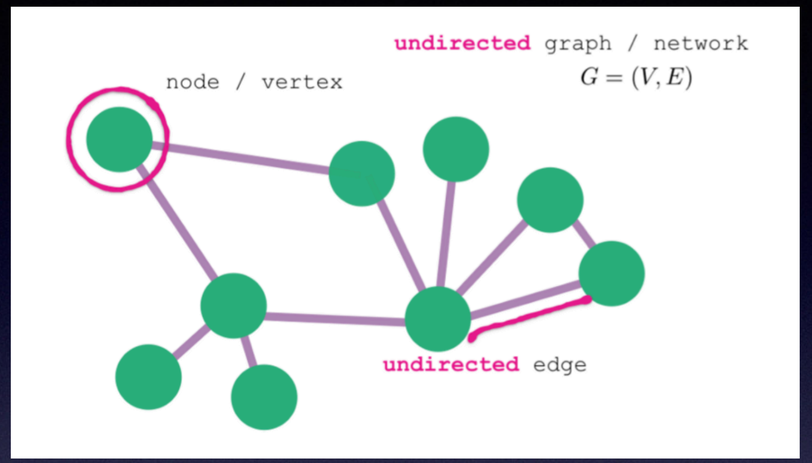
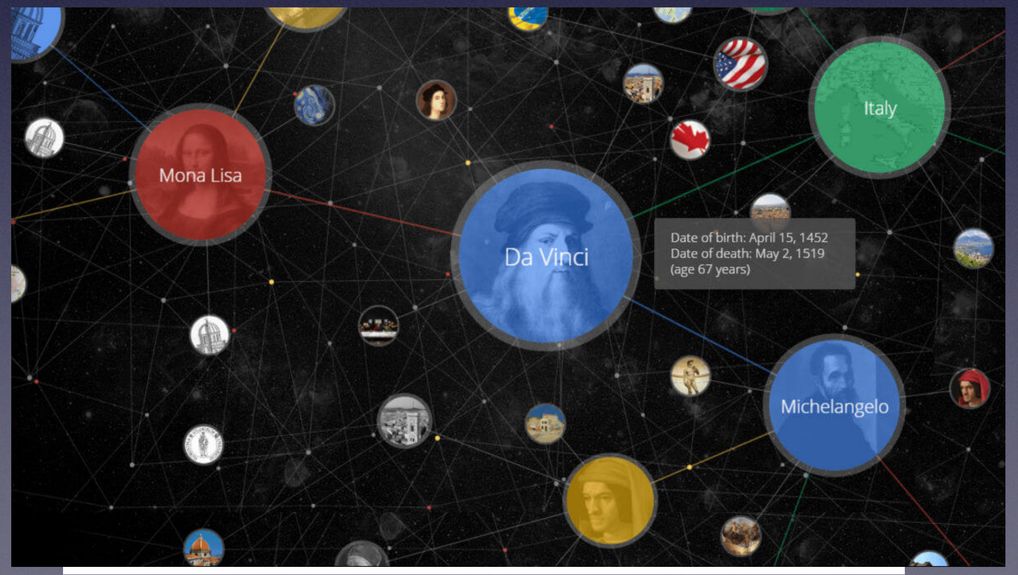
Graph Neural Network and its applications

Juncheng Dong

Definition of Graph/Network

- A graph contains nodes and edges. Nodes are connected by edges.
- Edges can have directions and types.
- Network data are very common in real life (e.g social network, traffic network, molecular network, knowledge graph etc.)
- Graph neural network, as a research area, studies how to apply deep learning on graph data





Setup/Input to GNN

- Adjacency Matrix A. A(i, j) = 1 if node i, node j are connected.
- Feature Matrix X. $X \in R^{NxD}$ where N is total number of nodes and D is dimension of feature vector.
- How features are collected is design choice. Features can be different in different networks. In social network, features can be user profile. In molecule network, features can be information of atoms.

Graph-related tasks

1. Node-level tasks

- Node classification/regression
- Link prediction
- Node embedding*

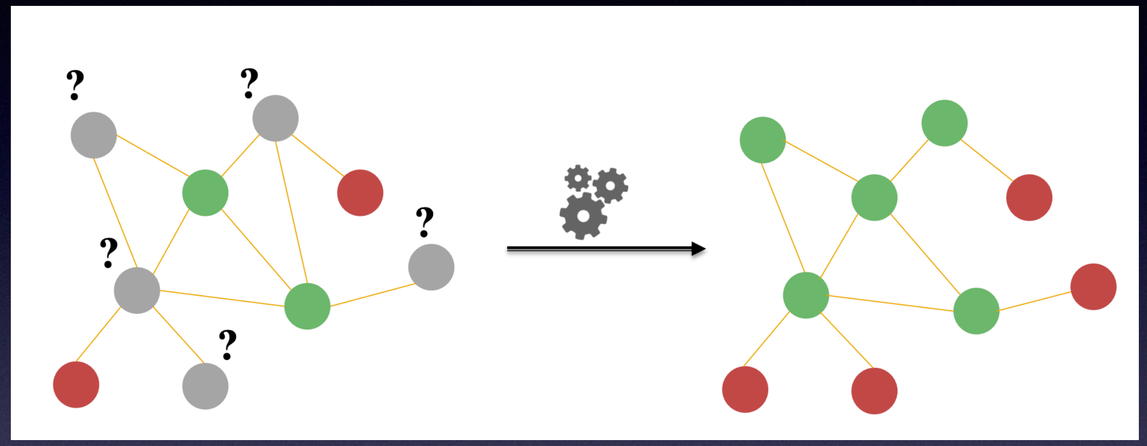
2. Graph-level tasks

- Graph classification/regression
- Graph generation
- Graph embedding*

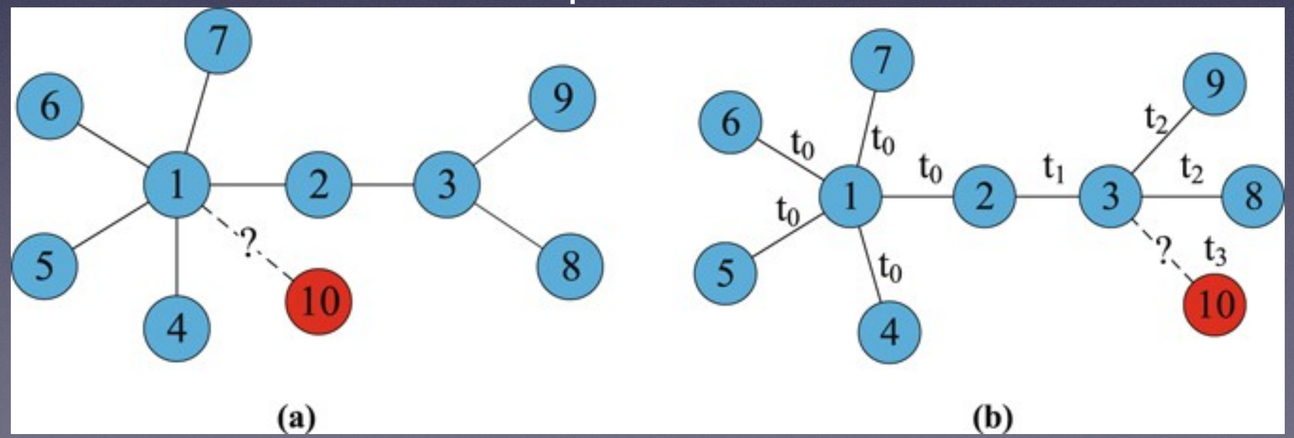
GNNs have various of applications in many different areas including computer vision, NLP, Chemistry, social social etc. Area-specific tasks are first transformed to graph-related tasks, then solved by GNNs

Node-level tasks

Node classification/regression

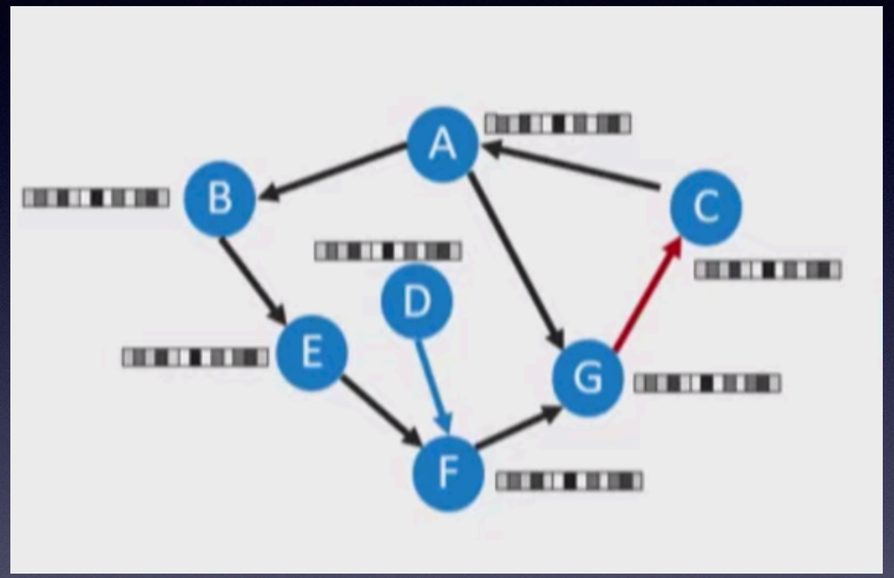


Link prediction

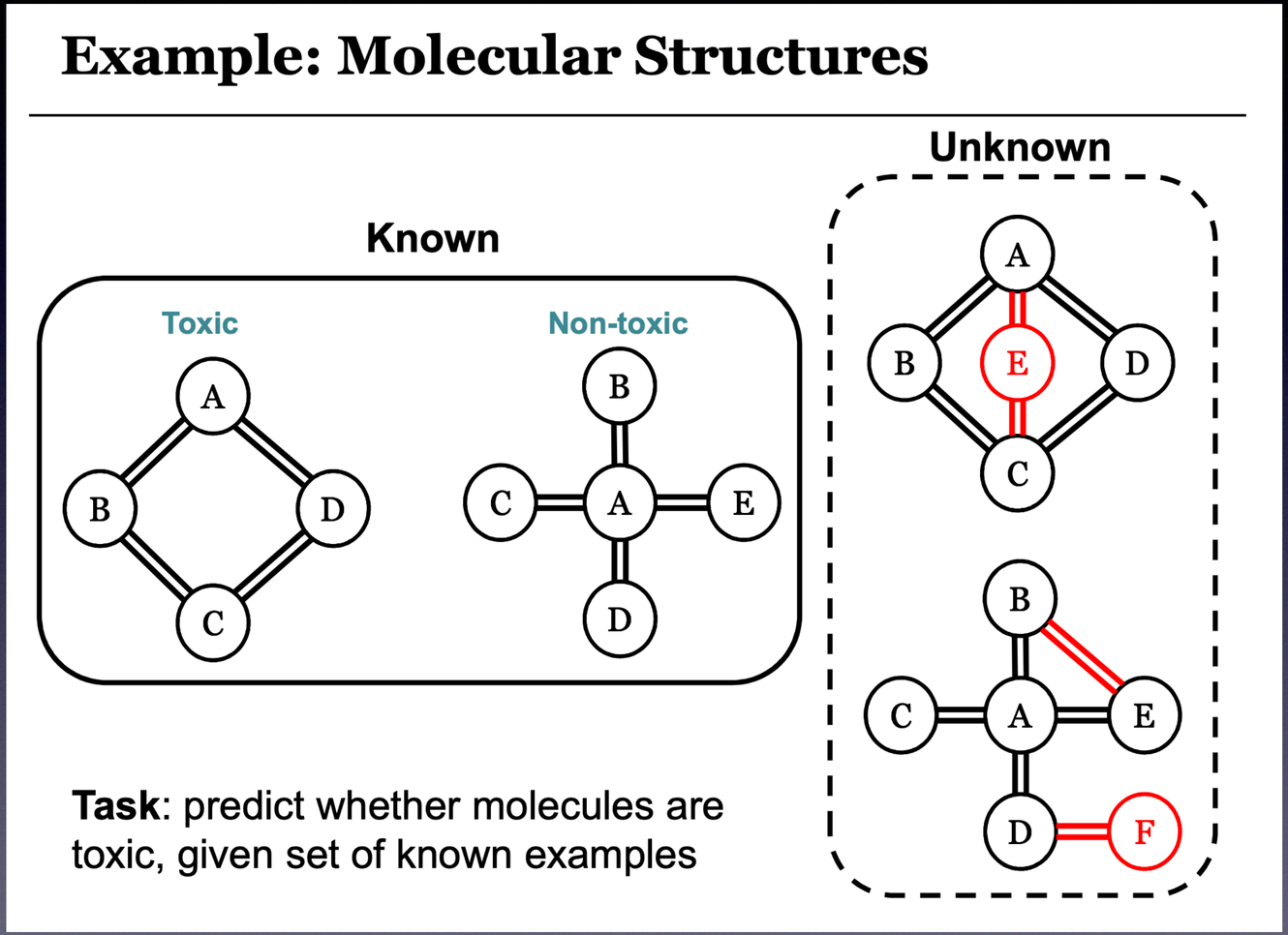


Social Network

- Nodes are users. Each user has a feature vector.
- Two nodes are connected if they are friends of each other
- We want to predict if two users will become friend (link prediction).
- If some of the users are labelled (fans of pop-music), we want to predict the preference of unlabelled users (node classification).



Graph-level tasks



1. Graph classification: to predict some properties of a molecule

2. Graph Generation: to generate some molecules with similar structure to known drugs

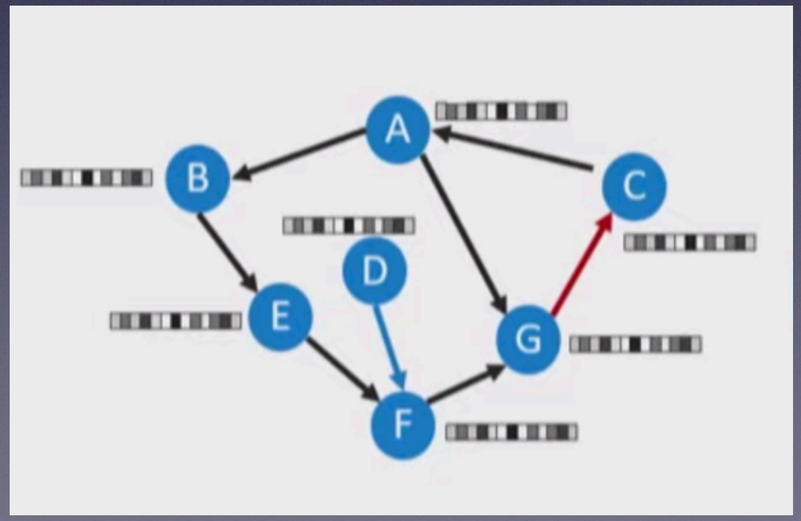
Comparison

Traditional Machine Learning

- Given data $\{x_i, y_i\}_{i=1}^{n}$, x is feature vector and y is the label
- Find the function f(.) that best predicts each data point's label
- Data points are assumed to be independent

Relational Data represented as graph

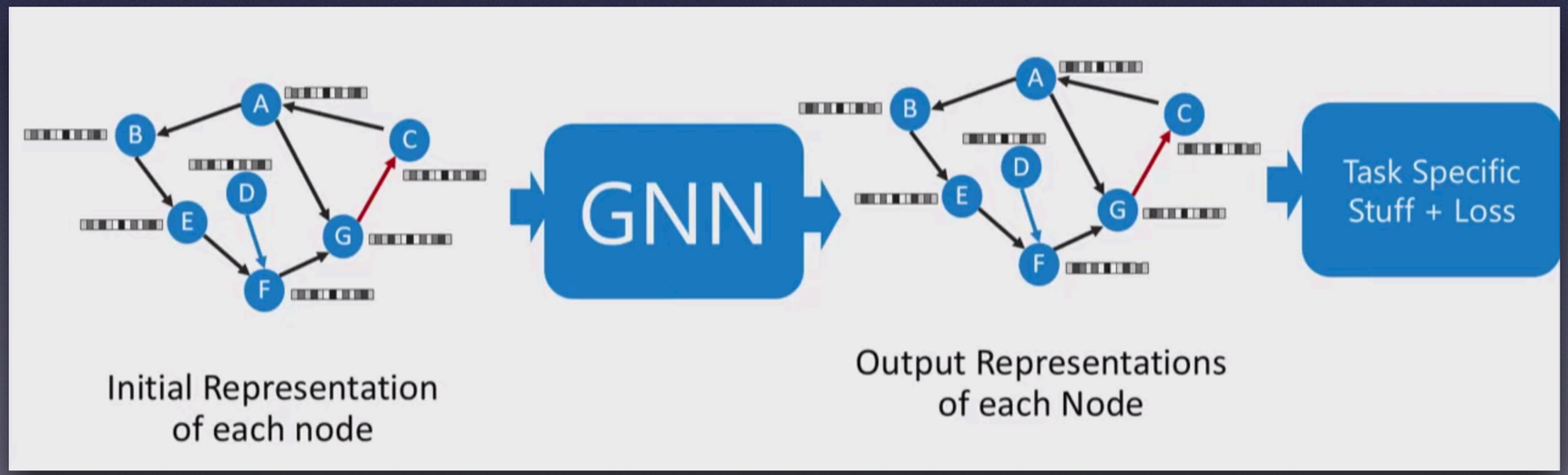
- Data points are connected and form a graph. Each node in graph is a data point. Closeness of data points are defined by their distance in graph



What GNN does

- Intuition/Key idea: Relational data is important. Neighbors' information is helpful to machine learning tasks such as classification. GNN learns to incorporate each node's own information with its neighbor's information.

- For any node v, given its feature x_v, and the network, we want to find representation h_v that not only contains its own information, but also its neighbors' information and its structural information in the graph.



Components of GNN

1. Representation Learning Network
2. Node-level tasks: output function + loss function
3. Graph-level tasks: readout function + output function + loss function

Hidden state and its update

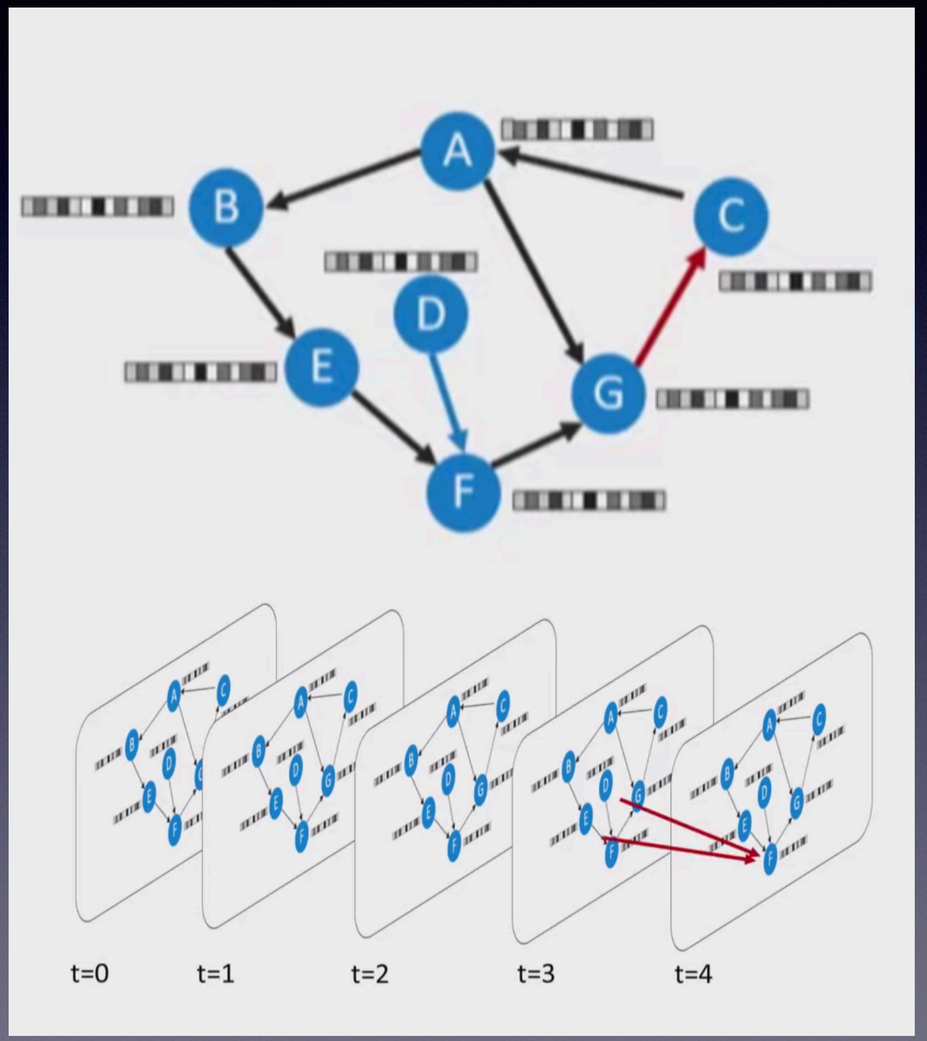
- Each node in graph has a hidden state which is initialized to its feature vector

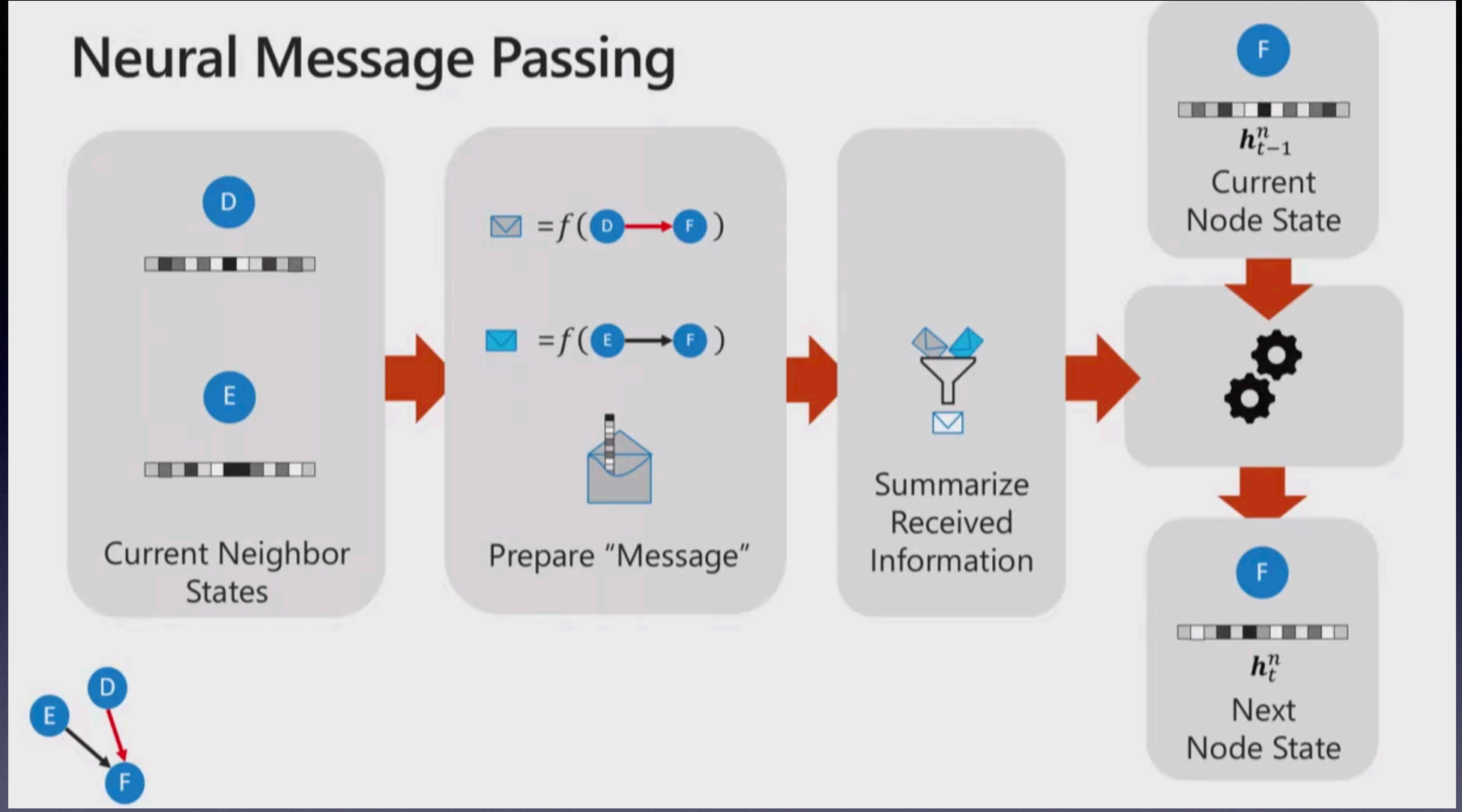
$$h_v^0 = x_v$$

- The hidden state of each node is updated simultaneously in each iteration by message passing between nodes

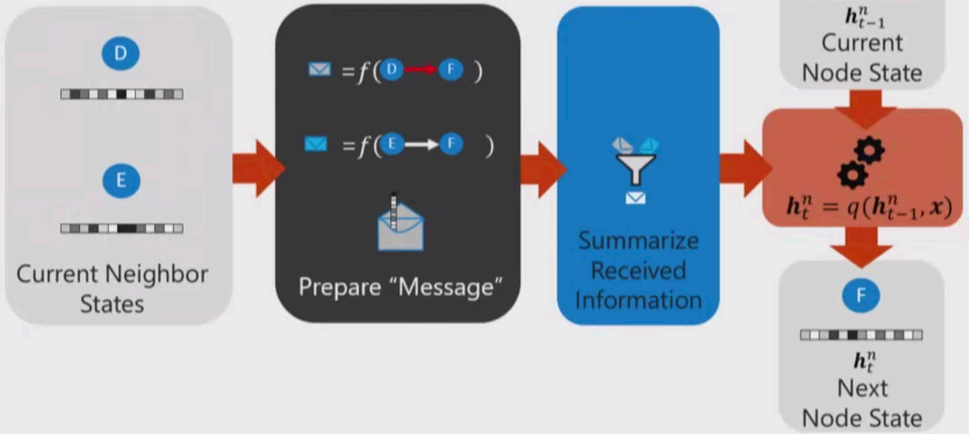
$$H^t = GNN(G, X, H^{t-1})$$

- After K iteration, GNN output the final representation (final hidden state) of nodes.

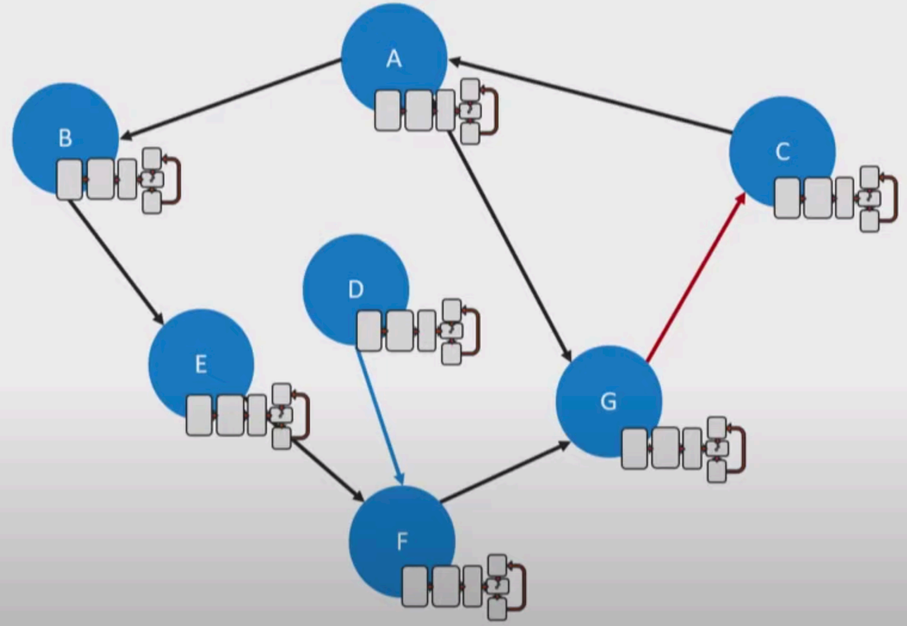




- Each node is initialized with hidden state h^0
- Hidden state is updated iteratively $h^1, h^2, h^3 \ldots$ by neural network
- Final representation h^K> Readout function o(.) (MLP)



$$h_t^n = q\left(h_{t-1}^n, \bigcup_{\substack{k \\ \forall n_j : n \to n_j}} f_t\left(h_{t-1}^n, k, h_{t-1}^{n_j}\right)\right)$$



Key steps:

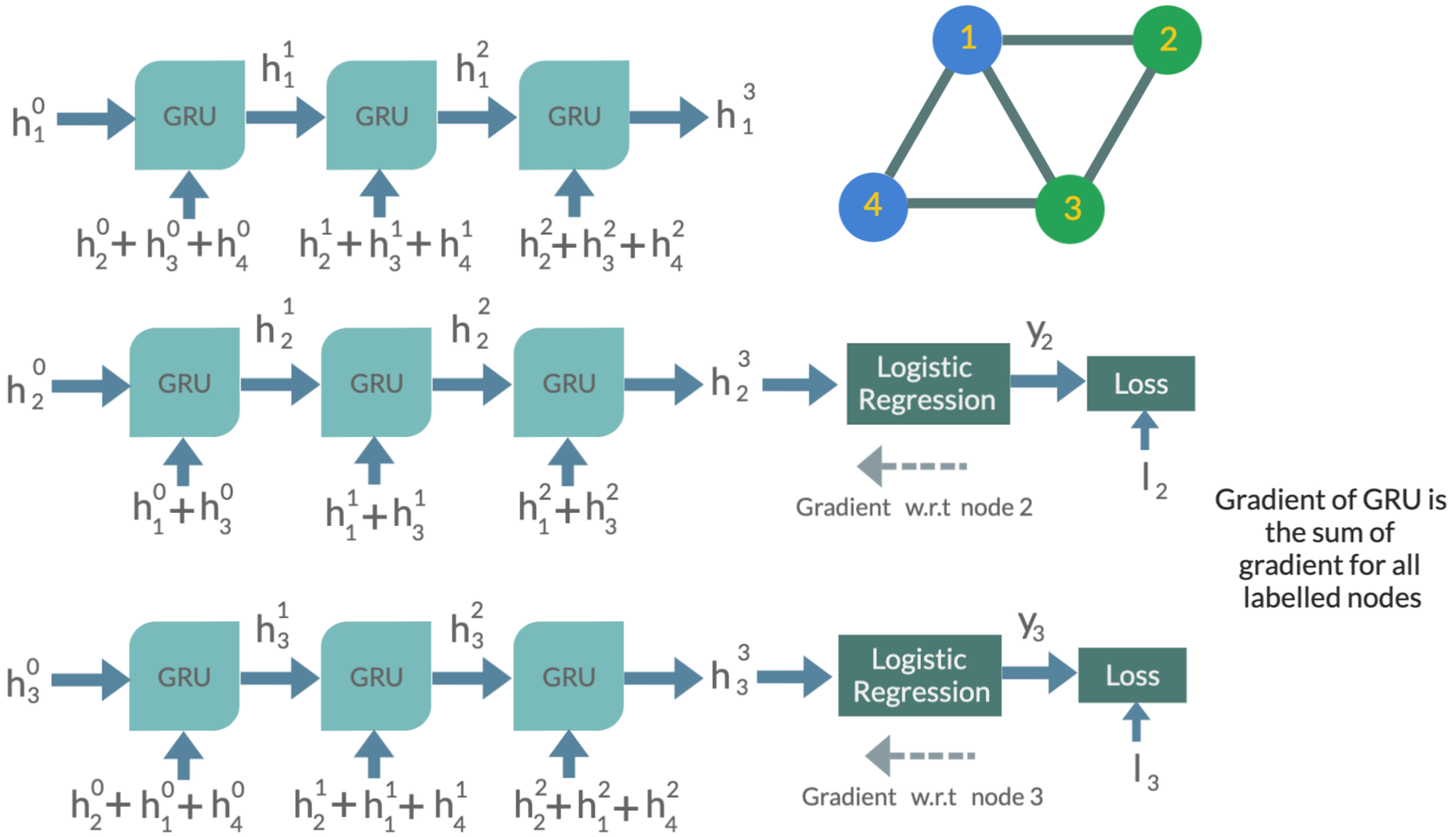
For each node:

1. Aggregate information from neighbors

2. Union the aggregated information with own information

3. Non-linear update of hidden state through neural network.

Gated GNN (2015)



Gradient of GRU is the sum of gradient for all labelled nodes

1. Representation learning through GRU
2. Output function: logistic regression
3. Loss function: cross entropy

The first GNN- GNN* 2009

$$h_v^t = \sum_{u \in neighbor(v)} f(x_v, x^e_{(v,u)}, x_u, h_u^{t-1})$$

$$o_v = g(h_v^T, x_v)$$

- Forces f(.) to be a contractive map (regularized NN) so that hidden representation can converge to a fixed point
- Limited performance due to contractive mapping and same set of weights in iterations

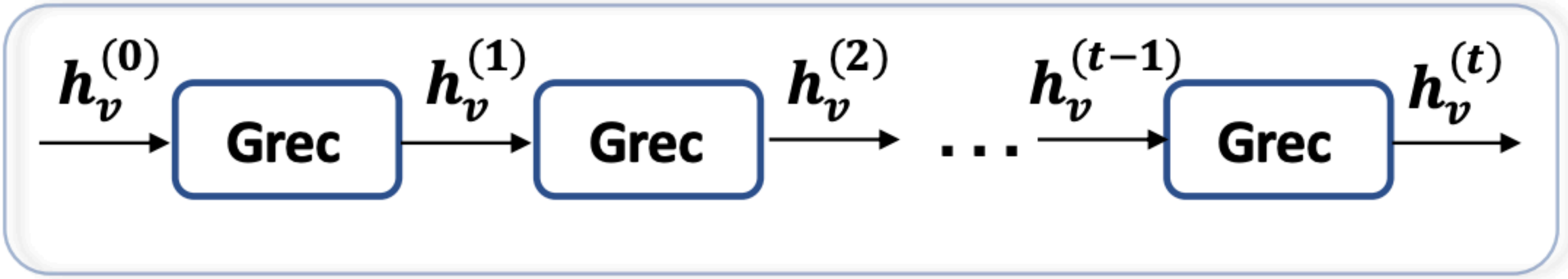
Gated GNN 2015

- Relaxed assumption of convergence
- Gated update of hidden state, trained by BPTT
- Enables sequential output
- BPTT and GRU takes large amount of memory so it is impractical to use on large graph

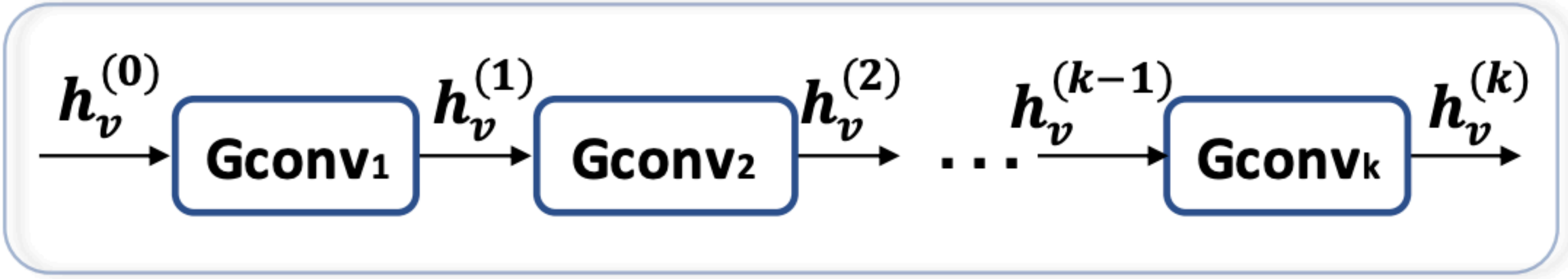
$$h_v^t = GRU(h_v^{t-1}, \sum_{u \in N(v)} h_u^{t-1})$$

Spatial-based ConvGNN

- GNN and Gated GNN are considered as Recurrent GNN because they use the same set of weights to update hidden states
- ConvGNN use individual weights for each iteration of update
- RecGNN: $h_v^t = f_\Theta(G, X, h_v^{t-1})$
- ConvGNN: $h_v^t = f_{\Theta^t}(G, X, h_v^{t-1})$



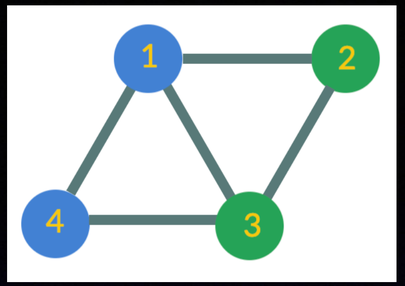
(a) Recurrent Graph Neural Networks (RecGNNs). RecGNNs use the same graph recurrent layer (Grec) in updating node representations.



(b) Convolutional Graph Neural Networks (ConvGNNs). ConvGNNs use a different graph convolutional layer (Gconv) in updating node representations.

GCN (2017)

1.Representation Learning through GCN



$$h_1^0 \rightarrow \boxed{a^1 = W^1 \sum_{u \in N(v_1)} \frac{h_u^0}{|N(v_1)|} + B^1 h_1^0 \quad Relu(a^1)} \rightarrow h_1^1 \rightarrow \boxed{a^2 = W^2 \sum_{u \in N(v_1)} \frac{h_u^1}{|N(v_1)|} + B^2 h_1^1 \quad Relu(a^2)} \rightarrow h_1^2$$

2. Output function

$$o_1 = \sigma(Wh_1^2 + b) \qquad o_1 = MLP(h_1^2)$$

3.. Loss function (Binary cross entropy)

$$L_1 = y_1 \log(o_1) + (1 - y_1)\log(1 - o_1)$$

1.Neural Network for Graph (NN4G) 2009

$$H^t = f(XW^t + \Theta^t \sum_{i=1}^{k-1} AH^i)$$

- f(.) can be any non-linear activation function
- Using different weights at each iteration
- Not popular at that time

2.Graph Convolution Network (GCN) 2017

$$H^t = f(\bar{A}H^{t-1}W^t) \qquad \bar{A} = I_n + D^{-1/2}AD^{-1/2}$$

- $\bar{A}$ Is the normalized adjacency matrix with self-loop
- f(.) can be any non-linear activation function (Relu)
- Most popular framework for its efficiency and flexibility

Variants/Improvements of GCN

- GCN:

$$h_v^t = f(\Theta^t \sum_{u \in N(v) U v} W_{u,v} h_u^{t-1}) \qquad W_{u,v} = \bar{A}_{u,v}$$

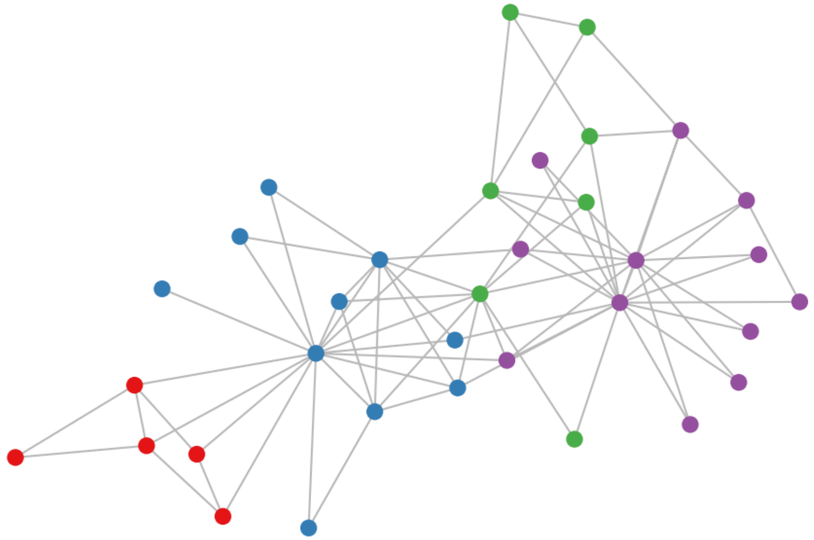
- Graph Isomorphism Network: $W_{u,v} = 1 / W_{u,u} = 1 + \epsilon$
- GraphSage: Sampling neighbors with equal weights
- Graph Attention Network: Adopts attention mechanisms to learn $W_{u,v}$

Classification of publications in citation network

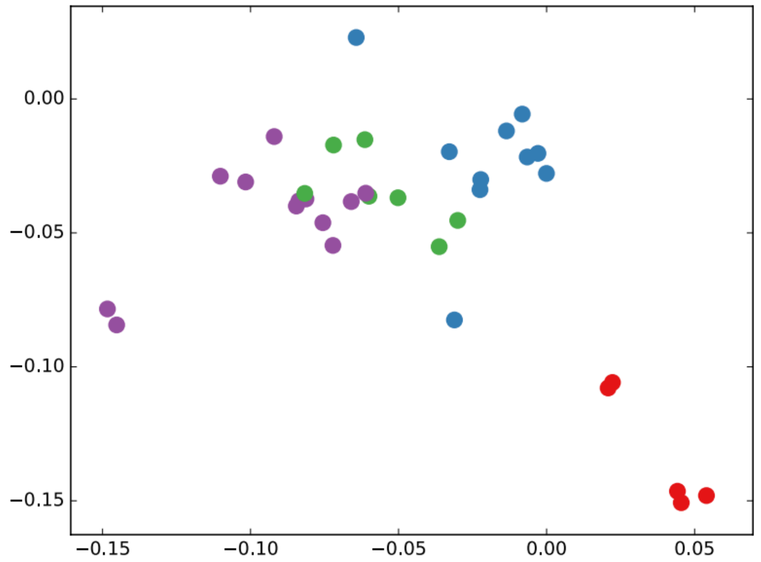
- Nodes are publications and an edge exists between pair of nodes if one publication cites the other.
- CiteCeer / Cora datasets - each publication has a feature vector and a class to predict

Description		Propagation model	Citeseer	Cora
Chebyshev filter (Eq. 5)	$K = 3$	$\sum_{k=0}^{K} T_k(\tilde{L}) X \Theta_k$	69.8	79.5
	$K = 2$		69.6	81.2
1st-order model (Eq. 6)		$X\Theta_0 + D^{-\frac{1}{2}} A D^{-\frac{1}{2}} X \Theta_1$	68.3	80.0
Single parameter (Eq. 7)		$(I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}}) X \Theta$	69.3	79.2
Renormalization trick (Eq. 8)		$\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} X \Theta$	**70.3**	**81.5**
1st-order term only		$D^{-\frac{1}{2}} A D^{-\frac{1}{2}} X \Theta$	68.7	80.5
Multi-layer perceptron		$X\Theta$	46.5	55.1

Representation learning without training



(a) Karate club network



(b) Random weight embedding

Figure 3: *Left*: Zachary's karate club network (Zachary, 1977), colors denote communities obtained via modularity-based clustering (Brandes et al., 2008). *Right*: Embeddings obtained from an untrained 3-layer GCN model (Eq. 13) with random weights applied to the karate club network. Best viewed on a computer screen.

Applications

Taxonomy of applications

- Structural Scenerios

  Networks are explicitly defined

  (social networks, molecule networks etc.)

- Non-Structural Scenerios

  First manually construct the graph, then apply GNN.

  (images, text, programs etc.)

Types of tasks on graph

1. Node classification/regression: predict property of nodes
2. Link prediction: predict whether two nodes are connected
3. Graph classification: predict property of the whole graph
4. Graph Generation: generate graphs similar to given graph

Structural Scenerios

Benchmark datasets

1. Citation Networks

Papers as nodes, citation as edges.

Cora - 2708 nodes/ 5429 edges

Pubmed - 19717 nodes/ 44338 edges

DBLP - 4107340 nodes/ 36624464 edges

2. Social Networks

Users as nodes, interaction as edges.

Reddit - 232965 nodes/ 11606919 edges

BlogCatalog - 10312 nodes/ 333983 edges

3. Bio-chemical Networks

NCI-1/NCI-9 - 4110/4127 molecular networks of chemical compounds, labeled as active to hinder the growth of human cancer cell

PPI - Protein Protein Interaction

QM9 - 133885 molecular networks of chemical compounds with 13 physical properties

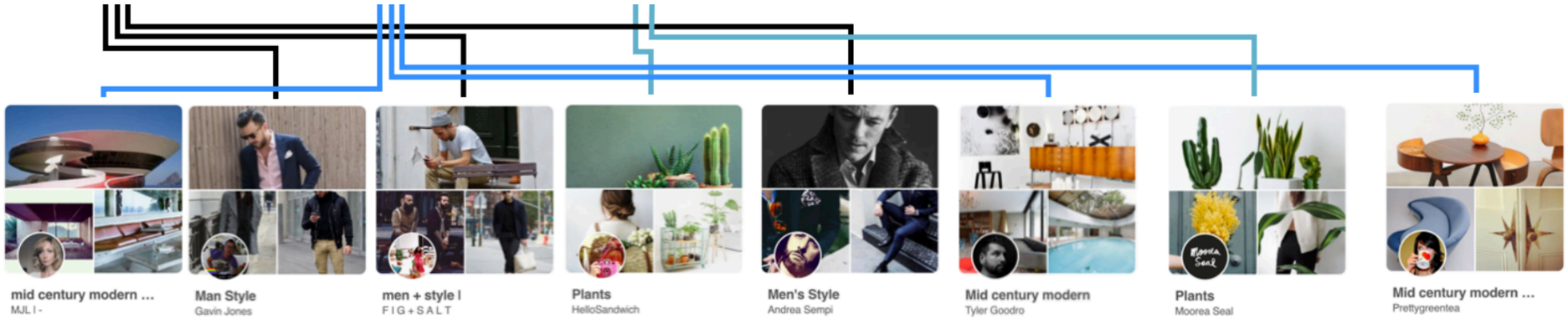
4. Traffic Networks

METR-LA - four months of traffic data collected at 207 monitors on highway at LA

Recommendation System- Pinterest

Pin: A visual bookmark someone has saved from the internet to a board they've created.

Pin: Image, text, link



Board: A collection of ideas (pins having something in common)

Recommendation System

Method	Hit-rate	MRR
Visual	17%	0.23
Annotation	14%	0.19
Combined	27%	0.37
max-pooling	39%	0.37
mean-pooling	41%	0.51
mean-pooling-xent	29%	0.35
mean-pooling-hard	46%	0.56
PinSage	**67%**	**0.59**

- Each item has features. Old-school recommendation systems only use item feature.
- With users' interaction history with items, we can perform graph-based recommendation (collaborative filtering).
- Link prediction
- State-of-art performance

Chemistry/Drug Discovery

1. Graph Classification

Given molecular network:

(1). Predict its chemical property (e.g stableness/harmfulness)

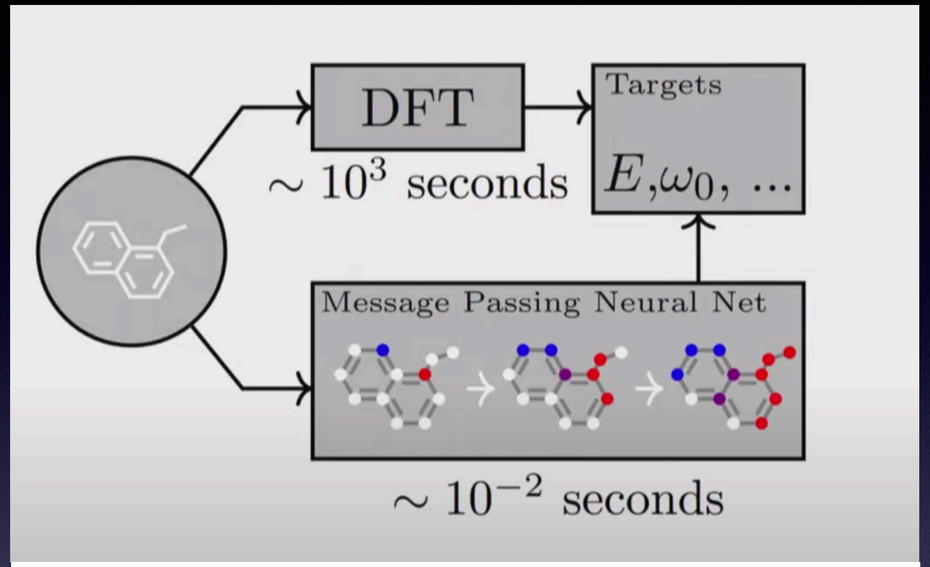
2. Graph Generation

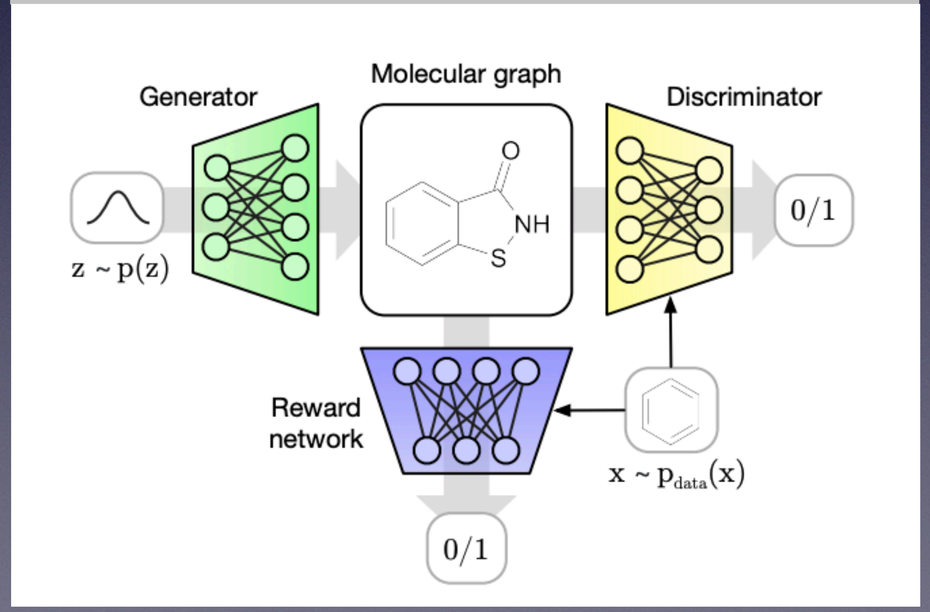
Generate molecular graph similar to drugs

3. Link prediction

Given protein interaction graph:

(1) Predict relationship between two proteins



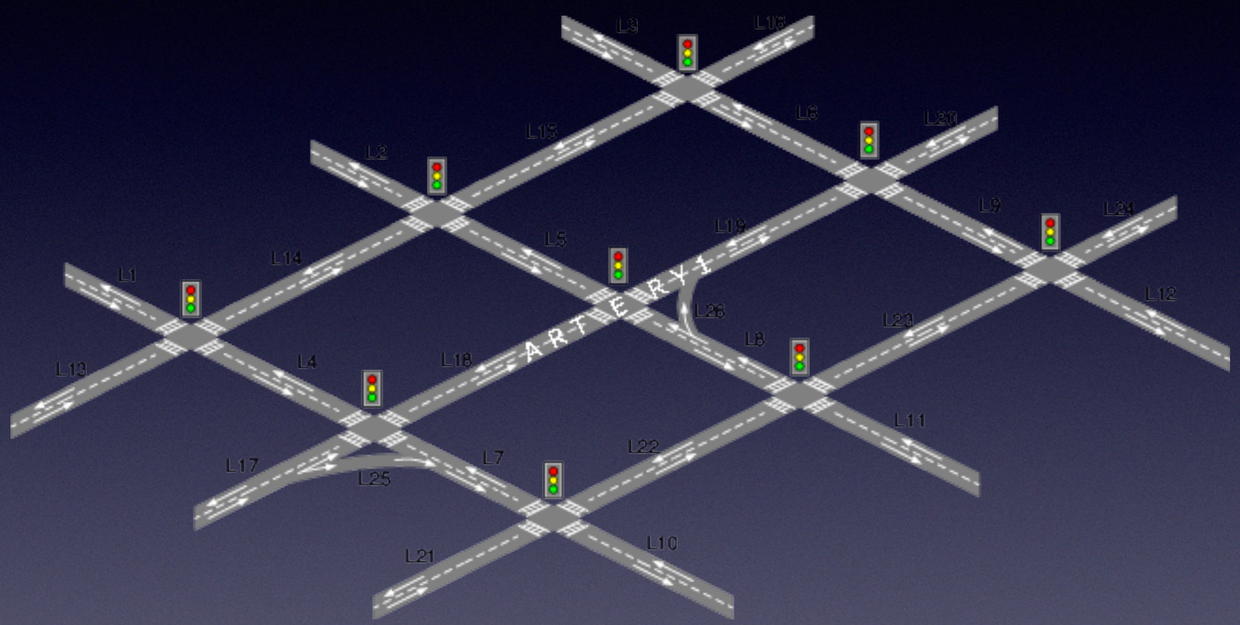


Traffic Prediction

- Models roads as network
- Given history data of traffic flow

$$X^1, X^2, X^2 \ldots X^T$$

- Predict future traffic flow X^{T+1}



Social Network

1. Node Classification

Predict personal attributes (e.g social influence)

Predict personal group/class

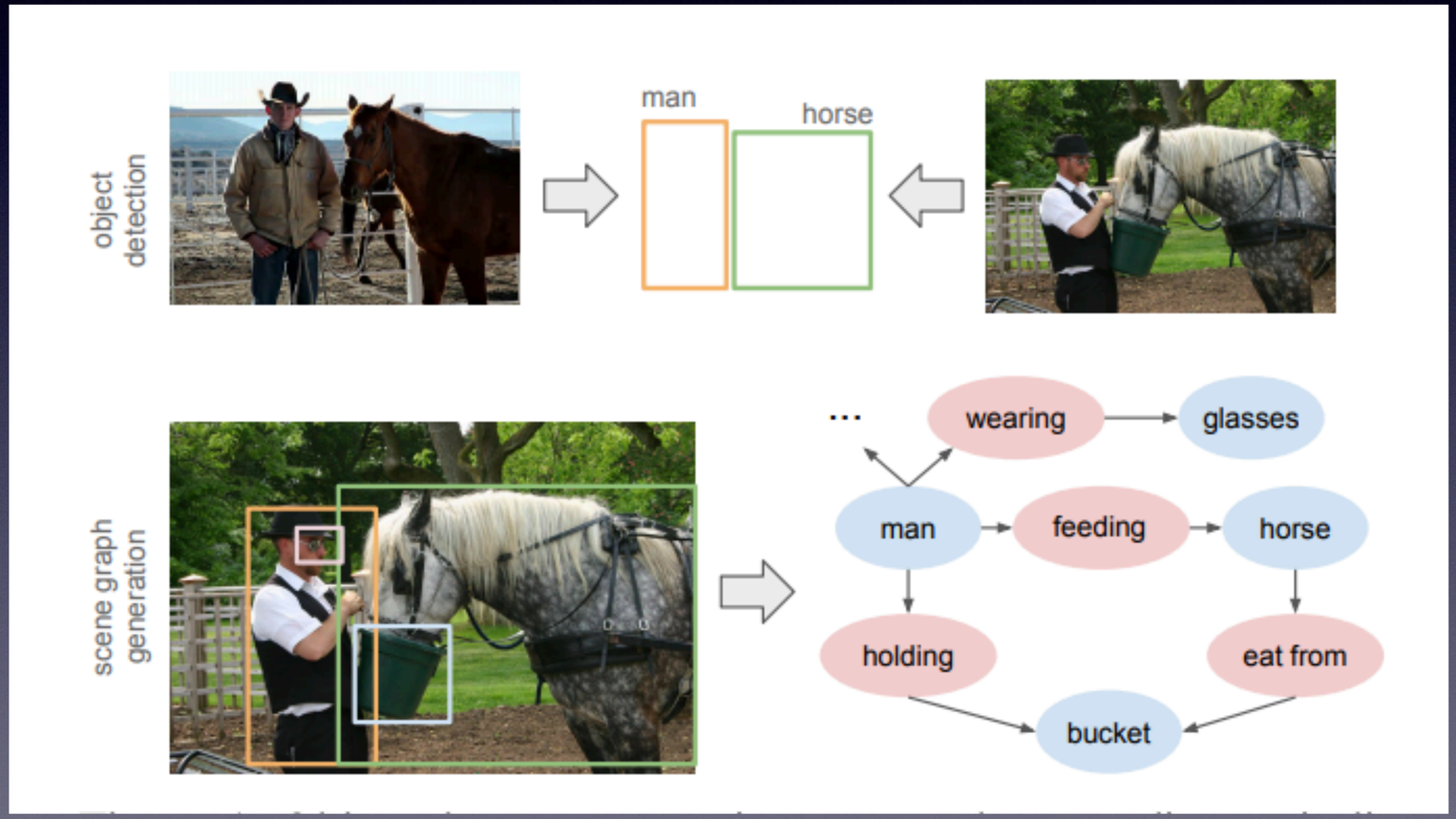
2. Link Prediction

Predict interaction between two people

Non-Structural Scenarios

Computer Vision

- Scene Graph Generation D. Xu, Y. Zhu, C. B. Choy, and L. Fei-Fei, "Scene graph generation by iterative message passing," in Proc. of CVPR, vol. 2, 2017



Based on Scene Graph:

1. Semantic Segmentation X. Qi, R. Liao, J. Jia, S. Fidler, and R. Urtasun, “3d graph neural networks for rgbd semantic segmentation,” in Proc. of CVPR, 2017, pp. 5199–5208.

2. Visual Reasoning X. Chen, L.-J. Li, L. Fei-Fei, and A. Gupta, “Iterative visual reasoning beyond convolutions,” in Proc. of CVPR, 2018.

3. Question Answering M. Narasimhan, S. Lazebnik, and A. Schwing, “Out of the box: Reasoning with graph convolution nets for factual visual question answering,” in Proc. of NeurIPS, 2018, pp. 2655–2666

4. The Scene graph generation can be reversed, given scene graph, generate a corresponding image J. Johnson, A. Gupta, and L. Fei-Fei, “Image generation from scene graphs,” in Proc. of CVPR, 2018.

5. Few-shot learning V. G. Satorras and J. B. Estrach, “Few-shot learning with graph neural networks,” in Proc. of ICLR, 2018.

Point Cloud

- Usage: Computer Graphics, LiDar technology
- Task: Given a set of 3-D data points

(1) Classify the whole point cloud

(2) Segment the point cloud into semantic parts

Other applications

1. Brian Network

J. Kawahara, C. J. Brown, S. P. Miller, B. G. Booth, V. Chau, R. E. Grunau, J. G. Zwicker, and G. Hamarneh, “Brainnetcnn: convolutional neural networks for brain networks; towards predicting neurodevelopment,” NeuroImage, vol. 146, pp. 1038–1049, 2017.

Predicting neurodevelopment of brain and identify possibly important brain connectivity

2. Program Verification

Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, “Gated graph sequence neural networks,” in Proc. of ICLR, 2015.

3. Combinatorial Optimization (Travel Salesman Problem)

T. H. Nguyen and R. Grishman, “Graph convolutional networks with argument-aware pooling for event detection,” in Proc. of AAAI, 2018, pp. 5900–5907.