DEEP LEARNING

Lecture 11: Deep Learning on Graphs

Dr. Yang Lu

Department of Computer Science and Technology

luyang@xmu.edu.cn

厦門大學信息学院(特色化示范性软件学院)
School of Informatics Xiamen University (National Characteristic Demonstration Software School)

厦门大学 计算机科学与技术系
Department of Computer Science and Technology, Xiamen University

GNN Applications

- User profiling

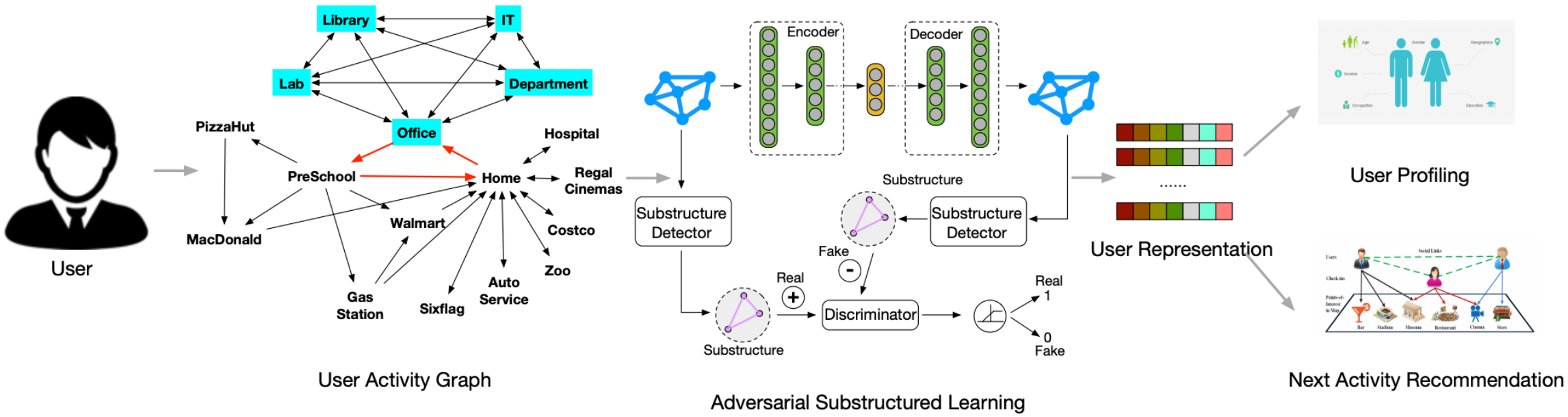


Image source: Wang, Pengyang, Yanjie Fu, Hui Xiong, and Xiaolin Li. "Adversarial substructured representation learning for mobile user profiling." In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 130-138. 2019.

GNN Applications

- Text classification

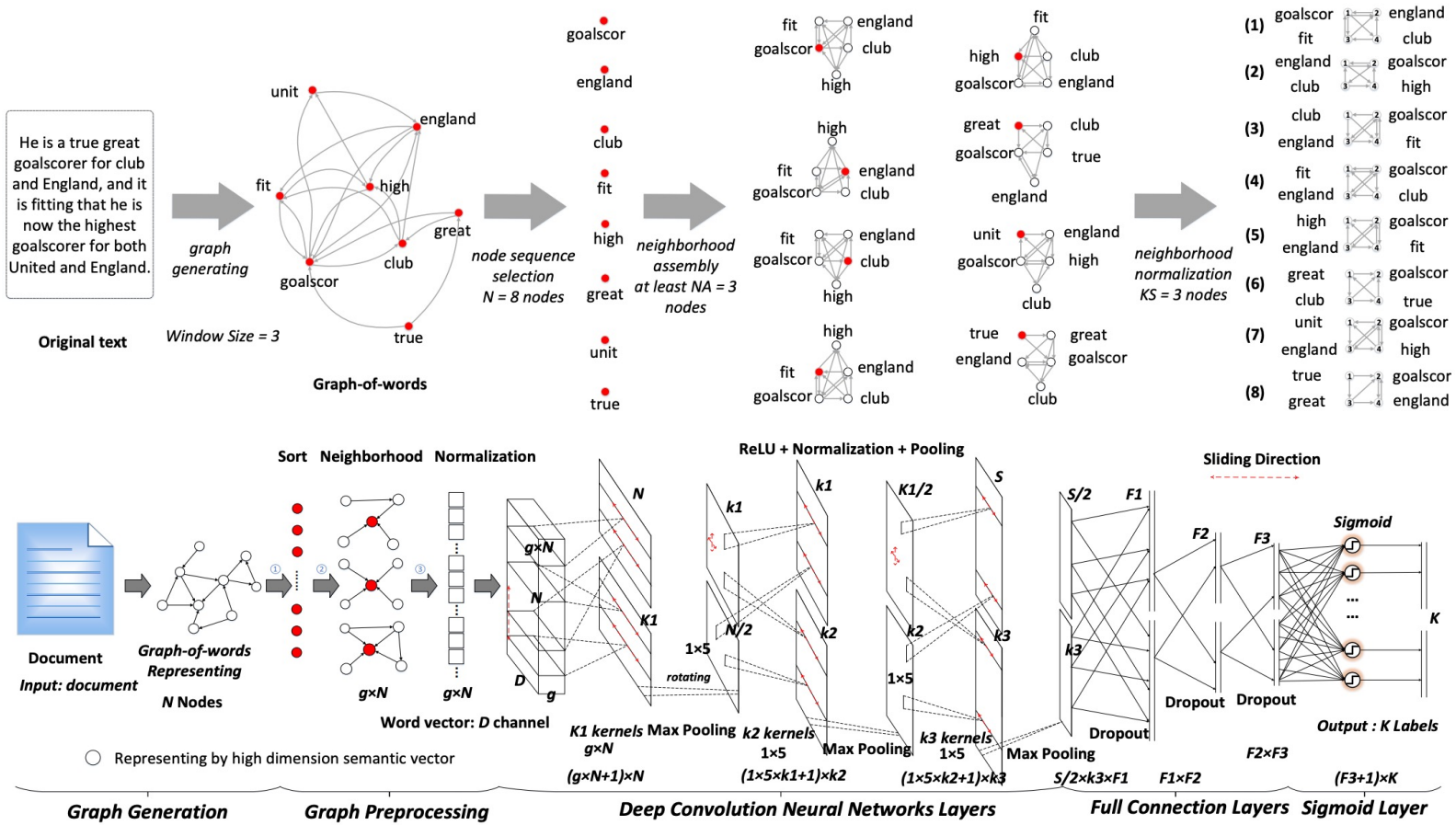


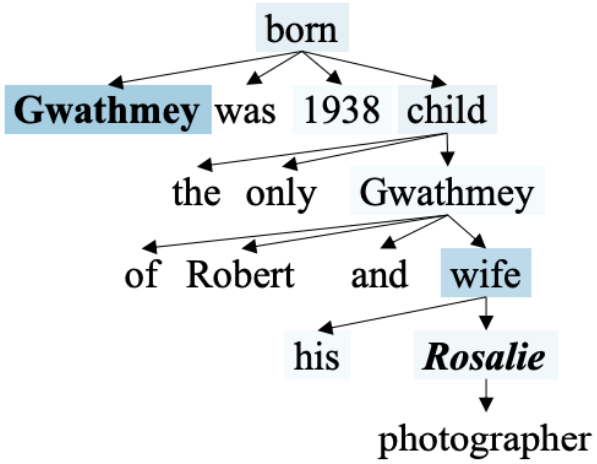
Image source: Peng, Hao, Jianxin Li, Yu He, Yaopeng Liu, Mengjiao Bao, Lihong Wang, Yangqiu Song, and Qiang Yang. "Large-scale hierarchical text classification with recursively regularized deep graph-cnn." In Proceedings of the 2018 World Wide Web Conference, pp. 1063-1072. 2018.

GNN Applications

Relation extraction

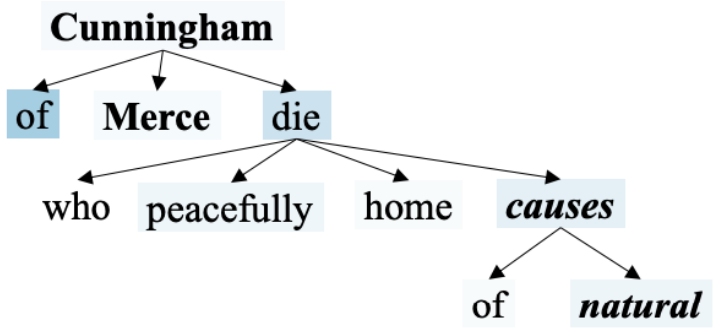
Relation: *per:parents*

Gwathmey was born in 1938, the only child of painter Robert Gwathmey and his wife, ***Rosalie***, a photographer.



Relation: *per:cause_of_death*

"It is with great sorrow that we note the passing of **Merce Cunningham**, who died peacefully in his home last night of ***natural causes***", the Cunningham Dance Foundation and the Merce Cunningham Dance Company said in a statement.



Relation: *per:employee_of*

Hwang, architect of the Pyongyang regime's ideology of "juche" or self-reliance, was once secretary of the ruling ***Workers' Party*** and a tutor to current leader Kim Jong-Il.

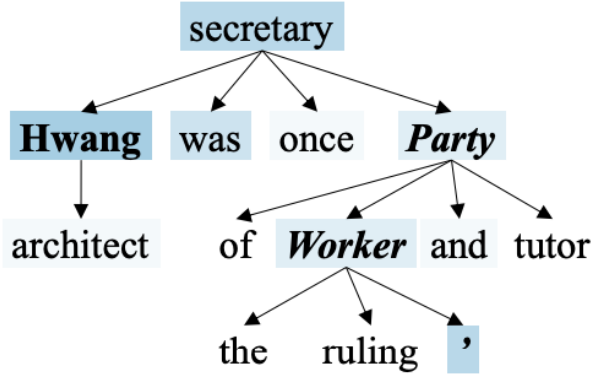


Image source: Zhang, Yuhao, Peng Qi, and Christopher D. Manning. "Graph convolution over pruned dependency trees improves relation extraction." arXiv preprint arXiv:1809.10185 (2018).

GNN Applications

- Zero-shot image classification

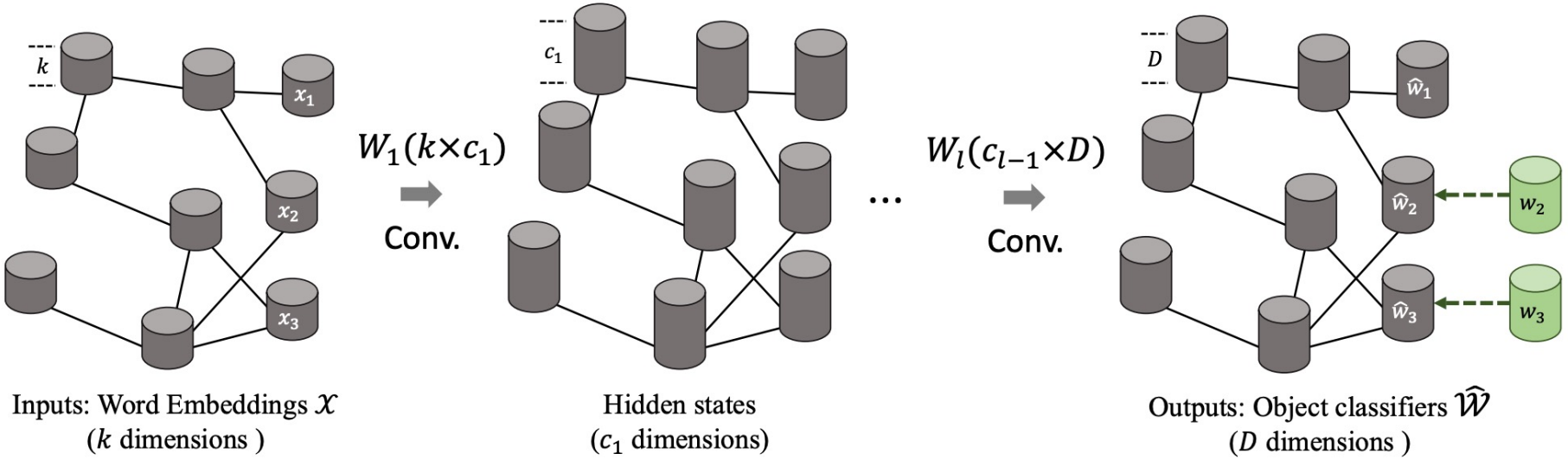


Image source: Wang, Xiaolong, Yufei Ye, and Abhinav Gupta. "Zero-shot recognition via semantic embeddings and knowledge graphs." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 6857-6866. 2018.

GNN Applications

- Point cloud semantic segmentation

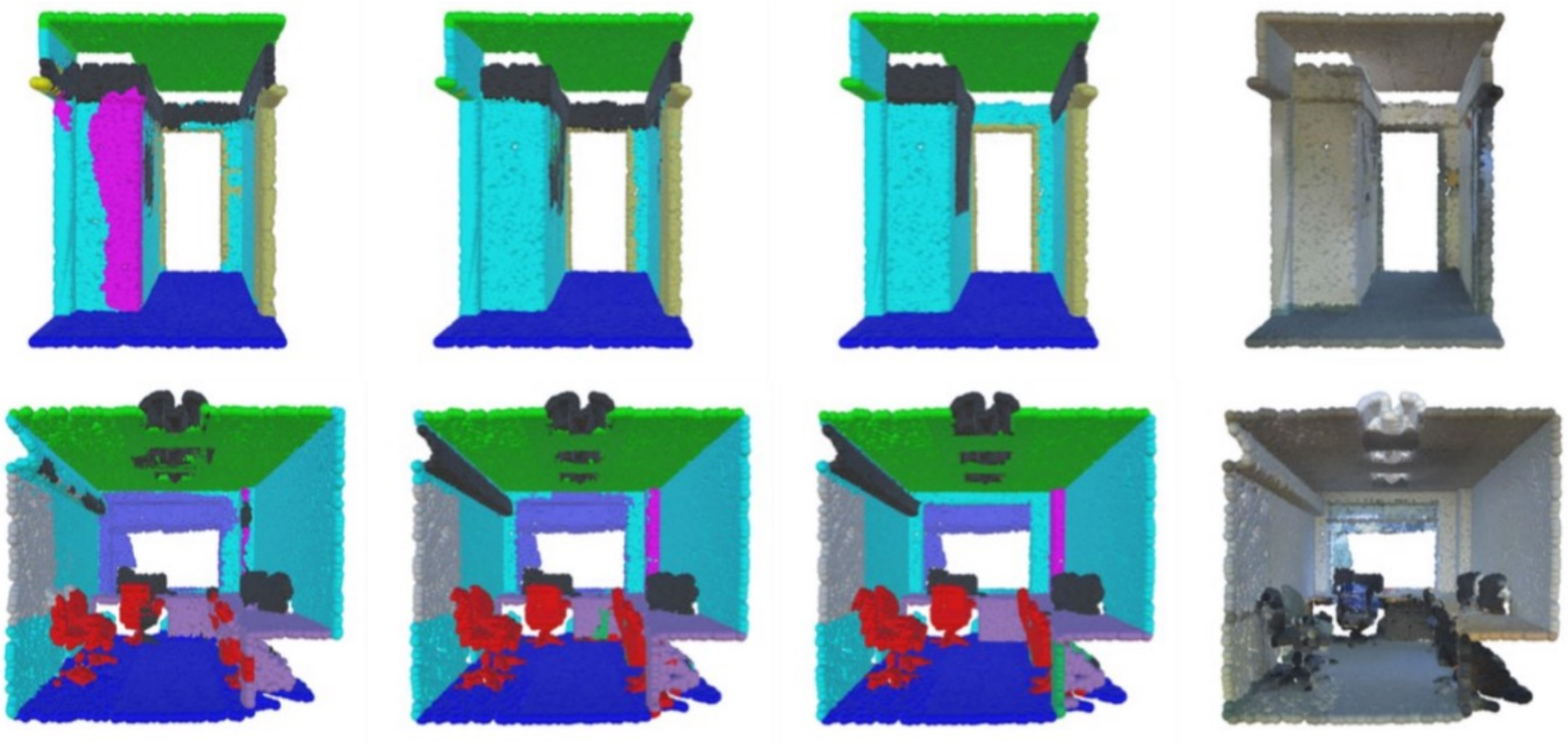


Image source: Wang, Yue, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. "Dynamic graph cnn for learning on point clouds." Acm Transactions On Graphics (tog) 38, no. 5 (2019): 1-12.

GNN Applications

- Visual question answering

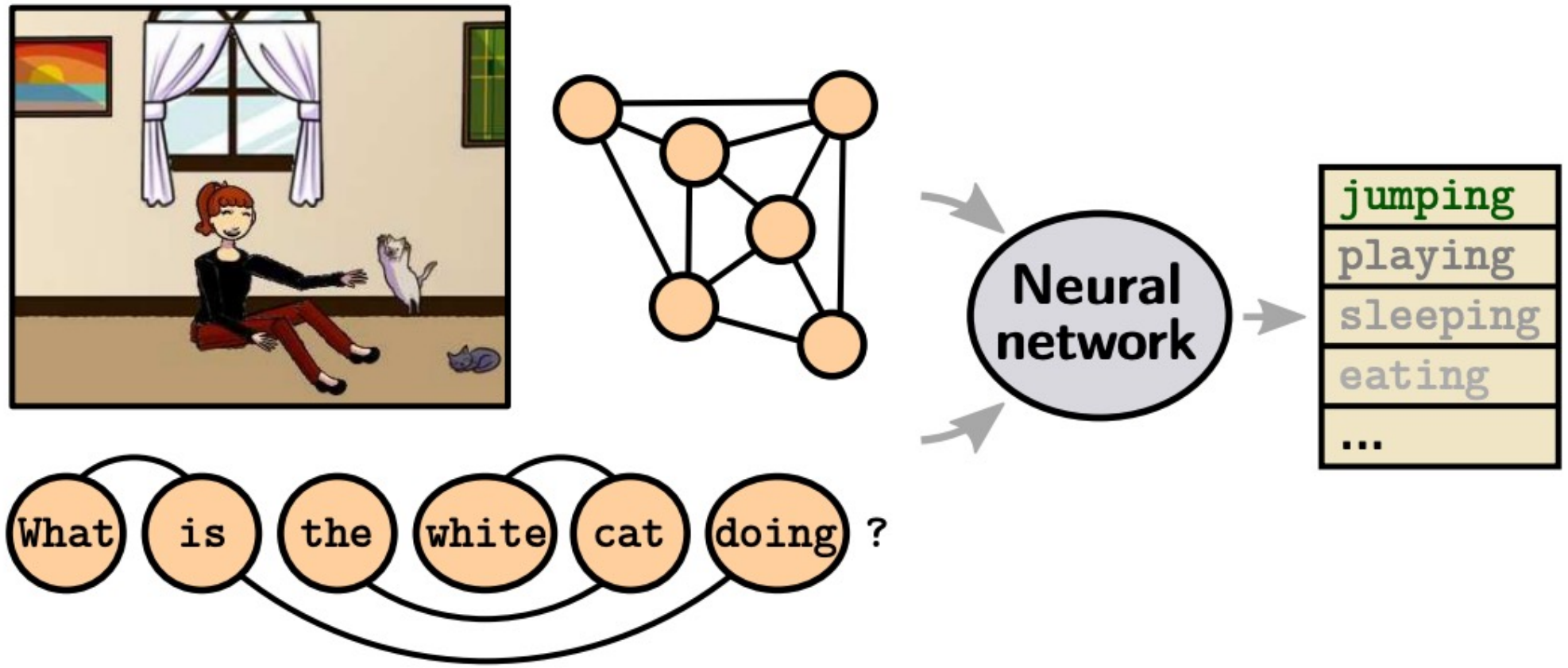
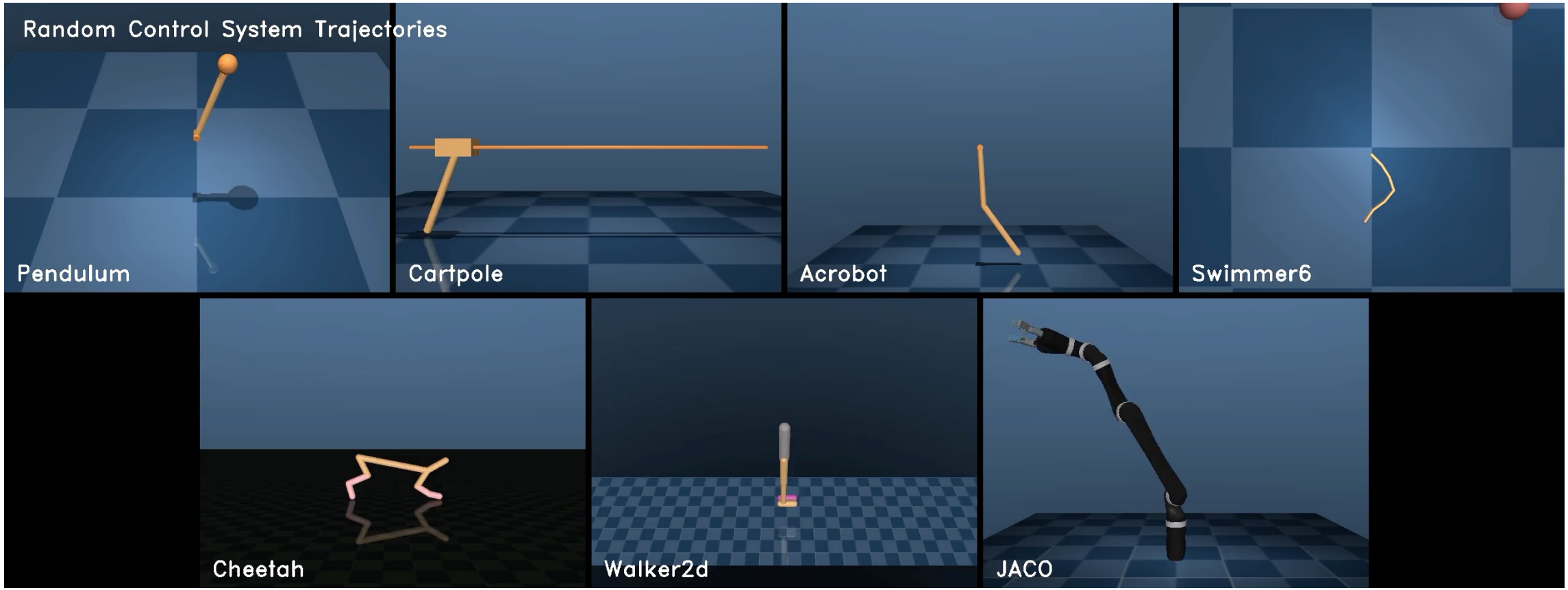


Image source: Teney, Damien, Lingqiao Liu, and Anton van Den Hengel. "Graph-structured representations for visual question answering." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1-9. 2017.

GNN Applications

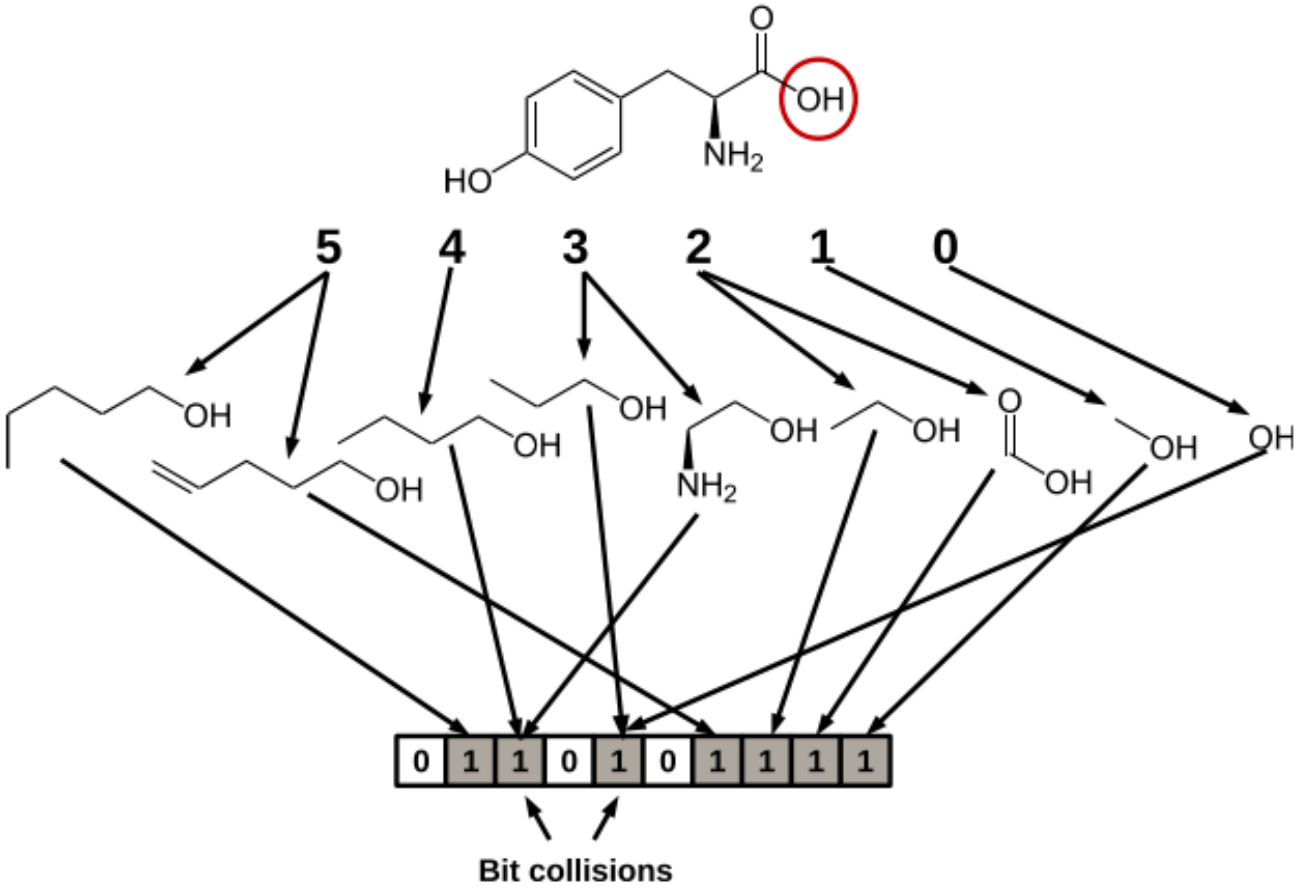
- Physics systems



Video source: Sanchez-Gonzalez, Alvaro, Nicolas Heess, Jost Tobias Springenberg, Josh Merel, Martin Riedmiller, Raia Hadsell, and Peter Battaglia. "Graph networks as learnable physics engines for inference and control." arXiv preprint arXiv:1806.01242 (2018).

GNN Applications

- Molecular fingerprints



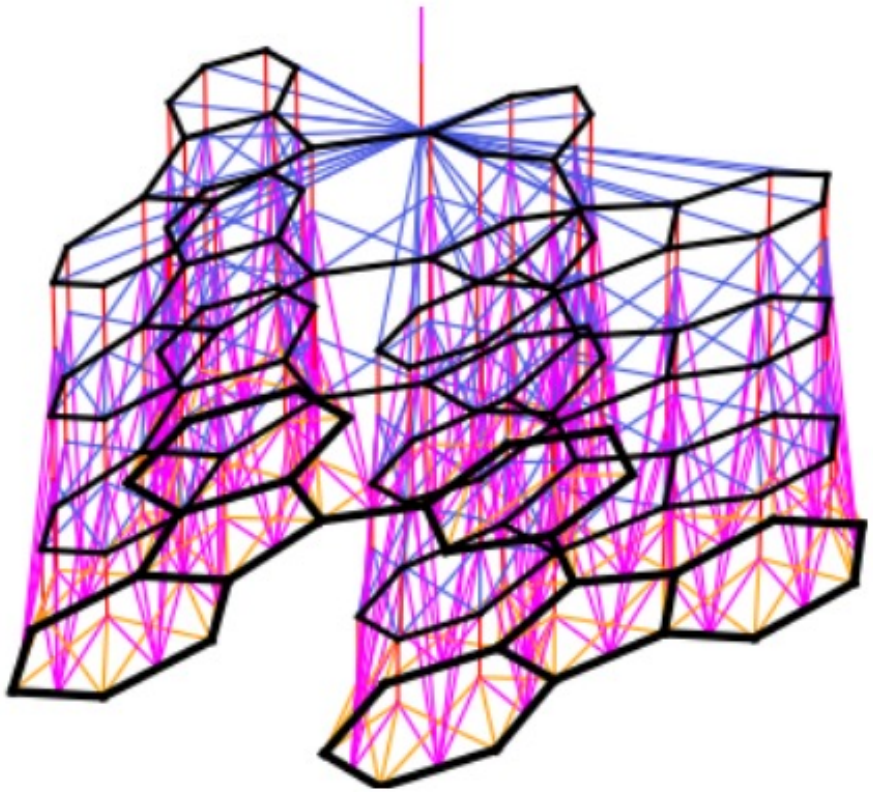


Image source: Duvenaud, David K., Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P. Adams. "Convolutional networks on graphs for learning molecular fingerprints." In Advances in neural information processing systems, pp. 2224-2232. 2015.
http://blog.molcalx.com.cn/2019/01/29/fingerprint.html

GNN Applications

- Protein interface prediction

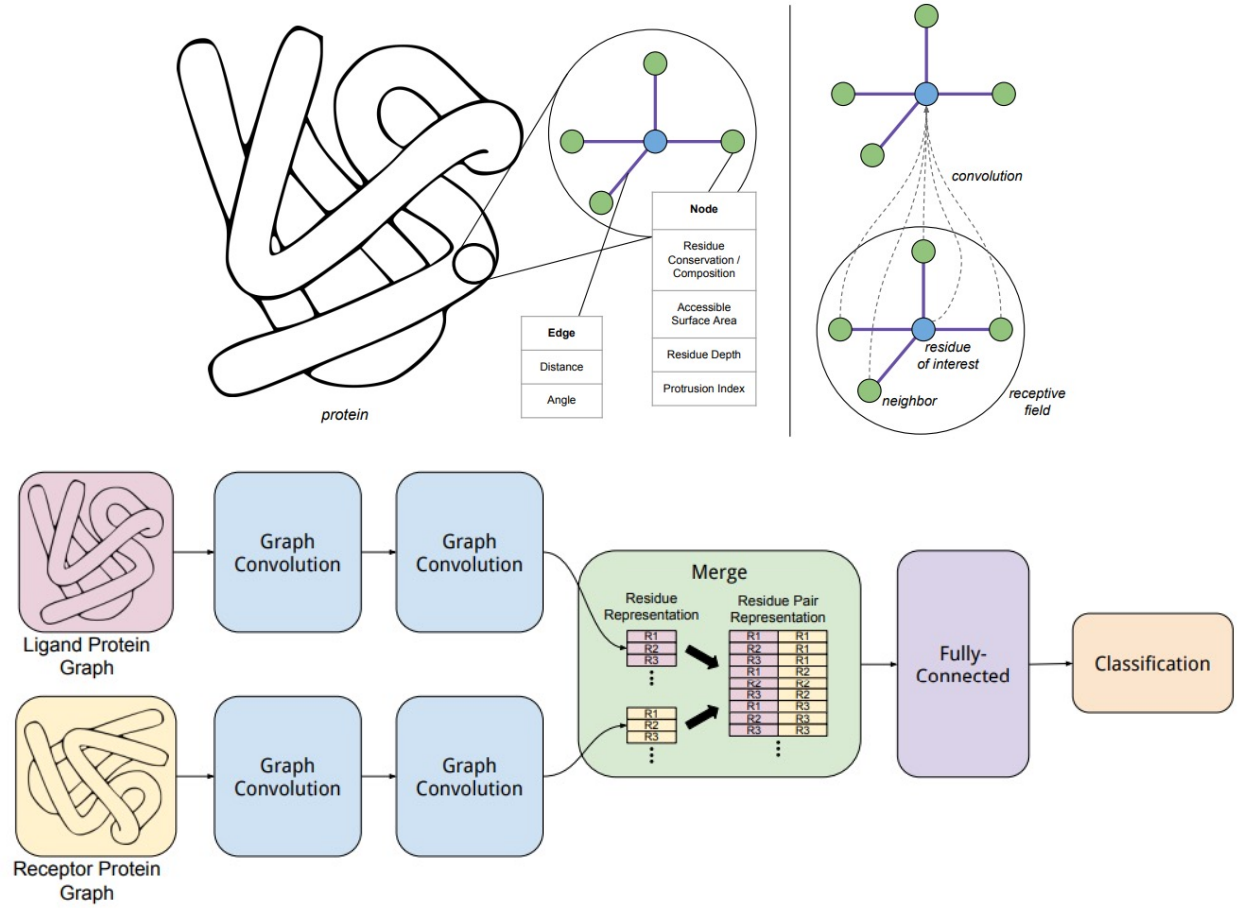


Image source: Fout, Alex, Jonathon Byrd, Basir Shariat, and Asa Ben-Hur. "Protein interface prediction using graph convolutional networks." In Advances in neural information processing systems, pp. 6530-6539. 2017.

GNN Applications

- Polypharmacy side effects

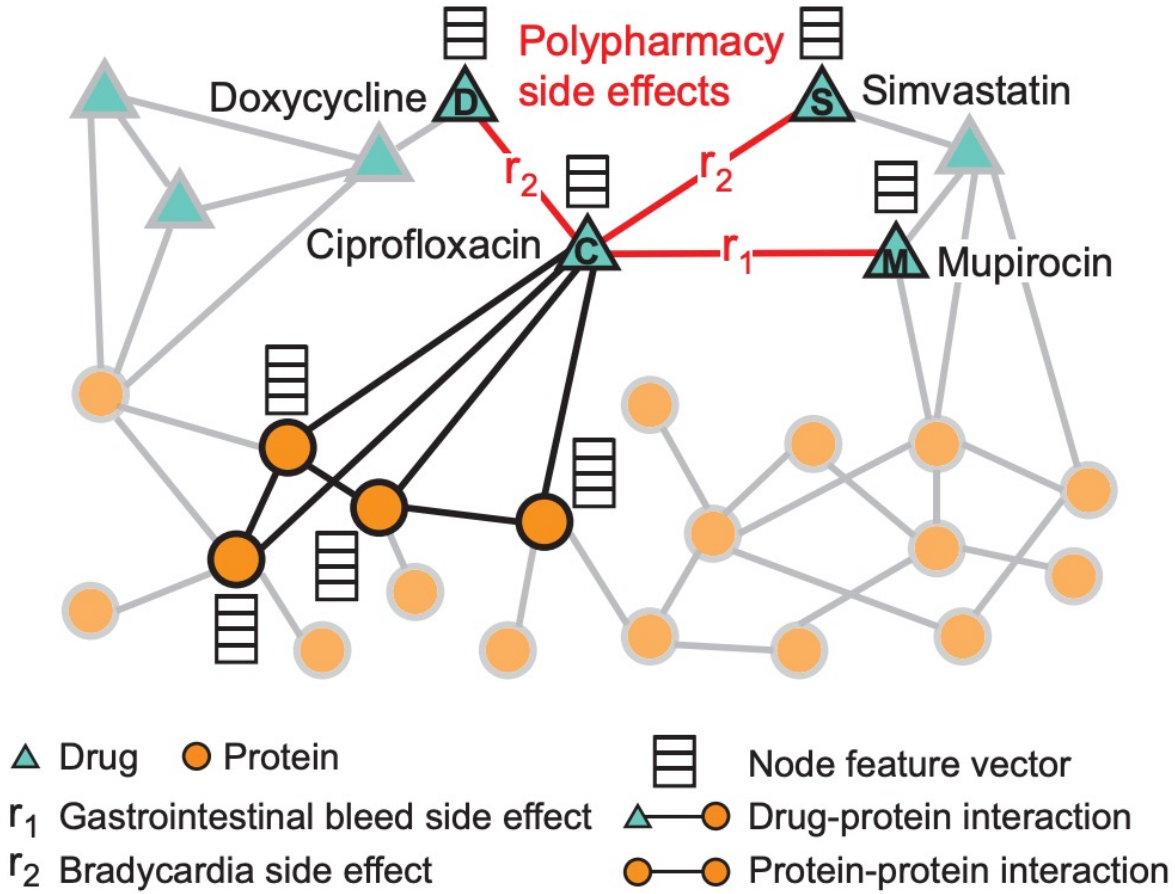


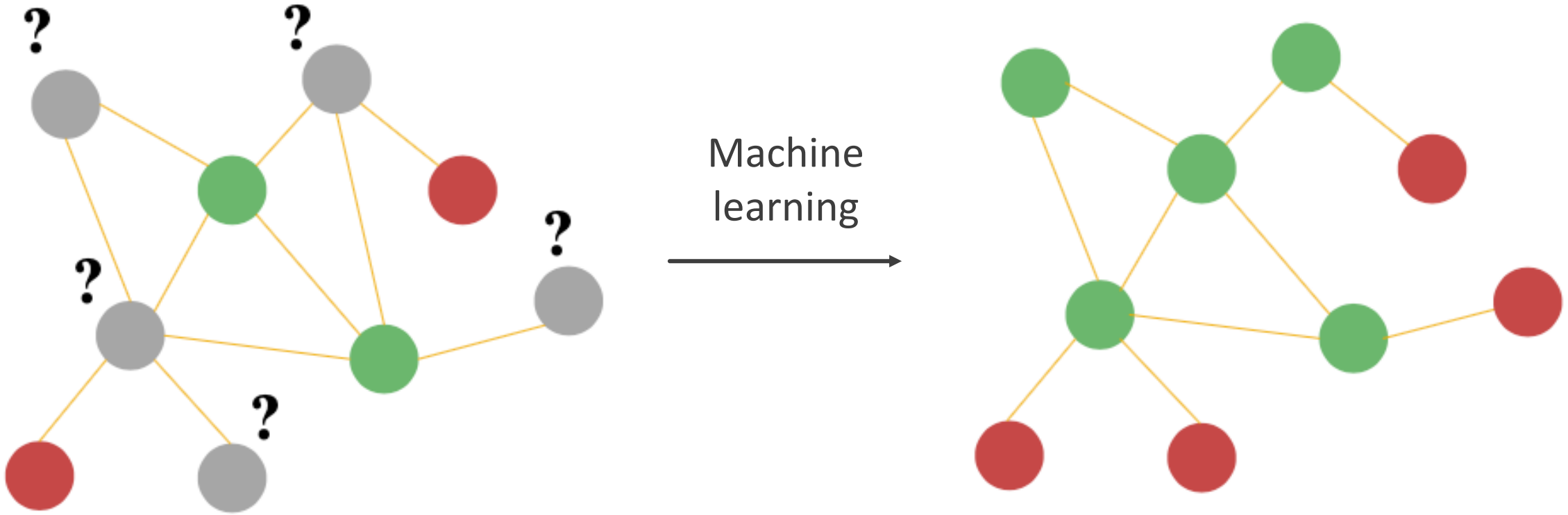
Image source: Zitnik, Marinka, Monica Agrawal, and Jure Leskovec. "Modeling polypharmacy side effects with graph convolutional networks." Bioinformatics 34, no. 13 (2018): i457-i466.

Outlines

- Graph Representation Learning
  - Deepwalk
  - LINE
  - Node2vec
- Graph Neural Networks
  - GCN
  - GraphSAGE
  - GAT
- Application to Recommender System
- Recent Advances

GRAPH REPRESENTATION LEARNING

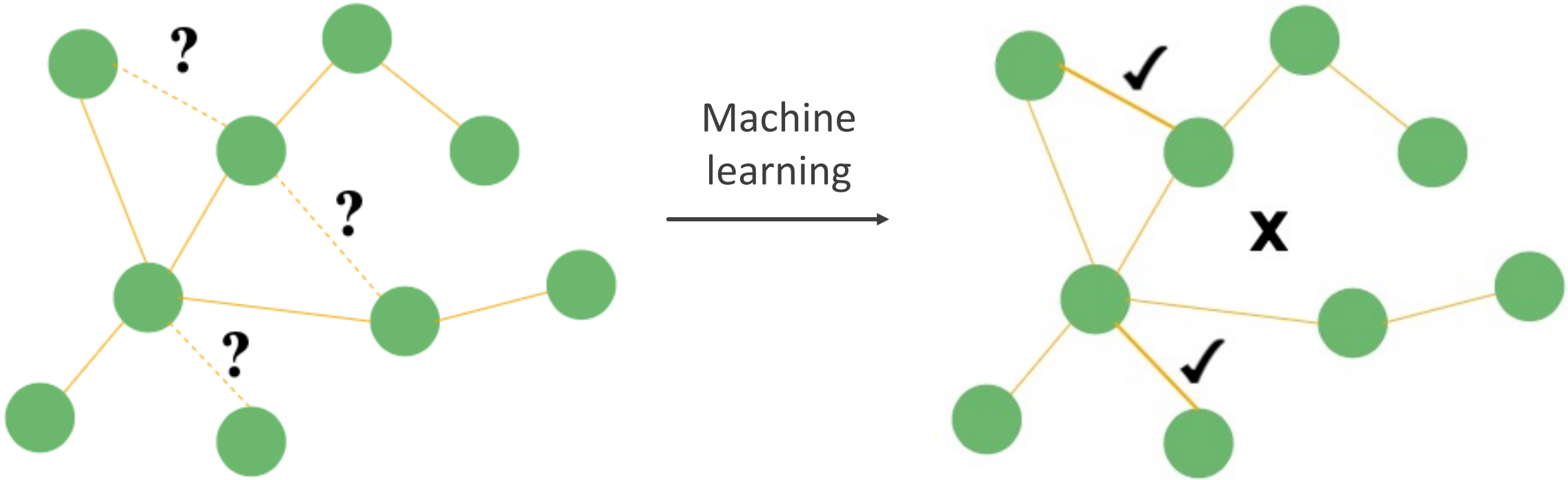
Graph Representation Learning



Node classification

Image source: Lecture 7, cs224w, Stanford University

Graph Representation Learning



Link prediction

Image source: Lecture 7, cs224w, Stanford University

Graph Representation Learning

- Given the graph, the only information we have is

$$G = \langle V, E \rangle$$

  and their corresponding labels.
- What are the features?
- We can do feature engineering:
  - degree (count of adjacent nodes);
  - mean of degrees of neighbor nodes;
  - number of triangles a node forms with other nodes;
  - …

Graph Representation Learning

- Goal: Efficiently learn task-independent features (embeddings) from graphs.

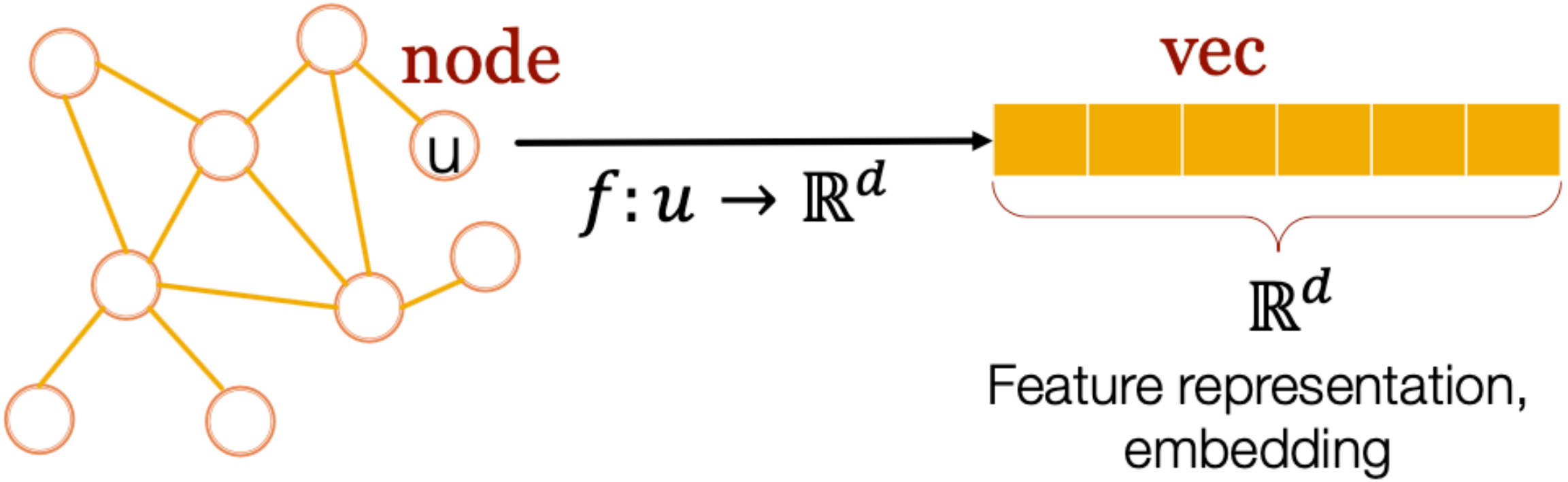


Image source: Lecture 7, cs224w, Stanford University

Graph Representation Learning

- Can we directly apply CNN or RNN on graphs?
- Probably no, because images and texts are structured.
  - Images are 2d matrices.
  - Texts are sequences.
- Graphs are far more complex.

Graph Representation Learning

Before the age of deep learning, we have some traditional machine learning methods:

- Locally Linear Embedding: low dimensional representations of each node can be represented by the linear combination of its neighbors.

$$\min \frac{1}{2} \sum_i \left| \boldsymbol{x}_i - \sum_j W_{ij} \boldsymbol{x}_j \right|^2$$

- Laplacian Eigenmaps: low dimensional representations of connected nodes are similar.

$$\min \frac{1}{2} \sum_{i,j} \left| \boldsymbol{x}_i - \boldsymbol{x}_j \right|^2 W_{ij}$$

- Graph Factorization: matrix factorization.

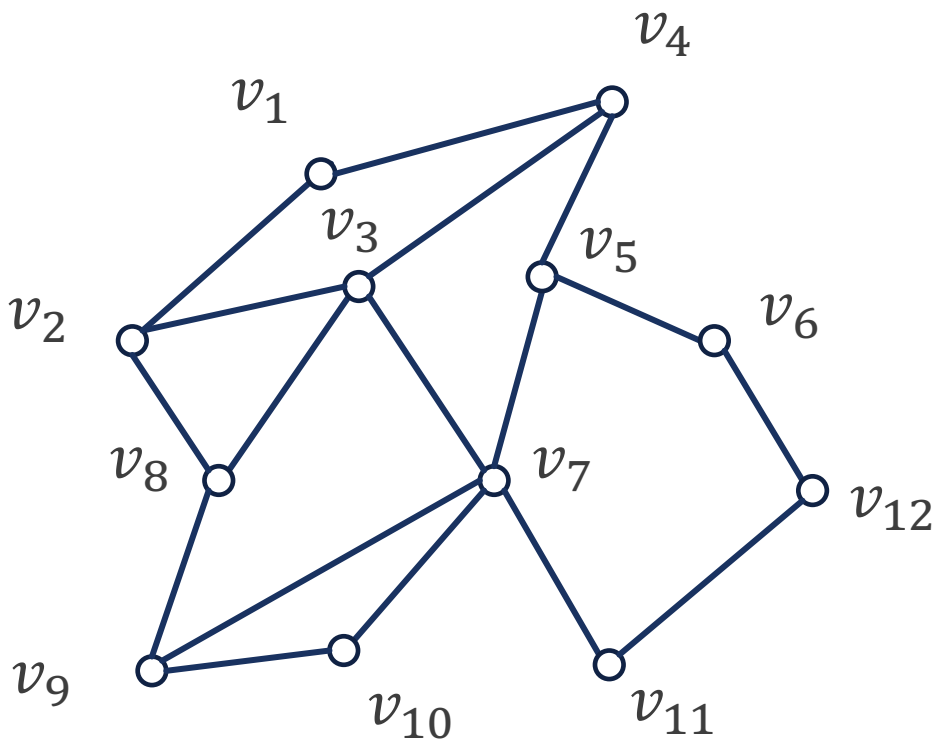
Graph Representation Learning

- Problem of these methods: can't scale!
- Key idea: If we assume that the connected nodes share similar properties (e.g. labels) in a graph, we should make their representations similar.
- Recall something?
- Word2vec and xxx2vec!
- But how to generate training pairs?

DEEPWALK

Deepwalk

Deepwalk: Online learning of social representations
B Perozzi, R Al-Rfou, S Skiena - Proceedings of the 20th ACM SIGKDD …, 2014 - dl.acm.org
… **DeepWalk**, a … **DeepWalk** generalizes recent advancements in language modeling and unsupervised feature learning (or deep learning) from sequences of words to graphs. **DeepWalk** …
☆ Save 99 Cite Cited by 10106 Related articles All 22 versions



(v_3, v_4, v_5, v_6)

(v_7, v_{10}, v_9, v_8)

Then what?

Skipgram!

Algorithm 1 DEEPWALK(G, w, d, γ, t)

Input: graph $G(V, E)$
window size w
embedding size d
walks per vertex γ
walk length t
Output: matrix of vertex representations $\Phi \in \mathbb{R}^{|V| \times d}$

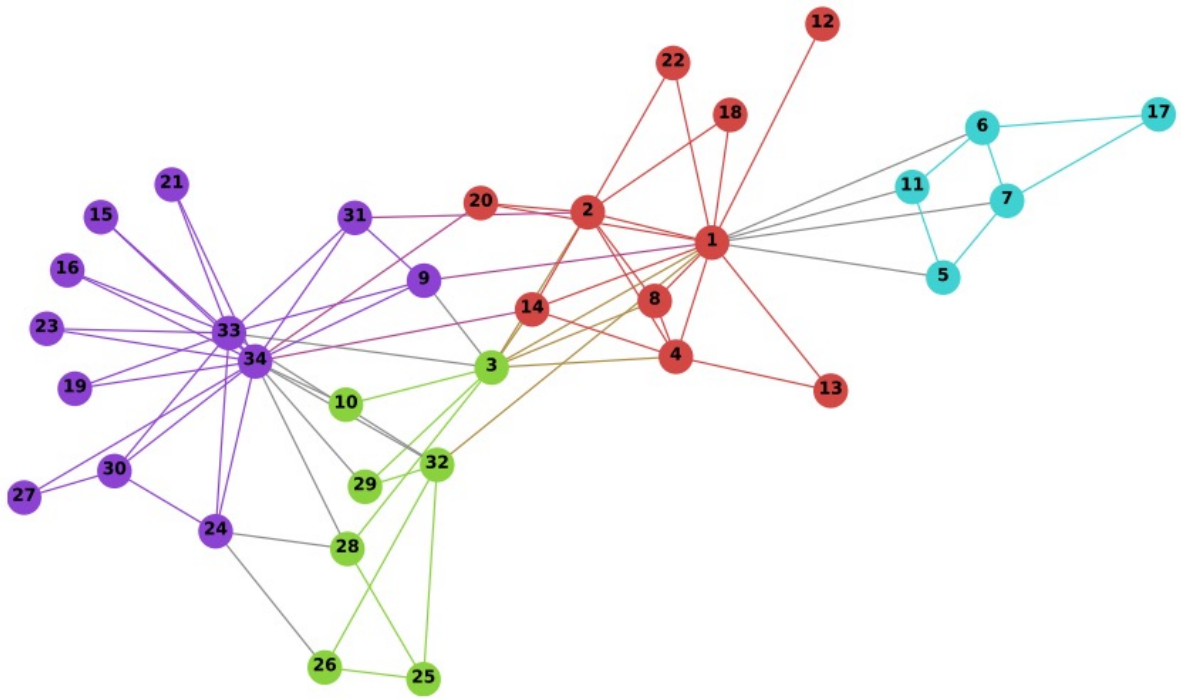
Initialization: Sample Φ from $\mathcal{U}^{|V| \times d}$
Build a binary Tree T from V
for $i = 0$ to γ **do**
  $\mathcal{O} = \text{Shuffle}(V)$
  for each $v_i \in \mathcal{O}$ **do**
    $\mathcal{W}_{v_i} = RandomWalk(G, v_i, \text{t})$
    SkipGram(Φ, $\mathcal{W}_{v_i}$, w)
  end for
end for

Algorithm 2 SkipGram(Φ, $\mathcal{W}_{v_i}$, w)

for each $v_j \in \mathcal{W}_{v_i}$ **do**
  for each $u_k \in \mathcal{W}_{v_i}[j - w : j + w]$ **do**
    $J(\Phi) = -\log \Pr(u_k \mid \Phi(v_j))$
    $\Phi = \Phi - \alpha * \frac{\partial J}{\partial \Phi}$
  end for
end for

Image source: Perozzi, Bryan, Rami Al-Rfou, and Steven Skiena. "Deepwalk: Online learning of social representations." In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 701-710. 2014.

Deepwalk



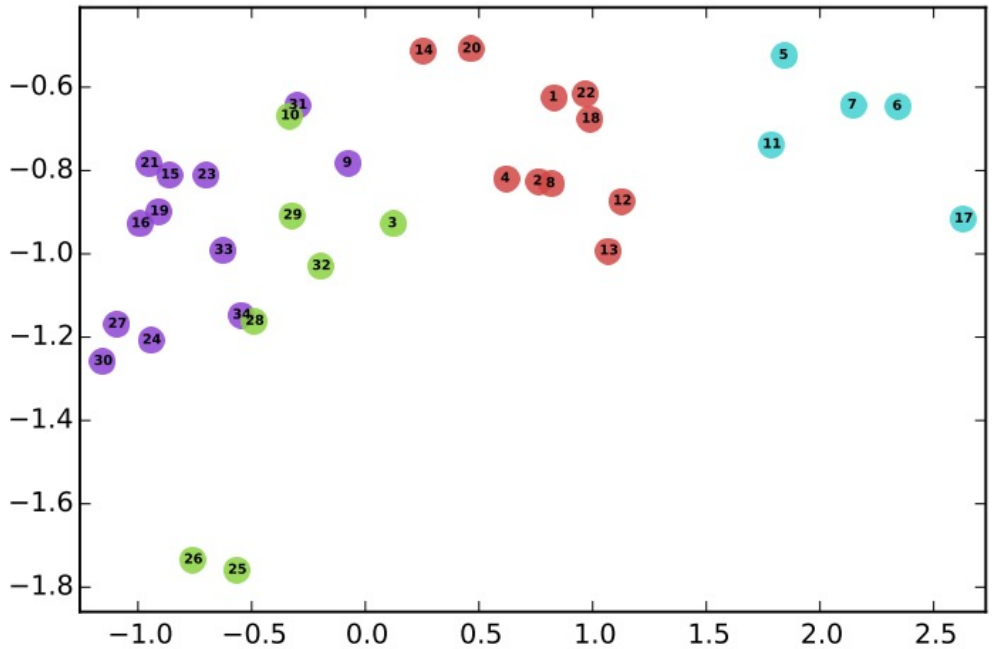
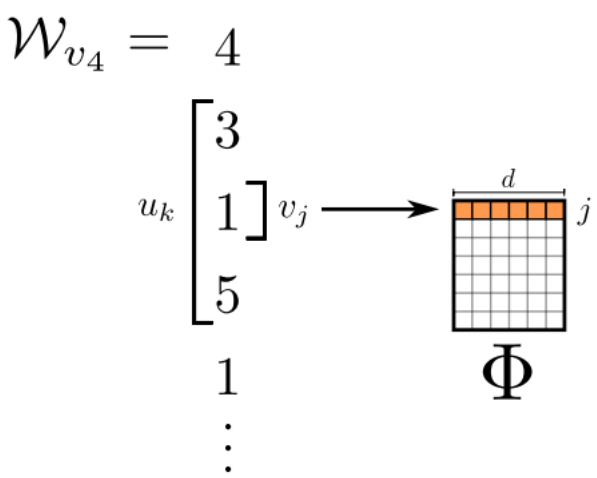


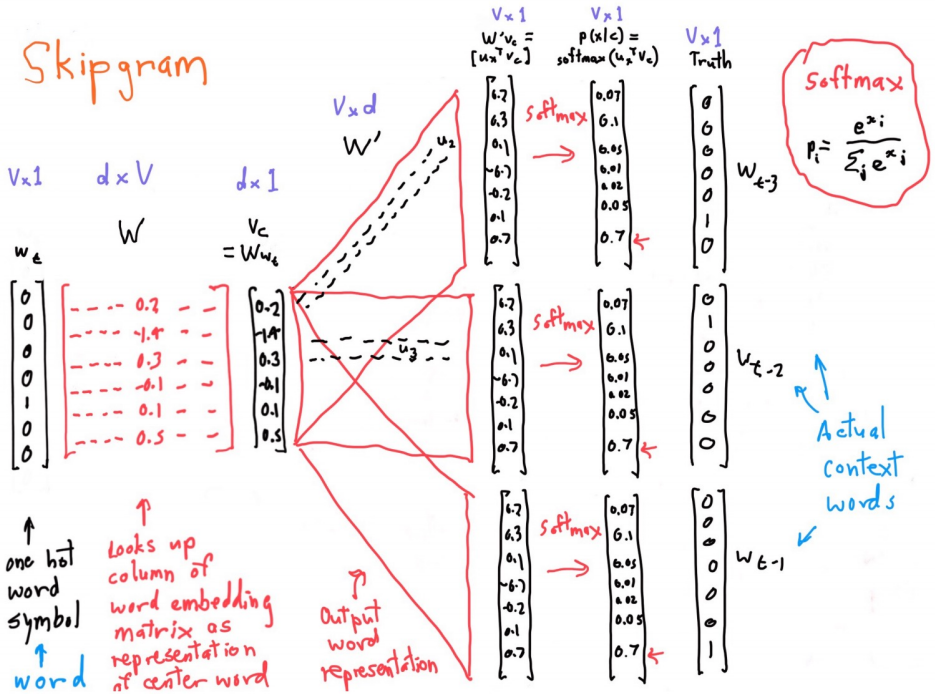
Image source: Perozzi, Bryan, Rami Al-Rfou, and Steven Skiena. "Deepwalk: Online learning of social representations." In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 701-710. 2014.

Difference

- Notice any difference between Deepwalk and Word2vec?



Deepwalk



Word2vec

Image source: Perozzi, Bryan, Rami Al-Rfou, and Steven Skiena. "Deepwalk: Online learning of social representations." In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 701-710. 2014.

Image source: Lecture 2, cs224n

Datasets

- Task: multi-label classification.
- Take Flickr as an example:
  - Nodes: users.
  - Links: following between users.
  - Categories: subscribe to different interest groups (e.g. black and white photos, or animals).

Data	BlogCatalog	Flickr	YouTube
Categories	39	195	47
Nodes (n)	10, 312	80, 513	1, 138, 499
Links (m)	333, 983	5, 899, 882	2, 990, 443
Network Density	6.3×10^{-3}	1.8×10^{-3}	4.6×10^{-6}
Maximum Degree	3, 992	5, 706	28, 754
Average Degree	65	146	5

Source: Tang, Lei, and Huan Liu. "Scalable learning of collective behavior based on sparse social dimensions." In Proceedings of the 18th ACM conference on Information and knowledge management, pp. 1107-1116. 2009.

Problems of Deepwalk

- Deepwalk is a pioneer work that builds a bridge between graph representation and word2vec.
- However, it is not specifically designed for graphs.
- How about directed graph? Weighted graph?

LINE

LINE

Line: Large-scale information network embedding
J Tang, M Qu, M Wang, M Zhang, J Yan… - Proceedings of the 24th …, 2015 - dl.acm.org
… We compare the **LINE** model with several existing graph embedding methods that are able to **scale** up to very **large** networks. We do not compare with some classical graph embedding …
☆ Save Cite Cited by 5808 Related articles All 16 versions

- First-order proximity in the real world data is not sufficient for preserving the global network structures.
- Second-order proximity is also very important.
  - It can be interpreted as nodes with shared neighbors being likely to be similar.
- The degree of overlap of two people's friendship networks correlates with the strength of ties between them.

LINE

- Vertex 6 and 7 should be placed closely in the low-dimensional space as they are connected through a strong tie.
- Vertex 5 and 6 should also be placed closely as they share similar neighbors.

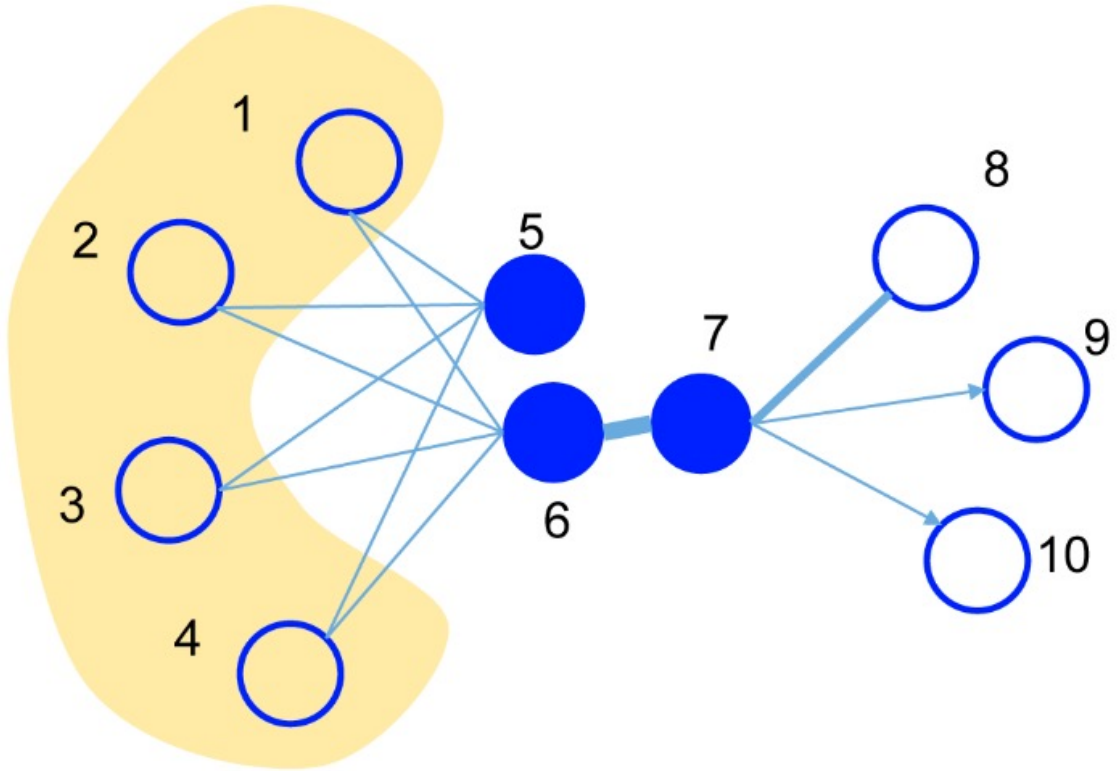


Image source: Tang, Jian, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. "Line: Large-scale information network embedding." In Proceedings of the 24th international conference on world wide web, pp. 1067-1077. 2015.

LINE with First-Order Proximity

- For each undirected edge (i, j), the joint probability between vertex v_i and v_j as follows:

$$p_1(v_i, v_j) = \frac{1}{1 + \exp(-\boldsymbol{u}_i^T \boldsymbol{u}_j)}.$$

- Use their edge weight as the label, W is total weight in the graph.

$$\hat{p}_1(v_i, v_j) = \frac{w_{ij}}{W}.$$

- Minimize the KL-divergence between p_1 and $\hat{p}_1$:

$$-\sum_{(i,j)\in E} w_{ij} \log p_1(v_i, v_j).$$

The constant W can be omitted in minimization

LINE with First-Order Proximity

- For each directed edge (i, j), we first define the conditional probability of "context" v_j generated by vertex v_i as:

$$p_1(v_j|v_i) = \frac{1}{1 + \exp(-\boldsymbol{u}_j'^T \boldsymbol{u}_i)}$$

- Here, we use difference representations for center and context just like Word2vec, why?

LINE with Second-Order Proximity

- If we consider the second-order proximity, v_j can be the neighbor of v_i's neighbor.

$$p_2(v_j|v_i) = \frac{\exp(\boldsymbol{u}_j'^T \boldsymbol{u}_i)}{\sum_{k=1}^{|V|} \exp(\boldsymbol{u}_k'^T \boldsymbol{u}_i)}$$

where $|V|$ is the number of vertices or "contexts."

LINE with Second-Order Proximity

- Similarly, minimize the KL-divergence:

$$-\sum_{j\in\mathcal{N}(i)} w_{ij}\log p_2\left(v_j|v_i\right).$$

where $\mathcal{N}(i)$ is the neighborhood of v_i, including first and second order.

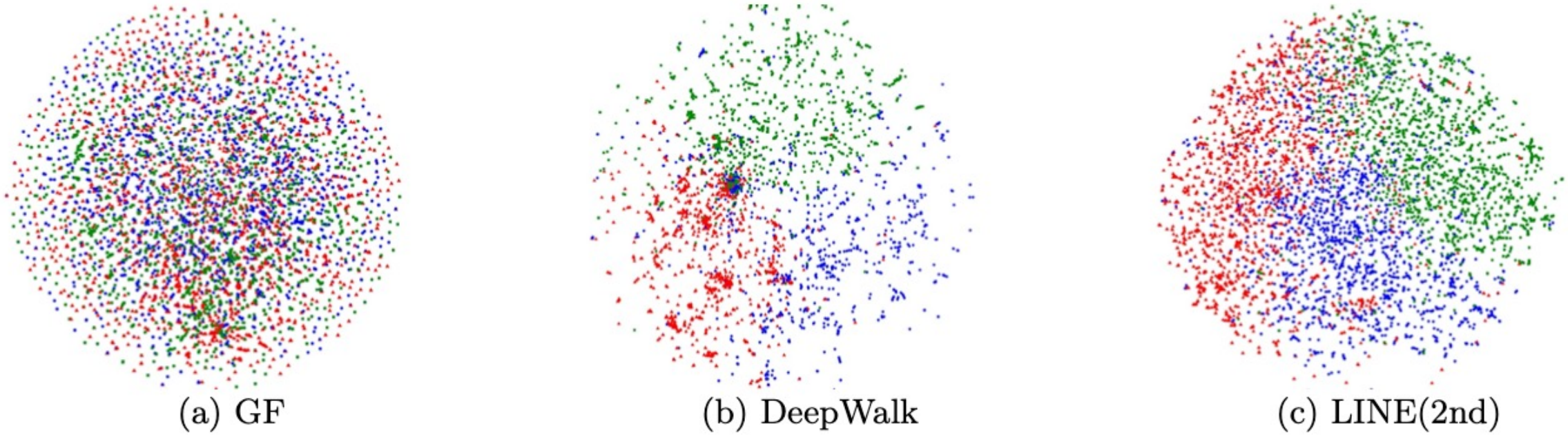
- w_{ij} depends on two situation:
  - v_j is the neighbor of v_i: w_{ij} is simply the weight.
  - v_j is the neighbor of neighbor of v_i: $w_{ij}=\sum_{k\in\mathcal{N}(i)} w_{ik}\frac{w_{kj}}{d_k}$, d_k is the out-degree of v_k.

Combining First-Order and Second-Order Proximities

Two ways:

- Train separately and then concatenate.
- Jointly train the objective function.

LINE



Visualization of the co-author network. The authors are mapped to the 2-D space using the t-SNE package with learned embeddings as input. Color of a node indicates the community of the author. Red: “data Mining,” blue: “machine learning,” green: “computer vision.”

Image source: Tang, Jian, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. "Line: Large-scale information network embedding." In Proceedings of the 24th international conference on world wide web, pp. 1067-1077. 2015.

Deepwalk vs LINE

- Deepwalk is actually a returnable DFS.
- LINE is a 2-level BFS.

Can we combine DFS and BFS?

NODE2VEC

Node2vec

node2vec: Scalable feature learning for networks
A Grover, J Leskovec - Proceedings of the 22nd ACM SIGKDD …, 2016 - dl.acm.org
… **node2vec**, an algorithmic framework for learning continuous feature representations for nodes in networks. In **node2vec**, … We demonstrate the efficacy of **node2vec** over existing state-of…
☆ Save Cite Cited by 10798 Related articles All 25 versions

- Motivation: It is now either DFS (Deepwalk) or BFS (LINE). It is too rigid to explore the network neighborhood.
- Can we make it flexible?

Node2vec

Jure Leskovec

FOLLOW

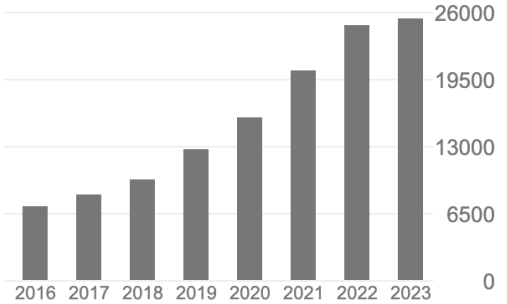
Professor of Computer Science, Stanford University
Verified email at cs.stanford.edu - Homepage

Data mining Machine Learning Graph Neural Networks Knowledge Graphs Complex Networks

TITLE	CITED BY	YEAR
Inductive representation learning on large graphs W Hamilton, Z Ying, J Leskovec Advances in neural information processing systems 30	12636	2017
node2vec: Scalable feature learning for networks A Grover, J Leskovec Proceedings of the 22nd ACM SIGKDD international conference on Knowledge …	10798	2016
How powerful are graph neural networks? K Xu, W Hu, J Leskovec, S Jegelka arXiv preprint arXiv:1810.00826	6074	2018
SNAP Datasets: Stanford large network dataset collection J Leskovec, A Krevl	4148	2014
Friendship and mobility: user movement in location-based social networks E Cho, SA Myers, J Leskovec Proceedings of the 17th ACM SIGKDD international conference on Knowledge …	3511	2011
Graphs over time: densification laws, shrinking diameters and possible explanations J Leskovec, J Kleinberg, C Faloutsos Proceedings of the eleventh ACM SIGKDD international conference on Knowledge …	3173	2005

Cited by VIEW ALL

	All	Since 2018
Citations	149434	109176
h-index	145	124
i10-index	340	316



Public access VIEW ALL

2 articles 140 articles

not available available

Based on funding mandates

Node2vec

- Nodes u and s_1 belonging to the same tightly knit community.
- Nodes u and s_6 in the two distinct communities share the same structural role of a hub node.
- Should u be similar to s_1 or s_6?
- Both, but in different perspective.

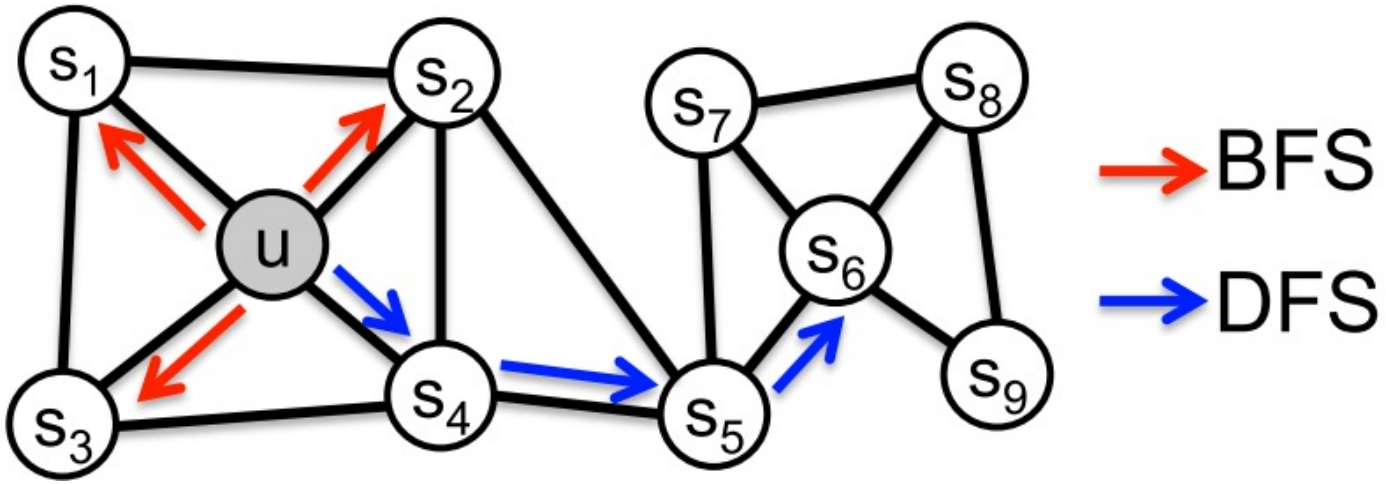


Image source: Grover, Aditya, and Jure Leskovec. "node2vec: Scalable feature learning for networks." In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 855-864. 2016.

Node2vec

- Real-world networks commonly exhibit a mixture of such equivalences.
- The representations should be flexible to have similar embeddings for:
  - nodes from the same network community;
  - nodes that share similar roles.

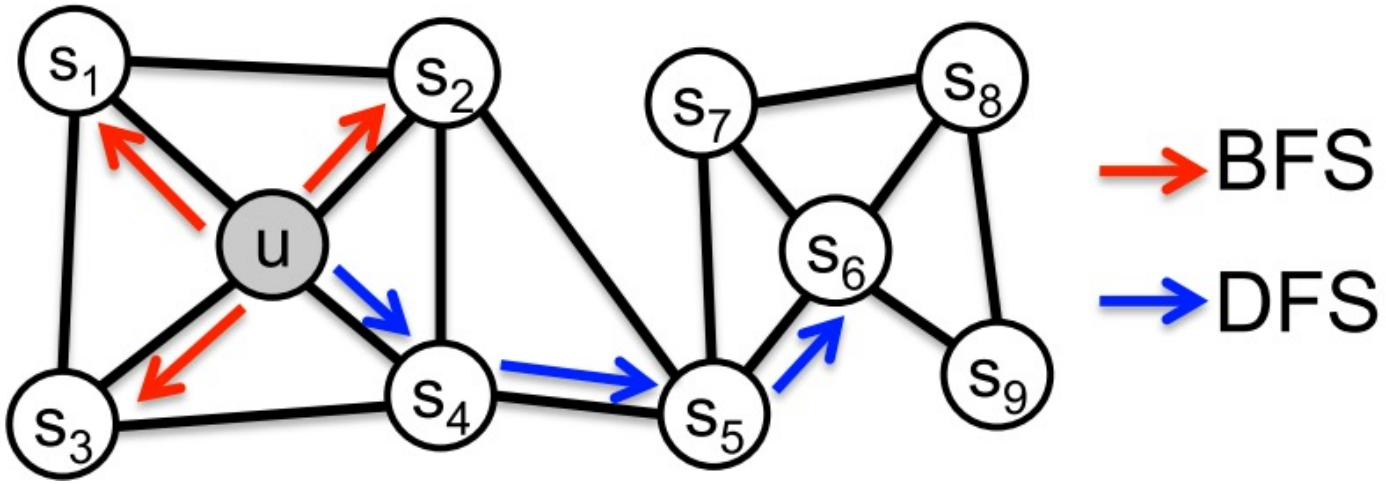


Image source: Grover, Aditya, and Jure Leskovec. "node2vec: Scalable feature learning for networks." In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 855-864. 2016.

Node2vec

- Idea: use flexible, biased random walks that can trade off between local and global views of the network.
- Walk of length 3 ($N_R(u)$ of size 3):
  - $N_{BFS}(u) = \{s_1, s_2, s_3\}$, local microscopic view.
  - $N_{DFS}(u) = \{s_4, s_5, s_6\}$, global macroscopic view.

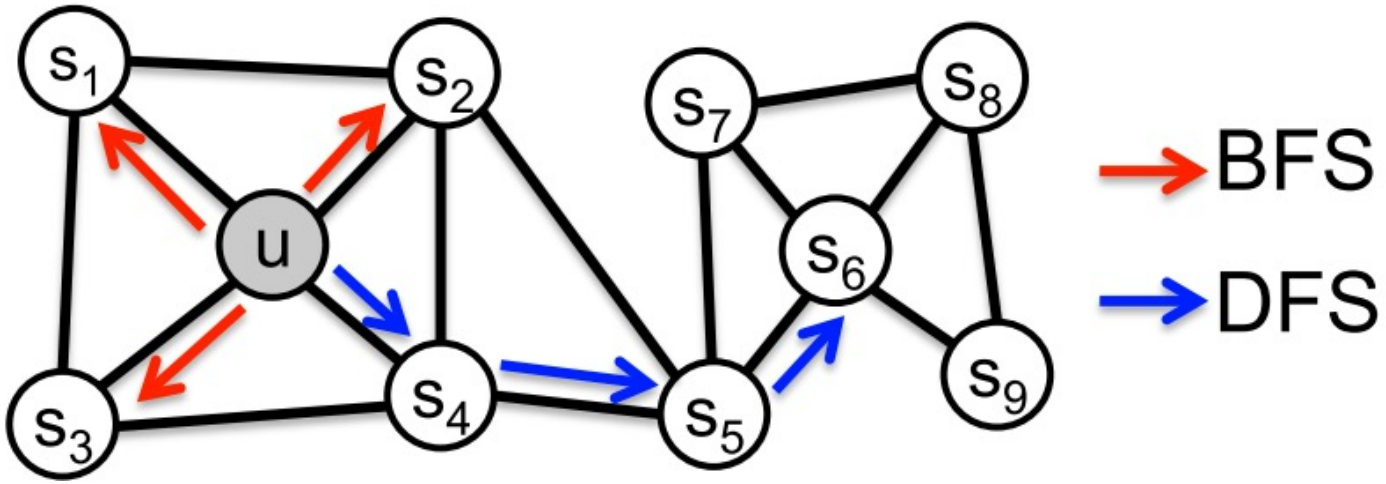


Image source: Grover, Aditya, and Jure Leskovec. "node2vec: Scalable feature learning for networks." In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 855-864. 2016.

Node2vec

- The probability from c_{i-1} to c_i is:

$$P(c_i = x | c_{i-1} = v) = \begin{cases} \frac{\pi_{vx}}{Z} & \text{if } (v, x) \in E \\ 0 & \text{otherwise} \end{cases}$$

- The first step is same for both DFS and BFS, by simply setting:

$$\pi_{vx} = w_{vx}$$

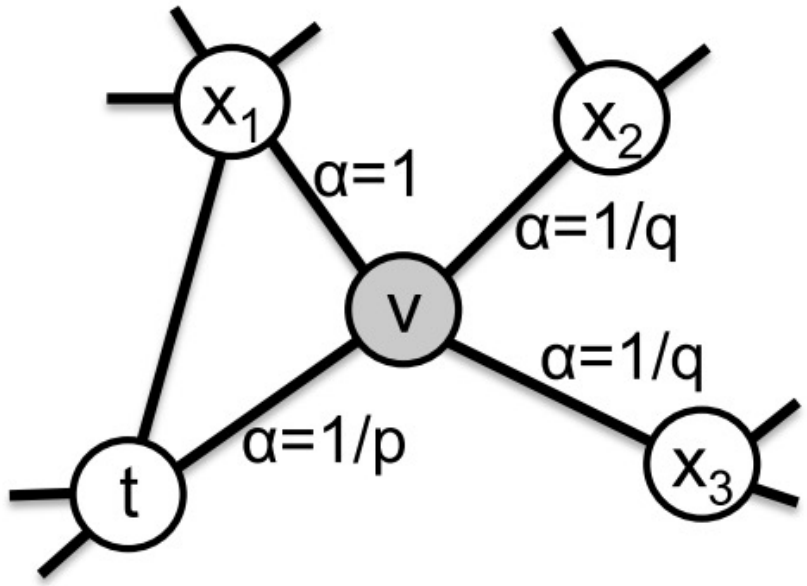
Node2vec

- Consider a random walk that just traversed edge (t, v) and now resides at node v.
- For the steps after the second step, we set

$$\pi_{vx} = \alpha_{pq}(t, x) w_{vx}$$

where

$$\alpha_{pq}(t,x) = \begin{cases} \frac{1}{p} & \text{if } d_{tx} = 0 \\ 1 & \text{if } d_{tx} = 1 \\ \frac{1}{q} & \text{if } d_{tx} = 2 \end{cases}$$



and $d_{tx} \in \{0,1,2\}$ denotes the shortest path distance between nodes t and x.

Image source: Grover, Aditya, and Jure Leskovec. "node2vec: Scalable feature learning for networks." In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 855-864. 2016.

Node2vec

- p and q are hyperparameters, to control how we move from the second step.
- Return parameter p:
  - Return back to the previous node.
- In-out parameter q:
  - Moving outwards (DFS) vs. inwards (BFS)
  - Intuitively, q is the "ratio" of BFS vs. DFS

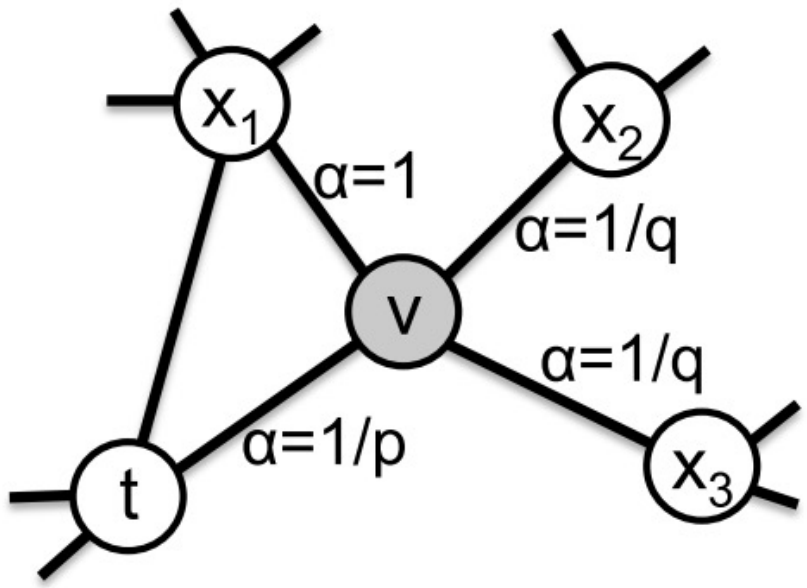


Image source: Grover, Aditya, and Jure Leskovec. "node2vec: Scalable feature learning for networks." In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 855-864. 2016.

Node2vec

Cases:

- p large, q large: non-returnable BFS.
- p large, q small: non-returnable DFS.
- p small, q large: returnable BFS.
- p small, q small: returnable DFS.
- $p = q = 1$: random walk.

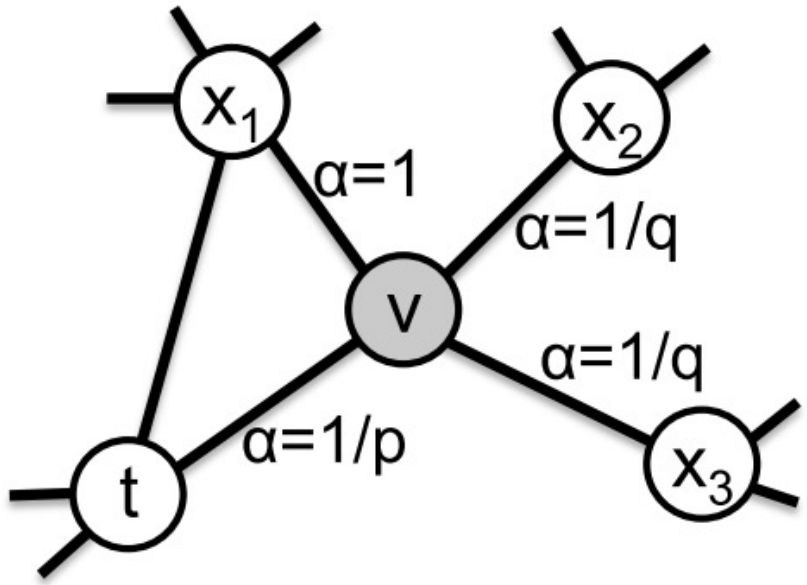
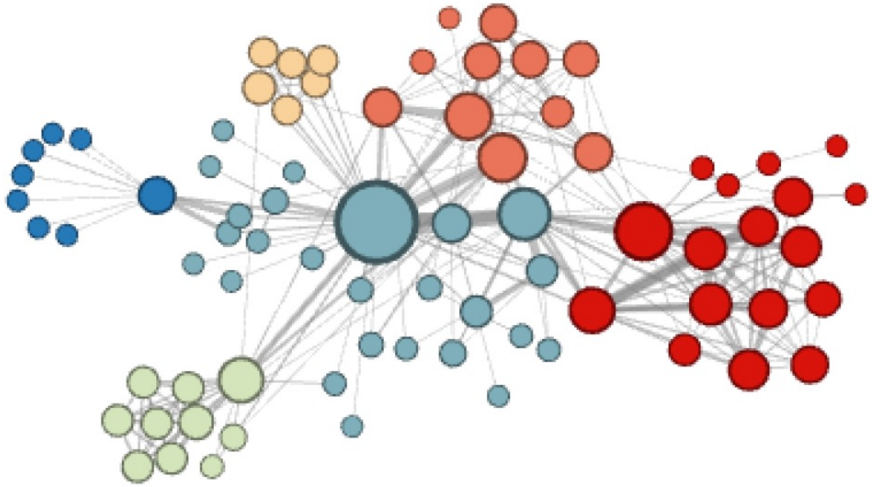


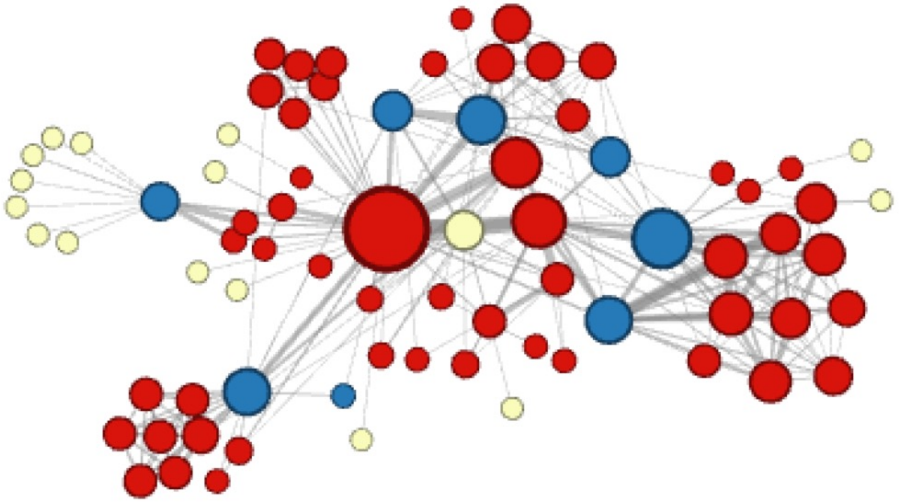
Image source: Grover, Aditya, and Jure Leskovec. "node2vec: Scalable feature learning for networks." In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 855-864. 2016.

Node2vec



DFS

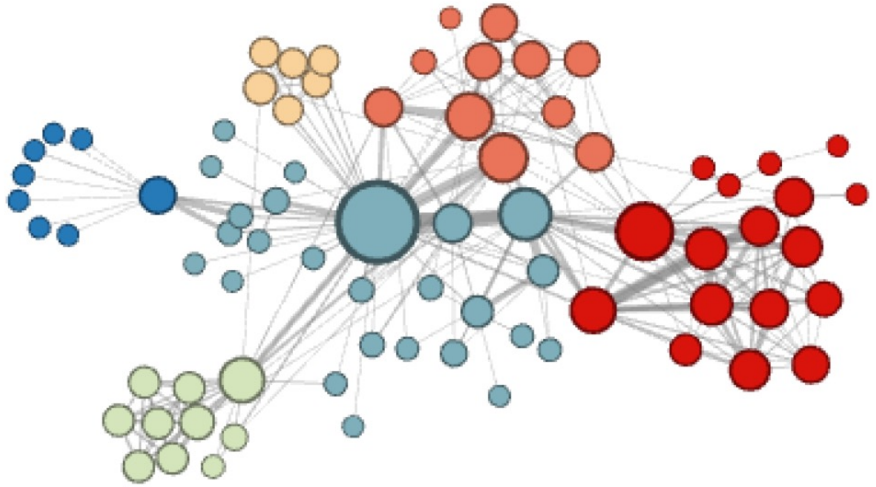
$p = 1, q = 0.5$



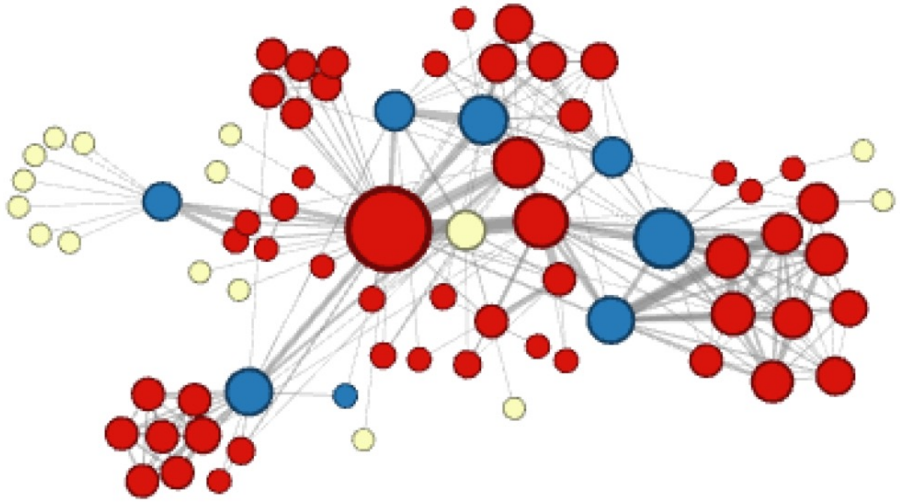
BFS

$p = 1, q = 2$

Node2vec



- DFS travels to the world, therefore know the difference.



- BFS only sees the neighborhood, therefore only know the difference between itself and its neighborhood.

How to Use Node Embeddings

After we obtain the embedding $\boldsymbol{z}_i$ for node i, how to use?

- Clustering/community detection: Clustering on nodes $\boldsymbol{z}_i$.
- Node classification: Predict label $f(\boldsymbol{z}_i)$ of node i based on $\boldsymbol{z}_i$.
- Link prediction: Predict edge (i, j) based on $f(\boldsymbol{z}_i, \boldsymbol{z}_j)$ by concatenate, avg, product, or take a difference between the embeddings:
  - Concatenate: $f(\boldsymbol{z}_i, \boldsymbol{z}_j) = g([\boldsymbol{z}_i, \boldsymbol{z}_j])$
  - Hadamard: $f(\boldsymbol{z}_i, \boldsymbol{z}_j) = g(\boldsymbol{z}_i \otimes \boldsymbol{z}_j)$
  - Sum/Avg: $f(\boldsymbol{z}_i, \boldsymbol{z}_j) = g(\boldsymbol{z}_i + \boldsymbol{z}_j)$
  - Distance: $f(\boldsymbol{z}_i, \boldsymbol{z}_j) = g(\|\boldsymbol{z}_i - \boldsymbol{z}_j\|_2)$

Shallow Encoders

Shallow encoders:

- One-layer of data transformation.
- A single hidden layer maps node u to embedding $\boldsymbol{z}_u$ by

$$\boldsymbol{z}_u = f\left(\boldsymbol{z}_v, v \in N_R(u)\right).$$

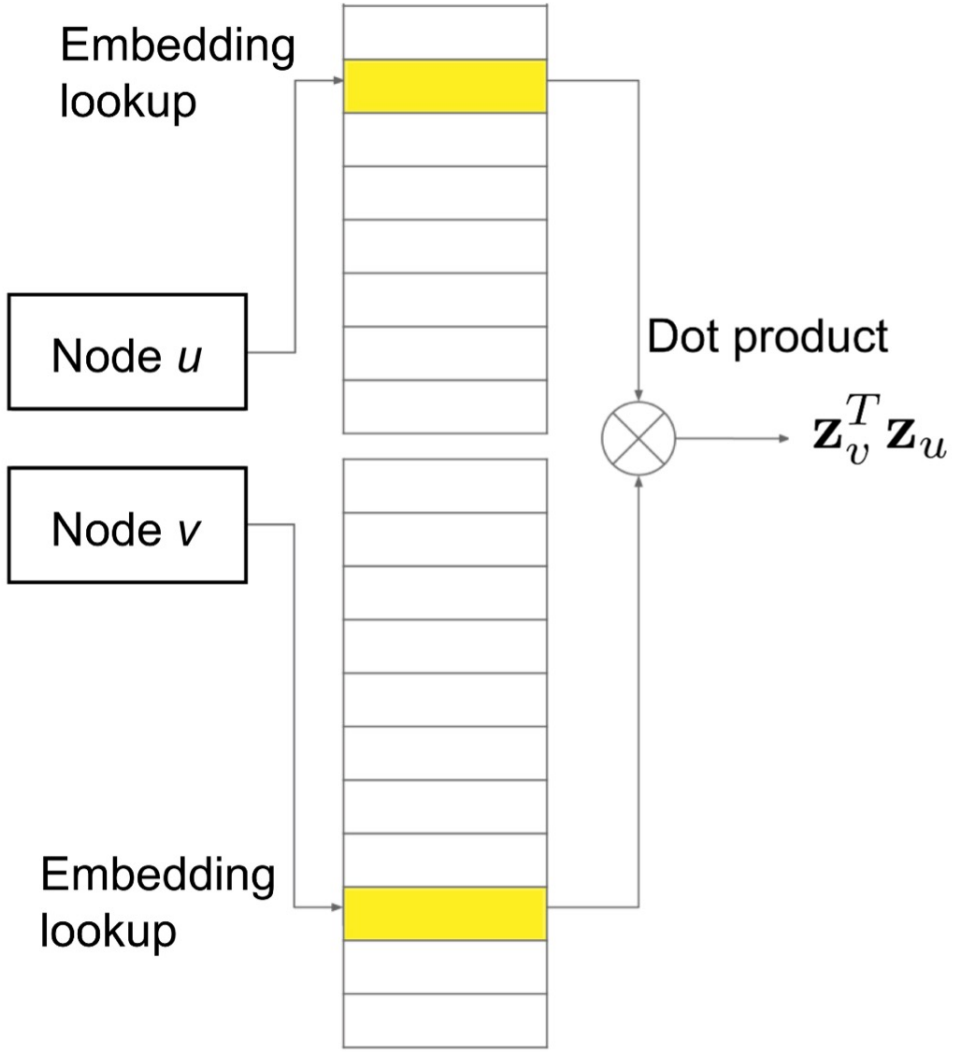


Image source: Lecture 8, cs224w, Stanford University

Shallow Encoders

Limitations of shallow embedding methods:

- No parameter sharing:
  - Every node has its own unique embedding.
- Transductive, not inductive:
  - Cannot generate embeddings for nodes that are not seen during training.
- Do not incorporate node features:
  - Many graphs have features that we can and should leverage.
- Separated from downstream tasks.
  - Training is not end-to-end.

GRAPH NEURAL NETWORKS

Deep Graph Encoder

- Instead of directly learning embedding, can we learn mapping to generate embedding?

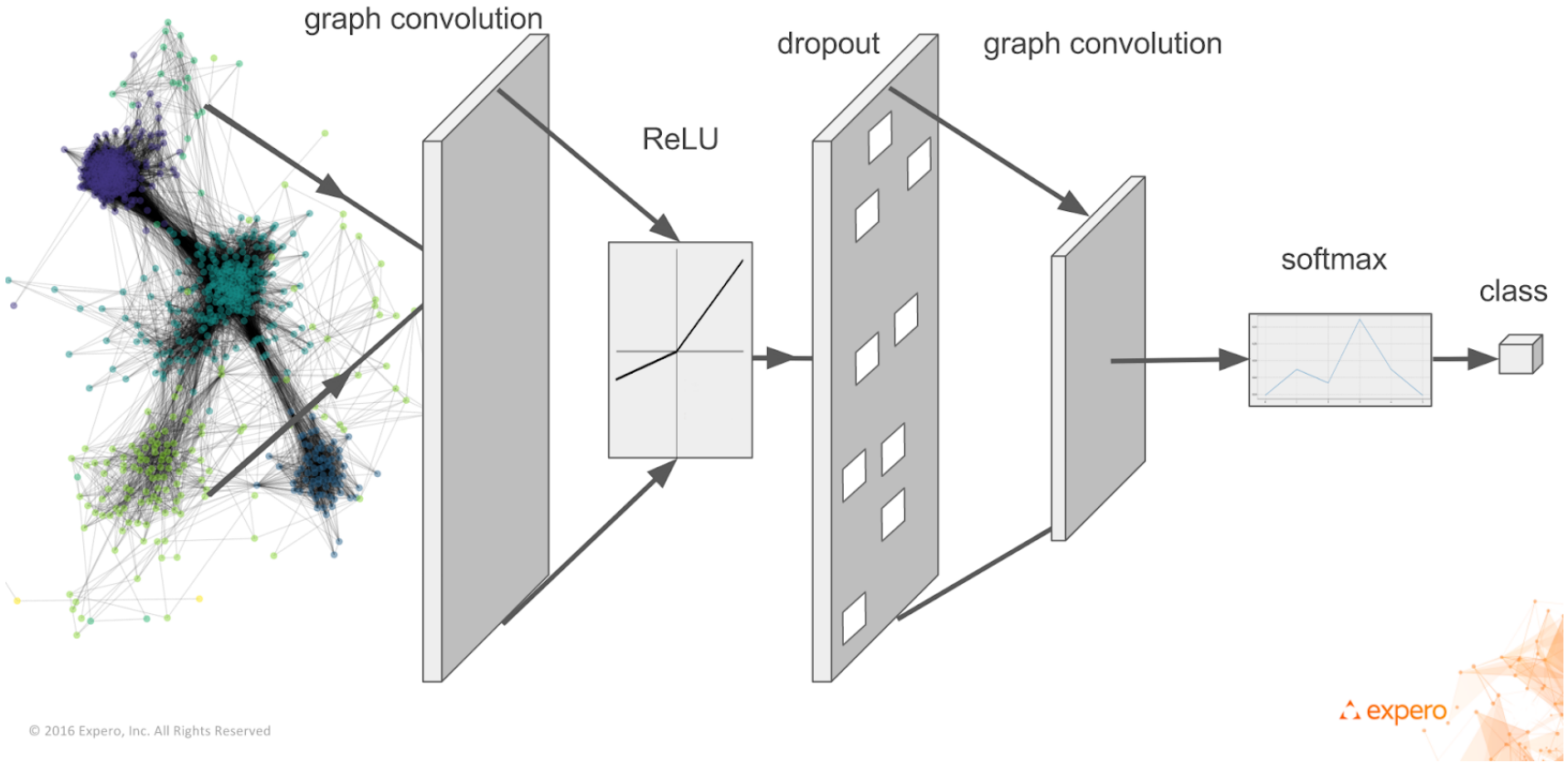


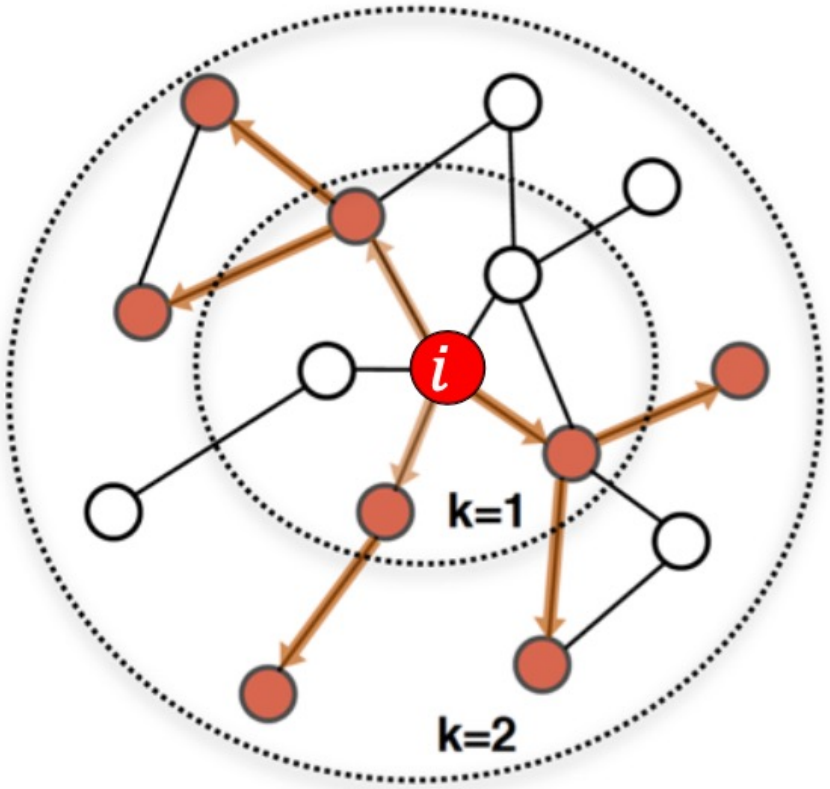
Image source: https://www.experoinc.com/post/node-classification-by-graph-convolutional-network

GCN

GCN

Semi-supervised classification with graph convolutional networks
TN Kipf, M Welling - arXiv preprint arXiv:1609.02907, 2016 - arxiv.org
We present a scalable approach for **semi-supervised** learning on graph-structured data that is based on an efficient variant of convolutional neural networks which operate directly on …
☆ Save 99 Cite Cited by 29001 Related articles All 23 versions »

- Idea: Node's neighborhood defines a computation graph.



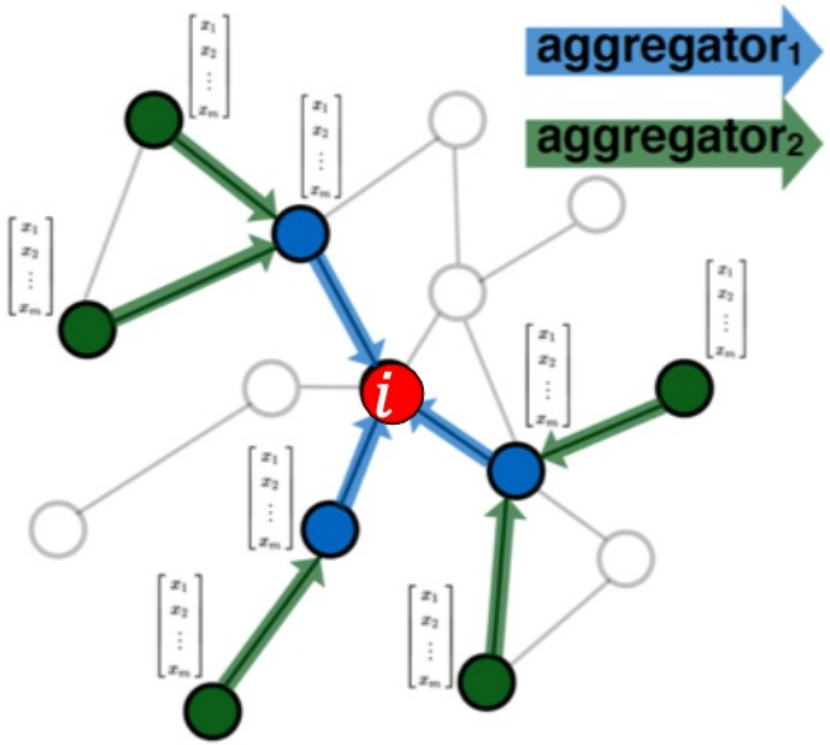


Image source: Hamilton, Will, Zhitao Ying, and Jure Leskovec. "Inductive representation learning on large graphs." In Advances in neural information processing systems, pp. 1024-1034. 2017.

GCN: Basic Setting

Assume we have a graph G:

- V is the vertex set.
- A is the adjacency matrix (assume binary).
- $X \in \mathbb{R}^{m \times |V|}$ is a matrix of node initial features.

- Node initial features:
  - Social networks: user profile, user image.
  - Biological networks: gene expression profiles, gene functional information.
  - No features: one-hot vector or constant vector.

GCN: Architecture

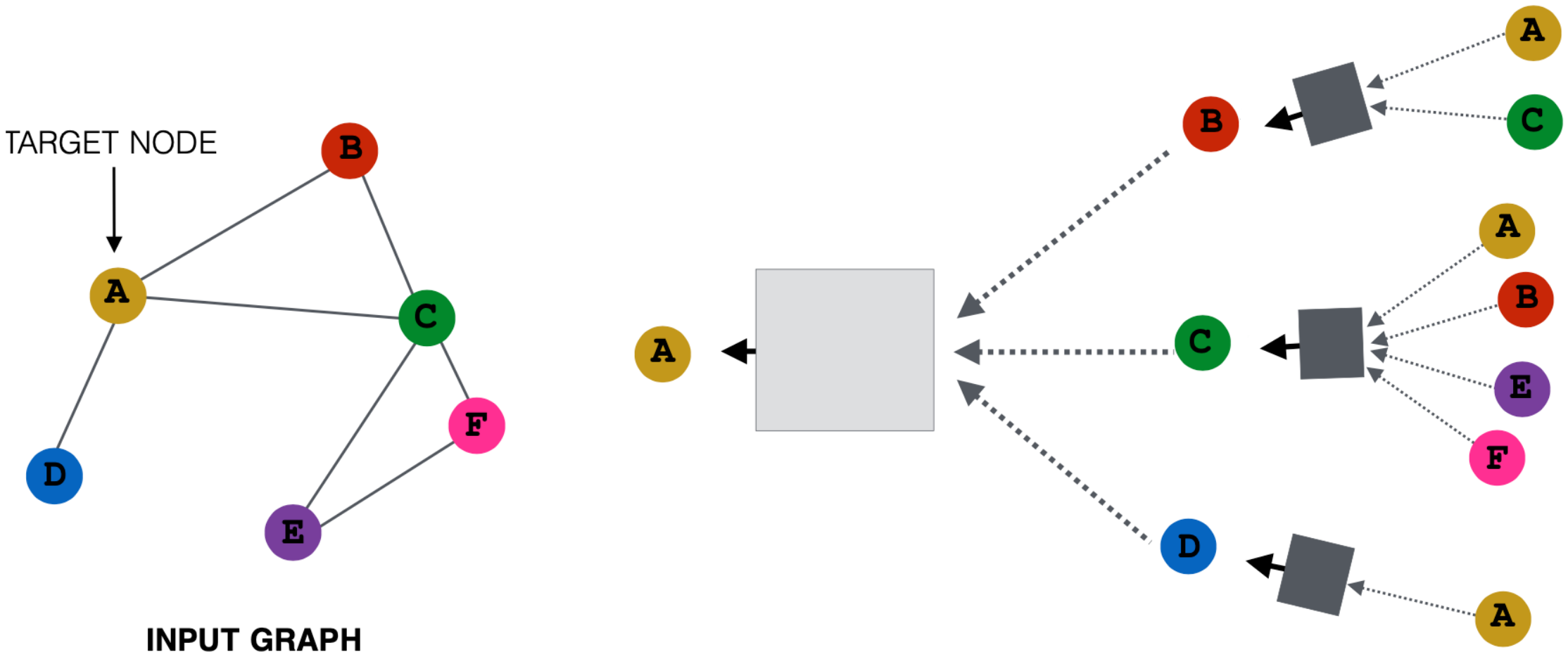
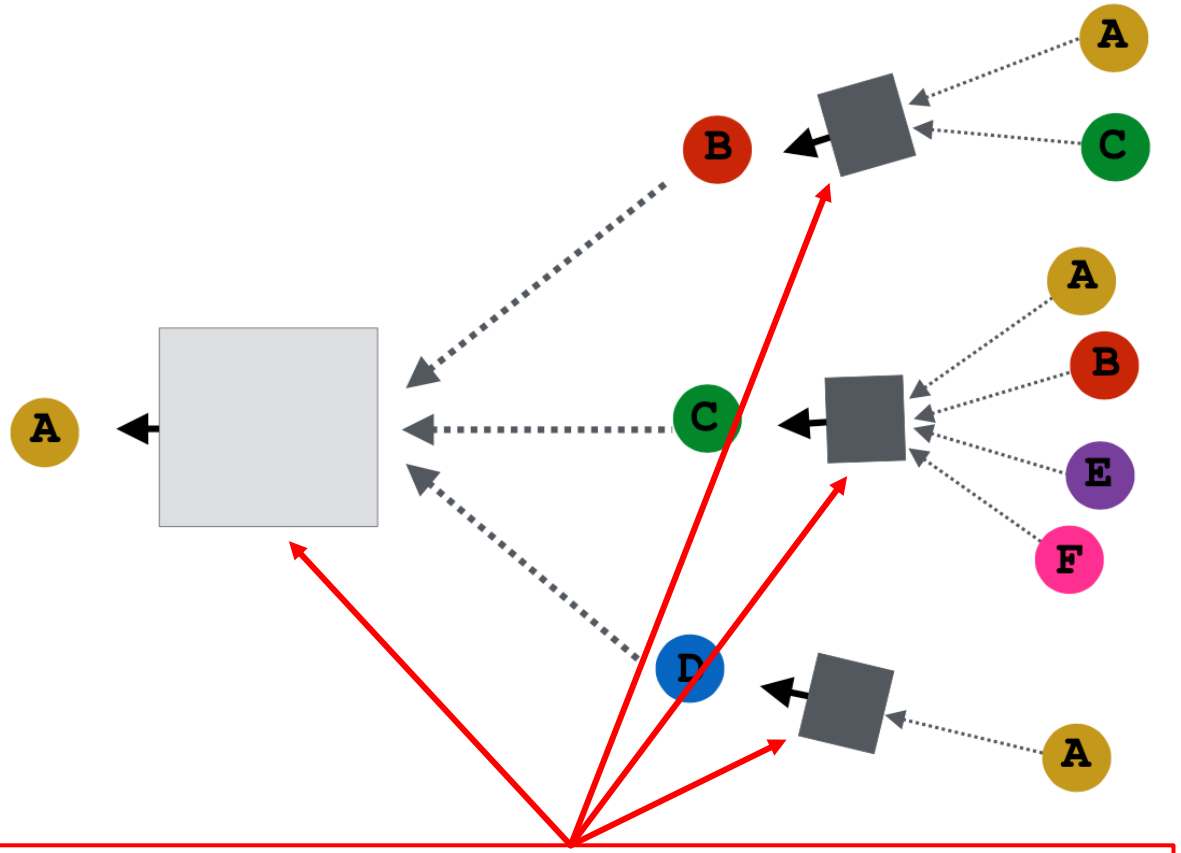


Image source: Ying, Rex, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. "Graph convolutional neural networks for web-scale recommender systems." In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 974-983. 2018.

GCN: Architecture



Average the information from the previous layer and apply neural network

Image source: Ying, Rex, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. "Graph convolutional neural networks for web-scale recommender systems." In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 974-983. 2018.

GCN: Multiple Layers

- Model can be of arbitrary depth:
  - Nodes have embeddings at each layer.
  - Layer-0 embedding of node u is its input feature, x_u.
  - Layer-K embedding gets information from nodes that are K hops away.

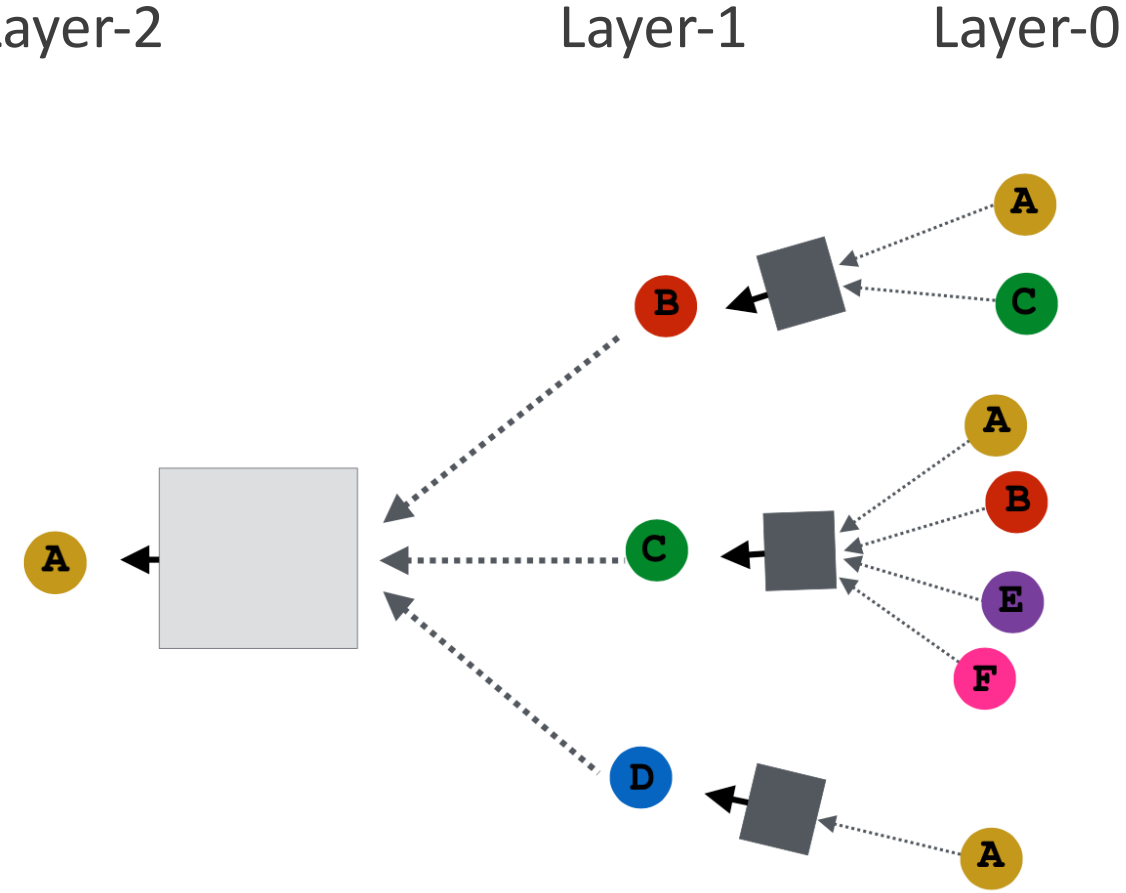


Image source: Ying, Rex, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. "Graph convolutional neural networks for web-scale recommender systems." In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 974-983. 2018.

GCN: Parameter Sharing

- Every node defines a computation graph based on its neighborhood!

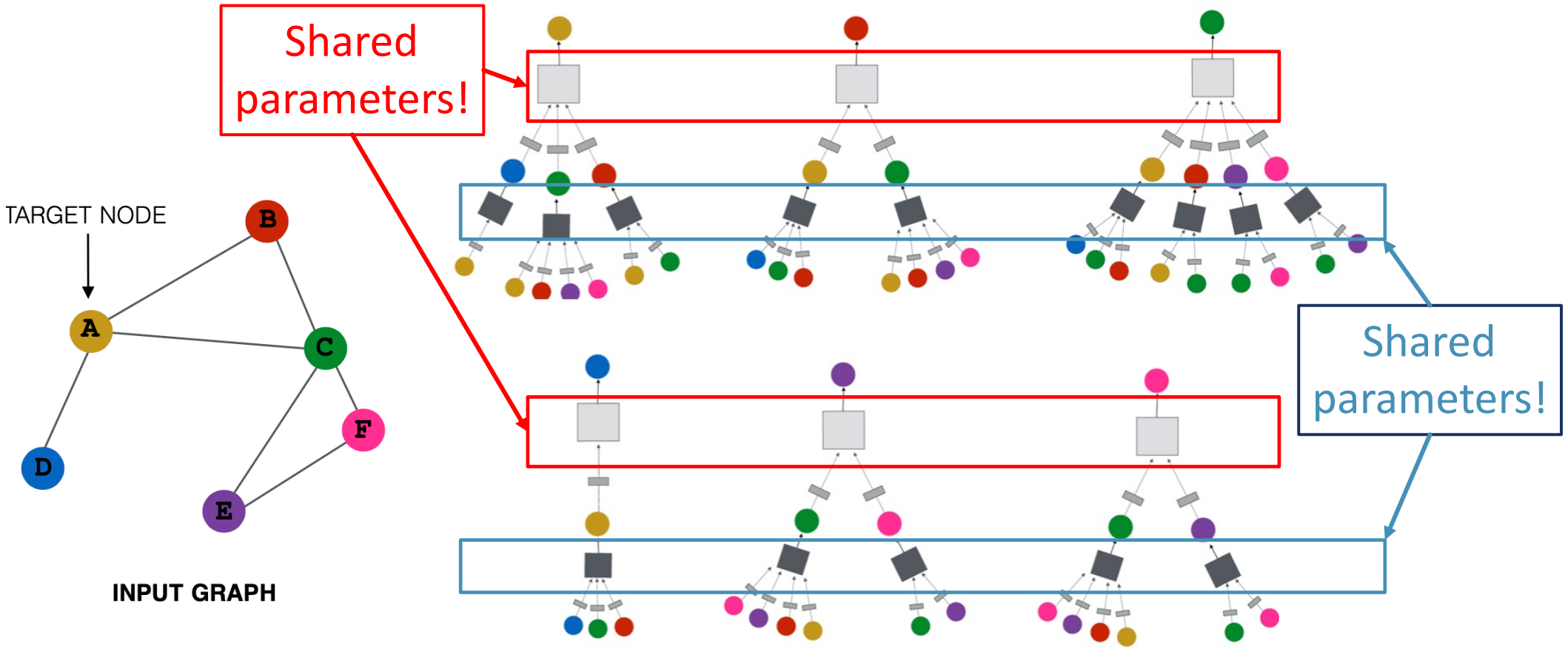


Image source: Ying, Rex, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. "Graph convolutional neural networks for web-scale recommender systems." In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 974-983. 2018.

GCN: Deep Encoder

- For each node v, its embedding at Layer-k is $\boldsymbol{h}_v^k$:

$$\boldsymbol{h}_v^0 = \boldsymbol{x}_v$$

$$\boldsymbol{h}_v^k = \sigma\left(\boldsymbol{W}_k \sum_{u \in N(v)} \frac{\boldsymbol{h}_u^{k-1}}{|N(v)|} + \boldsymbol{B}_k \boldsymbol{h}_v^{k-1}\right), k = 1, \ldots, K$$

$$\boldsymbol{z}_v = \boldsymbol{h}_v^K$$

  - $\boldsymbol{W}_k$ is the parameter at Layer-k for the averaged neighborhood of node v;
  - $\boldsymbol{B}_k$ is the parameter at Layer-k for node v itself.

GCN: Deep Encoder

- In original GCN paper, the neural network is represented by (sparse) matrix operations.

$$\boldsymbol{h}_v^k = \sigma\left(\boldsymbol{W}_k \sum_{u \in N(v)} \frac{\boldsymbol{h}_u^{k-1}}{|N(v)|} + \boldsymbol{B}_k \boldsymbol{h}_v^{k-1}\right)$$

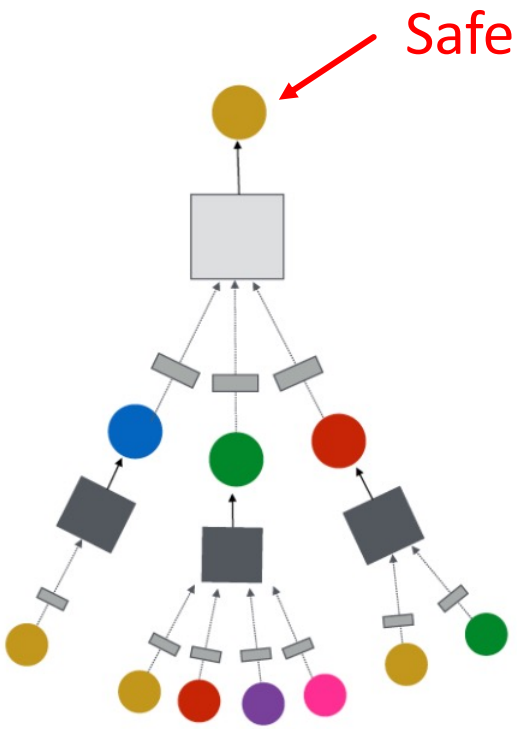
can be formulated as

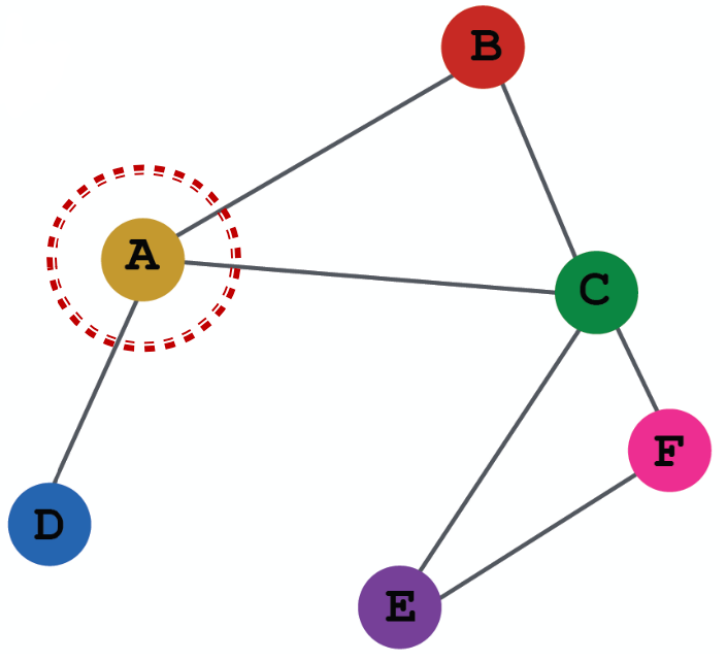
$$\boldsymbol{H}^k = \sigma(\widetilde{\boldsymbol{D}}^{-\frac{1}{2}} \widetilde{\boldsymbol{A}} \widetilde{\boldsymbol{D}}^{-\frac{1}{2}} \boldsymbol{H}^{k-1} \boldsymbol{W}_k)$$

where $\widetilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}_N$ is the adjacency matrix with added self-connections, $\widetilde{\boldsymbol{D}}_{ii} = \sum_j \widetilde{\boldsymbol{A}}_{ij}$ is the degree matrix.

GCN: Training

- Train in a supervised manner: Directly train the end-to-end model for a supervised task (e.g., node classification).





drug-drug interaction network

Image source: Ying, Rex, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. "Graph convolutional neural networks for web-scale recommender systems." In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 974-983. 2018.

GCN: Training

- Train in an unsupervised manner:
  - Use only the graph structure.
  - “Similar” nodes have similar embedding.
- How to find similar nodes?
  - Deepwalk, node2vec, …

GRAPHSAGE

GraphSAGE

Inductive representation learning on large graphs
W Hamilton, Z Ying, J Leskovec - Advances in neural …, 2017 - proceedings.neurips.cc
… Here we present GraphSAGE, a general, **inductive** … Instead of training individual embeddings for each node, we **learn** a … Our algorithm outperforms strong baselines on three **inductive** …
☆ Save 99 Cite Cited by 12636 Related articles All 22 versions »

- So far we have aggregated the neighbor messages by taking their (weighted) average.

$$\boldsymbol{h}_v^k = \sigma\left(\boldsymbol{W}_k \sum_{u \in N(v)} \frac{\boldsymbol{h}_u^{k-1}}{|N(v)|} + \boldsymbol{B}_k \boldsymbol{h}_v^{k-1}\right).$$

  It is very straightforward and simple.

- Can we make it more sophisticated to learn more latent information from a graph?

GraphSAGE

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

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

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

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

Generalizes the aggregation function

Replay sum by concat

Image source: Hamilton, Will, Zhitao Ying, and Jure Leskovec. "Inductive representation learning on large graphs." In Advances in neural information processing systems, pp. 1024-1034. 2017.

GraphSAGE

Mean aggregator

$$\text{AGGREGATE}_k = \sum_{u \in N(v)} \frac{\boldsymbol{h}_u^{k-1}}{|N(v)|}$$

- Nearly equivalent to the convolutional propagation rule used in GCN.
- This concatenation can be viewed as a simple form of a “skip connection” between the different layers.

GraphSAGE

LSTM aggregator

$$\text{AGGREGATE}_k = \text{LSTM}\left(\left[\boldsymbol{h}_u^{k-1}, \forall u \in \pi\left(N(v)\right)\right]\right)$$

- LSTMs have the advantage of larger expressive capability.
- Apply LSTM to random permutation of the node's neighbors $\pi\left(N(v)\right)$.

GraphSAGE

Pooling aggregator

$$\text{AGGREGATE}_k = \max(\{\sigma(\boldsymbol{W}_{pool}\boldsymbol{h}_{u_i}^k + \boldsymbol{b}), \forall u_i \in N(v)\})$$

where max is taken element-wise.

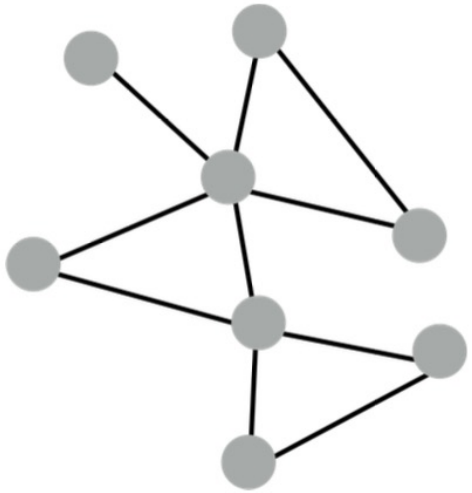
- By applying the max-pooling operator to each of the computed features, the model effectively captures different aspects of the neighborhood set.

Transductive vs. Inductive

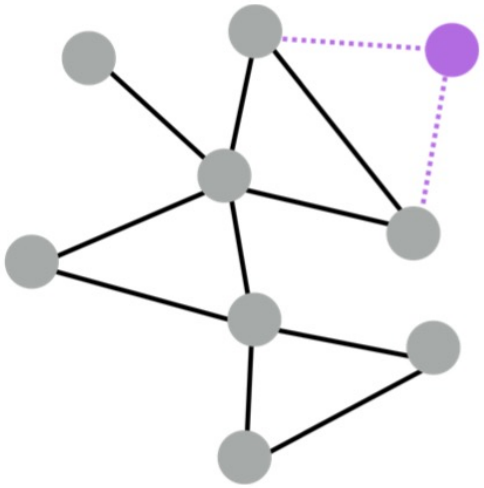
- GNN is usually in a semi-supervised learning manner.
  - The unlabelled node is involved during training.
- Semi-supervised learning can be grouped into two categories:
  - Transducrtive: The testing data is from the unlablled data.
  - Inductive: The testing data is unseen in training.

Inductive Capacity for New Nodes

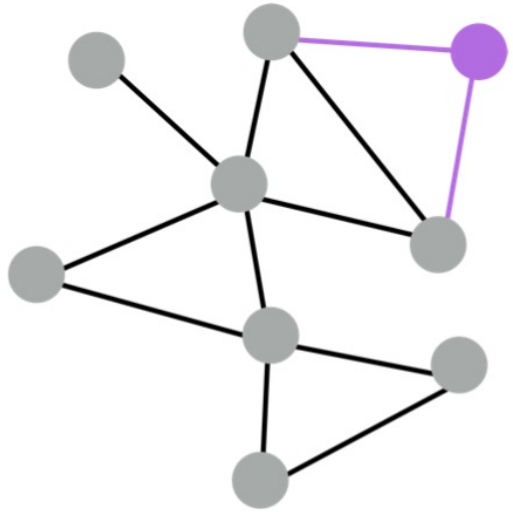
- Many application settings constantly encounter previously unseen nodes.
  - E.g. new user and new item in a recommendation system.



Train on known graph



New node arrives

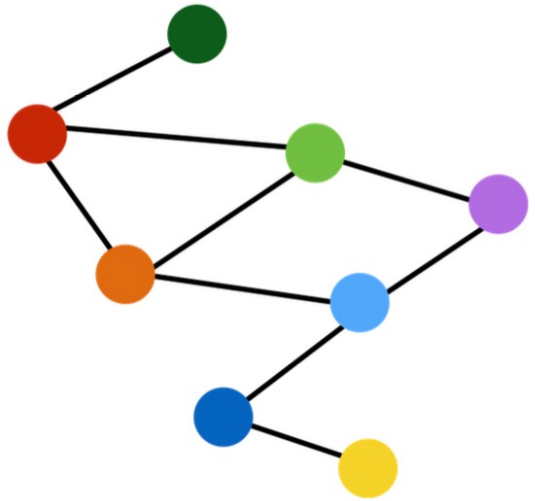


Generate embedding for new node

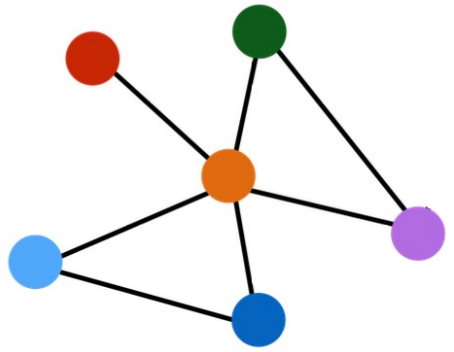
Image source: Lecture 8, cs224w, Stanford University

Inductive Capacity for New Graphs

- The trained GCN parameters can also be used to generalize to entirely unseen graphs.
  - E.g. train on protein interaction graph from model organism A and generate embeddings on newly collected data about organism B.



Train on one graph



Generalize to new graph

Image source: Lecture 8, cs224w, Stanford University

GAT

GAT

Graph attention networks
P Veličković, G Cucurull, A Casanova… - arXiv preprint arXiv …, 2017 - arxiv.org
… , we introduce an **attention**-based architecture to perform node classification of **graph**-structured data. The idea is to compute the hidden representations of each node in the **graph**, by …
☆ Save 99 Cite Cited by 8284 Related articles All 11 versions »

- Check the neighborhood aggregation of GCN again:

$$\boldsymbol{h}_v^k = \sigma\left(\boldsymbol{W}_k \sum_{u \in N(v)} \frac{\boldsymbol{h}_u^{k-1}}{|N(v)|} + \boldsymbol{B}_k \boldsymbol{h}_v^{k-1}\right).$$

- What is the weight of each neighbor $u \in N(v)$ that contributes to node v?

$$\frac{1}{|N(v)|}$$

- It simply assumes that all neighbors are equally important to node v.

GAT

- Can we simply learn a weight for each node in the graph?
  - Important node (e.g. with large degree) deserves large weight.
- Probably not.
- The importance of each node to each neighbor should be different.
- Goal: Specify arbitrary importance to different neighbors of each node in the graph.
- Idea: Compute embedding $\boldsymbol{h}_v^k$ of each node in the graph following an attention network.

GAT

- First compute attention coefficients of e_{vu} across node v, and its neighbor u based on their representation at layer $k-1$:

$$e_{vu} = a(\boldsymbol{W}_k \boldsymbol{h}_u^{k-1}, \boldsymbol{W}_k \boldsymbol{h}_v^{k-1})$$

  - e_{vu} indicates the importance of node u message to node v.
- The attention network a can just be a simple single-layer neural network:

$$a(\boldsymbol{p}, \boldsymbol{q}) = \boldsymbol{A}^T [\boldsymbol{p}, \boldsymbol{q}]$$

where $\boldsymbol{A}$ is a learnable parameter.

GAT

- Then normalize over all neighbors to get the weight α_{vu}:

$$\alpha_{vu} = \frac{\exp e_{vu}}{\sum_{k \in N(v)} \exp e_{vk}}$$

- The final attention-weighted aggregation is:

$$\boldsymbol{h}_v^k = \sigma\left(\sum_{u \in N(v)} \alpha_{vu} \boldsymbol{W}_k \boldsymbol{h}_u^{k-1} \right)$$

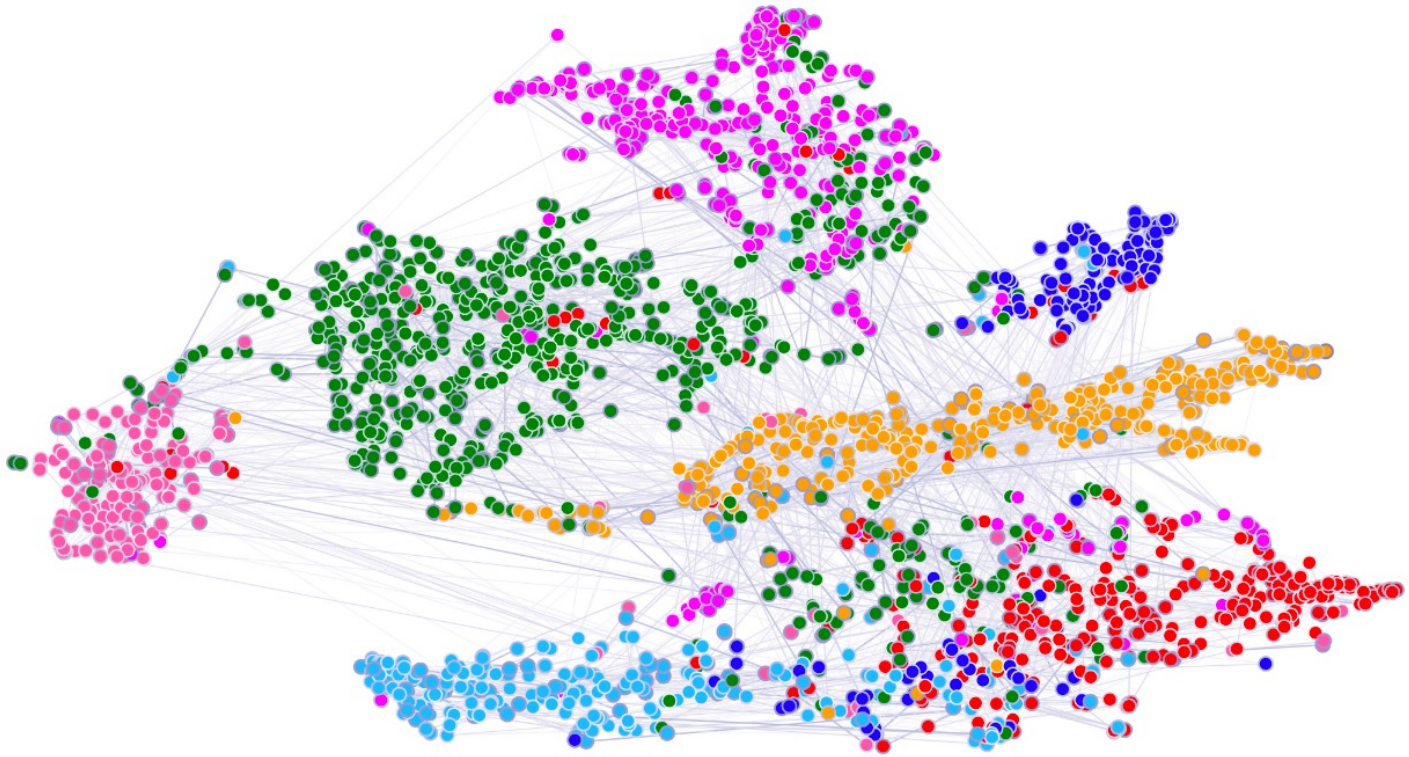
GAT: Multi-Head Attention

- Borrow the idea of multi-head attention from Transformer:

$$\boldsymbol{h}_v^k = \sigma\left(\sum_{t=1}^{T}\sum_{u\in N(v)} \alpha_{vu}^{(t)} \boldsymbol{W}_k^{(t)} \boldsymbol{h}_u^{k-1}\right)$$

  - We got T head and each head t has its own weights.

GAT



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 attention coefficient.

Image source: Veličković, Petar, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. "Graph attention networks." arXiv preprint arXiv:1710.10903 (2017).

APPLICATION TO RECOMMENDER SYSTEM

Pinterest

- Pinterest is an American image sharing and social media service.
- Users can save and discover images, GIFs and videos in the form of pinboards.
- 300M users, 4+B pins, 2+B pinboards.

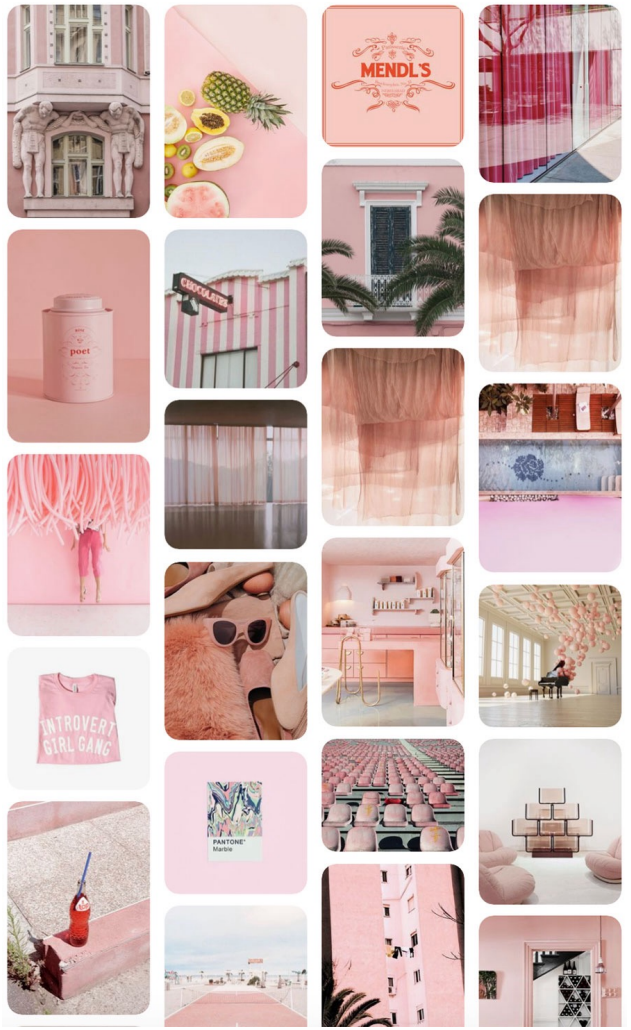
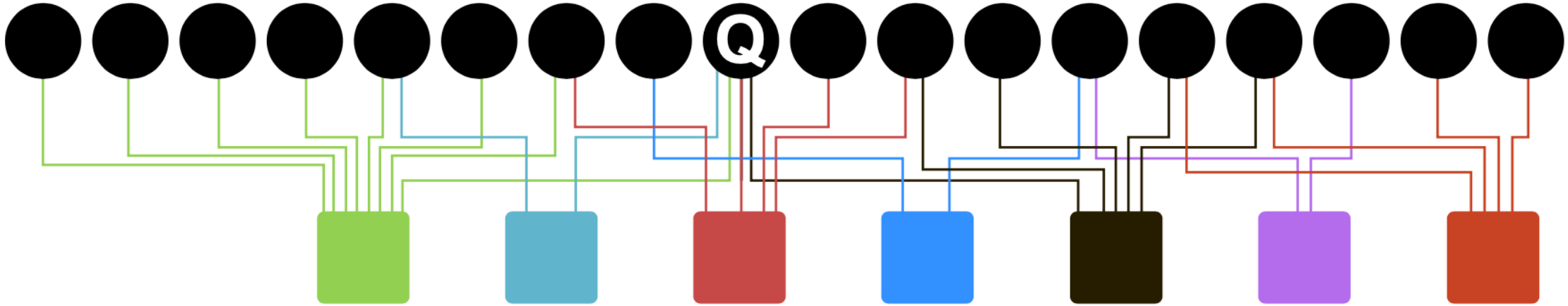


Image source: https://en.wikipedia.org/wiki/Pinterest
https://www.sfgirlbybay.com/2019/11/20/pinterest-practices-for-pros/

Pinterest Graph



- Graph: 2B pins, 1B boards, 20B edges.
  - Graph is dynamic: Need to apply to new pins and new boards without model retraining.
- Rich node features: images, text with pins.

Image source: Lecture 8, cs224w, Stanford University

PinSage

Graph convolutional neural networks for web-scale recommender systems
R Ying, R He, K Chen, P Eksombatchai… - Proceedings of the 24th …, 2018 - dl.acm.org
… Recent advancements in deep **neural networks** for **graph**-structured data … **Graph Convolutional Network** (GCN) algorithm PinSage, which combines efficient random walks and **graph** …
☆ Save 99 Cite Cited by 3121 Related articles All 8 versions

- Goal: Generate pin embeddings in a web-scale Pinterest graph containing billions of objects.
- Pin embeddings are essential to various tasks like pin recommendation, classification, clustering, ranking.
  - Services like “Related Pins”, “Search”, “Shopping”, “Ads”.

PinSage: Result

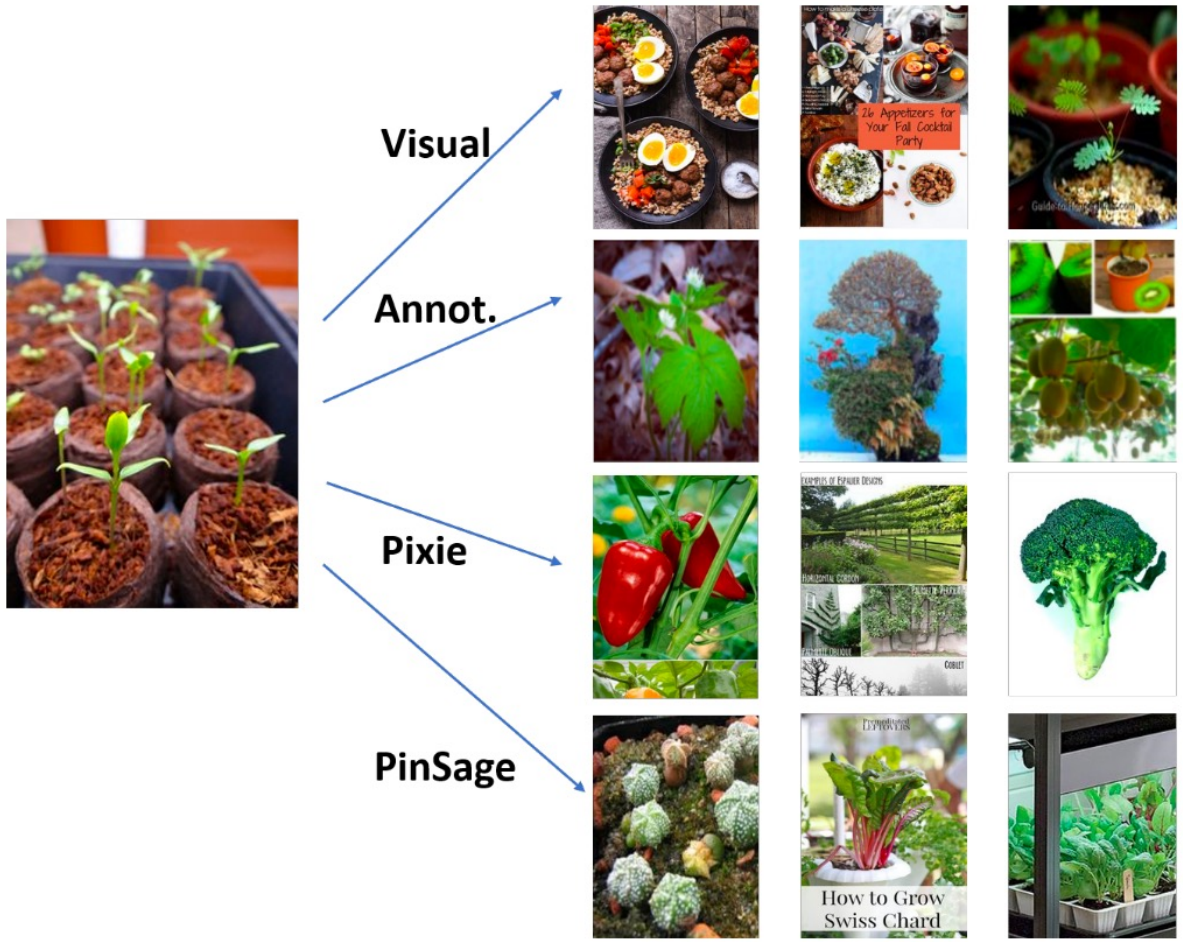


Image source: Ying, Rex, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. "Graph convolutional neural networks for web-scale recommender systems." In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 974-983. 2018.

PinSage: Result

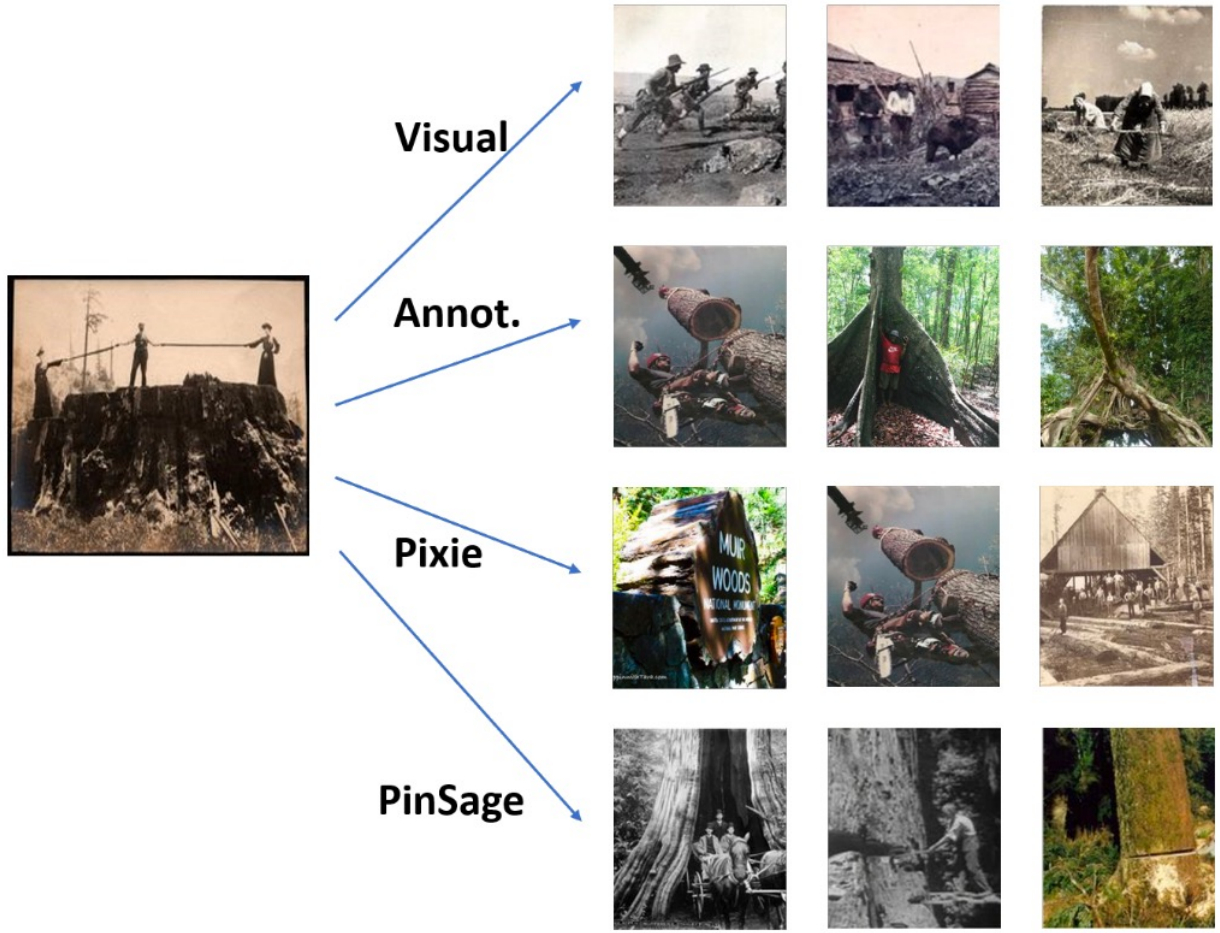


Image source: Ying, Rex, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. "Graph convolutional neural networks for web-scale recommender systems." In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 974-983. 2018.

RECENT ADVANCES

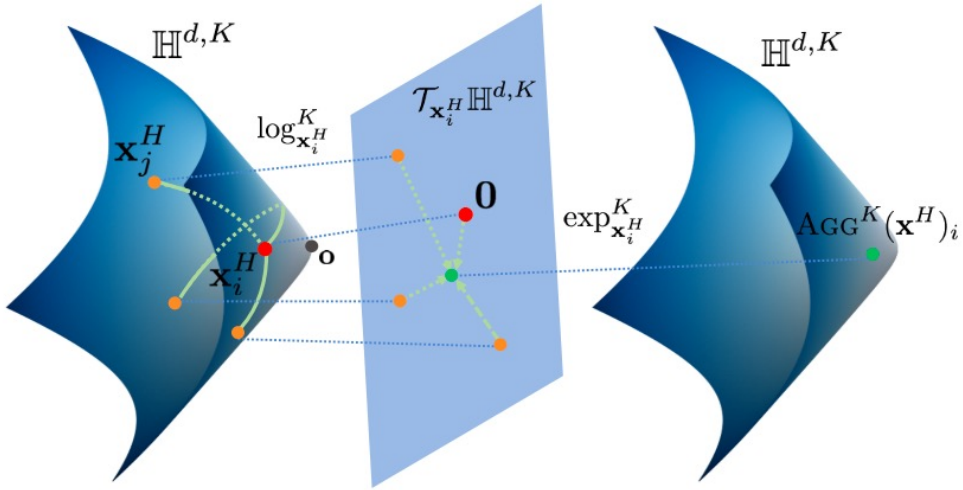
HGCN

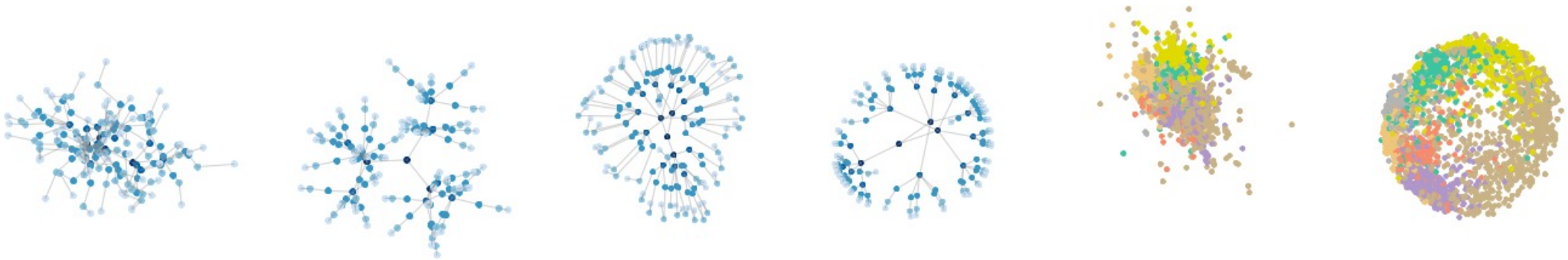
Hyperbolic graph convolutional neural networks

I Chami, Z Ying, C Ré… - Advances in neural …, 2019 - proceedings.neurips.cc

… and scale-free **graphs** in inductive settings: (1) We … **hyperbolic** space to transform input features which lie in Euclidean space into **hyperbolic** embeddings; (2) We introduce a **hyperbolic** …

☆ Save 99 Cite Cited by 541 Related articles All 17 versions »





(a) GCN layers. (b) HGCN layers. (c) GCN (left), HGCN (right).

Image source: Chami, Ines, Zhitao Ying, Christopher Ré, and Jure Leskovec. "Hyperbolic graph convolutional neural networks." In Advances in neural information processing systems, pp. 4868-4879. 2019.

GTN

Graph transformer networks

S Yun, M Jeong, R Kim, J Kang… - Advances in neural …, 2019 - proceedings.neurips.cc

… **Graph Transformer Network** (GTN) that learns to transform a heterogeneous input **graph** into useful meta-path **graphs** for each task and learn node representation on the **graphs** in an …

☆ Save 99 Cite Cited by 733 Related articles All 11 versions »

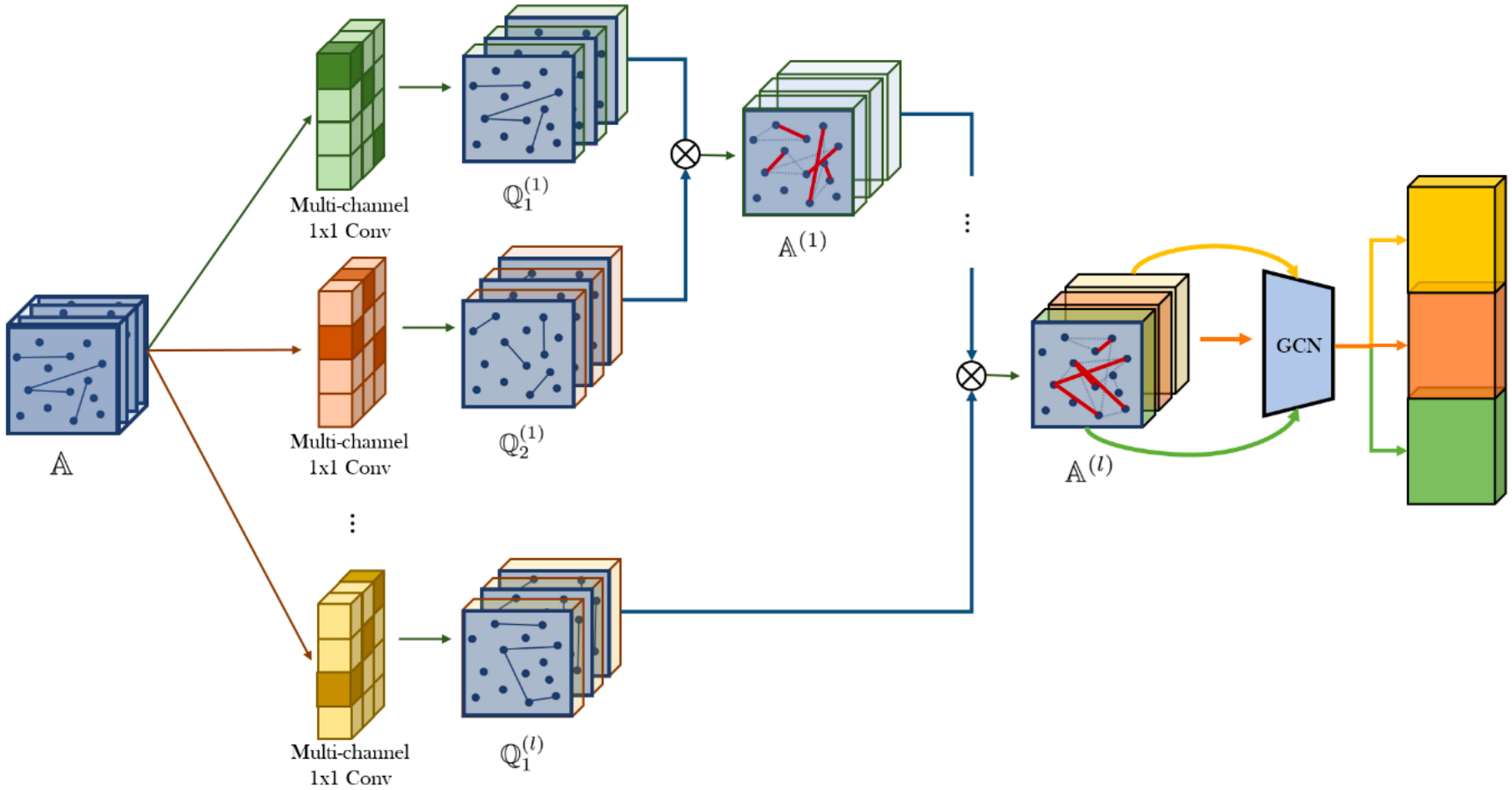


Image source: Yun, Seongjun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J. Kim. "Graph transformer networks." In Advances in Neural Information Processing Systems, pp. 11983-11993. 2019.

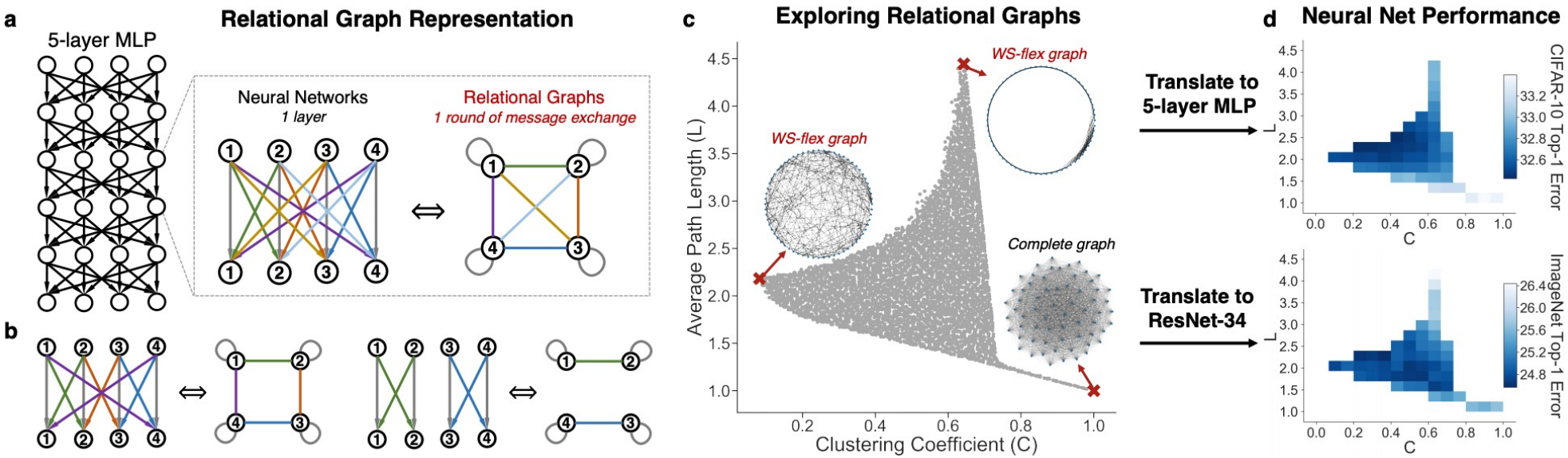
Relational Graph

Graph structure of neural networks

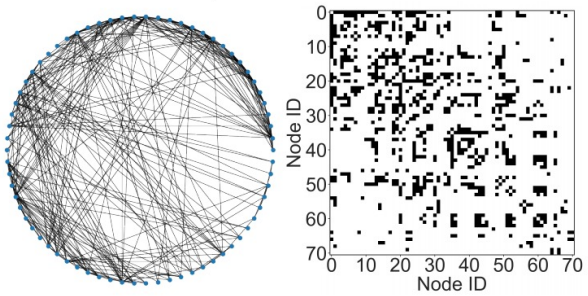
J You, J Leskovec, K He, S Xie - International Conference on …, 2020 - proceedings.mlr.press

… **graphs**. Here we systematically study the relationship between the **graph structure** of a **neural network** … of representing a **neural network** as a **graph**, which we call relational **graph**. Our …

☆ Save Cite Cited by 132 Related articles All 11 versions »



Biological neural network:
Macaque whole cortex



Artificial neural network:
Best 5-layer MLP

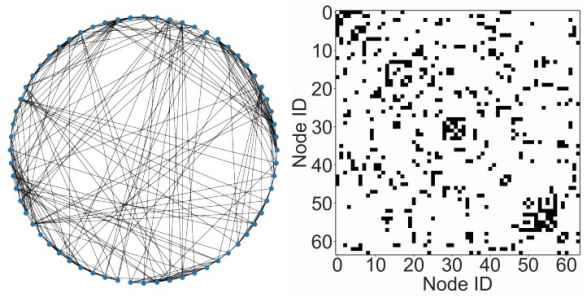


Image source: You, Jiaxuan, Jure Leskovec, Kaiming He, and Saining Xie. "Graph structure of neural networks." In International Conference on Machine Learning, pp. 10881-10891. PMLR, 2020.

Conclusion

After this lecture, you should know:

- What is a graph representation?
- How does random walk help generate graph representation?
- What kind of role do BFS and DFS play in node2vec?
- What is the basic architecture of GNN?
- How is attention applied to GNN?

Suggested Reading

- 深度学习中不得不学的Graph Embedding方法
- 关于Node2vec算法中Graph Embedding同质性和结构性的进一步探讨

Reference

- Tutorial at WWW 2019 on Representation Learning on Networks
- CS224W Machine Learning with Graphs

Thank you!

- Any question?
- Don’t hesitate to send email to me for asking questions and discussion. ☺