Brief Introduction to Graph Neural Networks

Zheng Wang

08/31/2021

1. A Comprehensive Survey on Graph Neural Networks

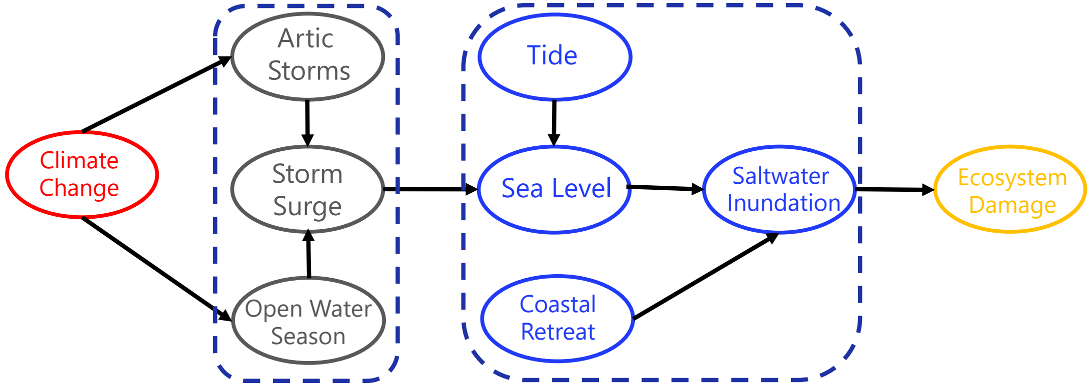
2. Graph neural networks: A review of methods and applications

Outline

- Background
- Taxonomy and representative methods
- Applications

Background

- Motivation
  - More available graph-structured data, e.g., entity interactions in recommendation system, molecular bioactivity in drug discovery
  - Development of CNN, RNN, autoencoders and other deep learning techniques
- Timeline (1997)
  - Recurrent GNNs (RecGNN)
  - Convolutional GNNs (ConvGNN)
  - Graph Autoencoders (GAEs)
  - Spatial-temporal GNNs (STGNNs)



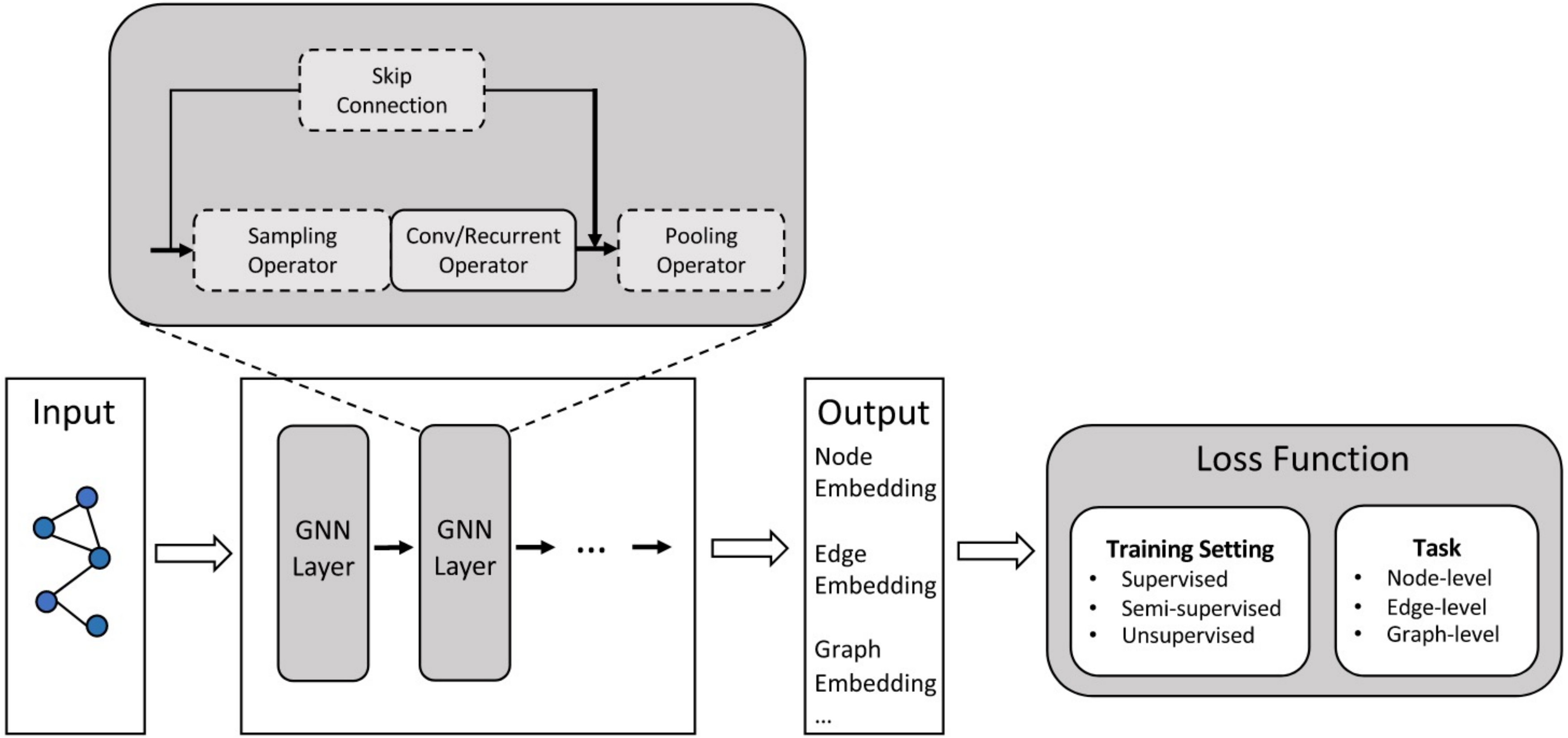
Definition

Definition 1 (Graph): A graph is represented as $G = (V, E)$ where V is the set of vertices or nodes (we will use nodes throughout the paper), and E is the set of edges. Let $v_i \in V$ to denote a node and $e_{ij} = (v_i, v_j) \in E$ to denote an edge pointing from v_j to v_i. The neighborhood of a node v is defined as $N(v) = \{u \in V | (v, u) \in E\}$. The adjacency matrix $\mathbf{A}$ is a $n \times n$ matrix with $A_{ij} = 1$ if $e_{ij} \in E$ and $A_{ij} = 0$ if $e_{ij} \notin E$. A graph may have node attributes $\mathbf{X}$ [1], where $\mathbf{X} \in \mathbf{R}^{n \times d}$ is a node feature matrix with $\mathbf{x}_v \in \mathbf{R}^d$ representing the feature vector of a node v. Meanwhile, a graph may have edge attributes $\mathbf{X}^e$, where $\mathbf{X}^e \in \mathbf{R}^{m \times c}$ is an edge feature matrix with $\mathbf{x}^e_{v,u} \in \mathbf{R}^c$ representing the feature vector of an edge (v, u).

Definition 2 (Directed Graph): A directed graph is a graph with all edges directed from one node to another. An undirected graph is considered as a special case of directed graphs where there is a pair of edges with inverse directions if two nodes are connected. A graph is undirected if and only if the adjacency matrix is symmetric.

Definition 3 (Spatial-Temporal Graph): A spatial-temporal graph is an attributed graph where the node attributes change dynamically over time. The spatial-temporal graph is defined as $G^{(t)} = (\mathbf{V}, \mathbf{E}, \mathbf{X}^{(t)})$ with $\mathbf{X}^{(t)} \in \mathbf{R}^{n \times d}$.

General GNN framework



1. Find graph structure.

2. Specify graph type and scale.

4. Build model using computational modules.

3. Design loss function.

Taxonomy

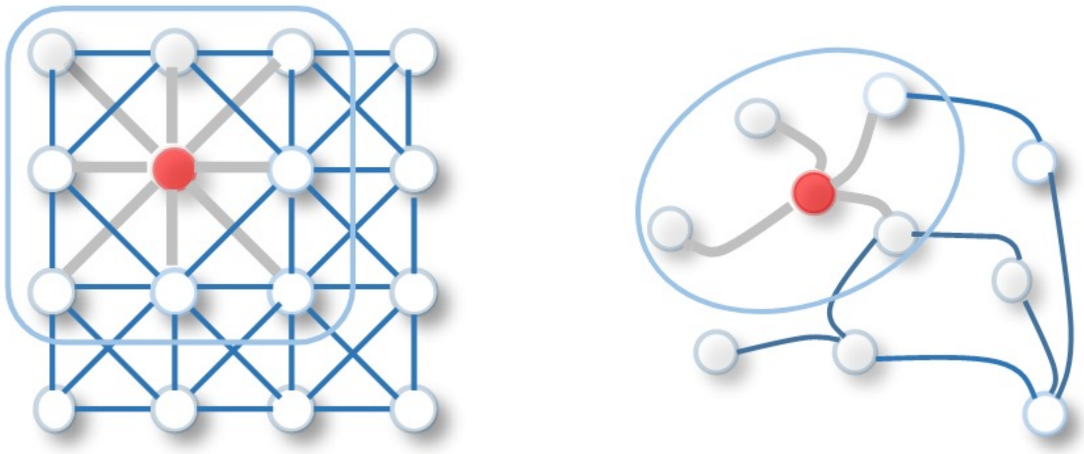
Category	
Recurrent Graph Neural Networks (RecGNNs)	
Convolutional Graph Neural Networks (ConvGNNs)	Spectral methods
	Spatial methods
Graph Autoencoders (GAEs)	Network Embedding
	Graph Generation
Spatial-temporal Graph Neural Networks (STGNNs)	

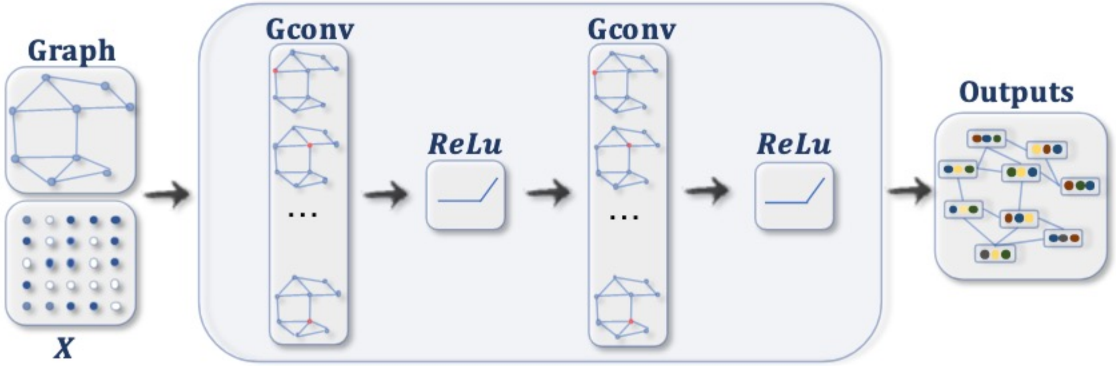
Recurrent Graph Neural Networks

- Convergence-based Method
  - GNN*
    $$\mathbf{h}_v = f\left(\mathbf{x}_v, \mathbf{x}_{co[v]}, \mathbf{h}_{\mathcal{N}_v}, \mathbf{x}_{\mathcal{N}_v}\right),$$
    $$\mathbf{o}_v = g(\mathbf{h}_v, \mathbf{x}_v),$$
  - Graph Echo State Network (GraphESN)
  - Stochastic Steady-state Embedding (SSE)
  - Lagrangian Propagation GNN (LP-GNN)
- Gate-based
  - Gated Graph Neural Network (GGNN)
  - Tree LSTM, Graph LSTM, S-LSTM

Variant	Aggregator	Updater
GGNN	$\mathbf{h}^t_{\mathcal{N}_v} = \sum_{k\in\mathcal{N}_v} \mathbf{h}_k^{t-1} + \mathbf{b}$	$\mathbf{z}_v^t = \sigma(\mathbf{W}^z\mathbf{h}^t_{\mathcal{N}_v} + \mathbf{U}^z\mathbf{h}_v^{t-1})$ $\mathbf{r}_v^t = \sigma(\mathbf{W}^r\mathbf{h}^t_{\mathcal{N}_v} + \mathbf{U}^r\mathbf{h}_v^{t-1})$ $\tilde{\mathbf{h}_v^t} = \tanh(\mathbf{W}\mathbf{h}^t_{\mathcal{N}_v} + \mathbf{U}(\mathbf{r}_v^t \odot \mathbf{h}_v^{t-1}))$ $\mathbf{h}_v^t = (1-\mathbf{z}_v^t) \odot \mathbf{h}_v^{t-1} + \mathbf{z}_v^t \odot \tilde{\mathbf{h}_v^t}$
Tree LSTM (Child sum)	$\mathbf{h}^{ti}_{\mathcal{N}_v} = \sum_{k\in\mathcal{N}_v} \mathbf{U}^i\mathbf{h}_k^{t-1}$ $\mathbf{h}^{tf}_{\mathcal{N}_v k} = \mathbf{U}^f\mathbf{h}_k^{t-1}$ $\mathbf{h}^{to}_{\mathcal{N}_v} = \sum_{k\in\mathcal{N}_v} \mathbf{U}^o\mathbf{h}_k^{t-1}$ $\mathbf{h}^{tu}_{\mathcal{N}_v} = \sum_{k\in\mathcal{N}_v} \mathbf{U}^u\mathbf{h}_k^{t-1}$	$\mathbf{i}_v^t = \sigma(\mathbf{W}^i\mathbf{x}_v^t + \mathbf{h}^{ti}_{\mathcal{N}_v} + \mathbf{b}^i)$ $\mathbf{f}_{vk}^t = \sigma(\mathbf{W}^f\mathbf{x}_v^t + \mathbf{h}^{tf}_{\mathcal{N}_v k} + \mathbf{b}^f)$ $\mathbf{o}_v^t = \sigma(\mathbf{W}^o\mathbf{x}_v^t + \mathbf{h}^{to}_{\mathcal{N}_v} + \mathbf{b}^o)$ $\mathbf{u}_v^t = \tanh(\mathbf{W}^u\mathbf{x}_v^t + \mathbf{h}^{tu}_{\mathcal{N}_v} + \mathbf{b}^u)$ $\mathbf{c}_v^t = \mathbf{i}_v^t \odot \mathbf{u}_v^t + \sum_{k\in\mathcal{N}_v} \mathbf{f}_{vk}^t \odot \mathbf{c}_k^{t-1}$ $\mathbf{h}_v^t = \mathbf{o}_v^t \odot \tanh(\mathbf{c}_v^t)$
Tree LSTM (N-ary)	$\mathbf{h}^{ti}_{\mathcal{N}_v} = \sum_{l=1}^{K} \mathbf{U}_l^i\mathbf{h}_{vl}^{t-1}$ $\mathbf{h}^{tf}_{\mathcal{N}_v k} = \sum_{l=1}^{K} \mathbf{U}_{kl}^f\mathbf{h}_{vl}^{t-1}$ $\mathbf{h}^{to}_{\mathcal{N}_v} = \sum_{l=1}^{K} \mathbf{U}_l^o\mathbf{h}_{vl}^{t-1}$ $\mathbf{h}^{tu}_{\mathcal{N}_v} = \sum_{l=1}^{K} \mathbf{U}_l^u\mathbf{h}_{vl}^{t-1}$	
Graph LSTM in (Peng et al., 2017)	$\mathbf{h}^{ti}_{\mathcal{N}_v} = \sum_{k\in\mathcal{N}_v} \mathbf{U}^i_{m(v,k)}\mathbf{h}_k^{t-1}$ $\mathbf{h}^{tf}_{\mathcal{N}_v k} = \mathbf{U}^f_{m(v,k)}\mathbf{h}_k^{t-1}$ $\mathbf{h}^{to}_{\mathcal{N}_v} = \sum_{k\in\mathcal{N}_v} \mathbf{U}^o_{m(v,k)}\mathbf{h}_k^{t-1}$ $\mathbf{h}^{tu}_{\mathcal{N}_v} = \sum_{k\in\mathcal{N}_v} \mathbf{U}^u_{m(v,k)}\mathbf{h}_k^{t-1}$	

Convolutional GNNs





Spectral-based ConvGNNs

- Definition

$O(n^2)$

$$\mathbf{x} *_G \mathbf{g} = \mathscr{F}^{-1}(\mathscr{F}(\mathbf{x}) \odot \mathscr{F}(\mathbf{g})) = \mathbf{U}(\mathbf{U}^T\mathbf{x} \odot \mathbf{U}^T\mathbf{g}),$$

$$\mathbf{g}_\theta = diag(\mathbf{U}^T\mathbf{g})$$

$$\mathbf{x} *_G \mathbf{g}_\theta = \mathbf{U}\mathbf{g}_\theta\mathbf{U}^T\mathbf{x}.$$

undirected graph

$$\mathbf{D}_{ii} = \sum_j (\mathbf{A}_{i,j}).$$

$$\mathbf{L} = \mathbf{I_n} - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}},$$

$$\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$$

$O(n^3)$

Spectral-based ConvGNNs

- ChebNet

$$\mathbf{g}_\theta = \sum_{i=0}^{K} \theta_i T_i(\tilde{\mathbf{\Lambda}})$$

$$\mathbf{x} *_G \mathbf{g}_\theta = \mathbf{U}(\sum_{i=0}^{K} \theta_i T_i(\tilde{\mathbf{\Lambda}}))\mathbf{U}^T\mathbf{x},$$

$$T_i(\tilde{\mathbf{L}}) = \mathbf{U}T_i(\tilde{\mathbf{\Lambda}})\mathbf{U}^T$$

$$\mathbf{x} *_G \mathbf{g}_\theta = \sum_{i=0}^{K} \theta_i T_i(\tilde{\mathbf{L}})\mathbf{x},$$

$$\tilde{\mathbf{\Lambda}} = 2\mathbf{\Lambda}/\lambda_{max} - \mathbf{I_n},$$

$$\tilde{\mathbf{L}} = 2\mathbf{L}/\lambda_{max} - \mathbf{I_n}$$

Chebyshev Polynmials:

$$T_i(\mathbf{x}) = 2\mathbf{x}T_{i-1}(\mathbf{x}) - T_{i-2}(\mathbf{x})$$

$$T_0(\mathbf{x}) = 1 \text{ and } T_1(\mathbf{x}) = \mathbf{x}.$$

- CayleyNet (Cayley polynomials)
- Graph Convolutional Network (GCN)
  - first-order approximation of ChebNet $\mathbf{x} *_G \mathbf{g}_\theta = \theta_0\mathbf{x} - \theta_1\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}\mathbf{x}.$

$$\mathbf{H} = \mathbf{X} *_G \mathbf{g}_{\mathbf{\Theta}} = f(\bar{\mathbf{A}}\mathbf{X}\mathbf{\Theta}), \quad \bar{\mathbf{A}} = \mathbf{I_n} + \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$$

Spatial-based ConvGNNs

- Neural Network for Graphs (NN4G)

$$\mathbf{H}^{(k)} = f(\mathbf{X}\mathbf{W}^{(k)} + \sum_{i=1}^{k-1} \mathbf{A}\mathbf{H}^{(k-1)}\mathbf{\Theta}^{(k)}),$$

- Diffusion Graph Convolution (DGC)

$$\mathbf{H} = \sum_{k=0}^{K} f(\mathbf{P}^k\mathbf{X}\mathbf{W}^{(k)}), \quad \mathbf{P} = \mathbf{D}^{-1}\mathbf{A}.$$

- PGC-DGCNN

$$\mathbf{H}^{(k)} = \|_{j=0}^{r} f((\tilde{\mathbf{D}}^{(j)})^{-1}\mathbf{S}^{(j)}\mathbf{H}^{(k-1)}\mathbf{W}^{(j,k)}),$$

- Message Passing Neural Networks

$$\mathbf{h}_v^{(k)} = U_k(\mathbf{h}_v^{(k-1)}, \sum_{u \in N(v)} M_k(\mathbf{h}_v^{(k-1)}, \mathbf{h}_u^{(k-1)}, \mathbf{x}_{vu}^e)),$$

Spatial-based ConvGNNs

- GraphSage

$$\mathbf{h}_v^{(k)} = \sigma(\mathbf{W}^{(k)} \cdot f_k(\mathbf{h}_v^{(k-1)}, \{\mathbf{h}_u^{(k-1)}, \forall u \in S_{\mathcal{N}(v)}\})),$$

- Graph Attention Network (GAT)

$$\mathbf{h}_v^{(k)} = \sigma(\sum_{u \in \mathcal{N}(v) \cup v} \alpha_{vu}^{(k)} \mathbf{W}^{(k)} \mathbf{h}_u^{(k-1)}),$$

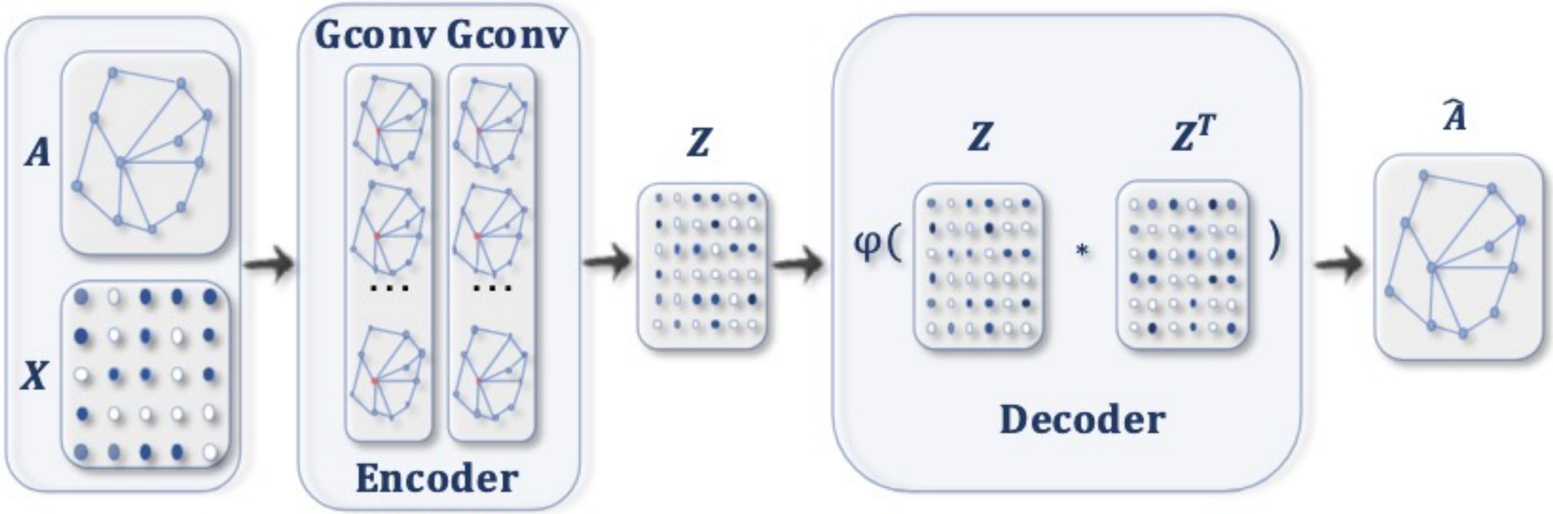
$$\alpha_{vu}^{(k)} = softmax(g(\mathbf{a}^T [\mathbf{W}^{(k)} \mathbf{h}_v^{(k-1)} || \mathbf{W}^{(k)} \mathbf{h}_u^{(k-1)})),$$

- Mixture Model Network (MoNet)
  - relative position

Spectral vs Spatial

- Spectral
  - Theoretical foundation in graph signal processing
  - Undirected graphs
  - Whole graph
  - Fixed graph
- Spatial
  - More scalable to large graphs, sampling
  - New nodes
  - Directed and undirected graphs

Graph Autoencoders



Graph Autoencoders

- Structural Deep Network Embedding (SDNE)

Encoder: $$L_{1st} = \sum_{(v,u)\in E} A_{v,u}||enc(\mathbf{x}_v) - enc(\mathbf{x}_u)||^2,$$

Decoder: $$L_{2nd} = \sum_{v\in V} ||(dec(enc(\mathbf{x}_v)) - \mathbf{x}_v) \odot \mathbf{b}_v||^2, \quad b_{v,u} = 1 \text{ if } A_{v,u} = 0,\ b_{v,u} = \beta > 1 \text{ if } A_{v,u} = 1$$

- GAE* (variational)

$$\mathbf{Z} = enc(\mathbf{X}, \mathbf{A}) = Gconv(f(Gconv(\mathbf{A}, \mathbf{X}; \mathbf{\Theta_1})); \mathbf{\Theta_2}),$$

$$\hat{\mathbf{A}}_{v,u} = dec(\mathbf{z}_v, \mathbf{z}_u) = \sigma(\mathbf{z}_v^T \mathbf{z}_u),$$

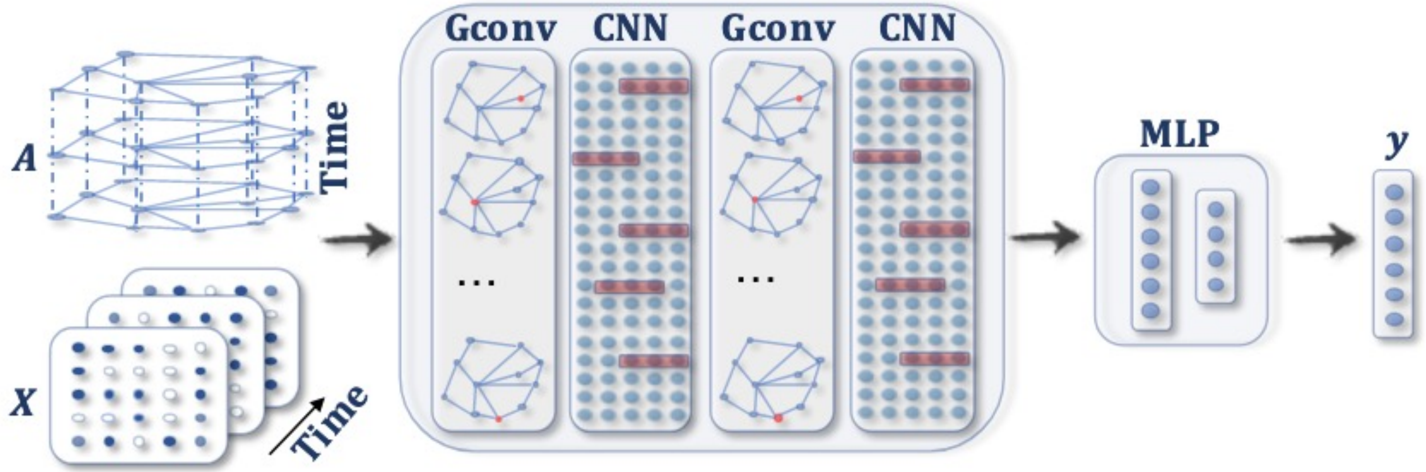
- Network Representationswith Adversarially Regularized Autoencoders (NetRA)

$$L = -E_{\mathbf{z}\sim P_{data}(\mathbf{z})}(dist(\mathbf{z}, dec(enc(\mathbf{z})))),$$

Spatial-temporal GNNs

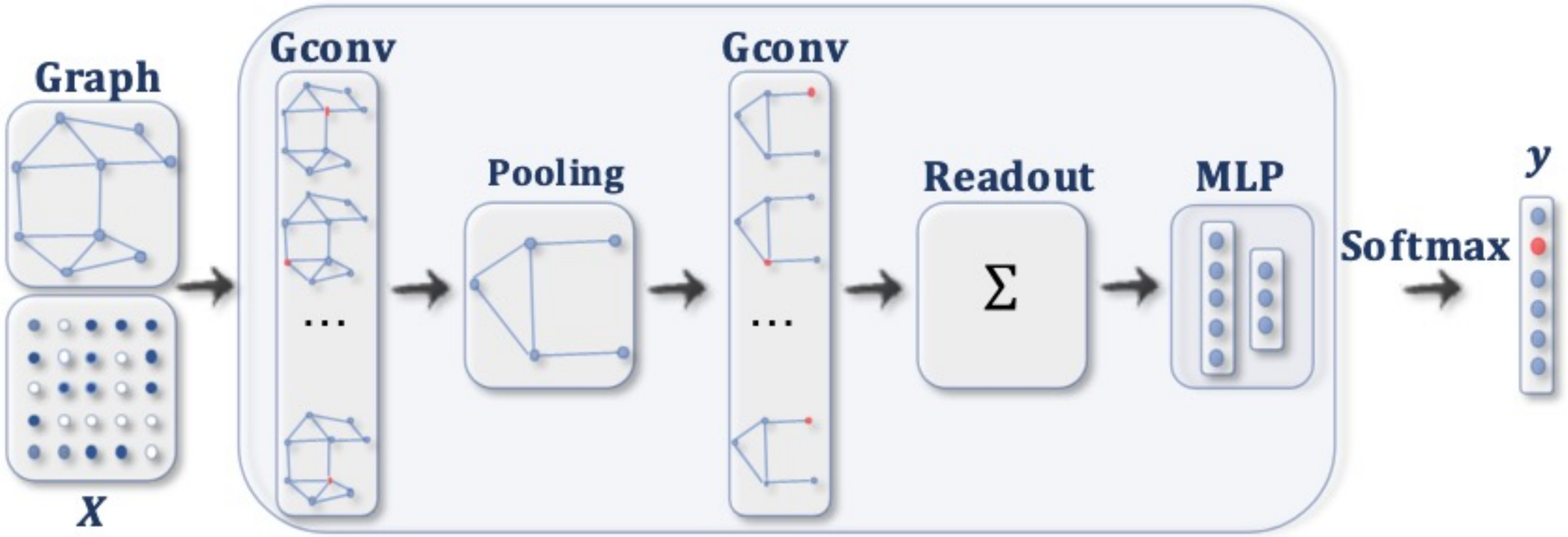
- General formulation

$$\mathbf{H}^{(t)} = \sigma(Gconv(\mathbf{X}^{(t)}, \mathbf{A}; \mathbf{W}) + Gconv(\mathbf{H}^{(t-1)}, \mathbf{A}; \mathbf{U}) + \mathbf{b}),$$

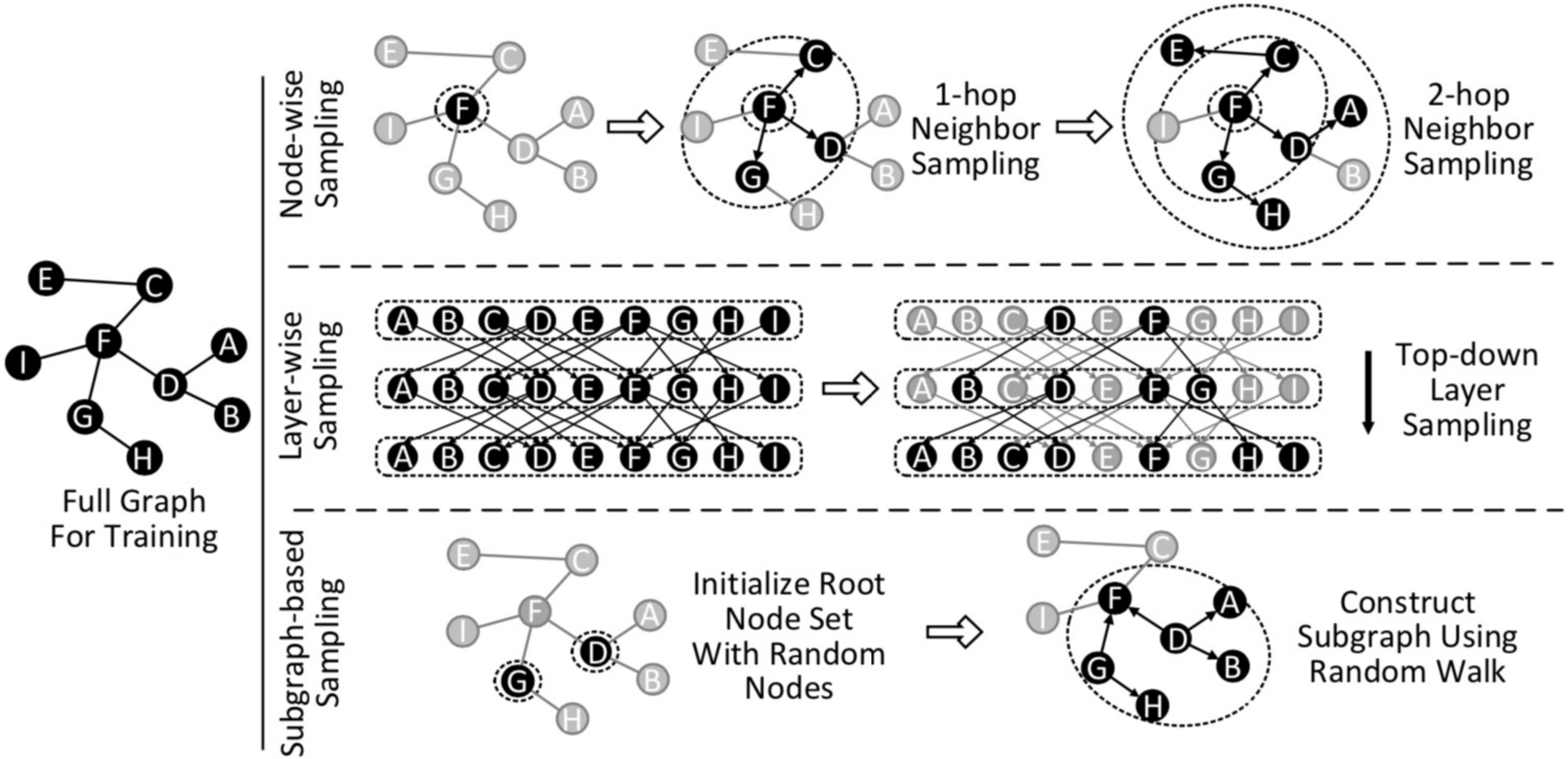


Other techniques

- Pooling
- Sampling



Sampling


Full Graph
For Training
Node-wise
Sampling
1-hop
Neighbor
Sampling
2-hop
Neighbor
Sampling
Layer-wise
Sampling
Top-down
Layer
Sampling
Subgraph-based
Sampling
Initialize Root
Node Set
With Random
Nodes
Construct
Subgraph Using
Random Walk

Pooling

- Simple pooling
  - max/min/mean/sum
- Set2Set
  - LSTM-based method
- SortPooling
  - sort and CNN
- Hierarchical pooling

Applications

- Semi-supervised learning for node-level classification
- Supervised learning for graph-level classification
- Computer Vision:
  - Recognizing semantic relationships between objects facili-tates the understanding of the meaning behind a visual scene
  - point clouds topological structure
- NLP
  - GNNs utilize the inter-relations of documents or words to inferdocument label
- Traffic
  - forecasting traffic speed, volume or thedensity of roads in traffic networks
- Recommender systems
  - link prediction
- Chemistry
  - study the graph structure of molecules/compounds

Area	Application	References
Graph Mining	Graph Matching	(Riba et al., 2018; Li et al., 2019b)
	Graph Clustering	(Zhang et al., 2019c; Ying et al., 2018b; Tsitsulin et al., 2020)
Physics	Physical Systems Modeling	(Battaglia et al., 2016; Sukhbaatar Ferguset al., 2016; Watters et al., 2017; Hoshen, 2017; Kipf et al., 2018; Sanchez et al., 2018)
Chemistry	Molecular Fingerprints	(Duvenaud et al., 2015; Kearnes et al., 2016)
	Chemical Reaction Prediction	Do et al. (2019)
Biology	Protein Interface Prediction	Fout et al. (2017)
	Side Effects Prediction	Zitnik et al. (2018)
	Disease Classification	Rhee et al. (2018)
Knowledge Graph	KB Completion	(Hamaguchi et al., 2017; Schlichtkrull et al., 2018; Shang et al., 2019)
	KG Alignment	(Wang et al., 2018b; Zhang et al., 2019d; Xu et al., 2019c)
Generation	Graph Generation	(Shchur et al., 2018b; Nowak et al., 2018; Ma et al., 2018; You et al., 2018a, 2018b; De Cao and Kipf, 2018; Li et al., 2018d; Shi et al., 2020; Liu et al., 2019; Grover et al., 2019)
Combinatorial Optimization	Combinatorial Optimization	(Khalil et al., 2017; Nowak et al., 2018; Li et al., 2018e; Kool et al., 2019; Bello et al., 2017; Vinyals et al., 2015b; Sutton and Barto, 2018; Dai et al., 2016; Gasse et al., 2019; Zheng et al., 2020a; Selsam et al., 2019; Sato et al., 2019)
Traffic Network	Traffic State Prediction	(Cui et al., 2018b; Yu et al., 2018; Zheng et al., 2020b; Guo et al., 2019)
Recommendation Systems	User-item Interaction Prediction	(van den Berg et al., 2017; Ying et al., 2018a)
	Social Recommendation	(Wu et al., 2019c; Fan et al., 2019)
Others (Structural)	Stock Market	(Matsunaga et al., 2019; Yang et al., 2019; Chen et al., 2018c; Li et al., 2020; Kim et al., 2019)
	Software Defined Networks	Rusek et al. (2019)
	AMR Graph to Text	(Song et al., 2018a; Beck et al., 2018)
Text	Text Classification	(Peng et al., 2018; Yao et al., 2019; Zhang et al., 2018d; Tai et al., 2015)
	Sequence Labeling	(Zhang et al., 2018d; Marcheggiani and Titov, 2017)
	Neural Machine Translation	(Bastings et al., 2017; Marcheggiani et al., 2018; Beck et al., 2018)
	Relation Extraction	(Miwa and Bansal, 2016; Peng et al., 2017; Song et al., 2018b; Zhang et al., 2018f)
	Event Extraction	(Nguyen and Grishman, 2018; Liu et al., 2018)
	Fact Verification	(Zhou et al., 2019; Liu et al., 2020; Zhong et al., 2020)
	Question Answering	(Song et al., 2018c; De Cao et al., 2019; Qiu et al., 2019; Tu et al., 2019; Ding et al., 2019)
	Relational Reasoning	(Santoro et al., 2017; Palm et al., 2018; Battaglia et al., 2016)
Image	Social Relationship Understanding	Wang et al. (2018c)
	Image Classification	(Garcia and Bruna, 2018; Wang et al., 2018d; Lee et al., 2018b; Kampffmeyer et al., 2019; Marino et al., 2017)
	Visual Question Answering	(Teney et al., 2017; Wang et al., 2018c; Narasimhan et al., 2018)
	Object Detection	(Hu et al., 2018; Gu et al., 2018)
	Interaction Detection	(Qi et al., 2018; Jain et al., 2016)
	Region Classification	Chen et al. (2018d)
	Semantic Segmentation	(Liang et al., 2016, 2017; Landrieu and Simonovsky, 2018; Wang et al., 2018e; Qi et al., 2017b)
Other (Non-structural)	Program Verification	(Allamanis et al., 2018; Li et al., 2016)

Open Problems

- Robustness: Adversarial attacks
- Interpretability
- Graph Pretraining
- Complex Graph Structures

Thanks!