Graph Structure Learning for Graph Neural Networks

Yu (Hugo) Chen

IEEE AIIOT 2022 Keynote Talk

June 8th, 2022

Meta

Outline

GNN Introduction

- Why Graphs?
- Graph Neural Networks: Foundations and Models

GSL Foundations

- Why Graph Structure Learning?
- Unsupervised GSL
- Supervised GSL

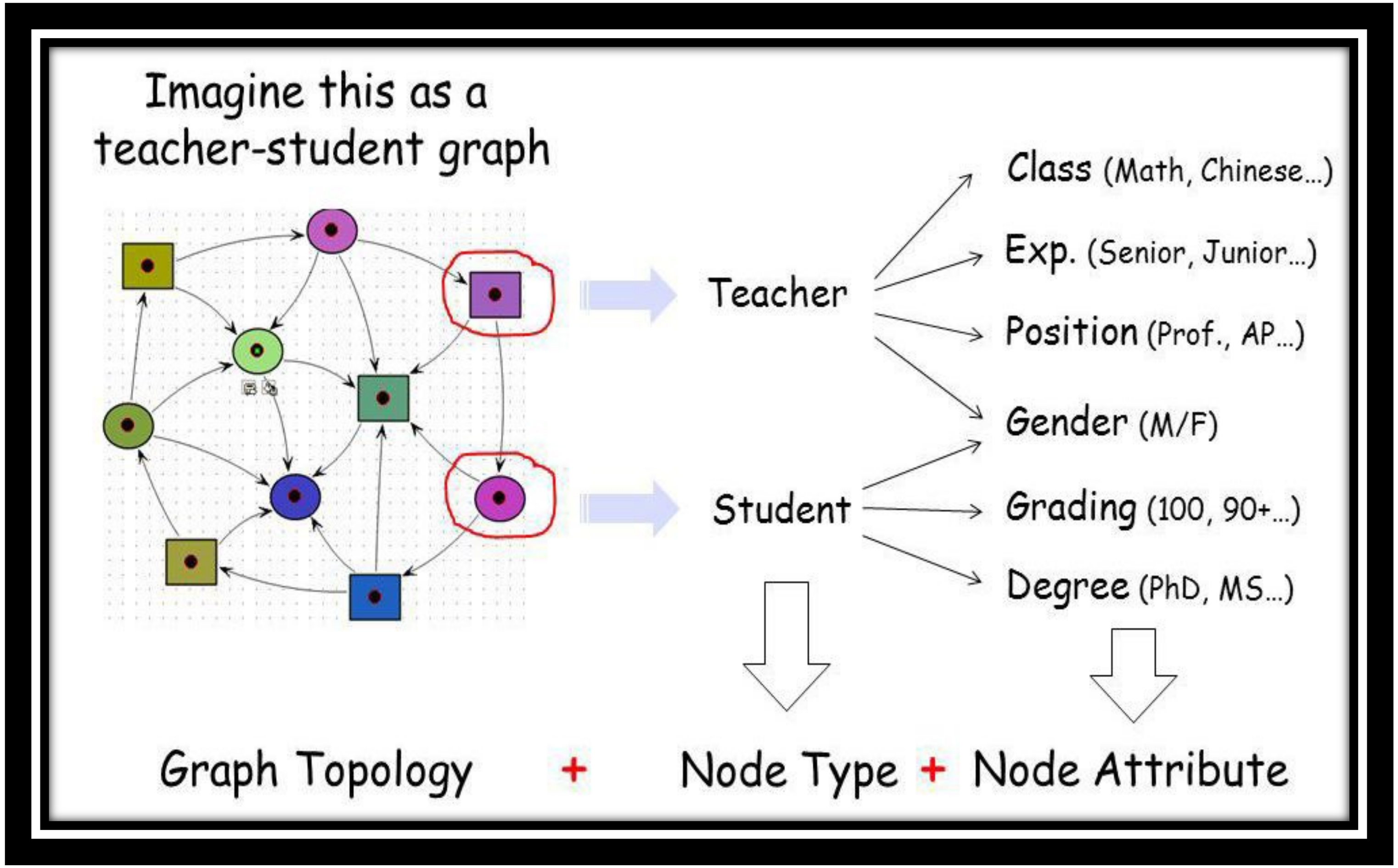
GSL4GNN

- Why GSL for GNNs?
- Learning Discrete Graph Structures for GNNs
- Learning Weighted Graph Structures for GNNs
- Connections to Other Problems
- Future Directions and Conclusions

GNN
Introduction

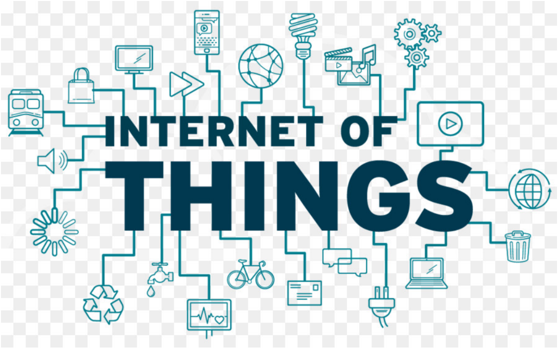
Why Graphs?

- Graphs are a general language for describing and modeling complex systems

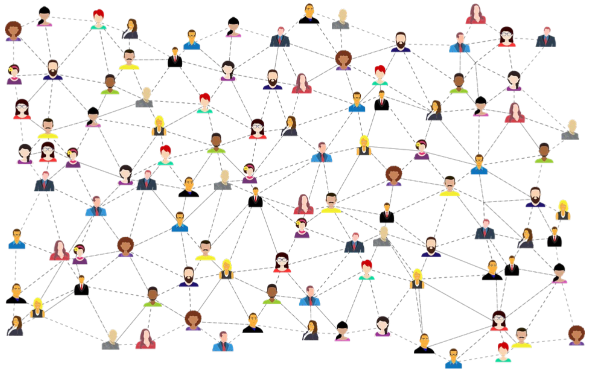


Graph = Graph Topology + Node Type + Node Attribute + Edge Type + Edge Attribute

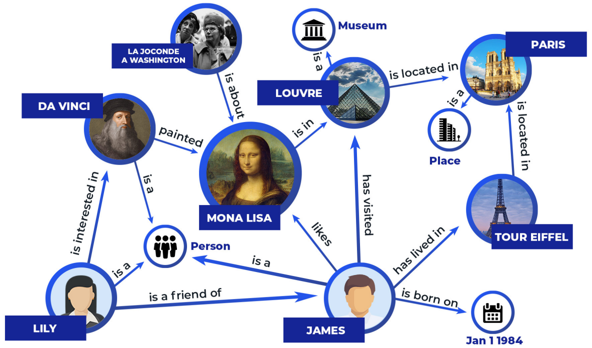
Graph-structured Data Are Ubiquitous



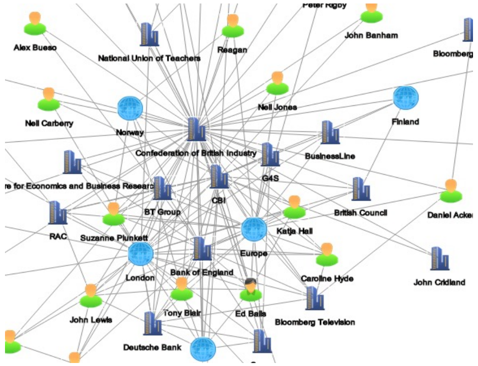
IOT graphs



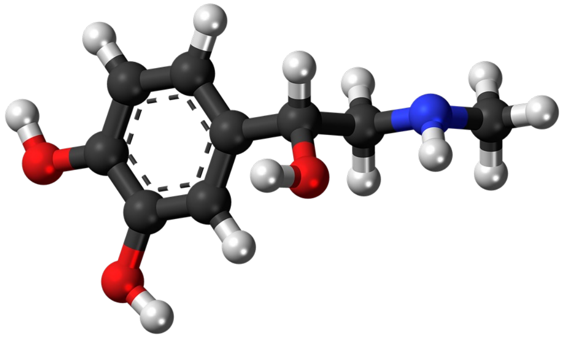
Social networks



Knowledge graphs



Financial transactions

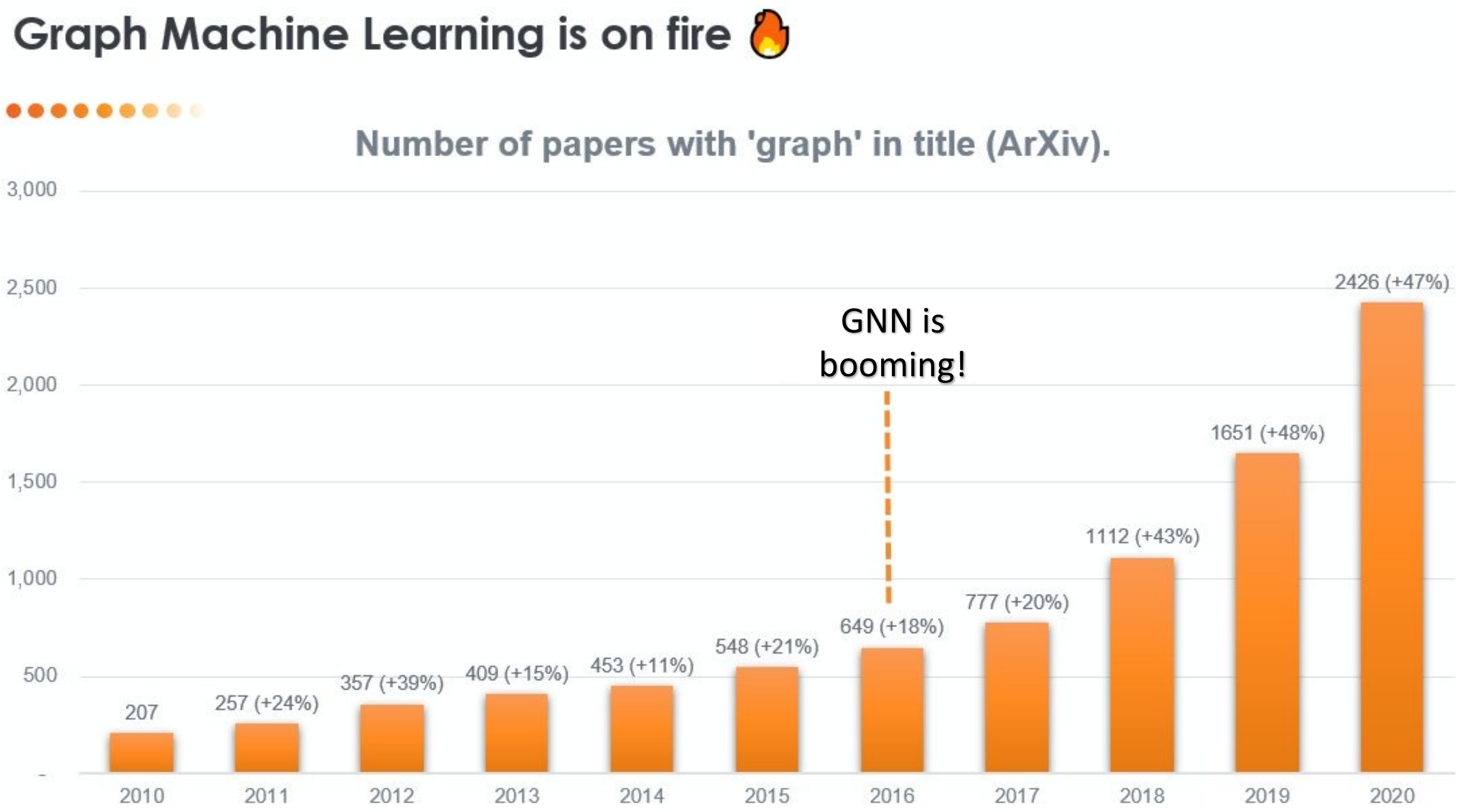


Biomedical graphs

Scene graphs

Graph = Graph Topology + Node Type + Node Attribute + Edge Type + Edge Attribute

Graph Machine Learning: Recent Trending



Graph Neural Networks: A Brief History

Year 2009

First GNN Paper

(Scarselli et al., 2009)

Year 2016, 2017

First GRU-based Modern

GNN paper: GGNN

First graph convolution-based paper GCN, Start a new era of GNNs history

Year 2017-2021

A series of Graph Convolution

(GCN), Message Passage

(MPNN, GraphSage, GIN),

Attention-based (GAT),

Unsupervised GNNs (Graph-

Autoencoder, graph-infomax)

Many new GNNs fast developed!

Year 2019-2021

GNN–based applications,

such as search, recommendation,

drug discovery, NLP, Transport…

Many open-source libraries

like DGL, Pytorch Geometric, DIG,

Graph4NLP, TorchDrug…

Year 2021

3 GNN books released simultaneously:

1) Prof. Liu (Tsinghua) et al.:

“Introduction to Graph Neural Networks”

2) Prof. Tang (MSU) et al.:

“Deep learning on graphs”

3) Prof. Hamilton (McGill):

Graph representation learning

Year 2022

Most comprehensive GNNs

Book (by Dr. L. Wu et al.):

"Graph Neural Networks:

Foundations, Frontiers,

Applications" by Springer

Machine Learning on Graphs Tasks

Classical ML tasks on graphs:

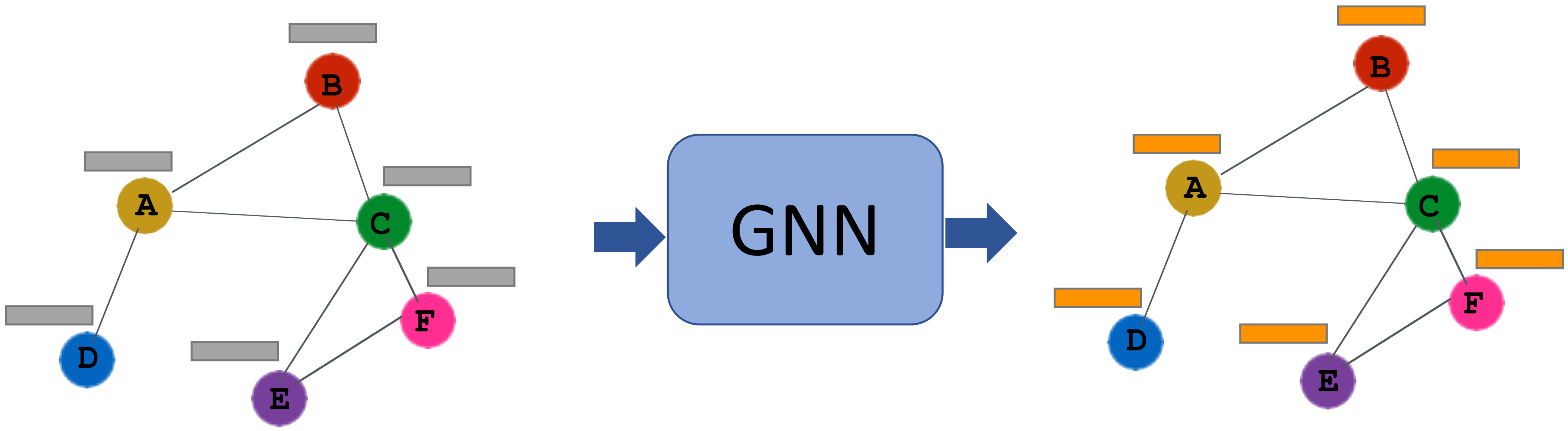
- Node classification
  - Predict a type of a given node
- Link prediction
  - Predict whether two nodes are linked
- Community detection
  - Identify densely linked clusters of nodes
- Graph similarity
  - How similar are two (sub)graphs

Recent ML tasks on graphs:

- Expressive power of GNNs
  - Theoretical understanding
- Scalability of GNNs
  - Sampling paradigms for scaling up
- Adversarial robustness of GNNs
  - Adversarial attacks and provable robustness
- Graph structure learning for GNNs
  - Learning optimal graph structures for GNNs

Wu, Lingfei, et al. "Graph Neural Networks: Foundations, Frontiers, and Applications."

Modeling Graphs with Graph Neural Networks

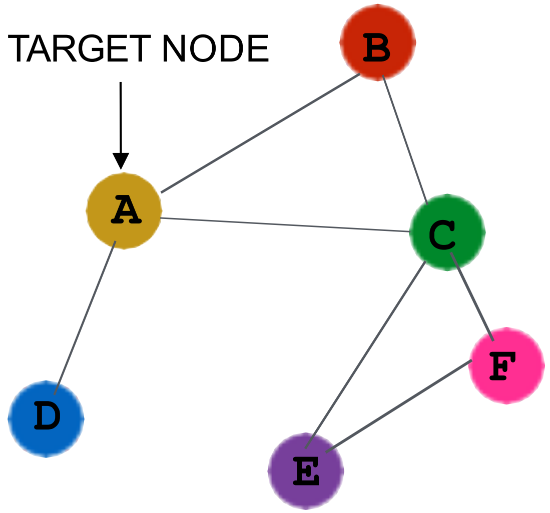


Input representations of nodes/edges

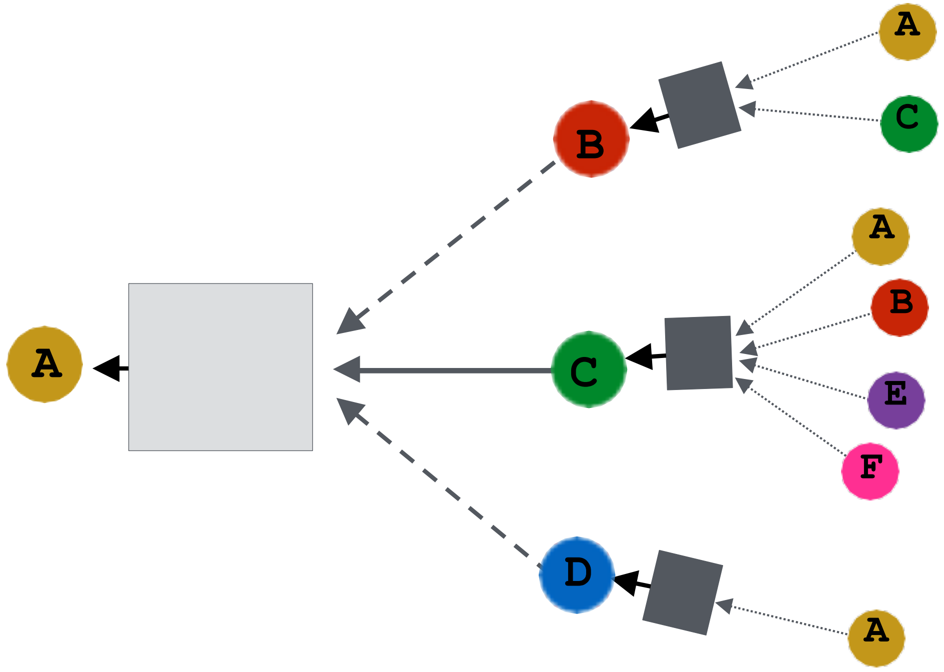
Updated representations of nodes/edges

Graph Neural Networks: Basic Model

- Key idea: Generate node embeddings based on local neighborhoods.

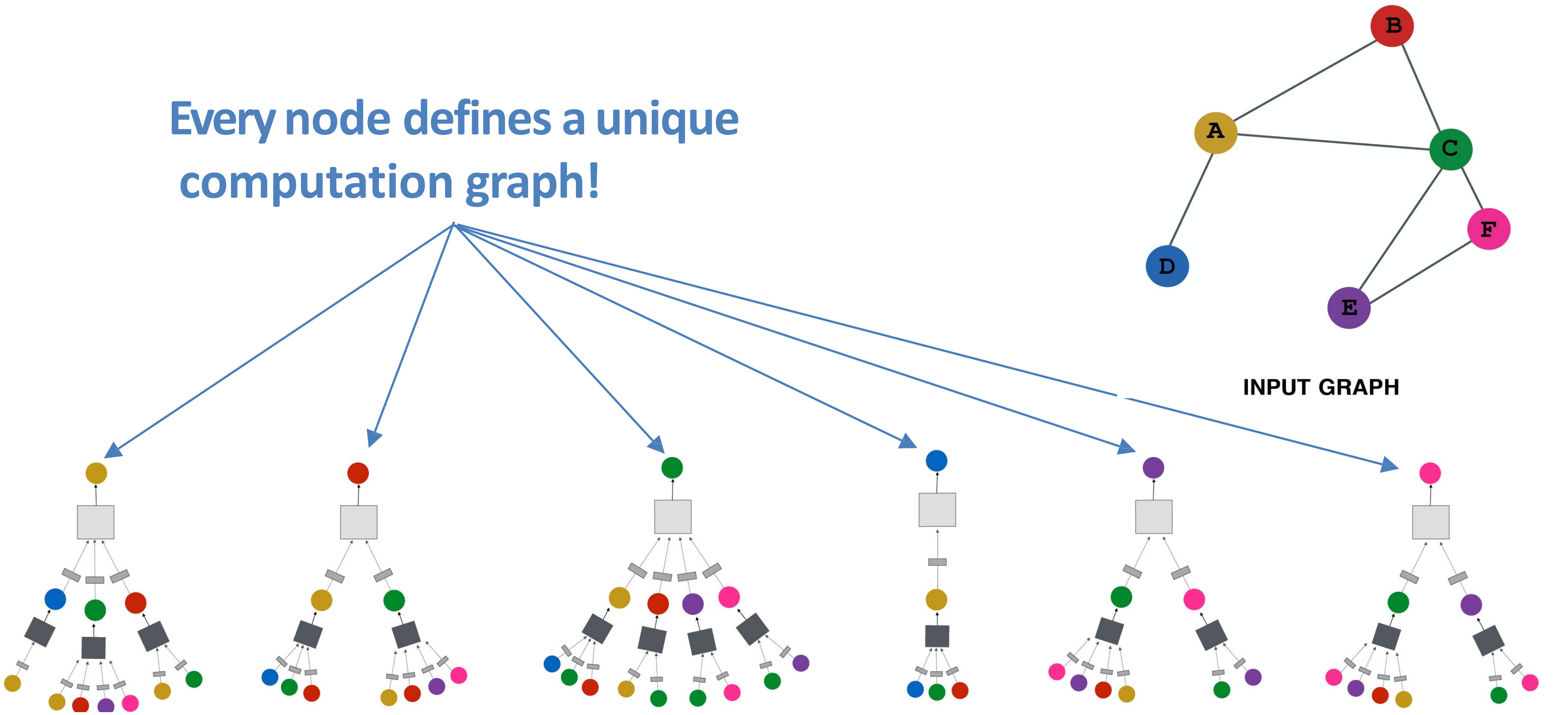


INPUT GRAPH



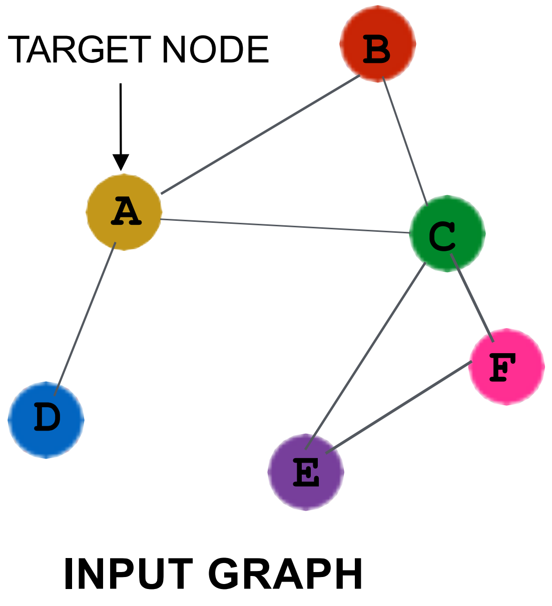
GNN Model: Neighborhood Aggregation

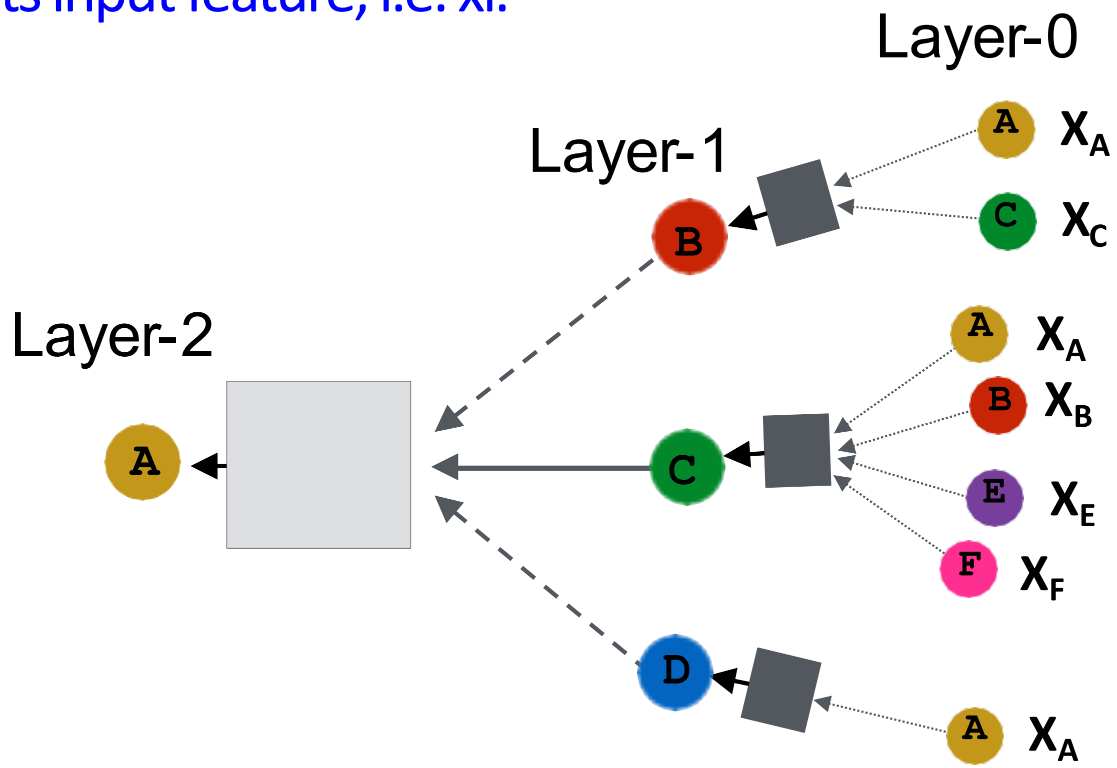
- Intuition: Network neighborhood defines a computation graph



GNN Model: Neighborhood Aggregation

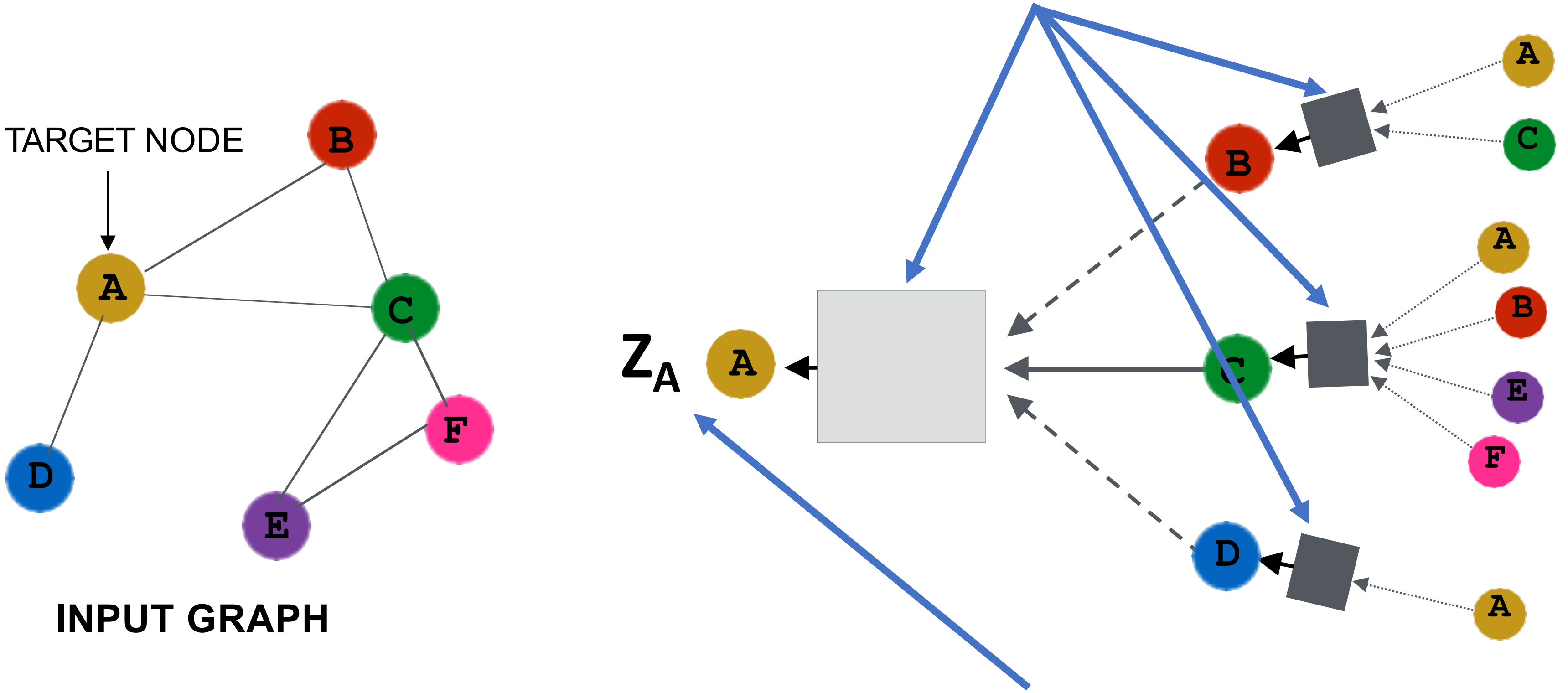
- Nodes have embeddings at each layer.
- Model can have arbitrary depth.
- “layer-0” embedding of node i is its input feature, i.e. xi.





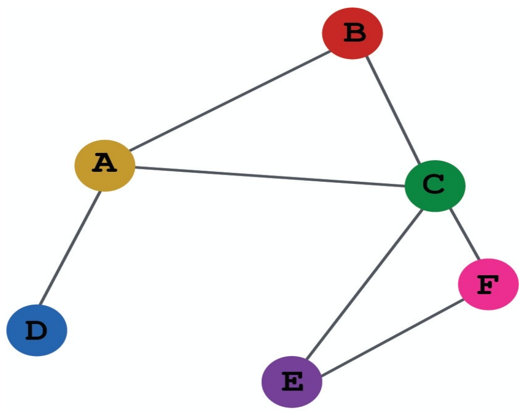
Overview of GNN Model

1) Define node aggregation and update functions



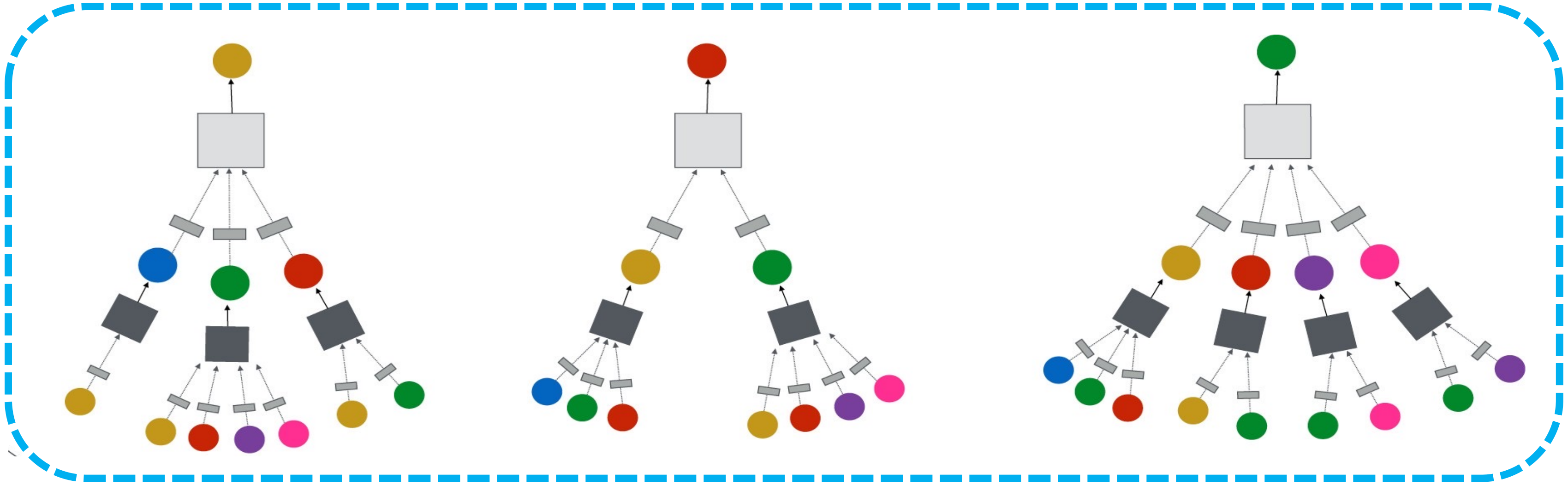
2) Define a loss function on the embeddings, **$L(z_v)$**

Overview of GNN Model



INPUT GRAPH

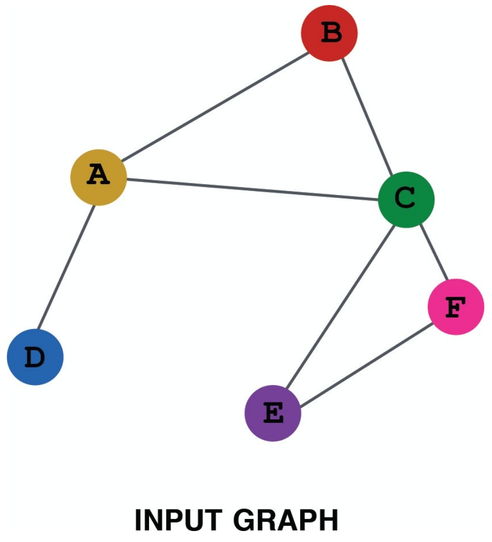
3) Train on a set of nodes, i.e., a batch of computation graphs

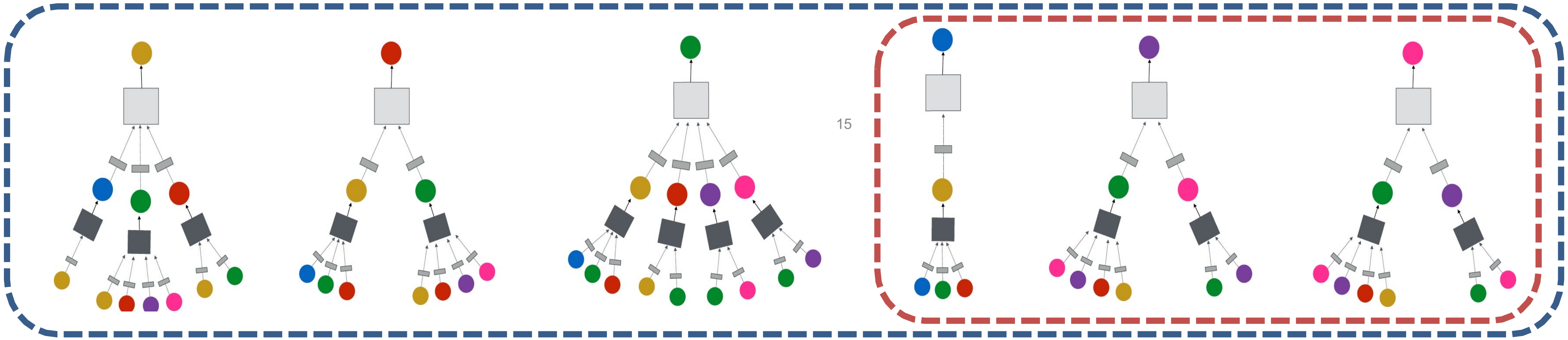


Overview of GNN Model

4) Generate embeddings for nodes as needed

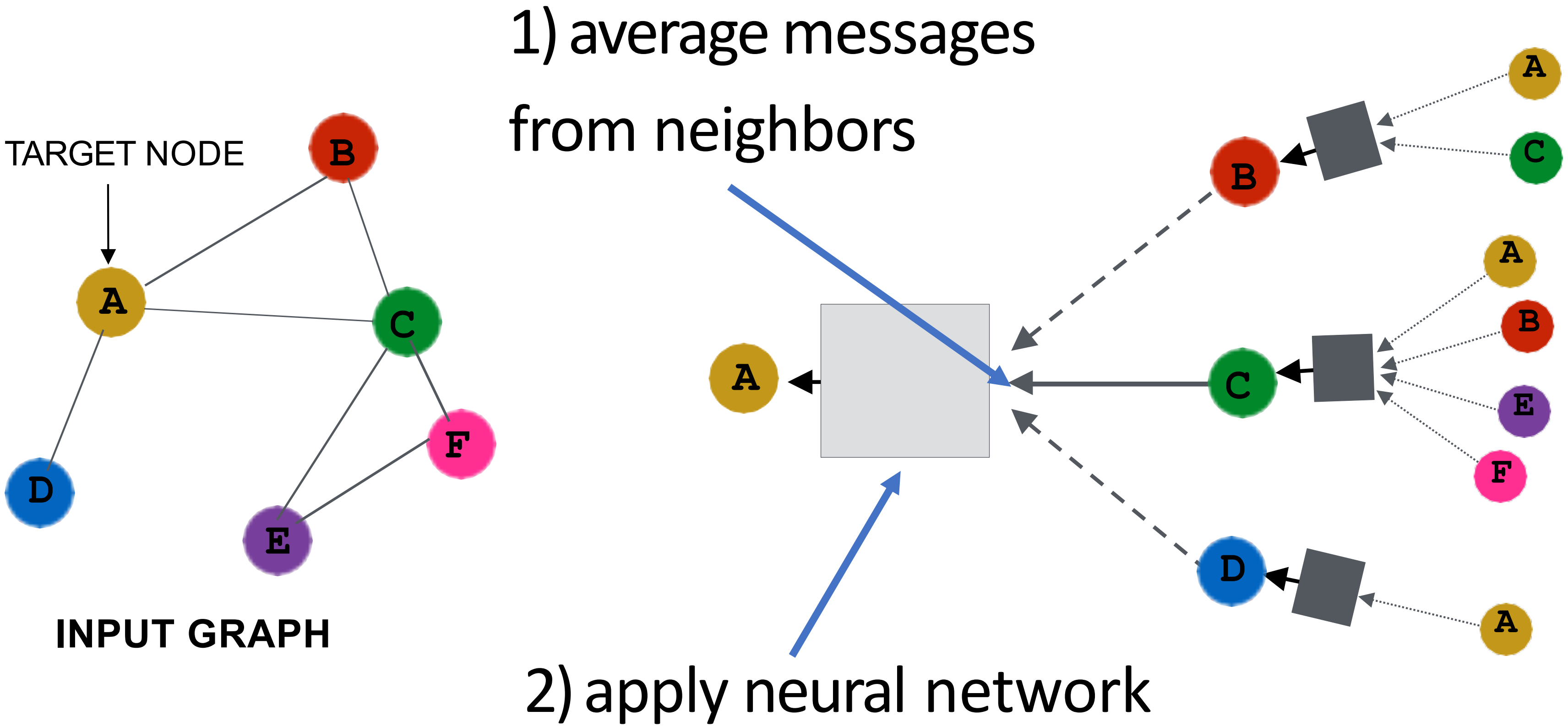
Even for nodes we never trained on!
– Inductive learning





GNN Model: A Case Study

- Basic approach: Average neighbor information and apply a neural network



GNN Model: A Case Study

- Basic approach: Average neighbor information and apply a neural network.

Initial "layer 0" embeddings are equal to node features

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

previous layer embedding of v

$$\mathbf{h}_v^k = \sigma\left(\mathbf{W}_k \sum_{u \in N(v)} \frac{\mathbf{h}_u^{k-1}}{|N(v)|} + \mathbf{B}_k \mathbf{h}_v^{k-1}\right), \quad \forall k > 0$$

kth layer embedding of v

non-linearity (e.g., ReLU or tanh)

average of neighbor's previous layer embeddings

Parameter sharing for all the nodes!

Graph Neural Networks: Foundations

- Learning node embeddings:

$$\mathbf{h}_i^{(l)} = f_{\text{filter}}(A, \mathbf{H}^{(l-1)})$$

A graph filter (f_{filter}); adjacency matrix (A); Updated node embeddings ($\mathbf{h}_i^{(l)}$); Input node embeddings ($\mathbf{H}^{(l-1)}$)

$f_{\text{filter}}(\cdot, \cdot)$:
- Spectral-based
- Spatial-based
- Attention-based
- Recurrent-based

- Learning graph-level embeddings:

$$A', \mathbf{H}' = f_{\text{pool}}(A, \mathbf{H})$$

A small graph w/ fewer nodes (A'); New node embeddings ($\mathbf{H}'$); Input graph (A); Input node embeddings ($\mathbf{H}$)

$f_{\text{pool}}(\cdot, \cdot)$:
- Flat Graph Pooling (i.e. Max, Ave, Min)
- Hierarchical Graph Pooling (i.e. Diffpool)

Graph Neural Networks: Popular Models

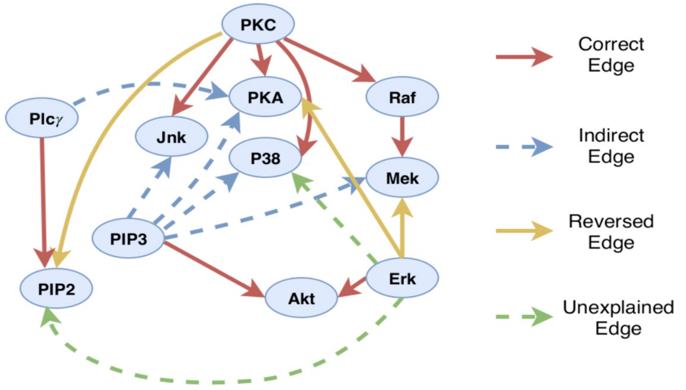
- Spectral-based Graph Filters
  - GCN (Kipf & Welling, ICLR 2017), Chebyshev-GNN (Defferrard et al. NIPS 2016)
- Spatial-based Graph Filters
  - MPNN (Gilmer et al. ICML 2017), GraphSage (Hamilton et al. NIPS 2017)
  - GIN (Xu et al. ICLR 2019)
- Attention-based Graph Filters
  - GAT (Velickovic et al. ICLR 2018)
- Recurrent-based Graph Filters
  - GGNN (Li et al. ICLR 2016)

GNN Model: Quick Summary

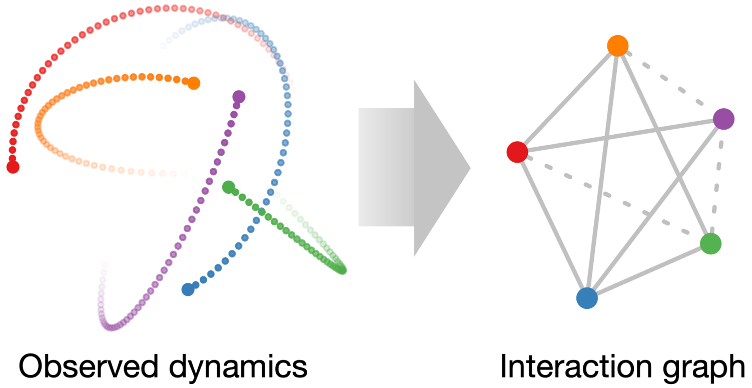
- Key idea: generate node embeddings by aggregating neighborhood information.
  - Allows for parameter sharing in the encoder
  - Allows for inductive learning

GSL
Foundations

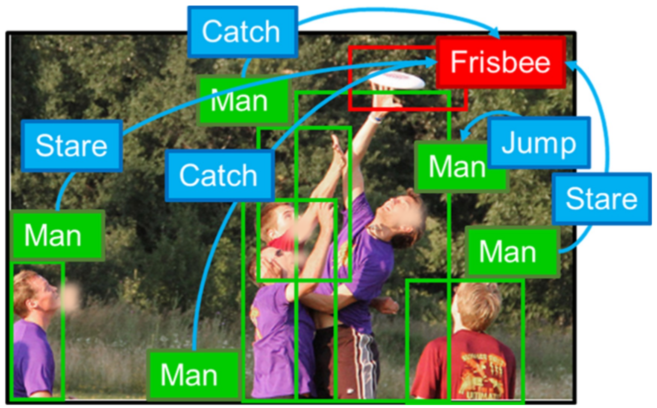
Why Graph Structure Learning?



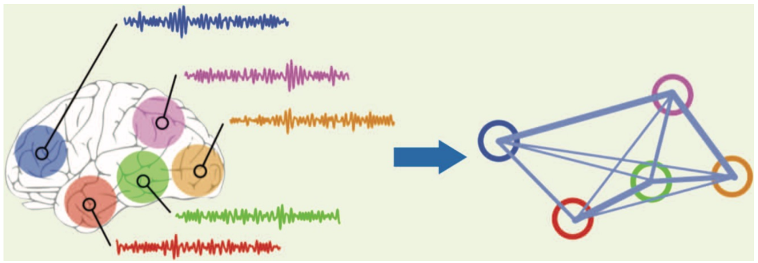
Estimating protein signaling network (Yu et al., ICML 2019)



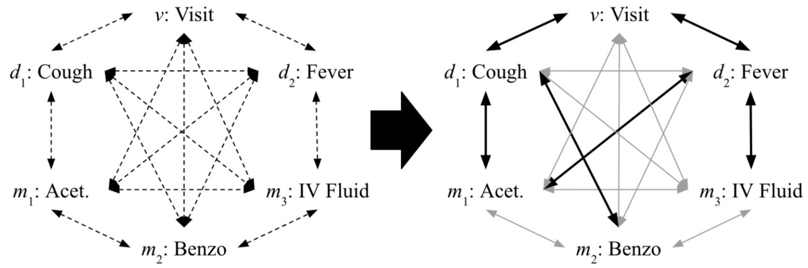
Relational inference for interacting systems (Kipf et al., ICML 2018)



Learning relationships among visual objects (Zhu et al., Multimedia Tools and Applications 2020)

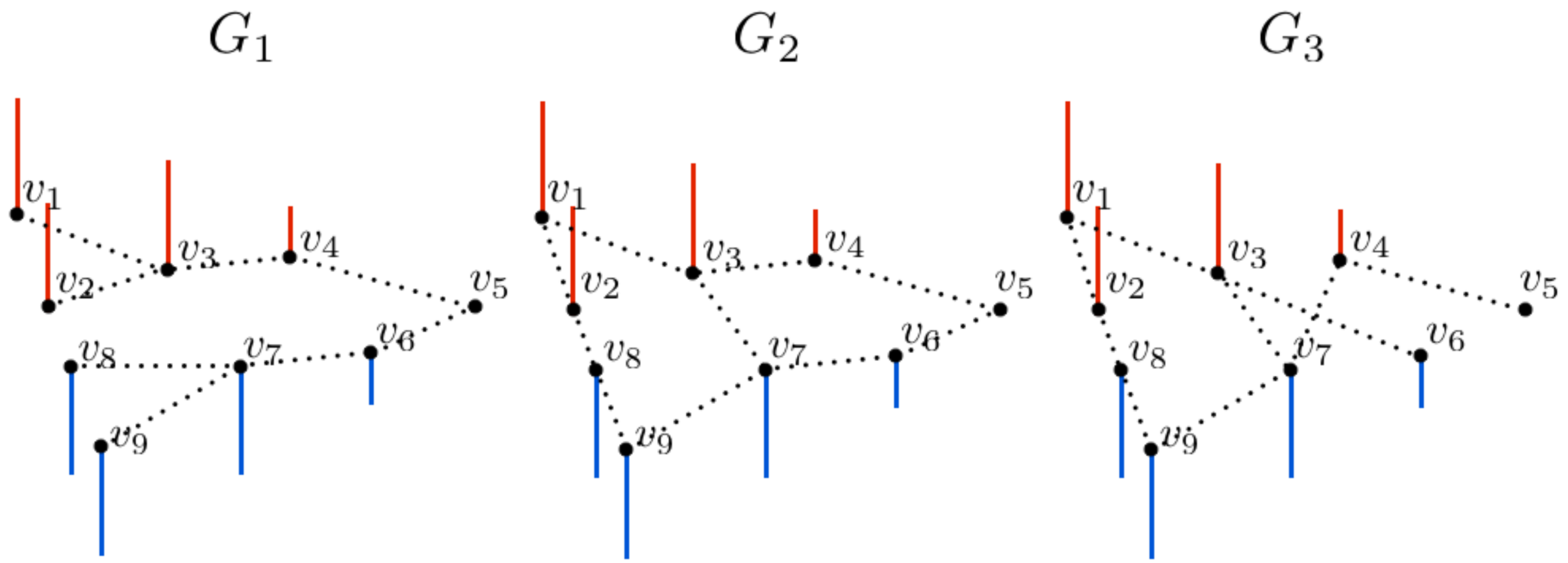


Inferring functional connectivity between different brain regions (Dong et al., IEEE Signal Processing Magazine 2019)



Learning the graphical structure of electronic health records (Choi et al., AAAI 2020)

Unsupervised GSL from Smooth Signals



Signals residing on graphs, graph G1 has the best smoothness property.

Dong et al. "Learning Laplacian Matrix in Smooth Graph Signal Representations". arXiv 2014.

Unsupervised GSL from Smooth Signals: Fitness

Node feature reconstruction using neighboring node features

$$\sum_i \boxed{||\mathbf{X}_i - \sum_j A_{i,j}\mathbf{X}_j||^2}$$

$$where \sum_j A_{i,j} = 1, \; A_{i,j} \geq 0$$

OR

Weighted sum of the squared distance from each node to the weighted average of its neighbors

$$\sum_i \boxed{||D_{i,i}\mathbf{X}_i - \sum_j A_{i,j}\mathbf{X}_j||^2} = ||\mathbf{LX}||_F^2$$

$$where \; D_{i,i} = \sum_j A_{i,j}$$

Wang et al. "Label propagation through linear neighborhoods". IEEE Transactions on Knowledge and Data Engineering 2007.

Daitch et al. "Fitting a graph to vector data". ICML 2009.

Unsupervised GSL from Smooth Signals: Smoothness

$$\Omega(\mathbf{A}, \mathbf{X}) = \frac{1}{2} \sum_{i,j} \boxed{A_{i,j} ||\mathbf{X}_i - \mathbf{X}_j||^2} = \text{tr}(\mathbf{X}^\top \mathbf{L} \mathbf{X})$$

Forcing neighboring vertices to have similar features

Belkin et al. "Laplacian eigenmaps and spectral techniques for embedding and clustering". NIPS 2002.

Unsupervised GSL from Smooth Signals: Connectivity and Sparsity

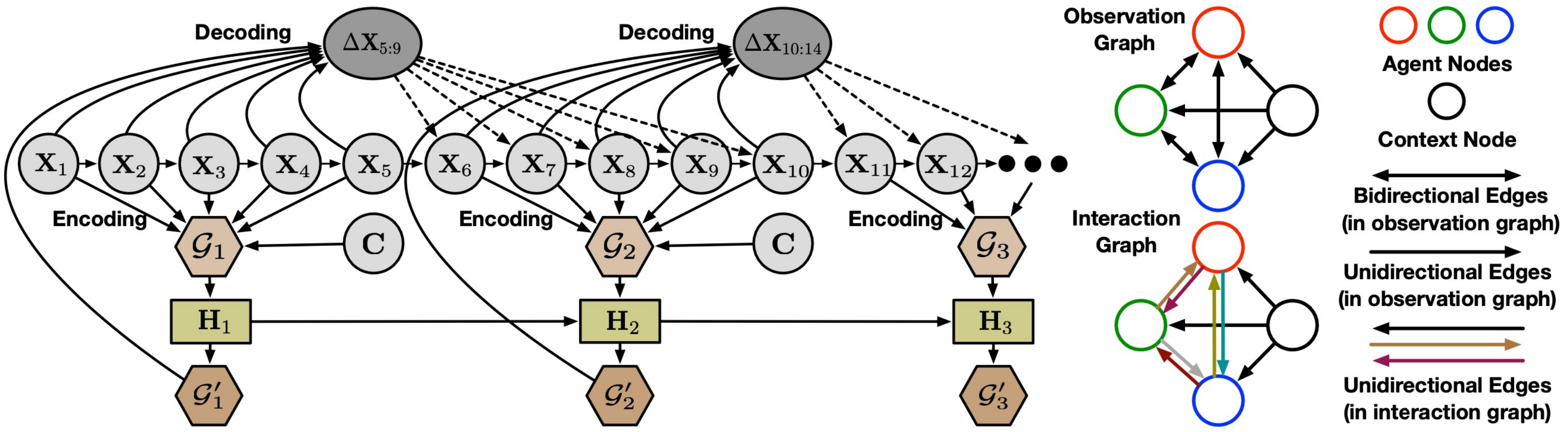
$$\underbrace{-\alpha \vec{1}^\top \log(\mathbf{A}\vec{1})}_{\text{Connectivity}} + \underbrace{\beta ||\mathbf{A}||_F^2}_{\text{Sparsity}}$$

Connectivity

Sparsity

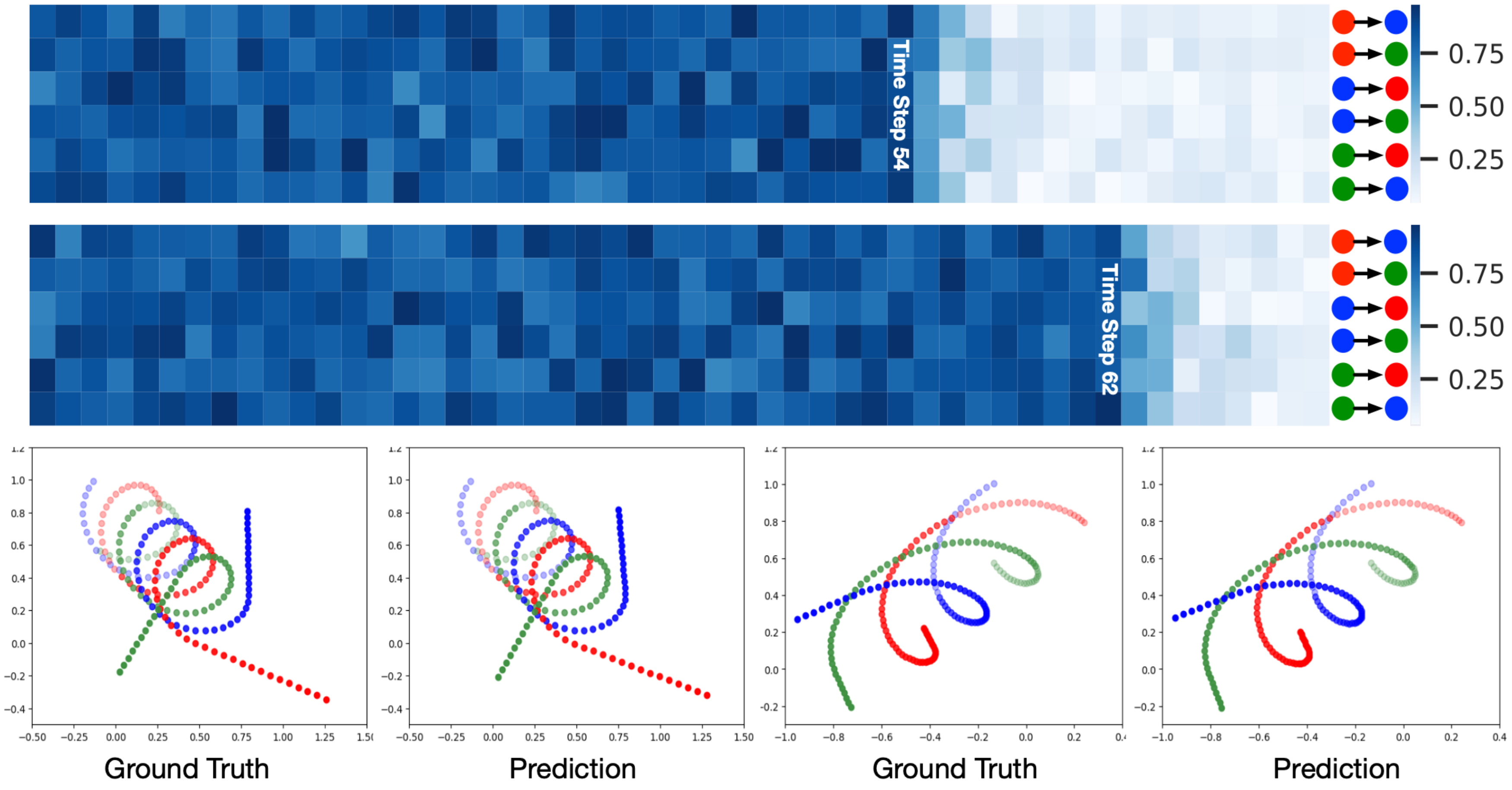
Kalofolias et al. "How to learn a graph from smooth signals". AISTATS 2016.

Supervised GSL for Interacting Systems [Li et al., NeurIPS 2020]



Li et al. "Evolvegraph: Multi-agent trajectory prediction with dynamic relational reasoning". NeurIPS 2020.

Supervised GSL for Interacting Systems [Li et al., NeurIPS 2020]

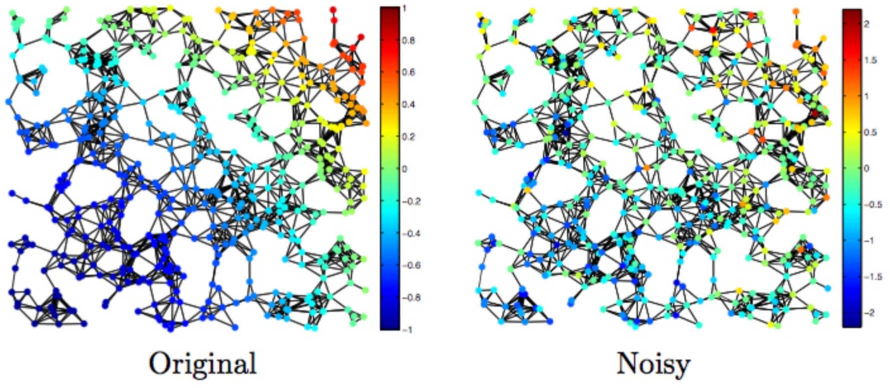


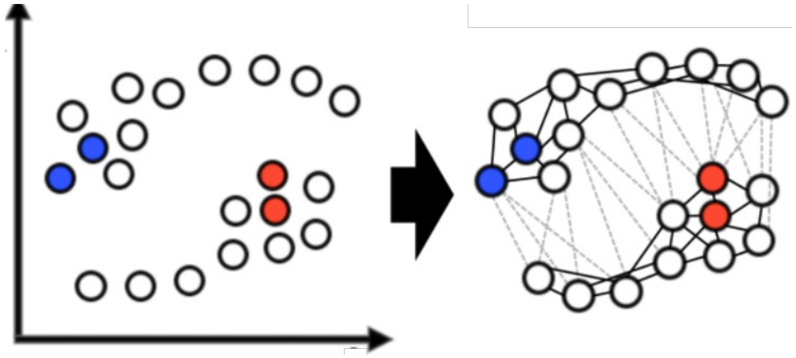
Visualization of latent interaction graph evolution and particle trajectories.

GSL4GNN Foundations

Why GSL for GNNs?

- GNNs are powerful, unfortunately, it requires graph-structured data available.
- Questionable if the given intrinsic graph-structures are optimal (i.e., noisy, incomplete, etc.) for downstream tasks.
- Many applications (e.g., NLP tasks) may only have non-graph structured data or even just the original feature matrix, requiring additional graph construction.

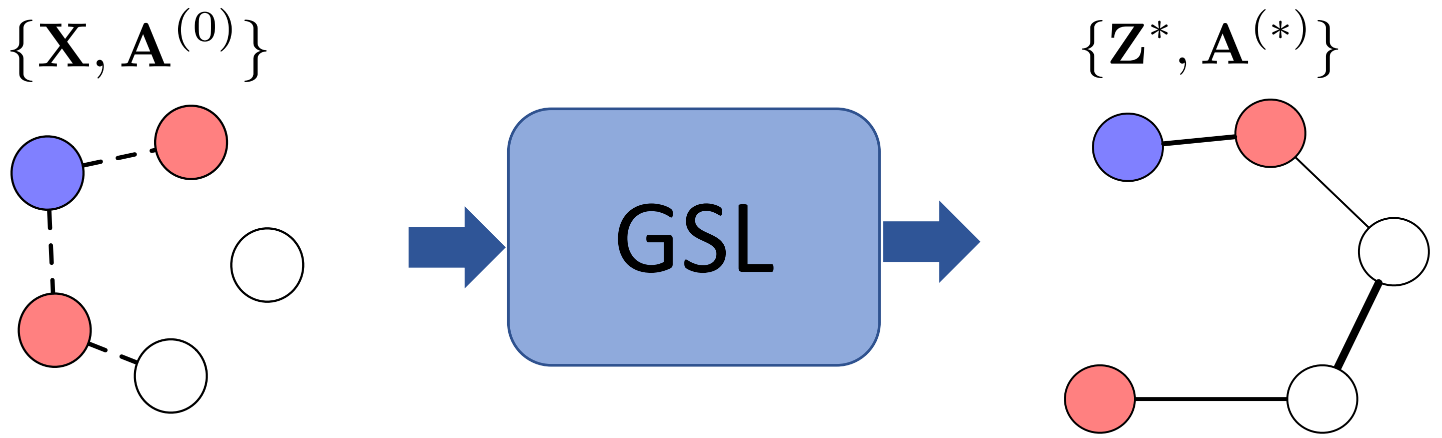




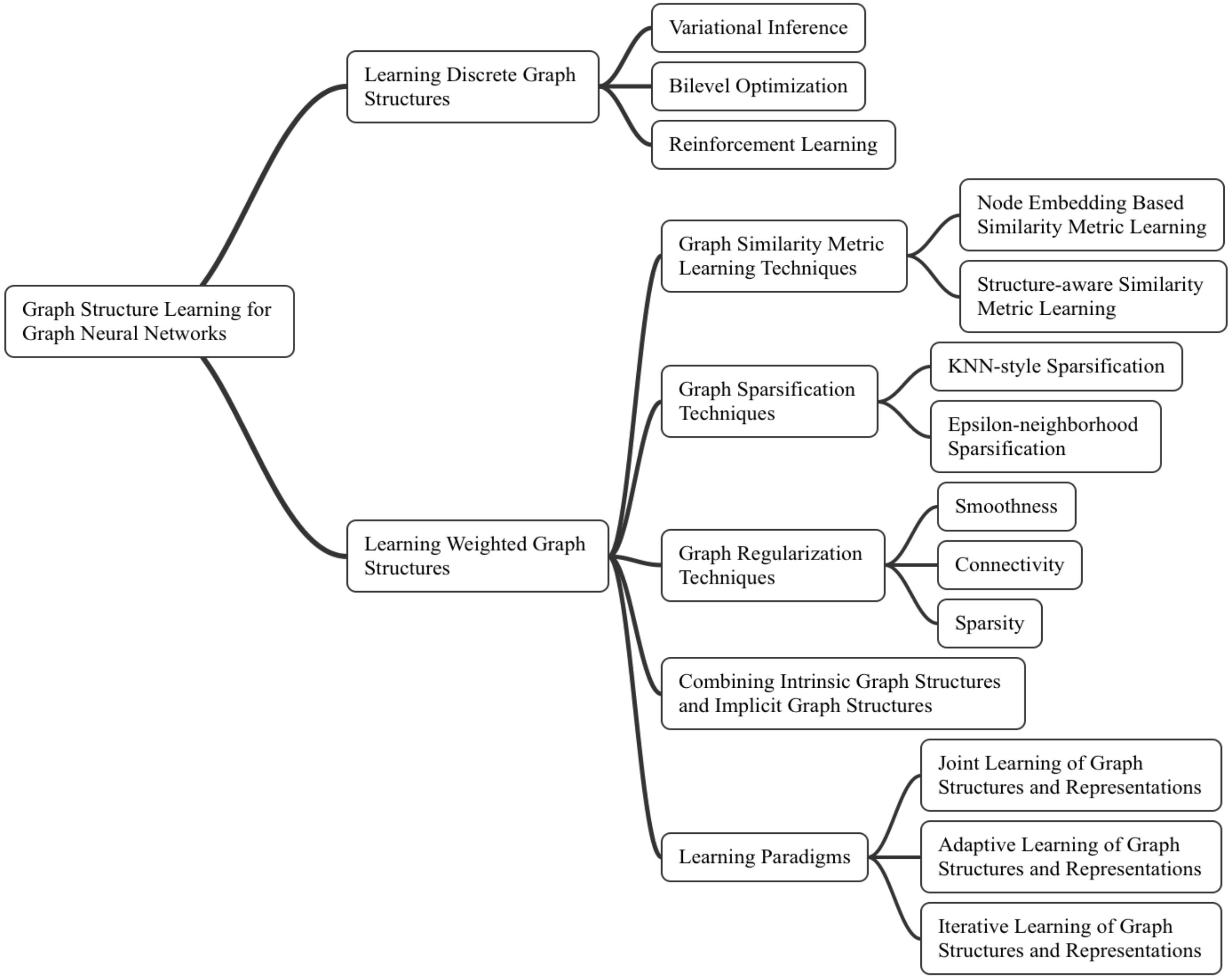
GSL4GNN Formulation

Input: a set of n nodes associated with a feature matrix $\mathbf{X} \in \mathbb{R}^{d \times n}$ and an (optional and potentially noisy) initial adjacency matrix $\mathbf{A}^{(0)} \in \mathbb{R}^{n \times n}$.

Output: an optimized adjacency matrix $\mathbf{A}^{(*)} \in \mathbb{R}^{n \times n}$ and node embedding matrix $\mathbf{Z}^{*} \in \mathbb{R}^{d' \times n}$ with respect to downstream task (i.e., task-dependent loss).



GSL4GNN Roadmap

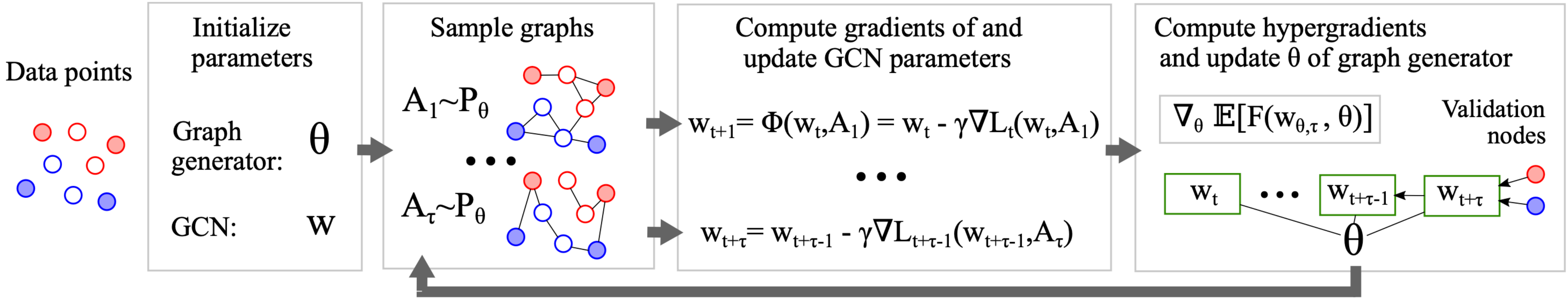


Learning Discrete Graph Structures for GNNs

Learning Discrete Graph Structures for GNNs

- Sampling a discrete graph structure from learned probabilistic adjacency matrix.
- Joint graph structure and GNN parameters optimization (non-differentiable, intractable to solve exactly) via
  - Variational inference
  - Bilevel optimization
  - Reinforcement Learning
- Non-trivial to extend to inductive learning setting.

Bilevel Optimization for GSL [Franceschi et al., ICML 2019]



$$\min_{\theta \in \overline{\mathcal{H}}_N} \mathbb{E}_{\mathbf{A} \sim \mathrm{Ber}(\theta)} [F(\mathbf{w}_\theta, \mathbf{A})]$$

Outer objective for graph structure optimization

$$\textit{such that } \mathbf{w}_\theta = \mathrm{argmin}_{\mathbf{w}} \; \mathbb{E}_{\mathbf{A} \sim \mathrm{Ber}(\theta)} [L(\mathbf{w}, \mathbf{A})]$$

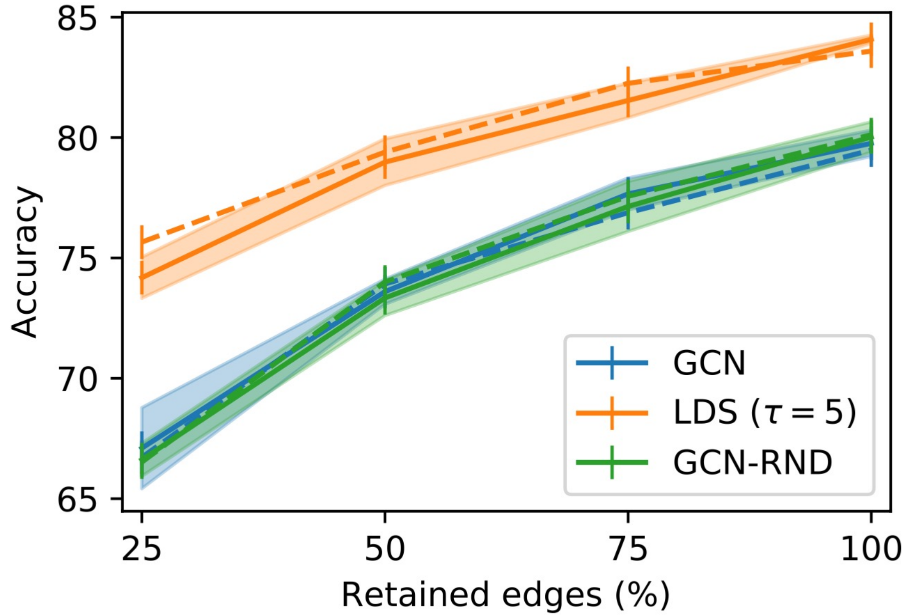
Inner objective for GNN parameters optimization

Franceschi et al. “Learning Discrete Structures for Graph Neural Networks”. ICML 2019.

Bilevel Optimization for GSL [Franceschi et al., ICML 2019]

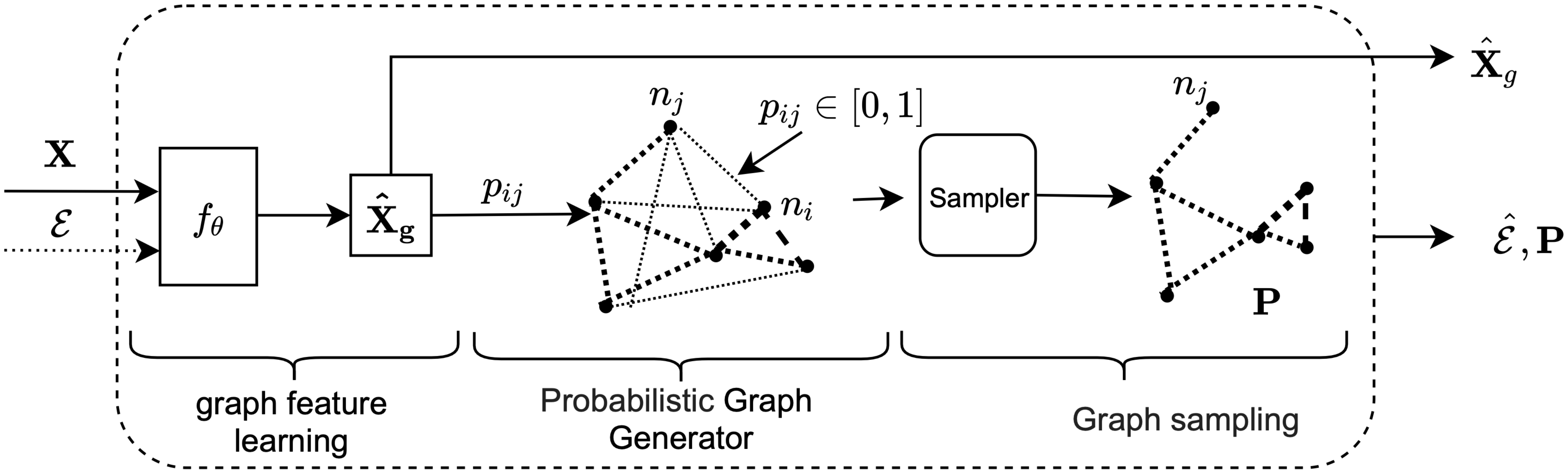
	Wine	Cancer	Digits	Citeseer	Cora	20news	FMA
LogReg	92.1 (1.3)	**93.3 (0.5)**	85.5 (1.5)	62.2 (0.0)	60.8 (0.0)	42.7 (1.7)	37.3 (0.7)
Linear SVM	93.9 (1.6)	**90.6 (4.5)**	87.1 (1.8)	58.3 (0.0)	58.9 (0.0)	40.3 (1.4)	35.7 (1.5)
RBF SVM	**94.1 (2.9)**	**91.7 (3.1)**	86.9 (3.2)	60.2 (0.0)	59.7 (0.0)	41.0 (1.1)	**38.3 (1.0)**
RF	93.7 (1.6)	**92.1 (1.7)**	83.1 (2.6)	60.7 (0.7)	58.7 (0.4)	40.0 (1.1)	**37.9 (0.6)**
FFNN	89.7 (1.9)	**92.9 (1.2)**	36.3 (10.3)	56.7 (1.7)	56.1 (1.6)	38.6 (1.4)	33.2 (1.3)
LP	89.8 (3.7)	76.6 (0.5)	**91.9 (3.1)**	23.2 (6.7)	37.8 (0.2)	35.3 (0.9)	14.1 (2.1)
ManiReg	90.5 (0.1)	81.8 (0.1)	83.9 (0.1)	67.7 (1.6)	62.3 (0.9)	**46.6 (1.5)**	34.2 (1.1)
SemiEmb	91.9 (0.1)	89.7 (0.1)	**90.9 (0.1)**	68.1 (0.1)	63.1 (0.1)	**46.9 (0.1)**	34.1 (1.9)
Sparse-GCN	63.5 (6.6)	72.5 (2.9)	13.4 (1.5)	33.1 (0.9)	30.6 (2.1)	24.7 (1.2)	23.4 (1.4)
Dense-GCN	90.6 (2.8)	90.5 (2.7)	35.6 (21.8)	58.4 (1.1)	59.1 (0.6)	40.1 (1.5)	34.5 (0.9)
RBF-GCN	90.6 (2.3)	**92.6 (2.2)**	70.8 (5.5)	58.1 (1.2)	57.1 (1.9)	39.3 (1.4)	33.7 (1.4)
kNN-GCN	93.2 (3.1)	**93.8 (1.4)**	**91.3 (0.5)**	68.3 (1.3)	66.5 (0.4)	41.3 (0.6)	**37.8 (0.9)**
kNN-LDS (dense)	**97.5 (1.2)**	**94.9 (0.5)**	**92.1 (0.7)**	**70.9 (1.3)**	**70.9 (1.1)**	**45.6 (2.2)**	**38.6 (0.6)**
kNN-LDS	**97.3 (0.4)**	**94.4 (1.9)**	**92.5 (0.7)**	**71.5 (1.1)**	**71.5 (0.8)**	**46.4 (1.6)**	**39.7 (1.4)**

Test accuracy (in percentage) on various node classification datasets.



Val/test accuracy (in percent) for the edge deletion scenarios on Cora.

Reinforcement Learning for GSL [Kazi et al., arXiv 2020]



Graph generator:

$$p_{i,j} = e^{-t||\mathbf{X}_i - \mathbf{X}_j||}$$

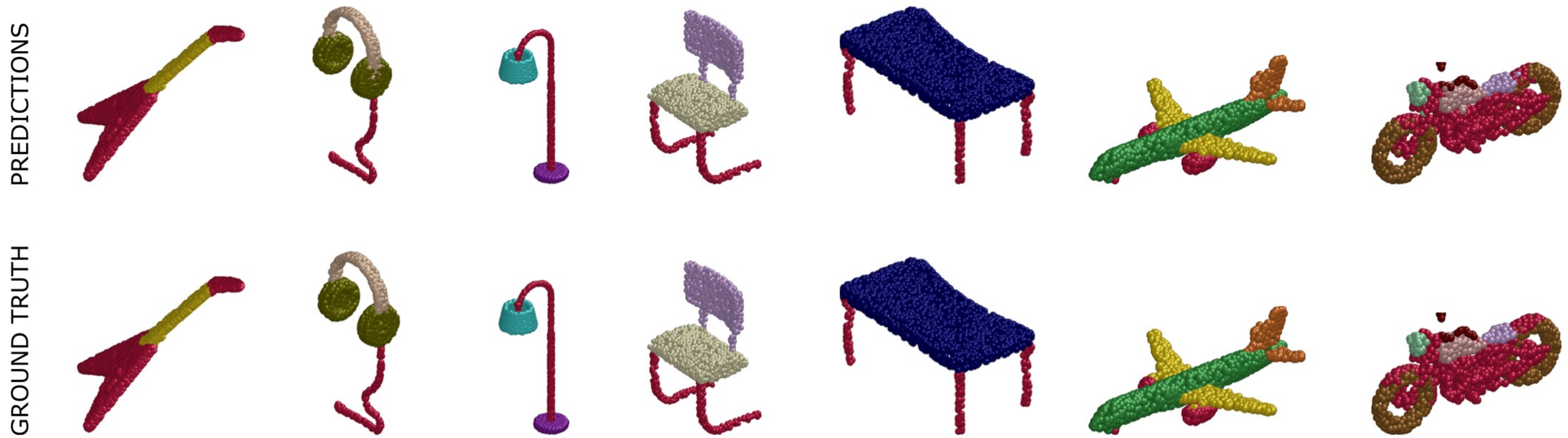
RL reward:

$$L_{graph} = \sum_{\alpha}^{Classes} \sum_{i \in \alpha} \delta_\alpha(y_i, \tilde{y}_i) \prod_{l=1}^{L} \prod_{j:(i,j) \in \hat{\mathcal{E}}^{(l)}} p_{ij}^{(l)},$$

$$\delta_\alpha(y_i, \tilde{y}_i) = \begin{cases} \mathrm{acc}_\alpha - 1 & \text{if } y_i = \tilde{y}_i \\ \mathrm{acc}_\alpha & \text{otherwise} \end{cases}$$

Kazi et al. "Differentiable Graph Module (DGM) for Graph Convolutional Networks". arXiv 2020.

Reinforcement Learning for GSL [Kazi et al., arXiv 2020]



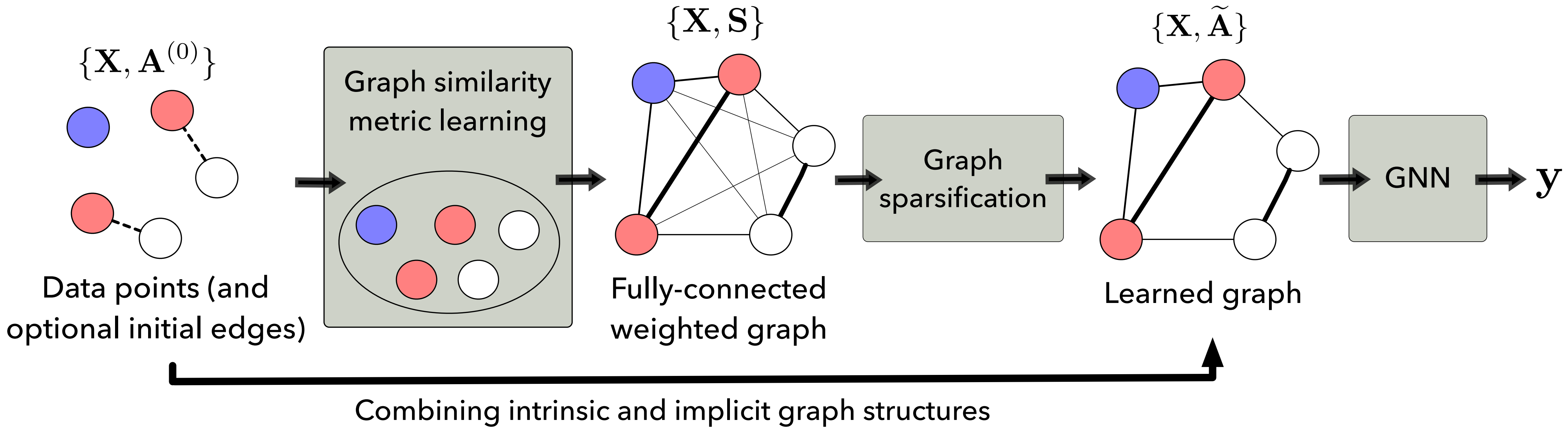
Point cloud segmentation results on ShapeNet dataset.

Learning Weighted Graph Structures for GNNs

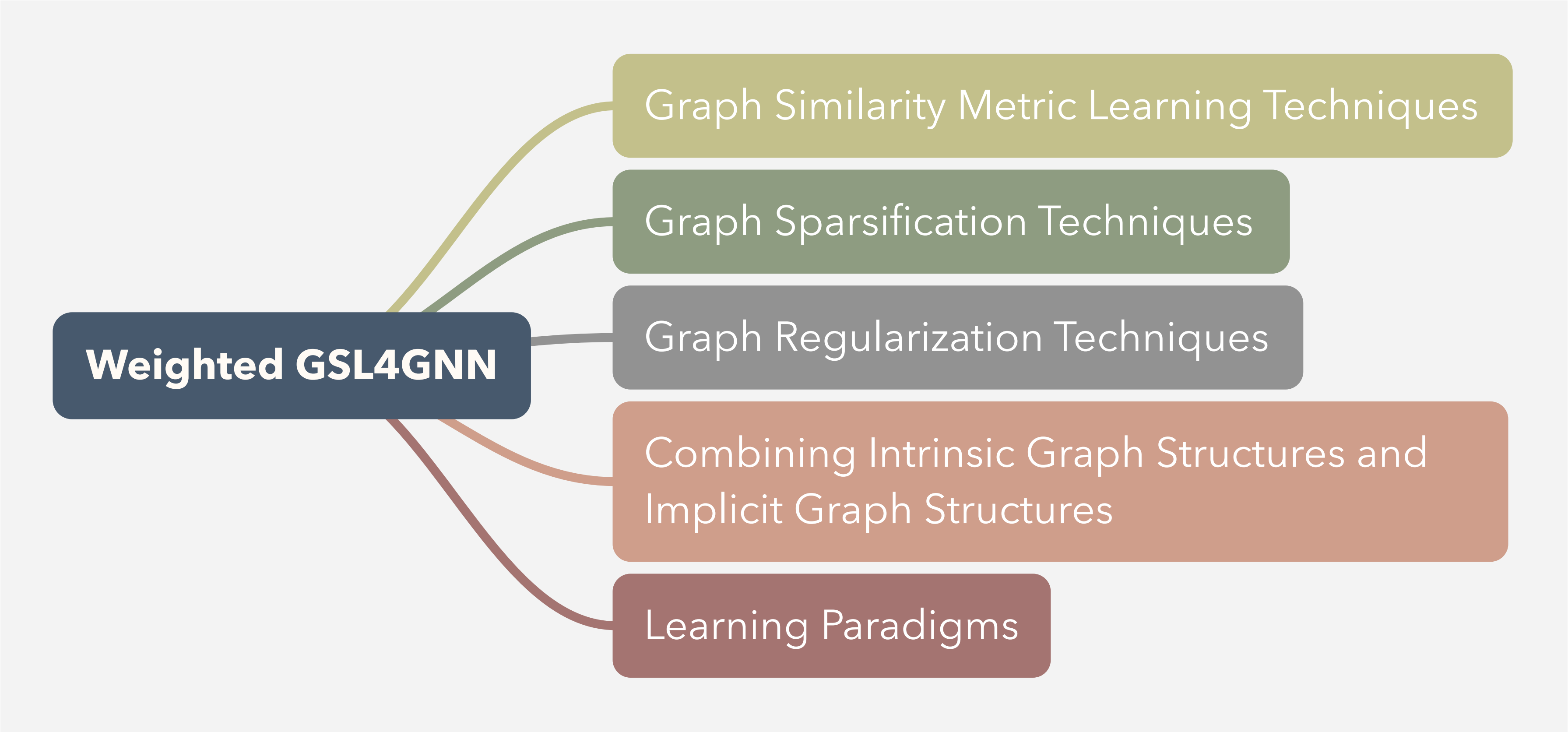
Learning Weighted Graph Structures for GNNs

- Learning a weighted adjacency matrix to represent graph structure.
- Joint graph structure and GNN parameters optimization (differentiable, more tractable) via SGD techniques.
- Weighted adjacency matrix captures richer information.
- Handling both transductive and inductive learning settings.

Weighted GSL4GNN Overview

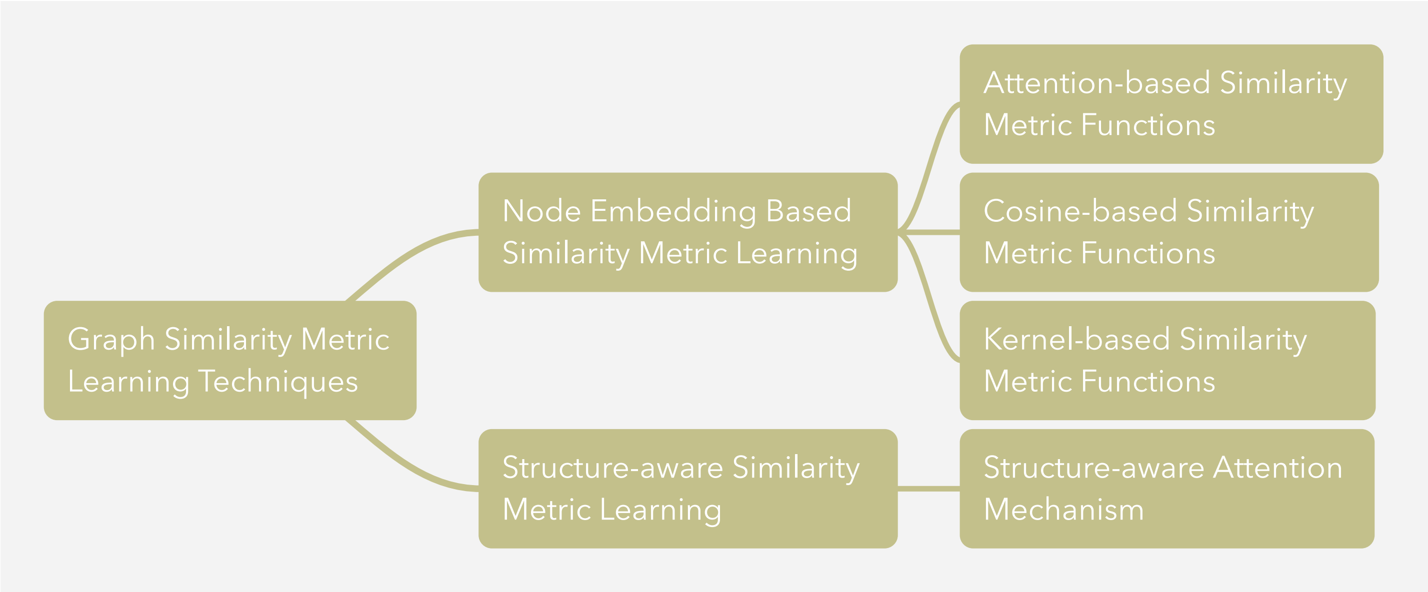


Weighted GSL4GNN Outline



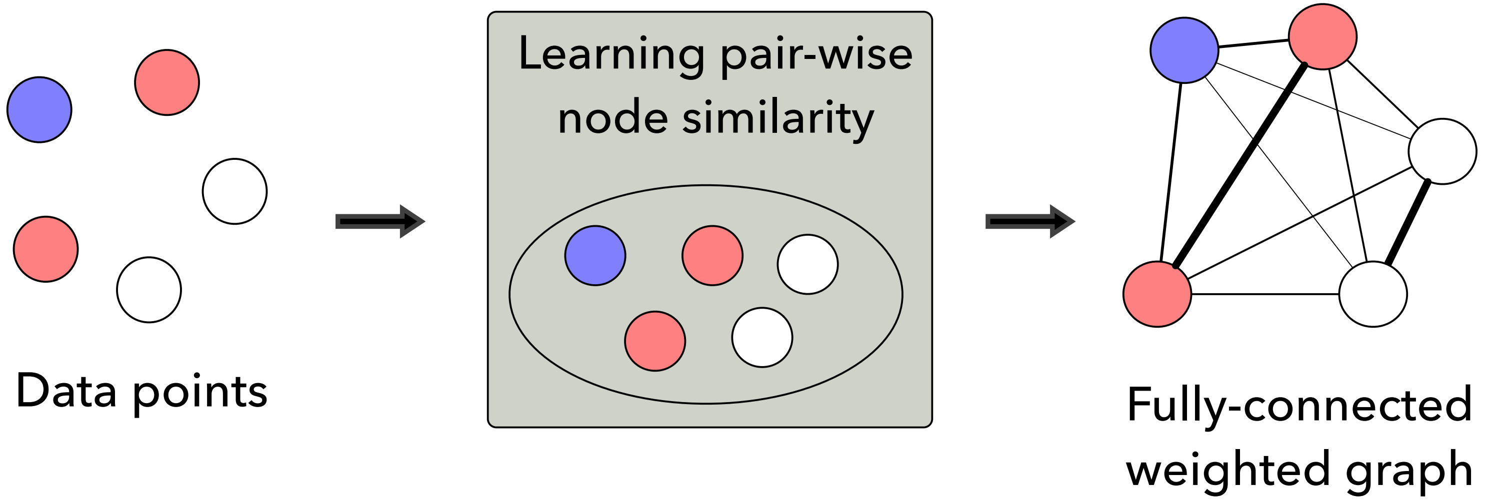
Graph Similarity Metric Learning Techniques

- Graph structure learning as similarity metric learning (in the node embedding space)
- Enabling inductive learning
- Various metric functions



Node Embedding Based Similarity Metric Learning

- Learning a weighted adjacency matrix by computing the pair-wise node similarity in the embedding space
- Common metrics functions
  - Attention-based similarity metric functions
  - Cosine-based similarity metric functions
  - Kernel-based similarity metric functions



Attention-based Similarity Metric Functions

Variant 1)

$$S_{i,j} = (\mathbf{v}_i \odot \mathbf{u})^T \mathbf{v}_j$$

Node feature vector

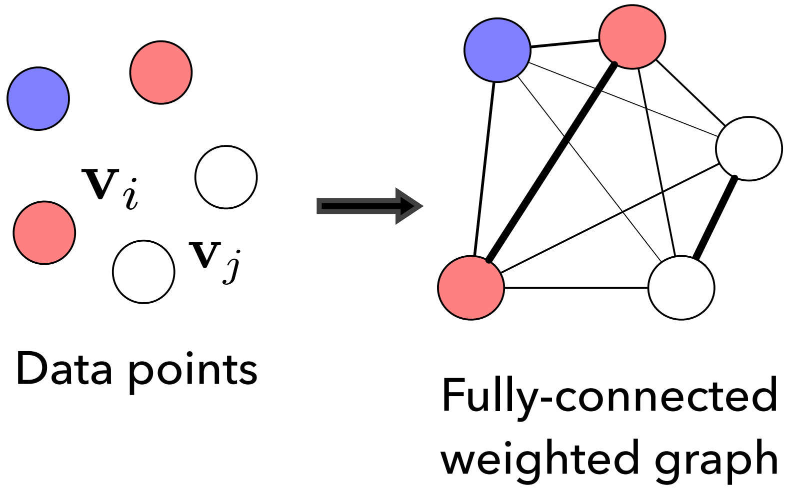
Non-negative learnable weight vector

Variant 2)

Enforcing sparsity

$$S_{i,j} = \mathrm{ReLU}(\mathbf{W}\mathbf{v}_i)^T \mathrm{ReLU}(\mathbf{W}\mathbf{v}_j)$$

Learnable weight matrix



Chen at al. *"GraphFlow: Exploiting Conversation Flow with Graph Neural Networks for Conversational Machine Comprehension". IJCAI 2020.*

Chen et al. *"Reinforcement Learning Based Graph-to-Sequence Model for Natural Question Generation". ICLR 2020.*

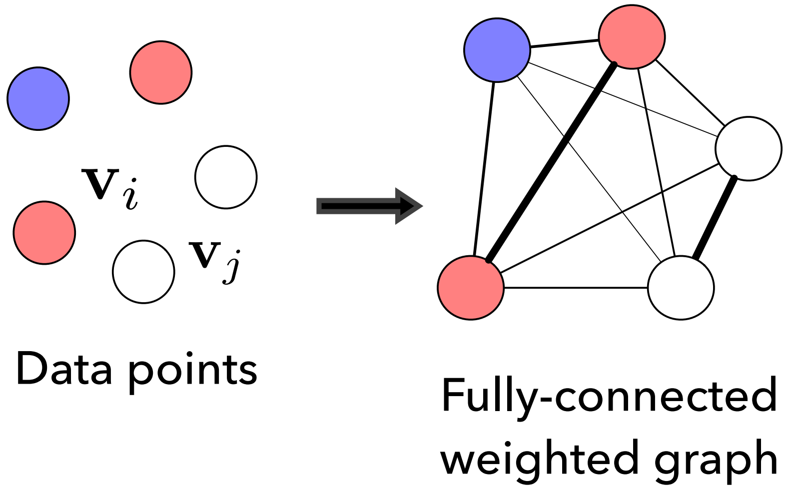
Cosine-based Similarity Metric Functions

$$S_{i,j}^{p} = \cos(\mathbf{w}_p \odot \mathbf{v}_i, \mathbf{w}_p \odot \mathbf{v}_j)$$

Learnable weight vector

$$S_{i,j} = \frac{1}{m}\sum_{p=1}^{m} S_{ij}^{p}$$

Multi-head similarity scores



Chen et al. “Iterative Deep Graph Learning for Graph Neural Networks: Better and Robust Node Embeddings”. NeurIPS 2020.

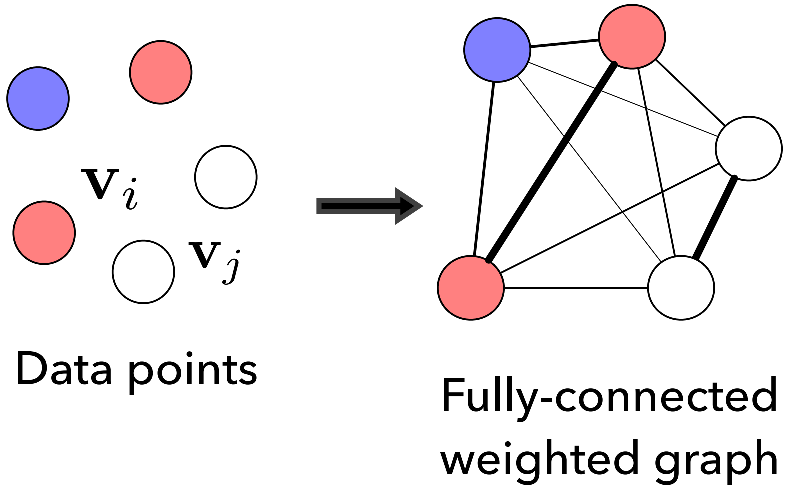
Kernel-based Similarity Metric Functions

Mahalanobis distance between node embeddings

$$d(\mathbf{v}_i, \mathbf{v}_j) = \sqrt{(\mathbf{v}_i - \mathbf{v}_j)^\top M (\mathbf{v}_i - \mathbf{v}_j)}$$

$$S(\mathbf{v}_i, \mathbf{v}_j) = \frac{-d(\mathbf{v}_i, \mathbf{v}_j)}{2\sigma^2}$$

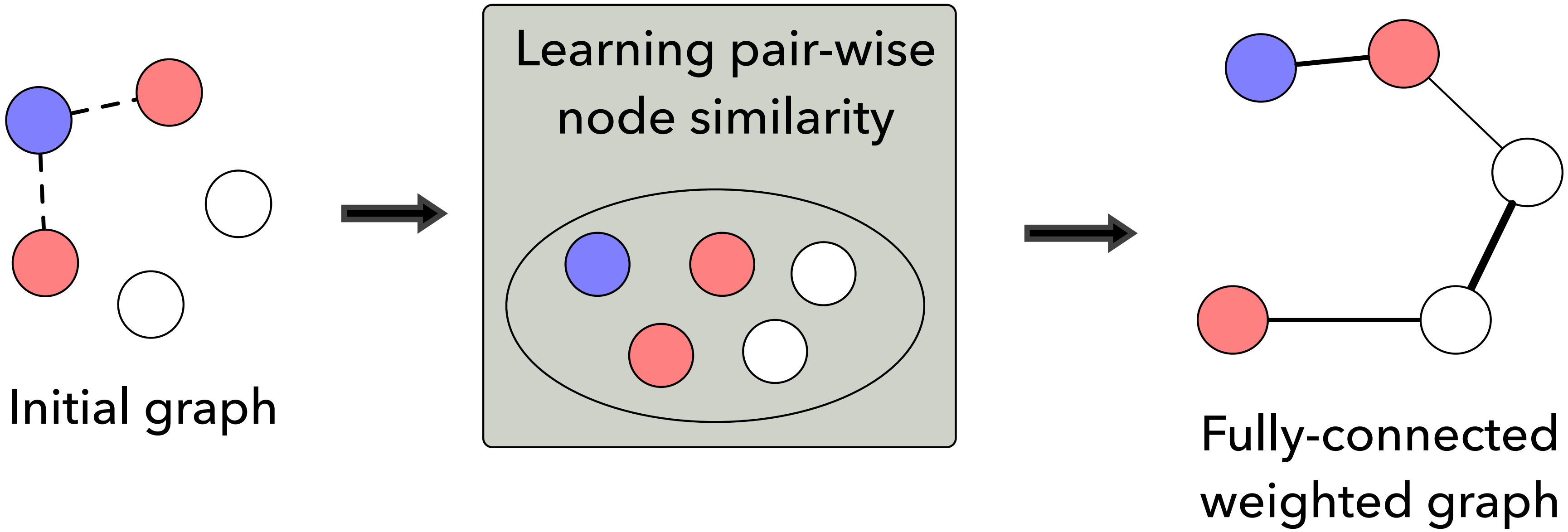
Gaussian kernel



Li et al. “Adaptive graph convolutional neural networks”. AAAI 2018.

Structure-aware Similarity Metric Learning

- Learning a weighted adjacency matrix by computing the pair-wise node similarity in the embedding space
- Considering existing edge information of the intrinsic graph in addition to the node information



Structure-aware Attention Mechanism

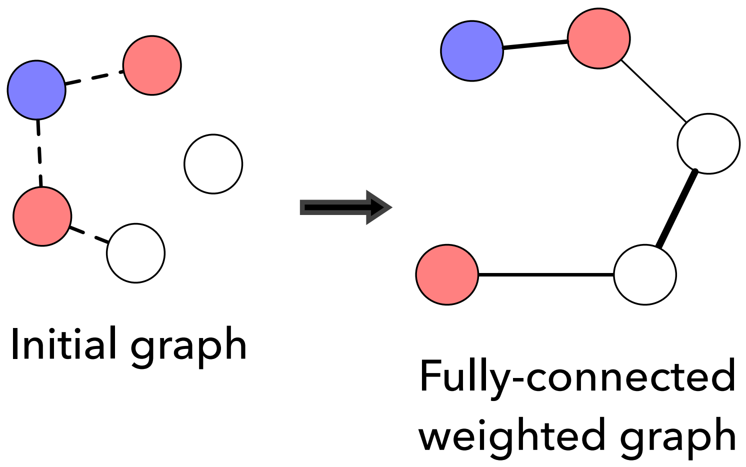
Variant 1)

$$S^l_{i,j} = \text{softmax}(\mathbf{u}^T \tanh(\mathbf{W}[\mathbf{h}^l_i, \mathbf{h}^l_j, \mathbf{v}_i, \mathbf{v}_j, \mathbf{e}_{i,j}]))$$

Edge embeddings

Variant 2)

$$S_{i,j} = \frac{\text{ReLU}(\mathbf{W}^Q \mathbf{v}_i)^T (\text{ReLU}(\mathbf{W}^K \mathbf{v}_i) + \text{ReLU}(\mathbf{W}^R \mathbf{e}_{i,j}))}{\sqrt{d}}$$

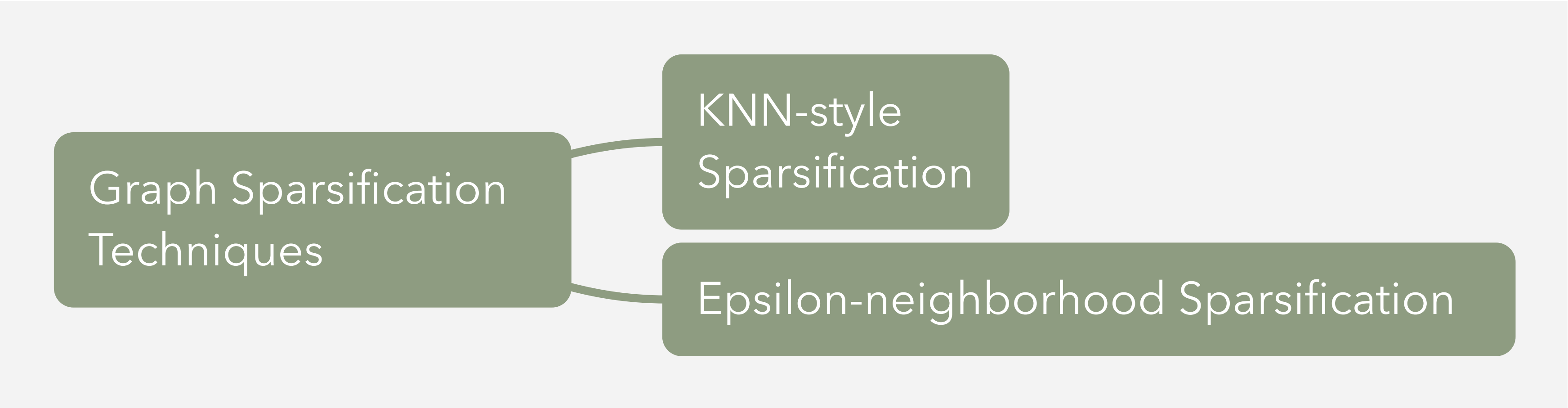


Liu et al. "Contextualized Non-local Neural Networks for Sequence Learning". AAAI 2019.

Liu et al. "Retrieval-Augmented Generation for Code Summarization via Hybrid GNN". ICLR 2021.

Graph Sparsification Techniques

- Similarity metric functions learn a fully-connected graph
- Fully-connected graph is computationally expensive and might introduce noise
- Enforcing sparsity to the learned graph structure
- Various techniques



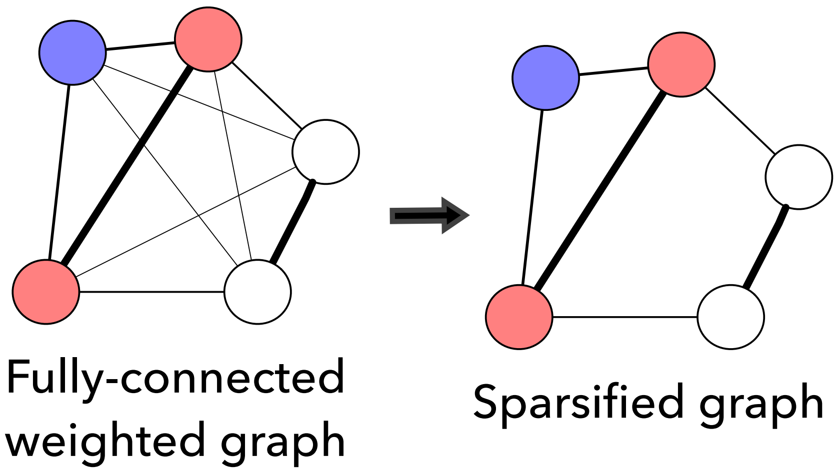
Common Graph Sparsification Options

Option 1) KNN-style Sparsification

$$\mathbf{A}_{i,:} = \text{topk}(\mathbf{S}_{i,:})$$

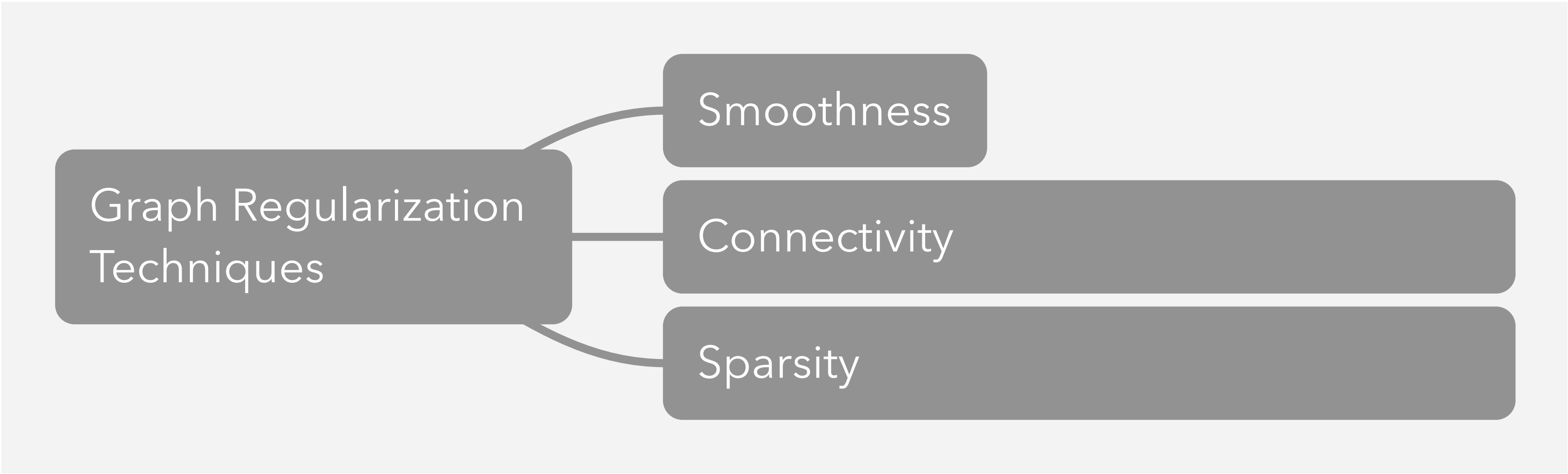
Option 2) epsilon-neighborhood Sparsification

$$A_{i,j} = \begin{cases} S_{i,j} & S_{i,j} > \varepsilon \\ 0 & \text{otherwise} \end{cases}$$



Graph Regularization Techniques

- Enforcing common graph properties to the learned graph structure
- Combining both task prediction loss and graph regularization loss
- Various graph properties



Graph Regularization Techniques

- Smoothness

$$\Omega(\mathbf{A}, \mathbf{X}) = \frac{1}{2n^2} \sum_{i,j} A_{i,j} ||\mathbf{X}_i - \mathbf{X}_j||^2 = \frac{1}{n^2} \mathrm{tr}(\mathbf{X}^\top \mathbf{L} \mathbf{X})$$

- Connectivity

$$\frac{-1}{n} \mathbf{1}^\top \log(\mathbf{A}\mathbf{1})$$

- Sparsity

$$\frac{1}{n^2} ||\mathbf{A}||_F^2$$

Borrowed from unsupervised GSL from smooth signals!

Combining Intrinsic and Implicit Graph Structures

- Intrinsic graph typically still carries rich and useful information
- Learned implicit graph is potentially a "shift" (e.g., substructures) from the intrinsic graph structure

$$\widetilde{A} = \lambda L^{(0)} + (1 - \lambda) \mathrm{f}(A)$$

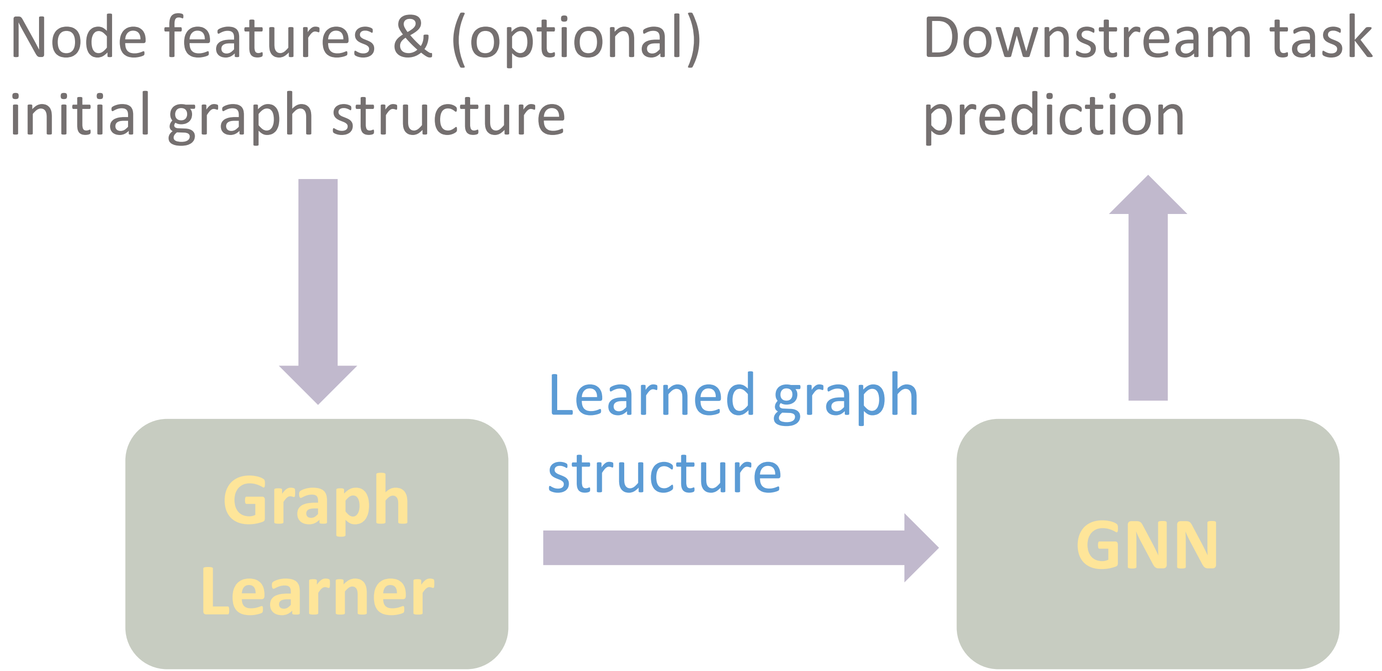
Normalized graph Laplacian

f(A) can be arbitrary operation, e.g., graph Laplacian, row-normalization

Li et al. "Adaptive Graph Convolutional Neural Networks". AAAI 2018.

Chen et al. "Iterative Deep Graph Learning for Graph Neural Networks: Better and Robust Node Embeddings". NeurIPS 2020.

Learning Paradigms: Joint Learning



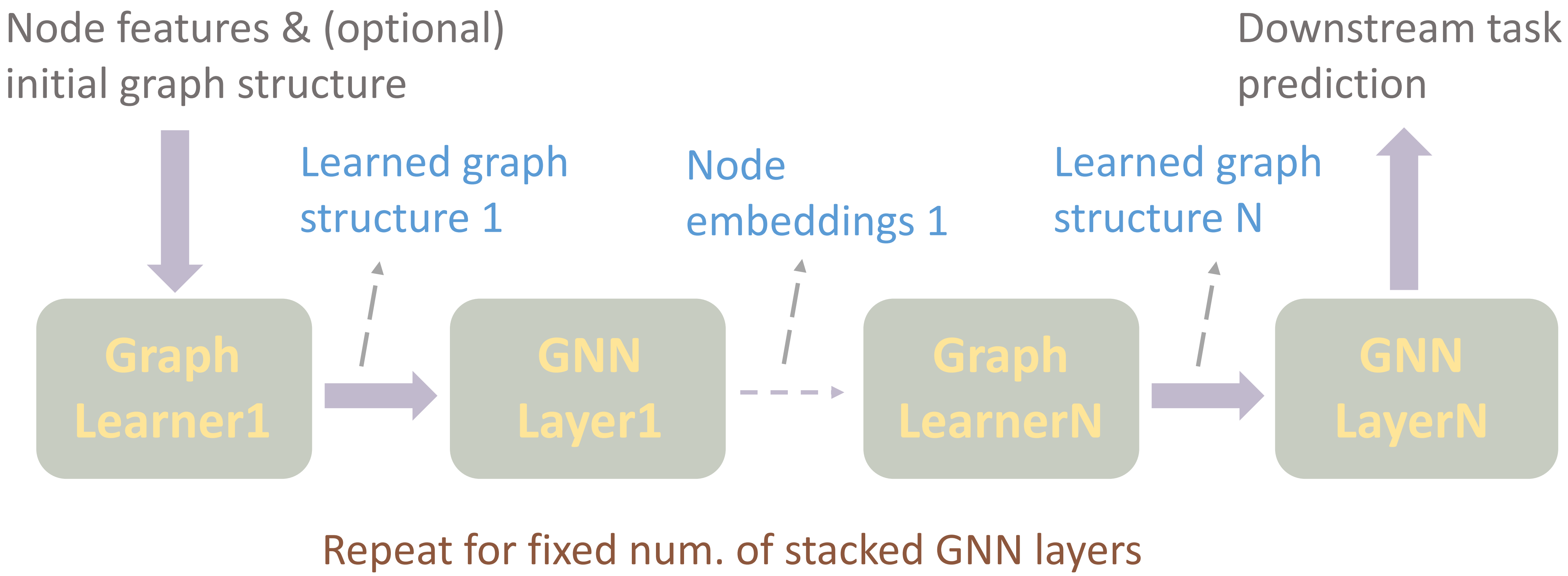
Chen at al. "GraphFlow: Exploiting Conversation Flow with Graph Neural Networks for Conversational Machine Comprehension". IJCAI 2020.

Chen et al. "Reinforcement Learning Based Graph-to-Sequence Model for Natural Question Generation". ICLR 2020.

Liu et al. "Contextualized Non-local Neural Networks for Sequence Learning". AAAI 2019.

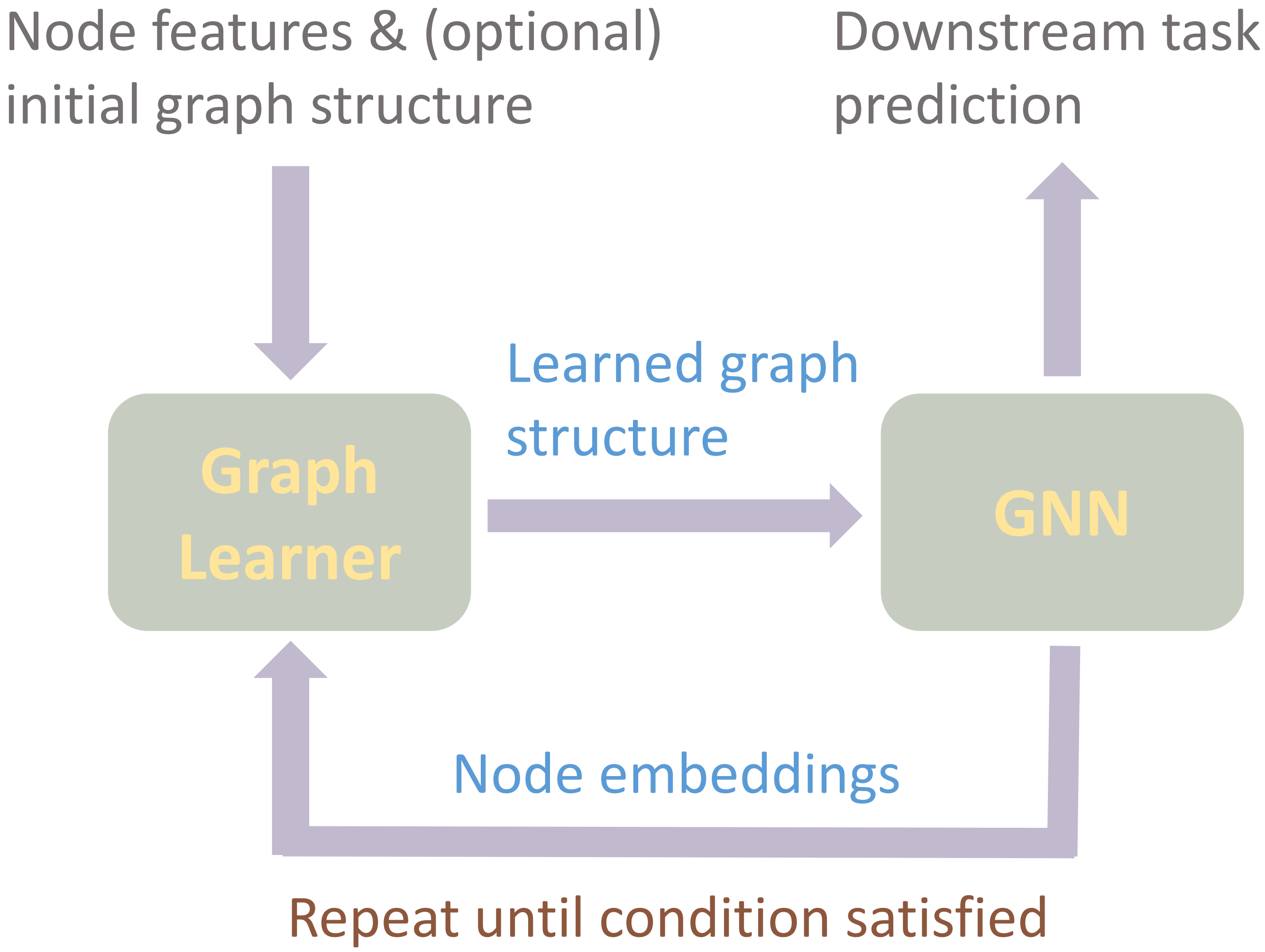
Liu et al. "Retrieval-Augmented Generation for Code Summarization via Hybrid GNN". ICLR 2021.

Learning Paradigms: Adaptive Learning



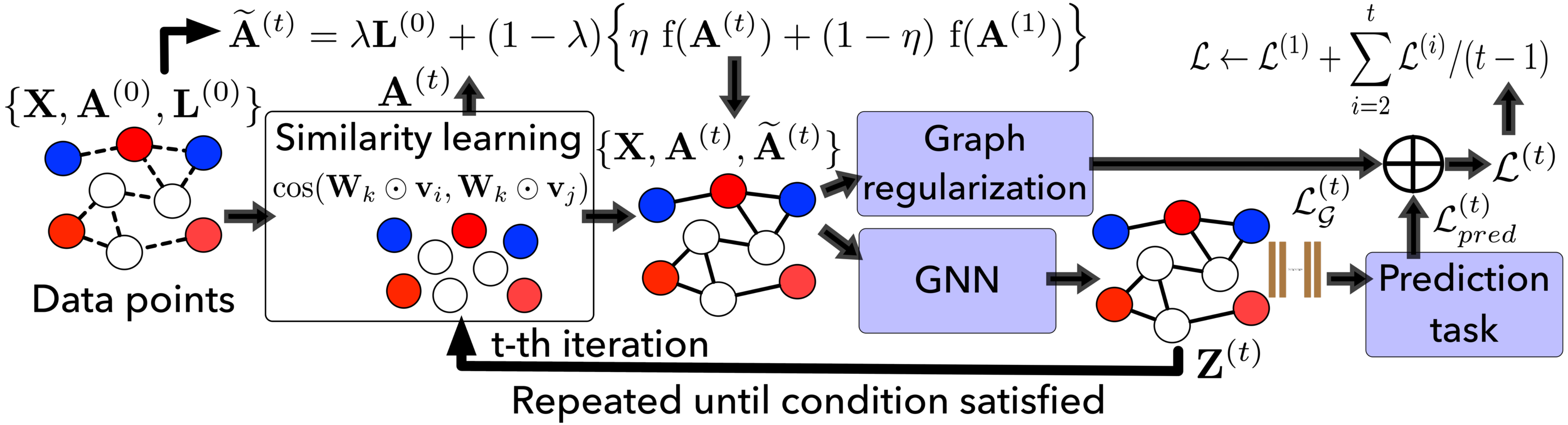
Li et al. “Adaptive Graph Convolutional Neural Networks”. AAAI 2018.

Learning Paradigms: Iterative Learning



Chen et al. “Iterative Deep Graph Learning for Graph Neural Networks: Better and Robust Node Embeddings”. NeurIPS 2020.

Iterative Deep Graph Learning [Chen et al., NeurIPS 2020]



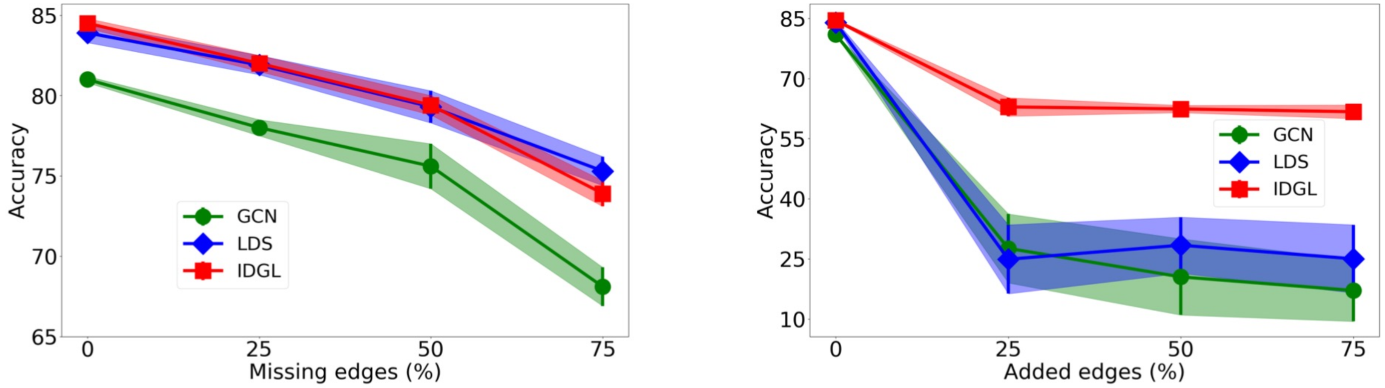
- GSL as similarity metric learning
- Graph regularization to control smoothness, sparsity and connectivity
- Iterative method to refine the graph structure and graph embeddings
- Better scalability (O(n^2) -> O(n)) using anchor-based approximation technique

Chen et al. “Iterative Deep Graph Learning for Graph Neural Networks: Better and Robust Node Embeddings”. NeurIPS 2020.

Iterative Deep Graph Learning [Chen et al., NeurIPS 2020]

Model	Cora	Citeseer	Pubmed	ogbn-arxiv	Wine	Cancer	Digits
GCN	81.5	70.3	79.0	71.7 (0.3)	—	—	—
GAT	83.0 (0.7)	72.5 (0.7)	79.0 (0.3)	—	—	—	—
GraphSAGE	77.4 (1.0)	67.0 (1.0)	76.6 (0.8)	71.5 (0.3)	—	—	—
APPNP	—	**75.7 (0.3)**	79.7 (0.3)	—	—	—	—
H-GCN	**84.5 (0.5)**	72.8 (0.5)	79.8 (0.4)	—	—	—	—
GCN+GDC	83.6 (0.2)	73.4 (0.3)	78.7 (0.4)	—	—	—	—
LDS	84.1 (0.4)	75.0 (0.4)	—	—	97.3 (0.4)	94.4 (1.9)	92.5 (0.7)
GCN_{kNN}*	—	—	—	—	95.9 (0.9)	94.7 (1.2)	89.5 (1.3)
GAT_{kNN}*	—	—	—	—	95.8 (3.1)	88.6 (2.7)	89.8 (0.6)
$\text{GraphSAGE}_{\text{kNN}}$*	—	—	—	—	96.5 (1.1)	92.8 (1.0)	88.4 (1.8)
LDS*	83.9 (0.6)	74.8 (0.3)	—	—	96.9 (1.4)	93.4 (2.4)	90.8 (2.5)
IDGL	**84.5 (0.3)**	74.1 (0.2)	—	—	97.8 (0.6)	**95.1 (1.0)**	93.1 (0.5)
IDGL-ANCH	84.4 (0.2)	72.0 (1.0)	**83.0 (0.2)**	**72.0 (0.3)**	**98.1 (1.1)**	94.8 (1.4)	**93.2 (0.9)**

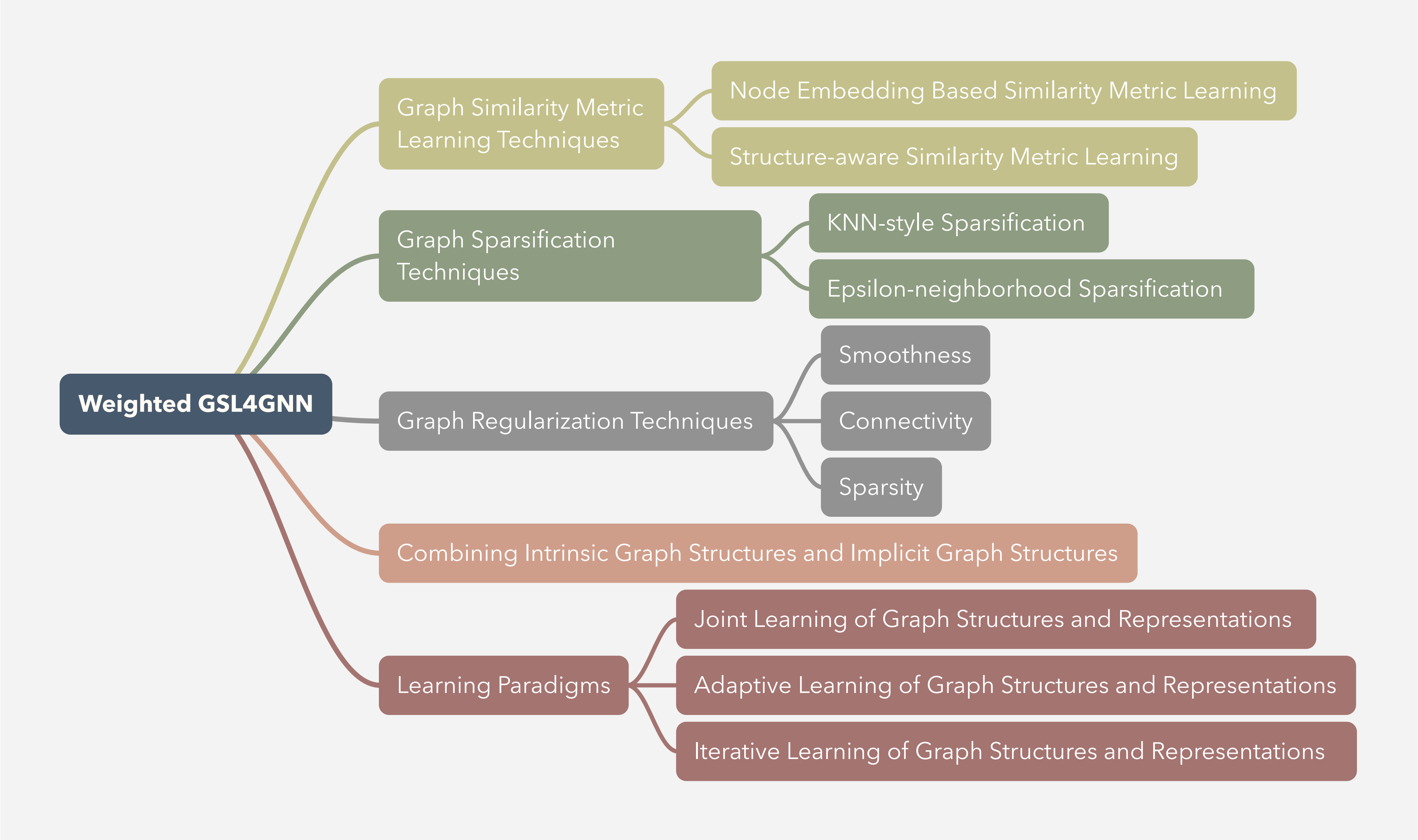
Node classification results.



(a) Edge deletion (b) Edge addition

Edge attack results on Cora.

Weighted GSL4GNN Summary



Connections to Other Problems

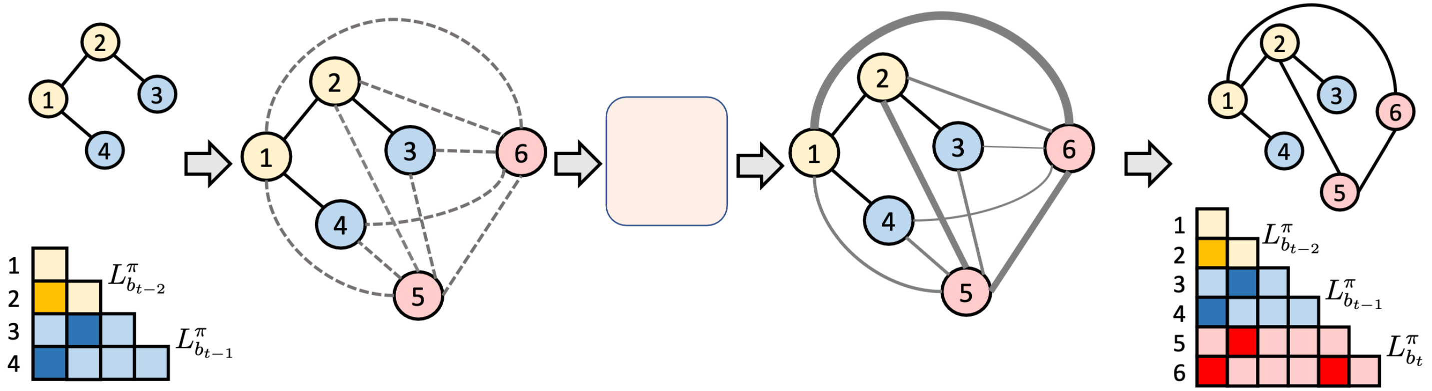
Connections to Other Problems: GSL as Graph Generation

Connections:

- Learning graphs from data

Differences:

- Graph generation: generating new graphs where both nodes and edges are added by sampling from the learned graph distribution.
- GSL: learning a graph structure given a set of node attributes.



Liao, Renjie. "Graph Neural Networks: Graph Generation." In Graph Neural Networks: Foundations, Frontiers, and Applications. 2022

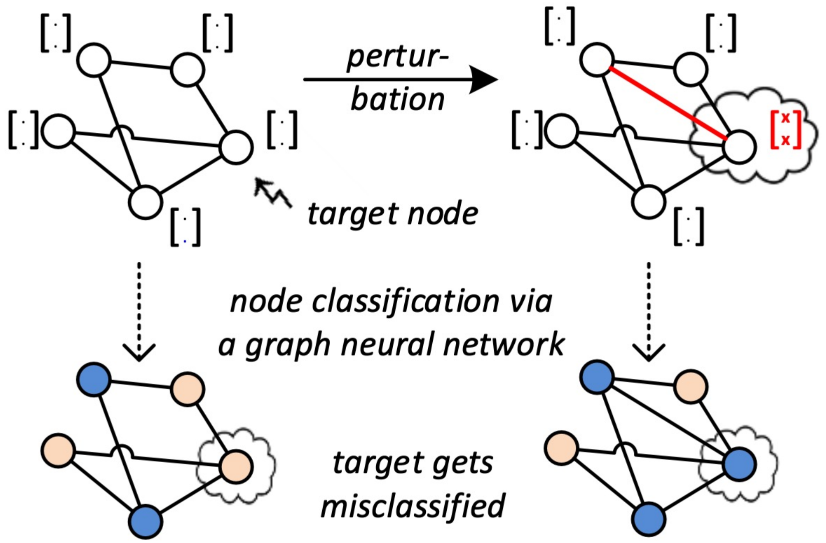
Connections to Other Problems: GSL for Graph Adversarial Defenses

Connections:

- Improving potentially error-prone (e.g., noisy or incomplete) input graphs
- Graph adversarial defenses can benefit from GSL techniques

Differences:

- Graph adversarial defenses: initial graph structure is available, but potentially poisoned by adversarial attacks
- GSL: initial graph structure is available or unavailable



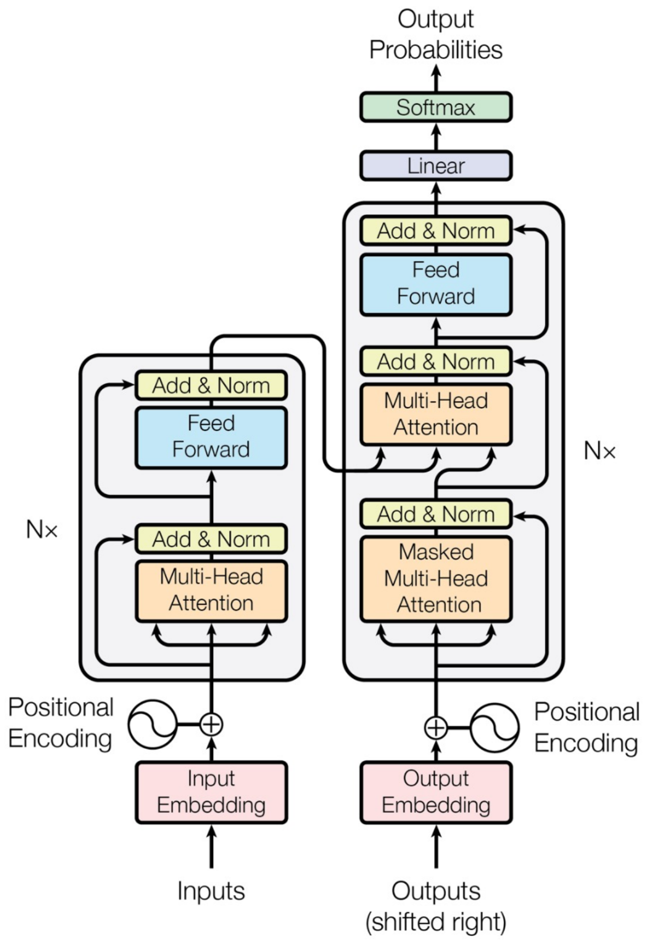
Connections to Other Problems: Transformers

Connections:

- Transformer models aim to learn a self-attention matrix between every pair of objects adaptively at each layer, similar to adaptive learning paradigm for weighted GSL

Differences:

- Vanilla Transformers don't handle graph-structured data (graph transformers combining transformers and GNNs)



Vaswani et al., *"Attention Is All You Need."* NIPS 2017.

GSL4GNN: Future Directions and Conclusions

Future Directions

- Robust GSL
  - Noisy initial graph structures and noisy node attributes
- Scalable GSL
  - Pair-wise node similarity computation is expensive and intractable for large graphs
  - Potential solutions: LSH/low-rank/random feature/kernel methods
- GSL for Heterogeneous Graphs
  - Heterogeneous graphs carry on richer information
  - Less explored

Conclusions

- GNNs are powerful machine learning tools for modeling graph-structured data
- GSL has been extensively studied in traditional machine learning
- GSL4GNN is a trending research area and critical for the success of GNN applications
- Open challenges in GSL4GNN

Resources

- Chen, Yu, and Lingfei Wu. "Graph Neural Networks: Graph Structure Learning." *Graph Neural Networks: Foundations, Frontiers, and Applications*. Springer, Singapore, 2022. 297-321. (website, video)
- Zhu, Yanqiao, et al. "Deep graph structure learning for robust representations: A survey." *arXiv preprint arXiv:2103.03036* (2021).
- Dong, Guimin, et al. “Graph Neural Networks in IoT: A Survey.” arXiv 2022.
- Wu, Lingfei, et al. “Deep Learning on Graphs for Natural Language Processing.” Tutorials at NAACL'21, SIGIR'21, KDD'21, IJCAI'21, AAAI’22 and TheWebConf’22. (website)

Thanks!

Q&A