	**6.36**	**8.56%**	3.64	7.65	10.62%	**3.16**	**6.41**	**8.72%**

Variants from Graph GRU

- Scores under "*t* min" are the scores at the *t*-th predicted frame.
- The Graph GRU models consistently give better results than baselines.

Medical Knowledge Graph Embedding

- **Medical Knowledge Graph is a Multi-Relational Graph**
  - **Entities: Medicines, Diseases, Drugs, Patients, Symptoms, Electronic Medical Records, etc.**
  - **Relations: Medical Knowledge**
  - **Applications: Medical Recommendation, Precision Medicine, Biomedical Link Predicti**



Medical Knowledge Graph Embedding

- **Research Question : How to do graph embedding for medical knowledge graphs?**

- **Observation: A patient with disease "pneumonia" has symptom "fever".**

  - Conventional model: "pneumonia" leads to "fever".

  - Our thinking: "fever" is common but not always presented in disease "pneumonia"!

$$P(symptom = fever \mid disease = pneumonia)$$

We are doing: *Probabilistic graph embedding for medical knowledge graphs*

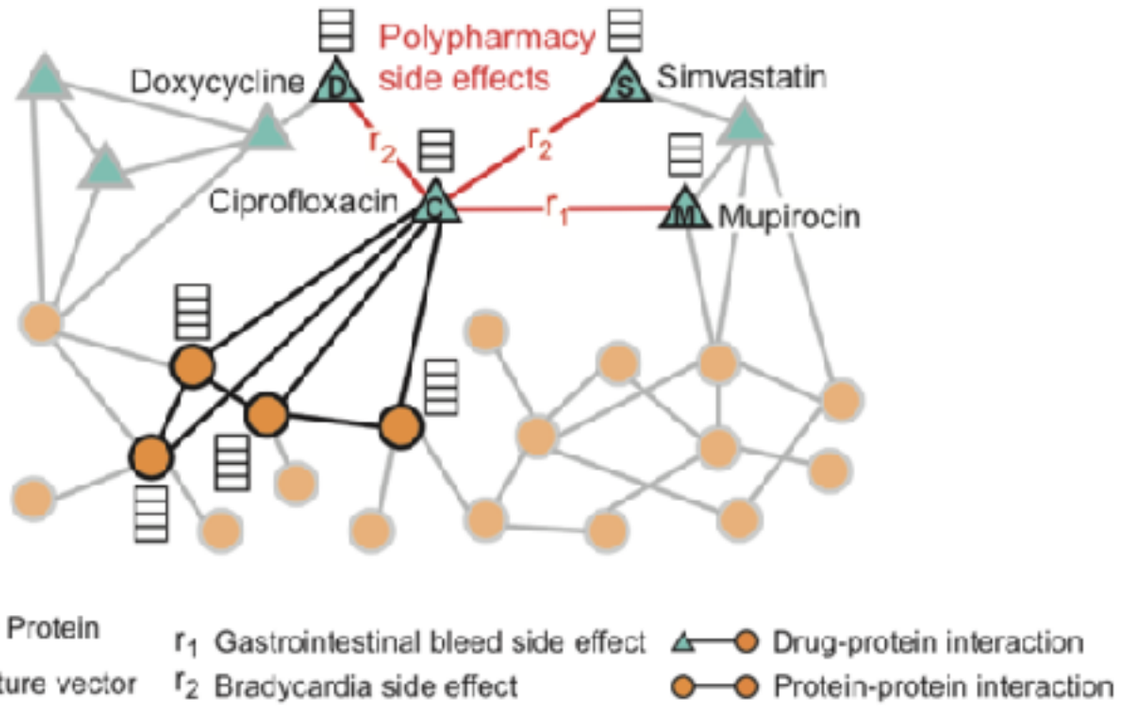




Drug-Drug Interaction (DDI) Prediction

- **Motivation**: Clinical studies cannot sufficiently and accurately identify DDI's in polypharmacy.

- How about using AI to help predict DDI's?

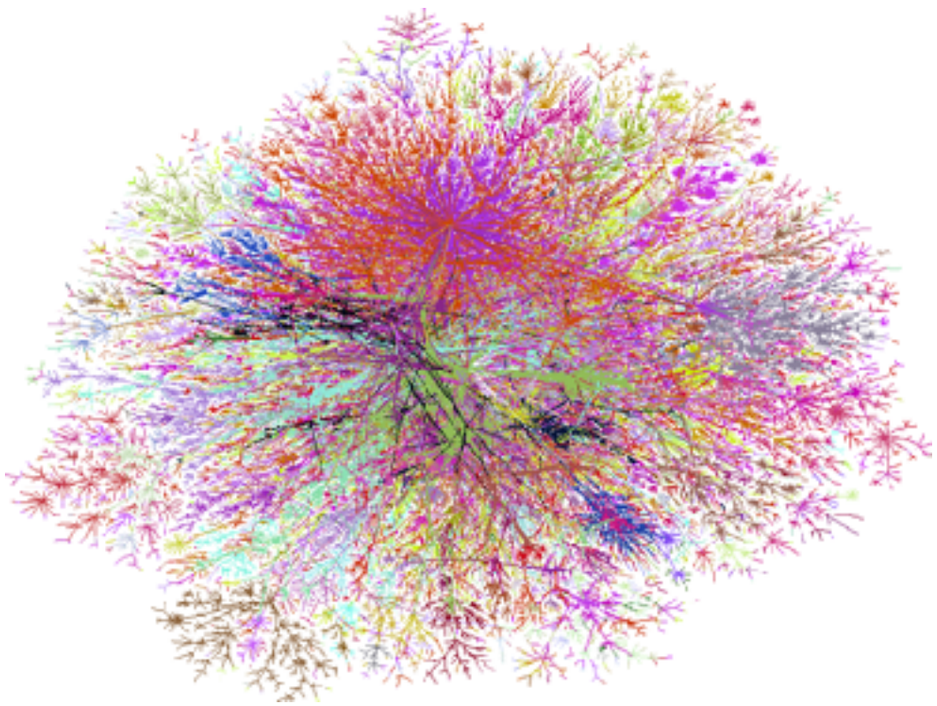


Zitnik M, Agrawal M, Leskovec J. Modeling polypharmacy side effects with GCN [J]. Bioinformatics, 2018, 34(13): i457-i466.

- **We are doing: Using medical knowledge graphs to predict DDI's through link prediction.**

Sampling Large Graphs

Graph Sampling



Sampling Algorithm





Original Graph (G)

$$P(G) \approx P(G')$$

Subgraph Sample (G')

Image: Kaskade, J.(2011, November 19).Big Data and Graphs = Big Graph?[Blog post]. Retrieved from https://jameskaskade.com

Sampling Large Graphs

Large Graph issues

- Ubiquity of large graph
- Millions of nodes and edges
- Complete graph requirement to study
- Space and computation issues

Research Question : Given a large real graph, how can we derive a representative sample preserving properties of original graph?

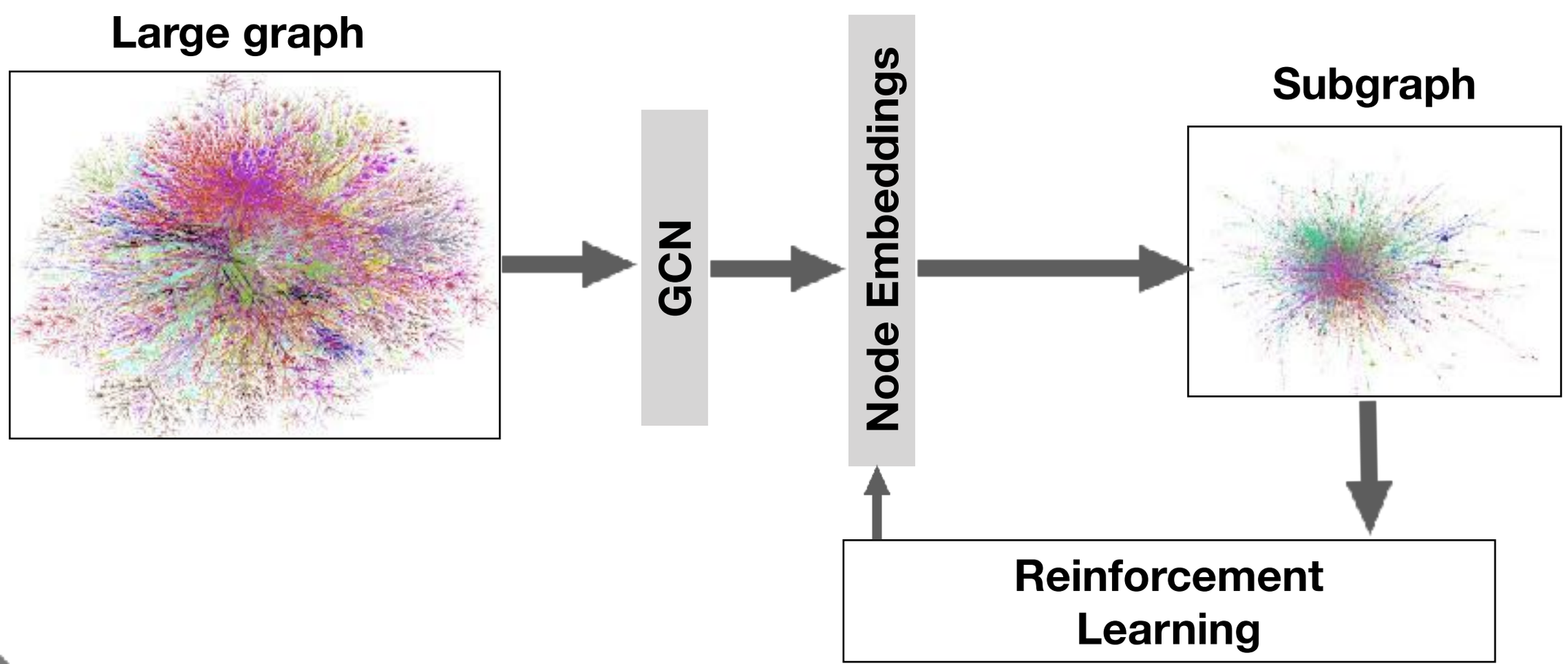




Sampling Large Graphs

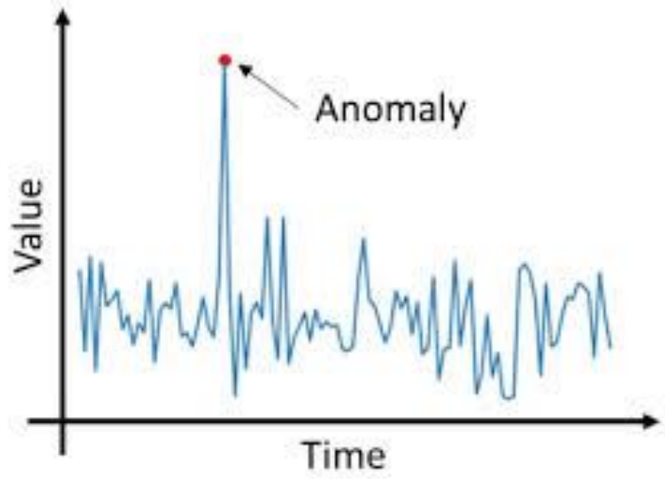
Proposed Solution

- Graph sampling using deep reinforcement learning



Anomaly Detection over Dynamic Graph

- **Anomalies are**: Outliers, Outbreaks, Events Changes, Fraud Detection, Noises, etc.



- **Problem description**: Given a **dynamic graph**, to find the graph objects (nodes/edges/substructures) over temporal dimension **that are rare and differ significantly** from the majority of the reference objects in the graph.

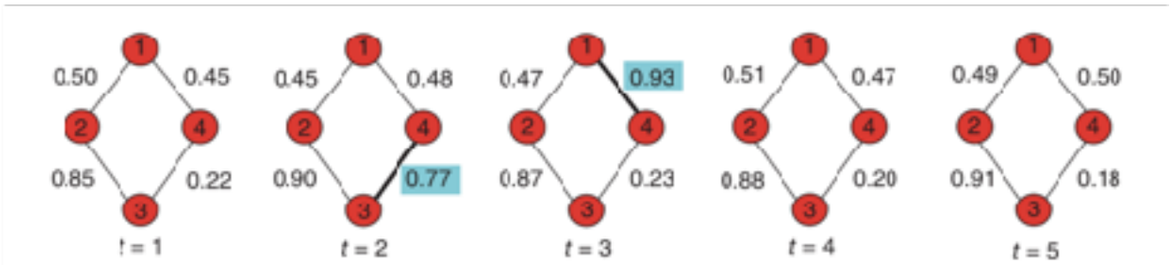




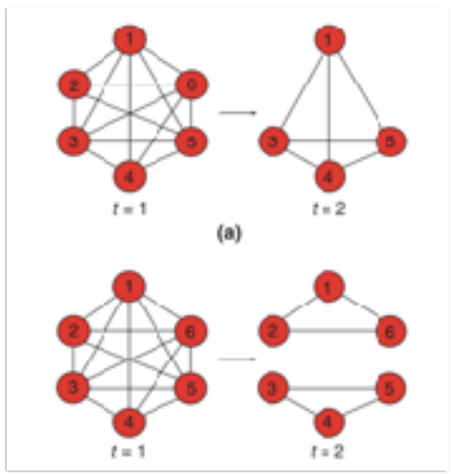
Anomaly Detection over Dynamic Graph

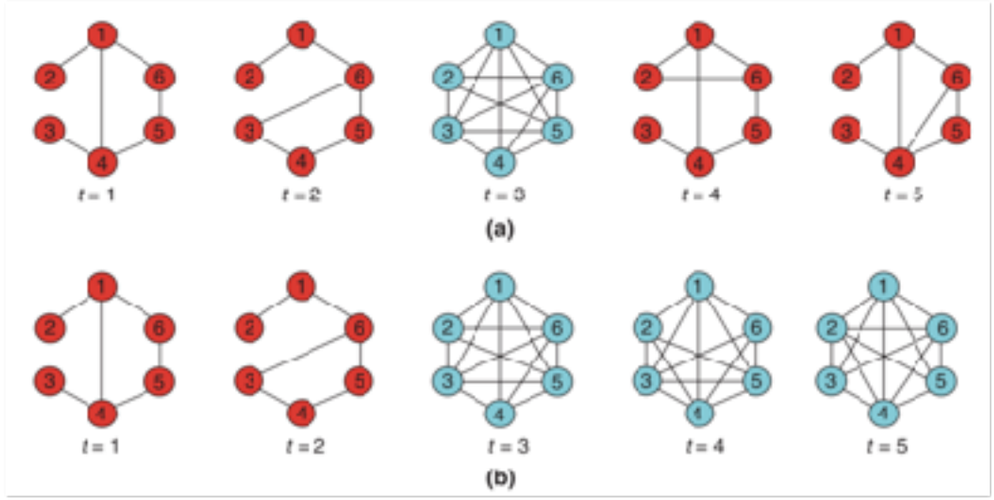
Content





Structure





Ranshous, S., Shen, S., Koutra, D., Harenberg, S., Faloutsos, C., & Samatova, N. F. (2015). Anomaly detection in dynamic networks: a survey. Wiley Interdisciplinary Reviews: Computational Statistics, 7(3), 223-247.

Anomaly Detection over Dynamic Graph

Challenges:

- Flexible and dynamic nature
- Insufficient labelled data
- Existing methods cannot perfectly update the representation dynamically





Summary & Future Directions

- Graph embedding (Graph Convolution Network) is a **powerful approach** to obtain **implicit** and **distributed** representation of a graph with **node (attributes)** and **link (relations)** information!

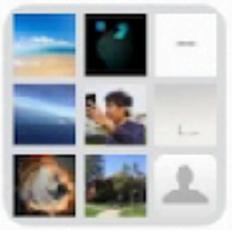
- **STAR-GCN** achieves SOTA results in **transductive and inductive** **rating prediction** [Zhang et al. UAI 2018]

- **Gated Attention Networks** achieve SOTA results in **inductive** **node classification** [Zhang et al. IJCAI 2019]

- Future Directions
  - **Temporal** and **heterogenous** graph neural networks
  - **Quantum random walk** algorithm for neighborhood sampling
  - **Sub-graph sampling**, **anomaly detection** in dynamic graphs
  - **Knowledge Graph** for Medical Academic **Citation Network**







认知图谱 2019

Source Code

- Code: https://github.com/jennyzhang0215/STAR-GCN

- [**Zhang et al. UAI 2018**] Zhang, Jiani, Xingjian Shi, Junyuan Xie, Hao Ma, Irwin King, and Dit-Yan Yeung. "*GAAN: Gated attention networks for learning on large and spatiotemporal graphs.*" UAI 2018.

- [**Zhang et al. IJCAI 2019**] Zhang, Jiani, Xingjian Shi, Shenglin Zhao, and Irwin King. "*STAR-GCN: Stacked and Reconstructed Graph Convolutional Networks for Recommender Systems*." IJCAI 2019.

THE WEB CONFERENCE | April 20-24 TAIPEI 2020

https://www2020.thewebconf.org

April 20-24, 2020

ABOUT AUTHORS ATTENDEES PROGRAMS SPONSORS



The Chinese University of Hong Kong

Thanks!