Data Quality-Aware Graph Machine Learning

Yu Wang\textsuperscript{1,6}  
Yijun Tian\textsuperscript{2}  
Tong Zhao\textsuperscript{3}  
Xiaorui Liu\textsuperscript{4}  
Jian Kang\textsuperscript{5}  
Tyler Derr\textsuperscript{1}

Vanderbilt University\textsuperscript{1}  
University of Notre Dame\textsuperscript{2}  
Snap Research\textsuperscript{3}  
North Carolina State University\textsuperscript{4}  
University of Rochester\textsuperscript{5}  
University of Oregon\textsuperscript{6}
Introduction and Background - Graph-Structured Data is Everywhere

Scientific Graph
- Protein
- Small Molecule
- Virus
- Brain Neural
- Phylogenetic Tree
- 3D Grid

Gas Network
- Power Network
- Communication Network
- Comcast Nationwide Fiber Optic Network
- Traffic Trace

Infrastructure Graph
- Gas Network
- Power Network
- Communication Network
- Comcast Nationwide Fiber Optic Network
- Traffic Trace

Social Interaction Graph
- Citation Network
- Transaction Network
- User-Entity Interaction Graph

Virtual Village with AI Agents

Knowledge Graph
- Garden
- Table
- Sink
- Start seeds
- Involve
- Project work
- Belong
- Co-view
- Co-purchase
- Bought by
- User
- Product Knowledge Graph

Reasoning / Planning Graph
- CoT/ToT/GoT
- ToolChain* Exploration Space
- WalkTo(bathroom)
- WalkTo(shower)
- WalkTo(bedroom)
- Find(shower)
- TurnOn(shower)
- Find(soap)
- Find(clothes)
- SwitchOn(shower)
- TurnOn(shower)
- Grab(soap)
Introduction and Background - Graph-based Tasks and Graph Machine Learning

<table>
<thead>
<tr>
<th>Node Classification/Regression</th>
<th>Community Detection</th>
<th>Network Dismantling</th>
<th>Network Alignment</th>
<th>Influence Maximization</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Node Classification/Regression Diagram" /></td>
<td><img src="image2" alt="Community Detection Diagram" /></td>
<td><img src="image3" alt="Network Dismantling Diagram" /></td>
<td><img src="image4" alt="Network Alignment Diagram" /></td>
<td><img src="image5" alt="Influence Maximization Diagram" /></td>
</tr>
</tbody>
</table>

- **Link Prediction/Edge Classification**
  - ![Link Prediction/Edge Classification Diagram](image6)

- **Graph Classification/Regression**
  - ![Graph Classification/Regression Diagram](image7)

---

Real-world graph data can have data quality challenges...

Garbage in, garbage out

Real-world graph data can have data quality challenges...

Garbage in, garbage out
Introduction and Background – Real-world Graphs have Data Quality Issues

Topological Issues
  e.g., Homophily vs Heterophily

Bias Issues
  e.g., bail decision making

Imbalance Issues
  e.g., labeled data in chemistry

Abnormal Graph Data

Criminal Associate Network
Introduction and Background – Model- vs. Data-Centric Methods

Find the **best model for the given fixed dataset**

Model-Centric

- Model architectures
- Loss functions/constraints
- Hyperparameter tuning
- ... etc.

Realize the **best dataset for the given prediction task**

Data-Centric

- Data Organization: Constructing graphs
- Data Integration: Improving node/edge features
- Data Cleaning: Confident learning
- ... etc.
Introduction and Background – Model- vs. Data-Centric Methods

Credit: MIT Introduction to Data-Centric AI course & Inspired by XKCD 2494 “Flawed Data”
Data Quality-Aware Graph Machine Learning

• Introduction and Background
• Topology Issues
• Imbalance Issues
• Short Break

• Bias and Fairness Issues
• Limited Labeled Data Issues
• Abnormal Graph Data Issues
• Summary
Outline

- Introduction and Background
  - Topology Issues
  - Imbalance Issues
  - Short Break
- Bias and Fairness Issues
  - Limited Labeled Data Issues
  - Abnormal Graph Data Issues
  - Summary
• Introduction and Background

• **Topology Issues**

• Imbalance Issues

• Short Break

• Bias and Fairness Issues

• Limited Labeled Data Issues

• Abnormal Graph Data Issues

• Summary
Topology Issues

• Global Positional Issues

• Local Topology Issues

• Missing Graph Issues

• Future Directions and Q&A
Topography Issues – Global Topology Issues – Labeled Node Influence

Degree -> Influence

If $d_i > d_j$, $v_i$ has higher influence than $v_j$ on training GNNs

$$x_i^{l+1} = \sigma \left( \sum_{j \in N_i} a_{ij} \left( W^l + W^l_{d_j} \right) x_j^l \right)$$

Position -> Influence

$$P = \alpha (I - (1 - \alpha) A')^{-1}$$

$$T_v = \mathbb{E}_{x \sim P_v} \left( \sum_{j \in [1,k], j \neq y_v} |C_j|^{-1} \sum_{i \in C_j} \mathbf{P}_{i,x} \right)$$

$$L = -|\mathcal{L}|^{-1} \sum_{v \in \mathcal{L}} \sum_{c=1}^k y_v^c \log p_v^c$$

Darker colors - Higher influences.


**Homophily vs Heterophily**

Birds of a feather flock together

**Ego-Neighbor Separation**
\[ r_v^k = \text{COMBINE}(r_v^{k-1}, \text{AGGR}({r_u^{k-1}: u \in \mathcal{N}_v})) \]

**Higher-order Neighbor**
\[ r_v^k = \text{COMBINE}(r_v^{k-1}, \text{AGGR}_1({r_u^{k-1}: u \in \mathcal{N}_v^1}), \text{AGGR}_2({r_u^{k-1}: u \in \mathcal{N}_v^2} \ldots )) \]

**Combination of Intermediate Representation**
\[ r_v^k = \text{COMBINE}(r_v^1, r_v^2, \ldots, r_v^K) \]

**Class belief propagation**
\[ B^k = B^0 + AB^{k-1}H \]

**Graph-level Homophily**
\[ h(G, \{y_i: i \in \mathcal{V}\}) = \frac{1}{|\mathcal{E}|} \sum_{(j,k) \in \mathcal{E}} 1(y_j = y_k) \]
Topography Issue – Local Topology Issues – Heterophily/Homophily

Homophily vs Heterophily

Birds of a feather flock together

- Ego
- 1-order
- 2-order

Graph-level Homophily

\[ h(G, \{y_i; i \in V\}) = \frac{1}{|E|} \sum_{(i,j) \in E} 1(y_j = y_k) \]

Ego-Neighbor Separation

\[ r_v^k = \text{COMBINE}(r_v^{k-1}, \text{AGGR}((r_u^{k-1}; u \in N_v))) \]

Higher-order Neighbor

\[ r_v^k = \text{COMBINE}(r_v^{k-1}, \text{AGGR}_1((r_u^{k-1}; u \in N_v^1), \text{AGGR}_2((r_u^{k-1}; u \in N_v^2) ...) )) \]

Combination of Intermediate Representation

\[ r_v^k = \text{COMBINE}(r_v^1, r_v^2, ..., r_v^K) \]

Class belief propagation

\[ B^k = B^0 + A B^{k-1} H \]

Graph Transition

Class Transition

\[ \begin{bmatrix}
0.1 & 0.3 & 0.6 \\
0.8 & 0.1 & 0.1 \\
0.7 & 0.3 & 0
\end{bmatrix} \]


Zhu, Jiong, et al. "Graph Neural Networks with Heterophily." AAAI 2021
Topology Issue – Local Topology Issues – Heterophily/Homophily

Across Different Graphs

Within the Same Graph

Class Neighborhood Distribution Distinguishability

High

Low

High

In homophily graph, GNNs > MLP on homophily nodes

In heterophily graph, GNNs > MLP on heterophily nodes

Ma, Yao, et al. "Is Homophily a Necessity for Graph Neural Networks?." ICLR 2021
Mao, Haitao, et al. "Demystifying Structural Disparity in Graph Neural Networks: Can One Size Fit All?." NeurIPS 2024

Focal Link is missing from training subgraph to testing subgraph

**Distribution Shift**

\[
\text{h}^{\text{mean}} = \frac{\text{h}^+ + \text{h}^-}{2}
\]

**Edge Mean**

\[
h^{\text{mean}} = w^+ h^+ + w^- h^-
\]

**Edge Attention**

\[
w^+ = \sigma(q^T\tanh(Wh^+ + b))
\]
Topology Issue — Local Topology Issues — # of Common Neighbor Shift

Link-centric Perspective

Node-centric Perspective

Collab

Time-based Split
Testing edges have more testing edges around

Wang, Xiyuan et al. "Neural Common Neighbor with Completion for Link Prediction." ICLR, 2024
**User/Item Interaction**

(a) User historical sequences

![Diagram](Image)

(b) Edges

![Diagram](Image)

(c) Multiplex Graph

![Diagram](Image)

---

**Multi-hop Reasoning**

Q: In what year was the creator of the current arrangement of the Simpson's Theme born?

S1: The Simpson's Theme was re-arranged during season 2, and the current arrangement by Alf Clausen was introduced at the beginning of season 3.

S2: Alf Heiberg Clausen (born March 28, 1941) is an American film and television composer.

A: March 28, 1941

---

**Sometimes Real-world Applications do not have Graphs!**

**But Graph can actually encode some useful information**

---


**Topology Issue – Missing Topology Issues**

**GNN embeddings**

\[ A_{ij}^p = \cos(w_p \odot v_i, w_p \odot v_j), A_{ij} = m^{-1} \sum_{p=1}^{m} a_{ij}^p \]

\[ r_{ik}^p = \cos(w_p \odot v_i, w_p \odot u_k), r_{ik} = m^{-1} \sum_{p=1}^{m} r_{ik}^p \]

**Real-world Graph is sparse!**

\[ a_{ij} = \begin{cases} A_{ij}, & a_{ij} < \varepsilon \\ 0, & a_{ij} \geq \varepsilon \end{cases} \]

\[ A^t = \lambda L^0 + (1 - \lambda)(\eta f(A^t) + (1 - \eta)f(A^1)) \]

\[ L^0 = (D^0)^{-0.5} A^0 (D^0)^{-0.5} \]

**Quadratic Computation \( \mathcal{O}(n^2) \)**

\[ f(A)_{ij} = A_{ij}/\sum_{j} A_{ij} \]

**Anchor Selection \( \mathcal{O}(nK), K \ll n \)**

\[ F^{0,\prime} = \Lambda^{-1} R^T F^0 \quad \text{Node} \rightarrow \text{Anchor} \]

\[ F^1 = \Lambda^{-1} R^T F^{0,\prime} \quad \text{Anchor} \rightarrow \text{Node} \]
Q&A and Future Work – Topology Issue

Global Topology Issue

Missing Topology Issue

Q: In what year was the creator of the current arrangement of the Simpson's Theme born?

$S_1$: The Simpson’s Theme was re-arranged during season 2, and the current arrangement by Alf Clausen was introduced at the beginning of season 3

$S_2$: Alf Heiberg Clausen (born March 28, 1941) is an American film and television composer.

A: March 28, 1941

Topology Issue of Complex Graphs

Traditional Discrete Domains

Set

Graph

No Relation

Pairwise Relations

$\circ$ : Nodes  \ $\rightarrow$ : Edges

Domains of Topological Deep Learning

Simplicial complex

Cellular complex

Combinatorial complex

Part-Whole Relations

Set-Type Relations

Hypergraph

is part of

not necessarily part of
Outline

- Introduction and Background
- Topology Issues
- **Imbalance Issues**
- Short Break
- Bias and Fairness Issues
- Limited Labeled Data Issues
- Abnormal Graph Data Issues
- Summary
Imbalance Issues

- Node-level Imbalance
- Graph-level Imbalance
- Edge-level Imbalance
- Future Directions and Q&A
Imbalance Issues – Node-level imbalance

SMOTE

- Feature Interpolation
- Train – Major
- Train – Minor
- Test – Major
- Test – Minor

Graph-structured data has both feature and edge

GraphSMOTE

- Feature Interpolation
- Edge Generation

$$nn(v) = \arg\min_{u} ||h_{u}^1 - h_{v}^1||, \text{ s.t. } Y_{u} = Y_{v}$$

$$h_{v'}^1 = (1 - \delta)h_{v}^1 + \delta h_{nn(v)}^1$$

$$A_{v'u} = \begin{cases} 1, & \text{if } E_{v'u} \geq \eta \\ 0, & \text{otherwise} \end{cases}$$

$$L_{edge} = ||E - A||_F^2$$

$$E_{vu} = \text{softmax}(\sigma(h_{v}^1Sh_{u}^1))$$

Imbalance Issues – Node-level imbalance

Neighborhood Memorization

\[ p(u|v_{mixed}) = \hat{\phi}p(u|v_{minor}) + (1 - \hat{\phi})p(u|v_{target}) \]

\[ 0.5 < \hat{\phi} = \frac{1}{1 + e^{-\phi}} < 1 \]

\[ \phi = KL(\sigma(o_{minor})||\sigma(o_{target})) \]

\[ o_{minor} = |N_v|^{-1} \sum_{u \in N_v} o_{minor} \]

Park, Joonhyung et al. "GraphENS: Neighbor-aware ego network synthesis for class-imbalanced node classification." ICLR 2021
Imbalance Issues – Graph-level imbalance

Drug Discovery

Malware Detection

Quantity Augmentation

Structure Augmentation

Graph-of-Graphs (GoG)

SPP - Structurally Similar Molecules tend to have similar properties

Conditional distribution by node masking

Conditional distribution by edge removing

Graphs of Graphs (GoG)

Constructed GoG demonstrates high homophily!

HTS Hit Ratio
0.05% to 0.5%

0.01% Google, 2% Android

Normal : Autism
36 : 1

Autism Statistics. 2023


ASD Brain Classification

Drug Discovery

HTS Hit Ratio
0.05% to 0.5%

0.01% Google, 2% Android

Malware Detection

Shortest Path
Weisfeiler Lehman

Edge homophily

Number of neighborhoods
Imbalance Issues – Graph-level imbalance

70 years, ~600 polymers, oxygen permeability, Polymer Gas Separation Membrane Database

Imbalance Graph Regression!

(1) Use Model to predict on unlabeled graphs and select those high-quality-one

\[
\sigma_i = \frac{1}{\text{Var} \left( \{ f(G(i,j)) \} \right)}.
\]

(2) Sample more for label interval with less training samples

(3) Anchor-based Mix-up

\[ a_i, z_i: \text{anchor-label and embedding} \]

\[
\begin{align*}
\tilde{h}(i,j) &= \lambda \cdot z_i + (1 - \lambda) \cdot h_j, \\
\tilde{y}(i,j) &= \lambda \cdot a_i + (1 - \lambda) \cdot y_j,
\end{align*}
\]
Q&A and Future Work – Imbalance Issues

Node-level Imbalance

Graph-level Imbalance

Retrieval Additional Supervision

Generate Additional Supervision

ASD Brain

Normal : Autism

36 : 1

HTS Hit Ratio

0.05% to 0.5%
Short Break (4 min)
Outline

• Introduction and Background

• Topology Issues

• Imbalance Issues

• Short Break

• Bias and Fairness Issues

• Limited Labeled Data Issues

• Abnormal Graph Data Issues

• Summary
Bias and Fairness Issues - Suicide Prevention

• Why suicide prevention?
  • Suicide is one of the leading causes of death in United States

• Existing prevention strategies disproportionately affect different groups

• Key question
  • How to correct the bias and ensure fairness on graphs?
Bias and Fairness Issues - Fairness Definition

• **Principle**
  - Lack of favoritism from one side or another

• **Rich fairness definitions**
  - Group fairness
    - Statistical parity
    - Equal opportunity
    - Equalized odds
    - Accuracy parity
    - ...
  - Individual fairness
  - Counterfactual fairness
  - Degree fairness (on graphs)

**Fairness definition**

<table>
<thead>
<tr>
<th>Two sides</th>
<th>Fairness definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Two demographic groups</td>
<td>Group fairness</td>
</tr>
<tr>
<td>Two data points</td>
<td>Individual fairness</td>
</tr>
<tr>
<td>A data point and its counterfactual version</td>
<td>Counterfactual fairness</td>
</tr>
<tr>
<td>Two group of nodes with same degree</td>
<td>Degree fairness</td>
</tr>
</tbody>
</table>
Bias and Fairness Issues

- Group Fairness on Graphs
- Individual Fairness on Graphs
- Degree Fairness on Graphs
- Future Directions and Q&A
Group Fairness: Statistical Parity

- Statistical parity = equal acceptance rate
  \[ \Pr_+ (\hat{y} = c) = \Pr_- (\hat{y} = c) \]
  - \( \hat{y} \): model prediction
  - \( \Pr_+ \): probability for the protected group
  - \( \Pr_- \): probability for the unprotected group
  - Also known as demographic parity, disparate impact

- Example: clinical trial participation

Group Fairness: Equal Opportunity

- **Equal opportunity = equal true positive rate**

\[
\Pr_+(\hat{y} = c | y = c) = \Pr_-(\hat{y} = c | y = c)
\]

- \(y\): true label
- \(\hat{y}\): model prediction
- \(\Pr_+\): probability for the protected group
- \(\Pr_-\): probability for the unprotected group

- **Example: clinical trial participation**

If hold for all classes, it is called **equalized odds**

Node classification algorithm

- **Approved**
  - \(\Pr_+(\hat{y} = \text{approved} | \text{male}) = 1\)
  - \(\Pr_+(\hat{y} = \text{approved} | \text{female}) = 1\)

- **Not Approved**

Fair result
Same true positive rate for male and female

Adversarial Learning for Fair Representation Learning

• **Statistical parity**
  • Independence between the learned embedding $\mathbf{z}$ and a sensitive attribute $a$
    $$\mathbf{z}_u \perp a_u, \forall \text{ node } u$$
    where $a_u$ is the sensitive value of node $u$

• **Formulation**
  • Mutual information minimization
    $$I(\mathbf{z}_u, a_u) = 0, \forall \text{ node } u$$
  • Analogous to statistical parity in classification task
  • Fail to predict $a_u$ using $\mathbf{z}_u$ \(\leftarrow\) no information about $a_u$ in $\mathbf{z}_u$

• **Solution**
  • Adversarial learning
  • Encoder: encode node into low-dimensional embedding space for downstream tasks
  • Discriminator: fail to predict $a_u$ using $\mathbf{z}_u$

Limitation #1: Full Access to Sensitive Attribute Information

- **Adversarial learning**
  - Minimize a task-specific loss function to learn ‘good’ representations
  - Maximize the error of predicting sensitive feature to learn ‘fair’ representations

- **Limitations**
  - Require the sensitive attribute of all training nodes to train a good discriminator
  - Ignore the fact that sensitive information is hard to obtain due to privacy

- **Question**
  - What if we only have limited sensitive attribute information?

---

FairGNN: Additional Supervision Signal

- **Observation**
  - Adversarial learning is unstable to train ← even worse with limited sensitive attribute
  - Failure to converge may also cause discrimination

- **Key idea**
  - Additional prerequisite of independence for additional supervision
  - Independence → zero covariance

- **Solution**
  - Pseudo sensitive attribute from a sensitive attribute estimator
    - Not embedding from encoder
    - Offer pseudo-label for covariance minimization
  - Absolute covariance minimizer to minimize absolute covariance between model prediction \( \hat{\mathbf{y}} \) and pseudo sensitive attribute \( \hat{s} \)
    \[
    \mathcal{L}_R = |\text{cov}(\hat{s}, \hat{\mathbf{y}})| = |\mathbb{E}[(\hat{s} - \mathbb{E}[\hat{s}])(\hat{\mathbf{y}} - \mathbb{E}[\hat{\mathbf{y}}])]|
    \]
  - Absolute covariance to avoid minimizing negative covariance

FairGNN: Overall Framework

• Overall loss function

\[ \mathcal{L} = \mathcal{L}_C + \mathcal{L}_E - \alpha \mathcal{L}_A + \beta \mathcal{L}_R \]

• Intuition
  - \( \mathcal{L}_C \): classification loss (e.g., cross entropy) for learning representative node representation
  - \( \mathcal{L}_E \): sensitive attribute estimation loss for generating accurate pseudo sensitive attribute information
  - \( \mathcal{L}_A \): adversarial loss for debiasing the learned node representation
  - \( \mathcal{L}_R \): covariance minimizer to stabilize the adversary training
BeMap: Fair Topology View Generation

**Motivation**
- Message passing could be unfair

- Theoretical analysis
  node embedding = fair embedding + bias residual

- Empirical evidence
  - Predict node sensitive attribute using embeddings learned from GCN and MLP (no MP)

- Method: BeMap
  - (In every training epoch) neighbor sampling for balanced neighborhood and MP on it
  - Up to 80% bias reduction
  - Comparable or even better classification accuracy
  - More details in the paper

---

Bias and Fairness Issues

• Group Fairness on Graphs

• Individual Fairness on Graphs

• Degree Fairness on Graphs

• Future Directions and Q&A
Individual Fairness

• Definition
  • Similar individuals should have similar outcomes
  • Rooted in Aristotle’s conception of justice as consistency

• Formulation: Lipschitz inequality (most common)
  \[ d_1(M(x), M(y)) \leq L d_2(x, y) \]

  • \( M \): a mapping from input to output
  • \( d_1 \): distance metric for output
  • \( d_2 \): distance metric for input
  • \( L \): a constant scalar
InFoRM: Individual Fairness on GRaph Mining

• Research questions
  RQ1. Measure: how to quantitatively measure individual bias?
  RQ2. Algorithms: how to ensure individual fairness?
  RQ3. Cost: what is the cost of individual fairness?
InFoRM Measure: Quantifying Individual Bias

• **Principle**
  - Similar nodes → similar mining results

• **Mathematical formulation**
  \[
  \|Y[i, :] - Y[j, :]\|_F^2 \leq \frac{\epsilon}{S[i,j]} \quad \forall i, j = 1, ..., n
  \]
  
  - If \(S[i,j]\) is high, \(\frac{\epsilon}{S[i,j]}\) is small → push \(Y[i, :]\) and \(Y[j, :]\) to be more similar
  
  - Inequality should hold for every pairs of nodes \(i\) and \(j\) → too restrictive

• **Relaxed criteria**
  \[
  \sum_{i=1}^{n} \sum_{j=1}^{n} \|Y[i, :] - Y[j, :]\|_F^2 S[i,j] \leq m\epsilon
  \]
  
  - \(m\): number of edges in the graph
  
  - \(\delta = m\epsilon\)

(1) For any node pair \((i,j)\)
\[
\|Y[i, :] - Y[j, :]\|_F^2 S[i,j] \leq \epsilon
\]

(2) Sum it up for all node pairs

\[
2Tr(Y^T L_S Y) \leq \delta
\]

Overall individual bias of the graph


Alternative Measure: Ranking-Based Individual Fairness

- **Key challenge in InFoRM measure**
  - Lipschitz condition (used in InFoRM)
    \[ d_1(M(x), M(y)) \leq L d_2(x, y) \]
  - Distance comparison fails to calibrate between different individuals

- **Definition**
  - Given
    - (1) the node similarity matrix \( S_G \) of the input graph \( G \)
    - (2) the similarity matrix \( S_Y \) of the GNN output \( \hat{Y} \)
  - \( \hat{Y} \) is individually fair if, for each node \( i \), it satisfies that
    \[ \text{ranking list derived by } S_G[i, :] = \text{ranking list derived by } S_Y[i, :] \]

InFoRM Measure: Mitigating Individual Bias

• Graph mining workflow

- Input graph $A$
- Mining model with parameter $\theta$
- Mining results $Y$

Debiasing methods

- **Debiasing the input graph:**
  \[
  \min_Y J = \frac{1}{2} \| \tilde{A} - A \|_F^2 + \alpha \text{Tr}(Y^T L_S Y)
  \]
  s.t. $\partial_Y l(\tilde{A}, Y, \theta) = 0$

- **Debiasing the mining model:**
  \[
  \min_Y J = l(A, Y, \theta) + \alpha \text{Tr}(Y^T L_S Y)
  \]

- **Debiasing the mining results:**
  \[
  \min_Y J = \| Y - \overline{Y} \|_F^2 + \alpha \text{Tr}(Y^T L_S Y)
  \]

Individual bias (InFoRM measure)

InFoRM Cost: Characterizing Individual Bias

• **Main focus**
  - Debiasing the mining results (model-agnostic)

• **Given**
  - A graph with \( n \) nodes and adjacency matrix \( A \)
  - A node-node similarity matrix \( S \)
  - Vanilla mining results \( \bar{Y} \)
  - Debiased mining results \( Y^* = (I + \alpha S)^{-1}\bar{Y} \)

• If \( \|S - A\|_F = \Delta \), we have
  \[
  \|\bar{Y} - Y^*\|_F \leq 2\alpha \sqrt{n} \left( \Delta + \sqrt{\text{rank}(A)\sigma_{\text{max}}(A)} \right) \|\bar{Y}\|_F
  \]

• **Key factors**
  - The number of nodes \( n \) (i.e., size of the input graph)
  - The difference \( \Delta \) between \( A \) and \( S \)
  - The rank of \( A \) could be small due to (approximate) low-rank structures in real-world graphs
  - The largest singular value of \( A \) could be small if \( A \) is normalized

Bias and Fairness Issues

- Group Fairness on Graphs
- Individual Fairness on Graphs
- Degree Fairness on Graphs
- Future Directions and Q&A
Degree Fairness: Definition and Motivation

- **Definition**
  - Nodes of different degrees should have balanced utility on a graph mining task

- **Example: online advertising**
  - (A small portion of) celebrities often enjoy high-quality model performance
  - (A large portion of) grassroot users often suffer from bad model performance
Degree Unfairness: Pitfall of Graph Neural Networks

• **Given**
  - (1) $\mathcal{G} = (A, X)$
  - (2) Any test node $i$ in $\mathcal{G}$ with label $c$
  - (3) A graph learning model $M$ which output (before softmax) $Z$
  - (4) Any wrong prediction $c' \neq c$

• **Our results**
  - Misclassification rate
    \[
    \text{Pr}(\Pr(\hat{y} = c | i, M) > \Pr(\hat{y} = c'|M, i)) \leq \frac{1}{1 + R_{i,c'}}
    \]
  - $R_{i,c'} = \frac{\text{Var}[Z[i,c'] - Z[i,c]]}{\text{Var}[Z[i,c'] - Z[i,c]]}$ (reciprocal of measure of dispersion from economics)
  - $R_{i,c'}$ is positively correlated with the degree of node $i$

• **Conclusion**
  - High-degree nodes often have lower misclassification rate!
Causes #1: High-Degree Nodes with High Influence in Node Embeddings

• Given
  • $\mathcal{V}_{\text{labeled}}$: a set of labeled nodes
  • $W^{(L)}$: the weight of $L$-th layer in an $L$-layer GCN
  • $d_i$: degree of node $i$
  • $x_i$: input node feature of node $i$
  • $h_i^{(L)}$: output embeddings of node $i$ learned by the $L$-layer GCN

• Influence of node $i$ on GCN training
  $$ S(i) = \sum_{k \in \mathcal{V}_{\text{labeled}}} \left\| \mathbb{E} \left[ \frac{\partial h_i^{(L)}}{\partial x_k} \right] \right\| \propto \sqrt{d_i} \| W^{(L)} \| \sum_{k \in \mathcal{V}_{\text{labeled}}} \sqrt{d_k} $$

• Remark
  • For two nodes $i$ and $j$, if $d_i > d_j$, then $S(i) > S(j)$
    → Node with higher degree will have higher influence on GCN training
Solution #1: Degree-Specific Graph Convolution

• Key idea
  • Degree-specific weights to encode degree information

• Given
  • \( d_i \): the degree of node \( i \)
  • \( W_{d_j}^{(l)} \): the degree-specific weight w.r.t. degree of node \( j \)

• Degree-specific graph convolution

\[
h_i^{(l+1)} = \sigma \left( \sum_{j \in N_i \cup \{i\}} a_{ij} \left( W^{(l)} + W_{d_j}^{(l)} \right) h_j^{(l)} \right)
\]

• DEMO-Net \( \rightarrow W_{d_j}^{(l)} \) is generated randomly
• SL-DSGCN \( \rightarrow W_{d_j}^{(l)} \) is generated using a recurrent neural network
Causes #2: High-Degree Nodes with High Influence in Gradient

- **Gradient of loss w.r.t. weight**
  \[ \frac{\partial J}{\partial W^{(l)}} = \sum_{i=1}^{n} d_{\widehat{A}}(i) \Pi_i^{(\text{col})} = \sum_{j=1}^{n} d_{\widehat{A}}(j) \Pi_j^{(\text{row})} \]
  
  - \( \widehat{A} = \widehat{D}^{-\frac{1}{2}} (A + I) \widehat{D}^{-\frac{1}{2}} \rightarrow \) symmetric normalization kernel
  - \( \Pi_i^{(\text{col})} \) and \( \Pi_j^{(\text{row})} \rightarrow \) the directions for gradient descent
  - \( d_{\widehat{A}}(i) \) and \( d_{\widehat{A}}(j) \rightarrow \) the importance of the direction
  - High degree \( \rightarrow \) more focus on that direction

- **Symmetric normalization**
  - Normalize the largest eigenvalue but not degree
  - High degree in \( A \rightarrow \) high degree in \( \widehat{A} \)
Solution #2: Graph Normalization

- **Key idea**
  - Mitigate impacts of node degree by normalizing it to constant (i.e., 1)
  - Normalize the graph to a doubly stochastic graph

- **Sinkhorn-Knopp (SK) algorithm**
  - Iteratively normalize row and columns
  - *(Our result)* SK always finds the *unique* doubly stochastic form of symmetric normalization kernel

- **Fair gradient computation**
  \[
  \left( \frac{\partial J}{\partial W^{(l)}} \right)_{\text{fair}} = \left( H^{(l-1)} \right)^T \hat{A}_\text{DS} \frac{\partial J}{\partial E^{(l)}}
  \]
  - \( \hat{A}_\text{DS} \rightarrow \) doubly-stochastic normalization of \( \hat{A} \)

- **RawlsGCN family**
  - RawlsGCN-Graph: during data pre-processing, compute \( \hat{A}_\text{DS} \) and treat it as the input of GCN
  - RawlsGCN-Grad: during optimization (in-processing), treat \( \hat{A}_\text{DS} \) as a normalizer to equalize the importance of node influence
Bias and Fairness Issues

• Group Fairness on Graphs
• Individual Fairness on Graphs
• Degree Fairness on Graphs
• Future Directions and Q&A
Future Direction #1: Fairness beyond Plain and Static Graphs

• Observation
  • Real-world graphs are often dynamic and/or multi-sourced

• Research questions
  • How to ensure fairness for multiple type of nodes/edges or multi-graphs?
  • How to efficiently update the fair mining results at each timestamp?
  • How to characterize the impact of graph dynamics and multiple sources over the bias measure?
Preliminary Work: Dynamic Group Fairness in Recommender Systems

• Observation
  • performance disparity is getting larger over time

• Method: FADE
  - Model-agnostic
  - Fine-tuning with newly observed data
  - Periodically re-training to keep historical information
  - Linear complexity w.r.t. # new data

• Theory
  • Fine-tuning is better than re-training for fairness over time
    
    $\text{Re-training}$
    
    \[
    \ell^{\text{test}}_{\text{test}} = \ell^{\text{test}}_{\text{test}} + 2 \frac{m_{\text{new}} \ell_{\text{test}}}{n_{\text{test}} - m_{\text{test}}} \ln \left( \frac{1}{\delta} \right)
    \]

    $\text{Fine-tuning}$
    
    \[
    \ell^{\text{test}}_{\text{test}} = \ell^{\text{test}}_{\text{test}} + 2 \frac{m_{\text{new}} \ell_{\text{test}}}{n_{\text{test}} - m_{\text{test}}} \ln \left( \frac{1}{\delta} \right)
    \]

• Results
  • Fairness over time, small accuracy decrease

Yoo, H., Zeng, Z., Kang, J., Liu, Z., Zhou, D., Wang, F., ... & Tong, H. Ensuring user-side fairness in dynamic recommender systems. WWW 2024
Future Direction #2: Fairness on Graphs → Fairness with Graphs

• Fairness on graphs
  • Graph as data
  • Nodes = entities
  • Social networks → nodes = users
  • Citation networks → nodes = papers
  • Web graph → nodes = webpages

• Fairness with graphs
  • Graph as context
  • Nodes = models/datasets/modalities

• Example: supply chain

1. Demand + supply for medical resources
2. Models to allocate medical resources

• How can we leverage demand + supply + model collectively for fair supply chain?
Future Direction #3: Benchmark and Evaluation Metrics

• **Observation**
  • No consensus on the experimental settings for fair graph learning
  • Which data to compare? What sensitive attribute to consider?
  • Which evaluation metrics for each type of fairness?

• **Consequences**
  • Different settings for different research works
  • Hardly fair comparison among fair graph learning methods
  • Hardly deployable methods in real-world scenarios

• **Call for community effort**
  • Evaluation benchmark for consistent experimental settings and fair comparison
  • Collection of large-scale, realistic, but challenging dataset for evaluation
• Introduction and Background
• Topology Issues
• Imbalance Issues
• Short Break

• Bias and Fairness Issues
• Limited Labeled Data Issues
• Abnormal Graph Data Issues
• Summary
Limited Labeled Data Issues

- Graph Data Augmentations
- Self-supervised Learning on Graphs
Wikipedia: Techniques used to increase the amount of data by adding *slightly modified* copies of already existing data or *newly created* synthetic data from existing data.

- Why data augmentation?
  - It helps reduce overfitting when training a machine learning model.
  - The acquisition of labeled graph data can be expensive.
Wikipedia: Techniques used to increase the amount of data by adding *slightly modified* copies of already existing data or *newly created* synthetic data from existing data.
Wikipedia: Techniques used to increase the amount of data by adding *slightly modified* copies of already existing data or *newly created* synthetic data from existing data.

Image sources:
https://amitness.com/2020/05/data-augmentation-for-nlp/
Graph Data Augmentation

• Structure Augmentation
  • Drop/add nodes/edges, etc.

• Feature Augmentation
  • Mask off features, etc.

• Label Augmentation
  • Label propagation, etc.
Graph Data Augmentation

- Rule-based augmentations
  - Designed based on heuristic rules
  - Usually efficient and scalable
  - Simple and easy to implement
    - Commonly used in self-supervised learning

- Learned augmentations
  - Involve learning during augmentation
  - Augmented data better fits GML models
    - Better performances in supervised learning
## Rule-based Graph Data Augmentation Approaches

<table>
<thead>
<tr>
<th>Methodology</th>
<th>Representative Works</th>
<th>Task Level</th>
<th>Augmented Data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Node</td>
<td>Structure</td>
</tr>
<tr>
<td>Stochastic Dropping/Masking</td>
<td>DropEdge [87]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>DropNode [27]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>NodeDropping [127]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Feature Masking [100]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Feature Shuffling [106]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>DropMessage [23]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Subgraph Masking [127]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Subgraph Cropping/Substituting</td>
<td>GraphCrop [111]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>M-Evolve [145]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>MoCL [97]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Virtual Node</td>
<td>Graphormer [125]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>GNN-CM+/CM [45]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Mixup</td>
<td>Graph Mixup [115]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>tMixup [37]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>Graph Transparent [85]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>G-Mixup [39]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>SMOTE</td>
<td>GraphSMOTE [140]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>GATSMOTE [75]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>GNN-CL [70]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Diffusion</td>
<td>GDA [60]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Counterfactual Augmentation</td>
<td>CFLP [141]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Attribute Augmentation</td>
<td>LA-GNN [74]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>SR+DR [93]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Pseudo-labeling</td>
<td>Label Propagation [147]</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td></td>
<td>PTA [21]</td>
<td>✔</td>
<td>✔</td>
</tr>
</tbody>
</table>
DropEdge

- Dropout on edges: randomly remove some edges at the beginning of every training epoch.
  \[
  \tilde{A} = M \odot A
  \]
  \[
  M \in \{0, 1\}^{N \times N} \text{ s.t. } M_{i,j} = \text{Bernoulli}(\varepsilon)
  \]
- Prevents overfitting and over-smoothing.
Other Stochastic Masking/Dropping Methods

• **Node Dropping**
  • Randomly removing part of the nodes.

• **Feature Masking**
  • Randomly mask off node features.
  • Random row-shuffling on node feature matrix $X$.

• **Subgraph Masking**
  • Randomly mask off a connected subgraph.

• Mixup: generates a weighted combination of random pairs from the training data.

\[ \tilde{x} = \lambda x_i + (1 - \lambda) x_j, \]
\[ \tilde{y} = \lambda y_i + (1 - \lambda) y_j. \]

• Manifold Mixup: interpolating hidden states.

1. Graphon estimation:
2. Graphon Mixup:
3. Graph Generation:
4. Label Mixup:

\( \mathcal{G} \rightarrow W_G, \mathcal{H} \rightarrow W_H \)

\( W_\mathcal{I} = \lambda W_G + (1 - \lambda) W_H \)

\( \{ I_1, I_2, \cdots, I_m \} \overset{i.i.d.}{\sim} \mathcal{G}(K, W_\mathcal{I}) \)

\( y_\mathcal{I} = \lambda y_G + (1 - \lambda) y_H \)
## Learned Graph Data Augmentation Approaches

<table>
<thead>
<tr>
<th>Learned GDA</th>
<th>Graph Structure Learning</th>
<th>Graph Adversarial Training</th>
<th>Graph Rationalization</th>
<th>Automated Augmentation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GAug [140]</td>
<td>RobustTraining [125]</td>
<td>GREA [71]</td>
<td>AutoGDA [144]</td>
</tr>
<tr>
<td></td>
<td>LDS [28]</td>
<td>FLAG [63]</td>
<td></td>
<td>JOAO [130]</td>
</tr>
<tr>
<td></td>
<td>Eland [141]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Limitations of Rule-based Approaches

Do not leverage task information and could hurt the downstream performance

(a) Original graph.  
F1 Score: 92.4

(b) Random mod.  
F1 Score: 91.0
Learned Graph Data Augmentation Approaches

• Graph Structure Learning
  • Augment data with good graph structures

• Adversarial Training
  • Augment data with adversarial examples

• Rationalization
  • Augment data by changing graph environment

• Automated Augmentation
  • Automatically combine different augmentations
Graph Structure Learning

Graph Learning + Graph Convolution

What are better graph structures?

- "Noisy" edges should be removed
  - Inter-class edges
- "Missing" edges should be added
  - Intra-class edges

\[
M = \sigma \left( ZZ^T \right), \text{ where } Z = f_{GCL}^{(1)} \left( A, f_{GCL}^{(0)} (A, X) \right)
\]

\(M\) models node similarities
GAug: Interpolation and Sampling

\[ P_{ij} = \alpha M_{ij} + (1 - \alpha)A_{ij} \]

Bernoulli sampling

\[ A'_{ij} \]
Graph Self-supervised Learning

• **Graph Self-Supervised Learning** aims to learn generalizable node/edge/graph representations without using any human-annotated labels

  • **Graph Generative Modeling**
    - Learn generalizable representations by **reconstructing** the node features or/and graph structure

Qiu et al. GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training. KDD 2020
Graph Self-supervised Learning

• **Graph Self-Supervised Learning** aims to learn generalizable node/edge/graph representations **without** using any human-annotated labels

  • **Graph Generative Modeling**
    - Learn generalizable representations by **reconstructing** the node features or/and graph structure

  • **Graph Contrastive Learning (GCL)**
    - Create different views from the unlabeled input graph via data augmentation
Graph Self-supervised Learning

• Graph Self-Supervised Learning aims to learn generalizable node/edge/graph representations without using any human-annotated labels

• Graph Generative Modeling
  ➢ Learn generalizable representations by reconstructing the node features or/and graph structure

• Graph Contrastive Learning (GCL)
  ➢ Create different views from the unlabeled input graph via data augmentation
  ➢ Maximize the agreement between representations of different augmented views of the same instance

Qiu et al. GCC: Graph Contrastive Coding for Graph Neural Network Pre-Training. KDD 2020
Typical Unsupervised Graph Contrastive Learning

- **Graph Data Augmentation**
  - Create different views of each instance (e.g., node, subgraph)
  - Arbitrary graph data augmentation (e.g., edge dropping, feature masking)

- **Encoding Backbone**
  - Encode different augmented views
  - Shallow GNNs (e.g., 2-layer GCN)

- **Contrastive Loss**
  - Maximize the agreement between representations learned from different augmented views
  - Instance-level contrastive learning
Outline

• Introduction and Background
• Topology Issues
• Imbalance Issues
• Short Break

• Bias Issue
• Limited Labeled Data Issues
• Abnormal Graph Data Issues
• Summary
Abnormal Graph Data Issues

- Missing Features
- Adversarially Attacked Data
There are various solutions to deal with missing labels:

- Label propagation (LP)
- Self-supervised learning
- Unsupervised learning
- ...

What if we have missing features?

- Feature propagation

Missing Data

What if we have missing features?
- Feature propagation

**Algorithm 1 Feature Propagation**

1: **Input**: feature vector $\mathbf{x}$, diffusion matrix $\tilde{\mathbf{A}}$
2: $\mathbf{y} \leftarrow \mathbf{x}$
3: **while** $\mathbf{x}$ has not converged **do**
4: $\mathbf{x} \leftarrow \tilde{\mathbf{A}}\mathbf{x}$ \hspace{1em} $\triangleright$ Propagate features
5: $\mathbf{x}_k \leftarrow \mathbf{y}_k$ \hspace{1em} $\triangleright$ Reset known features
6: **end while**
**Comparison** of Feature Propagation to Label Propagation

**Feature Propagation:**
- Propagates features (continuous)
- Prediction is made by a GNN on top of the propagated features
- Uses features, and a low % of them being present is enough for good performance

**Label Propagation:**
- Propagates class labels (discrete)
- Prediction is obtained directly from propagating class labels
- Feature-agnostic

**Experiment Results**

Across different levels of missing features, Feature Propagation achieves the best performance.
Beyond missing features on graphs, can we solve the general missing data problem?

Two ways of approaching missing data problems:

- **Feature imputation**: missing feature values are estimated based on observed values
- **Label prediction**: downstream labels are learned directly from incomplete data

Issues:

- Existing methods fail to make full use of feature values from other observations
- Existing methods tend to make biased assumptions about the missing values by initializing them with special default values
Missing Data

GRAPE: reformulate the tasks as graph tasks

Data Matrix with Missing Values

<table>
<thead>
<tr>
<th></th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
<th>F4</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>O1</td>
<td>0.3</td>
<td>0.5</td>
<td>NA</td>
<td>0.1</td>
<td>y1</td>
</tr>
<tr>
<td>O2</td>
<td>NA</td>
<td>NA</td>
<td>0.6</td>
<td>0.2</td>
<td>y2</td>
</tr>
<tr>
<td>O3</td>
<td>0.3</td>
<td>NA</td>
<td>NA</td>
<td>0.5</td>
<td>?</td>
</tr>
</tbody>
</table>

Labels

GRAPE yields 20% lower mean absolute error for feature imputation, and 10% lower MAE for label prediction.

Observation: Small perturbations of the graph structure and node features lead to misclassification of the target

Zügner, Daniel et al. "Adversarial attacks on neural networks for graph data." KDD 2018
Adversarial Attacked Data

Can we leverage small data perturbations to **improve performance**?
Yes, adversarial training

Adversarial training is the process of crafting adversarial data points, and then injecting them into training data.

\[
\min_{\theta} \mathbb{E}_{(x,y) \sim D} \left[ \max_{\|\delta\|_p \leq \epsilon} L(f_{\theta}(x + \delta), y) \right]
\]

Find the optimal perturbation sample to achieve maximum loss

Find the optimal model parameters to resist the attack of perturbation sample

\(D\): distribution
\(\|\cdot\|_p\): \(l_p\)-norm distance metric
\(\epsilon\): perturbation budget

Kong, Kezhi, et al. "Robust optimization as data augmentation for large-scale graphs." CVPR 2022
Can we leverage small data perturbations to improve performance? Yes, adversarial training

<table>
<thead>
<tr>
<th>Backbone</th>
<th>ogbn-products Test Acc</th>
<th>ogbn-proteins Test ROC-AUC</th>
<th>ogbn-arxiv Test Acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCN</td>
<td>-</td>
<td>72.51±0.35</td>
<td>71.74±0.29</td>
</tr>
<tr>
<td>+FLAG</td>
<td>-</td>
<td>71.71±0.50</td>
<td>72.04±0.20</td>
</tr>
<tr>
<td>GraphSAGE</td>
<td>78.70±0.36</td>
<td>77.68±0.20</td>
<td>71.49±0.27</td>
</tr>
<tr>
<td>+FLAG</td>
<td>79.36±0.57</td>
<td>76.57±0.75</td>
<td>72.19±0.21</td>
</tr>
<tr>
<td>GAT</td>
<td>79.45±0.59</td>
<td>-</td>
<td>73.65±0.11</td>
</tr>
<tr>
<td>+FLAG</td>
<td>81.76±0.45</td>
<td>-</td>
<td>73.71±0.13</td>
</tr>
<tr>
<td>DeeperGCN</td>
<td>80.98±0.20</td>
<td>85.80±0.17</td>
<td>71.92±0.16</td>
</tr>
<tr>
<td>+FLAG</td>
<td>81.93±0.31</td>
<td>85.96±0.27</td>
<td>72.14±0.19</td>
</tr>
</tbody>
</table>
Can we leverage small data perturbations to improve robustness? Yes, adversarial training

A use case: training an MLP on graphs

Reason: to avoid the computation-intensive message passing mechanism

Adversarial Attacked Data

Can we leverage small data perturbations to **improve robustness**?
Yes, adversarial training

A **use case**: training an MLP on graphs

The problem of training an MLP on graphs: sensitive to features

Adversarial Attacked Data

Can we leverage small data perturbations to improve robustness?
Yes, adversarial training

A use case: training an MLP on graphs

The problem of training an MLP on graphs: sensitive to features

Overcome this problem with adversarial training

Adversarial Attacked Data

Can we leverage small data perturbations to improve robustness? Yes, adversarial training

A use case: training an MLP on graphs

GNN Teacher

Representational Similarity Distillation

Soft Label Distillation

MLP Student

Graph

Node Content Features

Node Position Features

Adversarial Feature Augmentation

NOSMOG is as robust as GNNs

Summary

• Introduction and Background
• Topology Issues
• Imbalance Issues
• Short Break

• Bias Issue
• Limited Labeled Data Issues
• Abnormal Graph Data Issues
• Summary
Summary

Data Quality Issues in Graph Machine Learning

- Topology Issues
  - Global Positional Issues
    - Degree Influence
    - Positions to Training Data
  - Local Topology Issues
    - Heterophily Issue
      - Distribution Shift in Target Edges
      - Distribution Shift in Number of Common Neighbors
    - Missing Topology Issues
      - Iterative Learning of Graph Structure and Node Embeddings
      - Application in Recommender Systems
      - Application in Document Question-answering

- Imbalance Issues
  - Node-Level Imbalance
    - Synthesizing Nodes
    - Synthesizing Edges
    - Synthesizing Neighbors
  - Graph-Level Imbalance
    - Quantity Augmentation
    - Structural Augmentation
    - Self-training with confidence selection
    - Anchor-based Mixup
  - Future Work
    - Edge-level Imbalance
    - Retrieval Augmentation
    - Generation Augmentation

- Bias and Fairness Issues
  - Group Fairness
    - Adversarial Learning
    - Structure Modification
    - Balance Message-passing
  - Individual Fairness
    - Measurement
    - Mitigation
    - Cost
  - Degree Fairness
    - Pitfall of GNNs on Low-degree Nodes
    - Degree-specific Graph Convolution
    - Graph Normalization
  - Future Direction
    - Fairness beyond Plain and Static Graphs
    - Fairness with Graphs
    - Benchmark and Evaluation Metrics

- Limited Labeled Data Issues
  - Rule-based Graph Data Augmentation
    - Stochastic Dropping and Masking
    - Mixup and G-Mixup
  - Learned Graph Data Augmentation
    - Graph Structure Learning
    - Rationalization
    - Automated Augmentation
  - Graph Self-supervised Learning
    - Generative Modeling
    - Constrastive Learning

- Abnormal Graph Data Issues
  - Missing Data Issue
  - Feature Propagation for Feature Imputation
  - Bipartite Graph between Instance and Feature
  - Adversarial Attacked Data
    - Adversarial Training for Robustness
    - Adversarial Training for Effective Distillation
Future Directions

- Topology Issue
- Imbalance Data Issue
- Bias and Fairness Issue
- Limited Labeled Data Issue
- Abnormal Graph Data Issue

Existing Data Processing is very time-consuming and labor extensive!
Summary

Intelligent Data Processing Tool

Data → Data Processing Tool → Better Data

Diagnosis
Limited Knowledge → Data Lake 1
Missing Feature → Data Lake 2
Imbalance → Data Lake 3
Discrimination
Out-of-distribution

Search

Retrieval
Data/Task/Issue → Data Lake

Diagnose
Data/Task → Output

Search
Data/Task/Issue/Data Lake → Data