Towards Data-centric Graph Machine Learning

Xin Zheng¹, Shirui Pan²

¹ Monash University
² Griffith University
Tutorial Outline

15 min
- Introduction & Overview
  - Why Data-centric Graph Machine Learning (DC-GML)
  - Framework of DC-GML

30 min
- Frontiers of Graph Data Enhancement
  - Graph Structure Enhancement
  - Graph Feature Enhancement
  - Graph Label Enhancement
  - Graph Size Enhancement

20 min
- Frontiers of Graph Data Exploitation
  - Overview of Graph Data Exploitation Strategies
  - Graph Self-supervised Learning

10 min
- Frontiers of Graph Data-centric MLOps
  - Overview of Graph MLOps
  - GNN Evaluation On OOD Graph Data

10 min
- Future Directions & Conclusion
Part 1: Introduction & Overview

- Why Data-centric Graph Machine Learning (DC-GML)
- Overview of DC-GML Framework
AI system = Code + Data

(model/algorithm)
What is data-centric AI?

“Data-centric AI (DCAI) is the discipline of systematically engineering the data used to build an AI system.” —Andrew Ng

Fig. 1. General comparison between (a) model-centric AI and (b) data-centric AI.
Why data-centric AI matters

An example:

Inspecting steel sheets for defects

<table>
<thead>
<tr>
<th></th>
<th>Steel defect detection</th>
<th>Solar panel</th>
<th>Surface inspection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>76.2%</td>
<td>75.68%</td>
<td>85.05%</td>
</tr>
<tr>
<td>Model-centric</td>
<td>+0% (76.2%)</td>
<td>+0.04% (75.72%)</td>
<td>+0.00% (85.05%)</td>
</tr>
<tr>
<td>Data-centric</td>
<td>+16.9% (93.1%)</td>
<td>+3.06% (78.74%)</td>
<td>+0.4% (85.45%)</td>
</tr>
</tbody>
</table>

Data-centric improves more than model-centric!

[1] A Chat with Andrew on MLOps: From Model-centric to Data-centric AI: https://www.youtube.com/watch?v=06-AZXmwHjo
Why data-centric AI matters

When model design becomes mature, the significance of both the size and quality of the data increases.

Core idea:

Engineering data to enable great “availability and quality” for serving and promoting model-related ML tasks.

Data-centric AI is attracting attentions…

- Exponentially growing DCAI research papers
- DCAI Courses, Workshops, Competitions
- AI Startups

Graphs: A typical & vital instantiation in DCAI

A Graph has nodes/vertices and edges:
- Nodes/vertices → a person in the social network
- Edges → Connection between people

Graphs have the ability of:
- Representing complex structural relationships among massive diverse entities in the real world

Example: A Social Network Graph
Graphs in real-world applications

- Social Networks
- Bibliography Networks
- Protein Interaction Networks
- Knowledge Graphs
- Chemical Compounds
- Traffic Networks
Data-centric graph machine learning (DC-GML) aims to:

- Process, analyze, and understand graph data in entire lifecycle
- Enhancing the quality
- Uncovering the insights
- Developing comprehensive representations
- Working collaboratively with graph ML models under graph MLOps
Why data-centric GML matters

- Taking graph OOD detection as example:

(a) Typical retraining-based graph OOD detection methods

(b) Our proposed data-centric framework for graph OOD detection.

<table>
<thead>
<tr>
<th>ID</th>
<th>OOD</th>
<th>Metric</th>
<th>GCLs</th>
<th>GCLs+</th>
<th>Improv.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENZYMES</td>
<td>PROTEIN</td>
<td>AUC↑</td>
<td>62.97</td>
<td>73.76</td>
<td>+17.14%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUPR↑</td>
<td>62.47</td>
<td>75.27</td>
<td>+20.49%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FPR95</td>
<td>93.33</td>
<td>88.33</td>
<td>-5.36%</td>
</tr>
<tr>
<td>IMDBM</td>
<td>IMDBB</td>
<td>AUC↑</td>
<td>80.52</td>
<td>83.84</td>
<td>+4.12%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUPR↑</td>
<td>74.43</td>
<td>80.16</td>
<td>+7.70%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FPR95</td>
<td>38.67</td>
<td>38.33</td>
<td>-0.88%</td>
</tr>
<tr>
<td>BZR</td>
<td>COX2</td>
<td>AUC↑</td>
<td>75.00</td>
<td>97.31</td>
<td>+29.75%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AUPR↑</td>
<td>62.41</td>
<td>97.17</td>
<td>+55.70%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FPR95</td>
<td>47.50</td>
<td>15.00</td>
<td>-68.42%</td>
</tr>
</tbody>
</table>

Model-centric GML method

Data-centric GML method and improvements

Overview of DC-GML Framework

Graph Data Improvement

- **Graph Structure Enhancement**
  - Graph Structure Learning // Graph Sparsification // Graph Diffusion

- **Graph Feature Enhancement**
  - Graph Feature Completion // Graph Feature Denoising

- **Graph Label Enhancement**
  - Graph Pseudo-labelling // Graph Label Denoising // Graph Class-imbalanced Sampling

- **Graph Size Enhancement**
  - Graph Size Reduction // Graph Data Augmentation

Graph Data Exploitation

- **Graph Self-supervised Learning**
- **Graph Semi-supervised Learning**
- **Graph Active Learning**
- **Graph Transfer Learning**

Graph Data Collection

- **Graph Data Crowdsourcing & Synthesis**

Graph Data Exploration

- **Graph Data Understanding, Visualization & Valuation**

Graph Data Maintenance

- **Graph Data Privacy & Security**
Resources

- More resources and details in our work
  - Survey paper: Towards Data-centric Graph Machine Learning: Review and Outlook
  - Github collection: [https://github.com/Data-Centric-GraphML/awesome-papers](https://github.com/Data-Centric-GraphML/awesome-papers)

Data-centric Graph ML Review & Outlook

DC-GML GitHub Collection
Part 2: Frontiers of
Graph Data Enhancement
Overview of Graph Data Enhancement

Core Strategy

aim to synthesize or modify graph data itself to improve availability and quality by comprehensively fixing potential issues of graph data.

Given a graph $G = (A, X, Y)$, with several essential components of:

1) graph structure $A$;
2) node/edge attribute features $X$;
3) node/graph annotated labels $Y$;
4) the holistic graph $G$ related scale
Outline for Graph Data Enhancement

- Overview of Graph Data Enhancement

- Techniques with Case Studies:
  - Graph Structure Enhancement
  - Graph Feature Enhancement
  - Graph Label Enhancement
  - Graph Size Enhancement
Graph Structure Enhancement

- **Graph Structure Learning**: add, remove, and reweight the edges on noisy or incomplete structures
- **Graph Sparsification**: prune the redundant edges to avoid over-dense structures
- **Graph Diffusion**: establish links with global and long-range structural interactions

Fig. 5. Illustration of graph structure enhancement methods.
Towards Unsupervised Deep Graph Structure Learning

--Case Study on Graph Structure Enhancement

Graph structure learning (GSL): learning graph structure from data when structure is missing or unreliable

Structure inference: Learning from non-structured data

Non-structured data → Learned structure

Structure refinement: Learning with structure-noisy graph data

Noisy graph structure → Learned structure

Towards Unsupervised Deep Graph Structure Learning

Existing methods: 

- Supervised graph structure learning

Limitation-1: High label reliance

Limitation-2: Biased edge distribution learning

Limitation-3: Sub-optimal to other downstream tasks

--Case Study on Graph Structure Enhancement

More practical scenario: **Unsupervised graph structure learning**

aim to optimize the graph structure as an independent task and without label-based supervision.
Comparison: **Supervised GSL** vs. **Unsupervised GSL**

**Supervised GSL**
- Node Labels
- Data
- Input
- GNN-based Model
- Improve
- Benefit
- Node Classification
- Task for Supervision

**Unsupervised GSL (Proposed)**
- Data
- Input
- GNN-based Model
- Improve
- Benefit
- Node Classification
- Node Clustering
- Link Prediction
- Downstream Tasks

**Advantages of UGSL:**
- Does not rely on labels
- Unbiased learning
- Task-agnostic

Towards Unsupervised Deep Graph Structure Learning

- Proposed framework - SUBLIME

To model and regularize the learned graph topology.

To provide a self-optimized supervision signal for GSL.

Towards Unsupervised Deep Graph Structure Learning

---Case Study on Graph Structure Enhancement

- **SUBLIME Performance on Node classification @ Structure Inference**

<table>
<thead>
<tr>
<th>Available Data for GSL</th>
<th>Method</th>
<th>Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Cora</td>
</tr>
<tr>
<td>-</td>
<td>LR</td>
<td>60.8±0.0</td>
</tr>
<tr>
<td>-</td>
<td>Linear SVM</td>
<td>58.9±0.0</td>
</tr>
<tr>
<td>-</td>
<td>MLP</td>
<td>56.1±1.6</td>
</tr>
<tr>
<td>-</td>
<td>GCN&lt;sub&gt;knn&lt;/sub&gt; [22]</td>
<td>66.5±0.4</td>
</tr>
<tr>
<td>-</td>
<td>GAT&lt;sub&gt;knn&lt;/sub&gt; [40]</td>
<td>66.2±0.5</td>
</tr>
<tr>
<td>-</td>
<td>SAGE&lt;sub&gt;knn&lt;/sub&gt; [15]</td>
<td>66.1±0.7</td>
</tr>
<tr>
<td>X, Y</td>
<td>LDS [12]</td>
<td>71.5±0.8</td>
</tr>
<tr>
<td>X, Y, A&lt;sub&gt;knn&lt;/sub&gt;</td>
<td>GRCN [53]</td>
<td>69.6±0.2</td>
</tr>
<tr>
<td>X, Y, A&lt;sub&gt;knn&lt;/sub&gt;</td>
<td>Pro-GNN [20]</td>
<td>69.2±1.4</td>
</tr>
<tr>
<td>X, Y, A&lt;sub&gt;knn&lt;/sub&gt;</td>
<td>GEN [45]</td>
<td>69.1±0.7</td>
</tr>
<tr>
<td>X, Y</td>
<td>IDGL [7]</td>
<td>70.9±0.6</td>
</tr>
<tr>
<td>X, Y</td>
<td>SLAPS [11]</td>
<td><strong>73.4±0.3</strong></td>
</tr>
<tr>
<td>A&lt;sub&gt;knn&lt;/sub&gt;</td>
<td>GDC [23]</td>
<td>68.1±1.2</td>
</tr>
<tr>
<td>X</td>
<td>SLAPS-2s [11]</td>
<td>72.1±0.4</td>
</tr>
<tr>
<td>X</td>
<td>SUBLIME</td>
<td><strong>73.0±0.6</strong></td>
</tr>
</tbody>
</table>

**Towards Unsupervised Deep Graph Structure Learning**

---

**Case Study on Graph Structure Enhancement**

- **SUBLIME Performance**
  - Node classification @ structure refinement

<table>
<thead>
<tr>
<th>Available Data for GSL</th>
<th>Method</th>
<th>Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cora</td>
<td>Citeseeer</td>
</tr>
<tr>
<td>-</td>
<td>GCN</td>
<td>81.5</td>
</tr>
<tr>
<td>-</td>
<td>GAT</td>
<td>83.0±0.7</td>
</tr>
<tr>
<td>-</td>
<td>SAGE</td>
<td>77.4±1.0</td>
</tr>
<tr>
<td>X, Y, A</td>
<td>LDS</td>
<td>83.9±0.6</td>
</tr>
<tr>
<td>X, Y, A</td>
<td>GRCN</td>
<td><strong>84.0±0.2</strong></td>
</tr>
<tr>
<td>X, Y, A</td>
<td>Pro-GNN</td>
<td>82.1±0.4</td>
</tr>
<tr>
<td>X, Y, A</td>
<td>GEN</td>
<td>82.3±0.4</td>
</tr>
<tr>
<td>X, Y, A</td>
<td>IDGL</td>
<td><strong>84.0±0.5</strong></td>
</tr>
<tr>
<td>A</td>
<td>GDC</td>
<td>83.6±0.2</td>
</tr>
<tr>
<td>X, A</td>
<td>SUBLIME</td>
<td><strong>84.2±0.5</strong></td>
</tr>
</tbody>
</table>

- **Node clustering @ structure refinement**

<table>
<thead>
<tr>
<th>Method</th>
<th>Cora</th>
<th>Citeseeer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>C-ACC</td>
<td>NMI</td>
</tr>
<tr>
<td>K-means</td>
<td>50.0</td>
<td>31.7</td>
</tr>
<tr>
<td>SC</td>
<td>39.8</td>
<td>29.7</td>
</tr>
<tr>
<td>GE</td>
<td>30.1</td>
<td>5.9</td>
</tr>
<tr>
<td>DW</td>
<td>52.9</td>
<td>38.4</td>
</tr>
<tr>
<td>DNGR</td>
<td>41.9</td>
<td>31.8</td>
</tr>
<tr>
<td>M-NMF</td>
<td>42.3</td>
<td>25.6</td>
</tr>
<tr>
<td>RMSC</td>
<td>46.6</td>
<td>32.0</td>
</tr>
<tr>
<td>TADW</td>
<td>53.6</td>
<td>36.6</td>
</tr>
<tr>
<td>VGAE</td>
<td>59.2</td>
<td>40.8</td>
</tr>
<tr>
<td>ARGA</td>
<td>64.0</td>
<td>44.9</td>
</tr>
<tr>
<td>MGAE</td>
<td>68.1</td>
<td>48.9</td>
</tr>
<tr>
<td>AGC</td>
<td>68.9</td>
<td><strong>53.7</strong></td>
</tr>
<tr>
<td>DAEGC</td>
<td><strong>70.4</strong></td>
<td>52.8</td>
</tr>
<tr>
<td>SUBLIME</td>
<td><strong>71.3</strong></td>
<td><strong>54.2</strong></td>
</tr>
</tbody>
</table>
Outline for Graph Data Enhancement

❖ Overview of Graph Data Enhancement

❖ Techniques with Case Studies:
  • Graph Structure Enhancement
  • **Graph Feature Enhancement**
  • Graph Label Enhancement
  • Graph Size Enhancement
Graph Feature Enhancement

- **Graph Feature Completion**: focuses on imputing the missing features
- **Graph Feature Denoising**: refining the noisy features.

Fig. 6. Illustration of graph feature enhancement methods.
Robust Graph Representation Learning for Local Corruption Recovery

-- Case study on Graph Feature Completion

- Graph node noise exists widely

- The question is:
  How to eliminate undesirable corruptions the input node attributes to enhance graph representation learning?

Robust Graph Representation Learning for Local Corruption Recovery

- Case study on Graph Feature Completion

- Framework of the proposed MAGNET

Robust Graph Representation Learning for Local Corruption Recovery

- First, mask matrix (M) generation

Robust Graph Representation Learning for Local Corruption Recovery

• Next, find a robust signal representation

This forms a standard formulation of problems that can be solved by ADMM. The associated augmented Lagrangian with respect to the objective function reads

\[
\mathcal{L}(U, Z; Y) := \|\nu Z\|_{p,G} + \frac{1}{2} \|M \odot (U - X)\|_{q,G} + \langle Y, Wu - Z \rangle + \frac{\gamma}{2} \|WU - Z\|^2
\]

where $\gamma > 0$. 

Robust Graph Representation Learning for Local Corruption Recovery

- Finally, learning robust GRL

Robust Graph Representation Learning for Local Corruption Recovery

-- Case study on Graph Feature Completion

Test the performance with node classification tasks

<table>
<thead>
<tr>
<th>Module</th>
<th>attribute injection</th>
<th>meta attack</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cora</td>
<td>Citepeer</td>
</tr>
<tr>
<td>clean</td>
<td>81.26±0.65</td>
<td>71.77±0.29</td>
</tr>
<tr>
<td>GCN</td>
<td>69.06±0.74</td>
<td>57.58±0.71</td>
</tr>
<tr>
<td>APPNP</td>
<td>68.46±0.81</td>
<td>60.04±0.59</td>
</tr>
<tr>
<td>GNNGuard</td>
<td>61.96±0.30</td>
<td>54.94±1.10</td>
</tr>
<tr>
<td>ELasticGNN</td>
<td>77.74±0.79</td>
<td>64.61±0.85</td>
</tr>
<tr>
<td>AIRGNN</td>
<td>76.22±1.75</td>
<td>62.14±0.82</td>
</tr>
<tr>
<td>MAGNETtrue</td>
<td>75.88±0.42</td>
<td>59.22±0.34</td>
</tr>
<tr>
<td>MAGNETgae</td>
<td>79.07±0.56</td>
<td>64.79±0.73</td>
</tr>
<tr>
<td>MAGNETtrue</td>
<td>78.48±0.67</td>
<td>68.55±0.74</td>
</tr>
</tbody>
</table>

- The three baseline graph smoothing methods fail to denoise local corruption within the input.
- MAGNET-gae outperforms its competitors and recovers at most 94% prediction accuracy from the perturbed attributes.
- An accurate mask approximation can push the prediction performance of graph representation up to MAGNET true's scores.

Outline for Graph Data Enhancement

- Overview of Graph Data Enhancement

- Techniques with Case Studies:
  - Graph Structure Enhancement
  - Graph Feature Enhancement
  - Graph Label Enhancement
  - Graph Size Enhancement
Real-world graphs are generally **sparsely and noisily labeled**

Noise in sparsely labeled graphs can **degrade** the performance of GNN:

✗ The size of labels is limited and GNN will overfit to noisy labels

✗ Noisy label information propagates to their unlabeled neighbors

Overview Graph Label Enhancement

- **Graph Pseudo-labelling**: enriching the label information to alleviate the scarce label issue
- **Graph Label Denoising**: removing the redundant noisy label information to clean the noisy labels
- **Graph Class-imbalanced Sampling**: downsampling majority and/or synthesizing minority class labels to tackle the class-imbalanced label issue

Fig. 7. Illustration of graph label enhancement methods.
NRGNN: Learning on Sparsely and Noisily Labeled Graphs

---Case Study on Graph Label Enhancement

Preliminary Analysis

- Linking **an unlabeled node** with **similar labeled nodes** belonging to the same class can increase the robustness against label noise.

- **Strategy**: Extend the label set with accurate pseudo labels by selecting the predictions with high confidence score

NRGNN: Learning on Sparsely and Noisily Labeled Graphs

**NRGNN Framework**

The Proposed NRGNN contains:

1) **Edge predictor**

   *Link unlabeled nodes with similar nodes having noisy/pseudo labels*

2) **Accurate pseudo label miner**

   *Obtain accurate pseudo labels with high confidence score*

3) **GNN classifier**

   *Provide robust predictions*

---

Outline for Graph Data Enhancement

- Overview of Graph Data Enhancement

- Techniques with Case Studies:
  - Graph Structure Enhancement
  - Graph Feature Enhancement
  - Graph Label Enhancement
  - Graph Size Enhancement
Graph Size Enhancement

- **Graph Size Reduction**: the oversized large-scale graphs with redundant information
- **Graph Data Augmentation**: small-scale graphs with limited data sources and insufficient information

Fig. 8. Illustration of graph data-centric size enhancement methods.
Graph Size Enhancement

- **Graph Size Reduction**: the oversized large-scale graphs with redundant information
- **Graph Data Augmentation**: small-scale graphs with limited data sources and insufficient information

Fig. 8. Illustration of graph data-centric size enhancement methods.
Background of Graph Condensation

What is graph condensation?

aim to reduce the size of a large-scale graph by synthesizing a small-scale condensed graph

→ the small-scale condensed graph achieves comparable test performance as the large-scale graph when training the same GNN model.

Background of Graph Condensation

Requirements, Advantages, & Applications

1) Why need GC [Requirements]?

Modelling large-scale graphs hinders GNN development with heavy costs

[X] Heavy costs on: graph data storage, computation, and memory

Table 1: Model serving space

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Model size</th>
<th>Training graph size</th>
<th>Training feature size</th>
<th>Total serving size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arxiv</td>
<td>1.4MB</td>
<td>5.9MB</td>
<td>46.5MB</td>
<td>53.8MB</td>
</tr>
<tr>
<td>Reddit</td>
<td>7.6MB</td>
<td>86.0MB</td>
<td>370.7MB</td>
<td>464.3MB</td>
</tr>
<tr>
<td>Product</td>
<td>4.8MB</td>
<td>87.2MB</td>
<td>78.6MB</td>
<td>170.6MB</td>
</tr>
<tr>
<td>Amazon2M</td>
<td>3.0MB</td>
<td>485.4MB</td>
<td>684.0MB</td>
<td>1.17GB</td>
</tr>
</tbody>
</table>

Background of Graph Condensation

Requirements, Advantages, & Applications

2) How GC benefit [Advantages]?
Using condensed graph as substitution to facilitate GNN training:
  • Alleviated graph data storage/computation/memory costs

3) What practical applications of GC [Applications]?
  • Graph Neural Architecture Search (GraphNAS)
    By searching on a small-scale condensed graph, accelerating new
    GNN architecture development in GraphNAS
  • Privacy Protection
  • Adversarial Robustness

Our Solution: Structure-free Graph Condensation

---Case Study on Graph Size Enhancement

- Existing works:

  \( \mathcal{T} = (X, A, Y) \rightarrow \mathcal{T}' = (X', A', Y'), \quad \text{GC.} \)

- Our SFGC:

  \( \mathcal{T} = (X, A, Y) \rightarrow S = (\bar{X}, \bar{I}, \bar{Y}) = S = (\bar{X}, \bar{Y}), \quad \text{SFGC.} \)

- Our Solution:
  
  ✓ Structure-free paradigm
  
  ✓ Long-range parameter matching schema

  • Only synthesizes a small scaled node set to train a GNN/MLP
  
  • Implicitly encodes topology structure into node attributes

Structure-free Graph Condensation

**Condensing large-scale graph into only node set without structures!**

---

**Figure 1. Overall pipeline of the proposed Structure-Free Graph Condensation (SFGC) framework**

- **Input:** large-scale T, GNN(T)
- **Output:** small-scale condensed S
  - S1: train expert GNN on large-scale T
  - S2-3: long-term meta training trajectory matching with condensed S
  - S4: update S
  - S5: dynamically evaluates S with a GNTK-based score

---

Experiments of SFGC

Generally, SFGC achieves the best performance on the node classification task with 13 of 15 cases (five datasets and three condensation ratios for each of them), illustrating the high quality and expressiveness of the condensed graph-free data synthesized by our SFGC.


--Case Study on Graph Size Enhancement

Table 1: Node classification performance (ACC±std) comparison between condensation methods and other graph size reduction methods with different condensation ratios. (Best results are in bold, and the second-bests are underlined.)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Citeseer</td>
<td>0.9%</td>
<td>52.2±0.4</td>
<td>54.4±4.4</td>
<td>57.1±1.5</td>
<td>52.4±2.8</td>
<td>66.8±1.5</td>
<td>71.4±0.8</td>
<td>70.5±1.2</td>
<td>71.4±0.5</td>
</tr>
<tr>
<td></td>
<td>1.8%</td>
<td>59.0±0.5</td>
<td>64.2±1.7</td>
<td>66.7±1.0</td>
<td>64.3±1.0</td>
<td>66.9±0.9</td>
<td>69.8±1.1</td>
<td>70.6±0.9</td>
<td>72.4±0.4</td>
</tr>
<tr>
<td></td>
<td>3.6%</td>
<td>65.3±0.5</td>
<td>69.1±0.1</td>
<td>69.0±0.1</td>
<td>69.1±0.1</td>
<td>66.3±1.5</td>
<td>69.4±1.4</td>
<td>69.8±1.4</td>
<td>70.6±0.7</td>
</tr>
<tr>
<td>Cora</td>
<td>1.3%</td>
<td>31.2±0.2</td>
<td>63.6±3.7</td>
<td>67.0±1.3</td>
<td>64.0±2.3</td>
<td>67.3±1.9</td>
<td>75.9±1.2</td>
<td>79.8±1.3</td>
<td>80.1±0.4</td>
</tr>
<tr>
<td></td>
<td>2.6%</td>
<td>65.2±0.6</td>
<td>72.8±1.1</td>
<td>73.4±0.4</td>
<td>73.2±1.2</td>
<td>67.6±3.5</td>
<td>75.7±0.9</td>
<td>80.1±0.6</td>
<td>81.7±0.5</td>
</tr>
<tr>
<td></td>
<td>5.2%</td>
<td>70.6±0.1</td>
<td>76.8±0.1</td>
<td>76.8±0.1</td>
<td>76.7±0.1</td>
<td>67.7±2.2</td>
<td>76.0±0.9</td>
<td>79.3±0.3</td>
<td>81.6±0.8</td>
</tr>
<tr>
<td>Ogbn-arxiv</td>
<td>0.05%</td>
<td>35.4±0.3</td>
<td>47.1±3.9</td>
<td>52.4±1.8</td>
<td>47.2±3.0</td>
<td>58.6±0.4</td>
<td>61.3±0.5</td>
<td>59.2±1.1</td>
<td>65.5±0.7</td>
</tr>
<tr>
<td></td>
<td>0.25%</td>
<td>43.5±0.2</td>
<td>57.3±1.1</td>
<td>58.6±1.2</td>
<td>56.8±0.8</td>
<td>59.9±0.3</td>
<td>62.4±0.4</td>
<td>63.2±0.3</td>
<td>66.1±0.4</td>
</tr>
<tr>
<td></td>
<td>0.5%</td>
<td>50.4±0.1</td>
<td>60.0±0.9</td>
<td>60.4±0.8</td>
<td>60.3±0.4</td>
<td>59.5±0.3</td>
<td>63.1±0.5</td>
<td>64.0±0.4</td>
<td>66.8±0.4</td>
</tr>
<tr>
<td>Flickr</td>
<td>0.1%</td>
<td>41.9±0.2</td>
<td>41.8±2.0</td>
<td>42.5±1.8</td>
<td>42.0±0.7</td>
<td>46.3±0.2</td>
<td>45.9±0.1</td>
<td>46.5±0.4</td>
<td>46.6±0.2</td>
</tr>
<tr>
<td></td>
<td>0.5%</td>
<td>44.5±0.1</td>
<td>44.0±0.4</td>
<td>43.9±0.9</td>
<td>43.2±0.1</td>
<td>45.9±0.1</td>
<td>45.0±0.2</td>
<td>47.1±0.1</td>
<td>47.0±0.1</td>
</tr>
<tr>
<td></td>
<td>1%</td>
<td>44.6±0.1</td>
<td>44.6±0.2</td>
<td>44.4±0.6</td>
<td>44.1±0.4</td>
<td>45.8±0.1</td>
<td>45.0±0.1</td>
<td>47.1±0.1</td>
<td>47.1±0.1</td>
</tr>
<tr>
<td>Reddit</td>
<td>0.05%</td>
<td>40.9±0.5</td>
<td>46.1±4.4</td>
<td>53.1±2.5</td>
<td>46.6±2.3</td>
<td>88.2±0.2</td>
<td>88.4±0.4</td>
<td>88.0±1.8</td>
<td>89.7±0.2</td>
</tr>
<tr>
<td></td>
<td>0.1%</td>
<td>42.8±0.8</td>
<td>58.0±2.2</td>
<td>62.7±1.0</td>
<td>53.0±3.3</td>
<td>89.5±0.1</td>
<td>89.3±0.1</td>
<td>89.6±0.7</td>
<td>90.0±0.3</td>
</tr>
<tr>
<td></td>
<td>0.2%</td>
<td>47.4±0.9</td>
<td>66.3±1.9</td>
<td>71.0±1.6</td>
<td>58.5±2.1</td>
<td>90.5±1.2</td>
<td>88.8±0.4</td>
<td>90.1±0.5</td>
<td>90.3±0.3</td>
</tr>
</tbody>
</table>
Part 3: Frontiers of
Graph Data Exploitation
Outline for Graph Data Exploitation

- Overview of Graph Data Exploitation

- Techniques with Case Studies:
  - Graph Self-supervised Learning
  - Graph Semi-supervised Learning
  - Graph Active Learning
  - Graph Transfer Learning
Overview of Graph Data Exploitation

Despite much effort on improving graph data quality, new graph data with high dynamics, complexity, diversity comes every day…

Core Question:

- What if directly graph data enhancement not feasible?
- What if after enhancement, it’s still not enough to instruct the graph model development?

Graph Data Exploitation

Graph Data

Enhanced Graph Data

How to Learn from Graph Data with Limited-availability & Low-quality?

Graph ML Models
Overview of Graph Data Exploitation

- Category of Graph Data Exploitation:
  - **Graph Self-Supervised Learning**
    - Input: Graph $G$ without Labels
    - Supervised: Self-generated supervision signals
    - Model: $G$ without Labels
  - **Graph Semi-Supervised Learning**
    - Input: Graph $G$ with Limited Labels
    - Supervised: Supervision from labels
    - Model: $G$ with Limited Labels
  - **Graph Active Learning**
    - Input: Graph $G$ with Initial Labels (iteration $= 0$)
    - Supervised: Supervision from labels
    - Model: Update $G$ with New Labels (iteration $= 1$)
  - **Graph Transfer Learning**
    - Supervised: Supervision from labels
    - Model: Testing $G$ (Target)

Strength of supervision signal:
- Weak: self-supervised
- Semi-supervised
- Active learning
- Transfer learning
- Strong
Outline for Graph Data Exploitation

- Overview of Graph Data Exploitation
- Techniques with Case Studies:
  - Graph Self-supervised Learning
  - Graph Semi-supervised Learning
  - Graph Active Learning
  - Graph Transfer Learning
Motivation of Graph Self-supervised Learning

When lacking of sufficient supervision signals, the potential problems are...

- Expensive cost of data collection and annotation
- Poor generalization
- Vulnerable to label-related adversarial attacks
Typical Categories of GSSL

(1) Generation-based

(2) Auxiliary Property-based

(3) Contrast-based

(4) Hybrid
Typical Categories of GSSL

1. Generation-based
   - Input Graph
   - Perturbation
   - Encoder $f_0$
   - Pretext Decoder $P_0$
   - Reconstructed Graph
   - Reconstruction Loss

2. Auxiliary Property-based
   - Input Graph
   - Property Extraction
   - Encoder $f_0$
   - Pretext Decoder $P_0$
   - Predicted Properties
   - Prediction Loss

3. Contrast-based
   - Input Graph
   - Augmentation
   - Encoder $f_0$
   - Pretext Decoder $P_0$
   - Representations
   - Target Agreements
   - Positive/Negative

4. Hybrid
   - Input Graph
   - Encoder $f_0$
   - Pretext Tasks
   - $P_{\phi_1}$
Existing Problems - Slow Computation with Node Comparison

Most contrastive-learning approaches

- rely on node-to-node comparison
- require heavy gradient computation

Existing Problems - Slow Computation with Node Comparison

- Existing typical Deep Graph Infomax (DGI) framework

**MI maximization** between nodes and **summary vector**

\[
\mathcal{L}_{\text{DGI}} = \frac{1}{2N} \left( \sum_{i=1}^{N} \log \mathcal{D}(z_i, \bar{s}) + \log(1 - \mathcal{D}(\bar{z}_i, s)) \right),
\]

Rethinking Existing DGI

Our important findings:

- Value in **summary vector** $s$ almost becomes constant vector with no variance
- DGI loss can be further simplified as BCE loss

<table>
<thead>
<tr>
<th>Activation</th>
<th>Statistics</th>
<th>Cora</th>
<th>CiteSeer</th>
<th>PubMed</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReLU/ReLU/PRelu</td>
<td>Mean</td>
<td>0.50</td>
<td>0.50</td>
<td>0.50</td>
</tr>
<tr>
<td></td>
<td>Std</td>
<td>1.3e-03</td>
<td>1.0e-04</td>
<td>4.0e-04</td>
</tr>
<tr>
<td></td>
<td>Range</td>
<td>1.4e-03</td>
<td>8.0e-04</td>
<td>1.5e-03</td>
</tr>
<tr>
<td>Sigmoid</td>
<td>Mean</td>
<td>0.62</td>
<td>0.62</td>
<td>0.62</td>
</tr>
<tr>
<td></td>
<td>Std</td>
<td>5.4e-05</td>
<td>2.9e-05</td>
<td>6.6e-05</td>
</tr>
<tr>
<td></td>
<td>Range</td>
<td>3.6e-03</td>
<td>3.0e-03</td>
<td>3.2e-03</td>
</tr>
</tbody>
</table>

Dataset: 0 0.2 0.4 0.6 0.8 1.0

Cora 70.3±0.7 82.4±0.2 82.3±0.3 82.5±0.4 82.3±0.3 82.5±0.1
CiteSeer 61.8±0.8 71.7±0.6 71.9±0.7 71.6±0.9 71.7±1.0 71.6±0.8
PubMed 68.3±1.5 77.8±0.5 77.9±0.8 77.7±0.9 77.4±1.1 77.2±0.9

Set $\epsilon$ to 1 for $s = \epsilon I = I$, and remove $w$ in $D(z_i, \tilde{s}) = z_i \cdot \tilde{s}$.

$$L_{DGI} = \frac{1}{2N} \left( \sum_{i=1}^{N} \log D(z_i, s) + \log(1 - D(\tilde{z}_i, s)) \right),$$

$$= \frac{1}{2N} \left( \sum_{i=1}^{N} \log(z_i \cdot s) + \log(1 - \tilde{z}_i \cdot s)) \right),$$

$$= \frac{1}{2N} \left( \sum_{i=1}^{N} \log(\text{sum}(z_i)) + \log(1 - \text{sum}(\tilde{z}_i)) \right),$$

Our Solution: **Group Discrimination (GD)**

--- Case Study on Graph Self-supervised Learning

Summarisation (e.g., sum):

\[
\begin{align*}
\mathcal{R}^{1 \times D} & \rightarrow \mathcal{R}^{1 \times 1}
\end{align*}
\]

- **Positive Group:**
  
  *Summarised Node representations generated with original or augmented graph.*

- **Negative Group:**
  
  *Summarised Node representations generated with corrupted graph.*

---

Our Solution: **Group Discrimination (GD)**

---Case Study on Graph Self-supervised Learning

Use a very simple **BCE loss** to conduct discrimination

A very simple binary classification task: discriminating positive/negative samples

\[ \mathcal{L}_{BCE} = -\frac{1}{2N} \left( \sum_{i=1}^{2N} y_i \log h_i + (1 - y_i) \log(1 - h_i) \right) \]

If positive → \( y = 1 \), else → \( y = 0 \)

where \( h_i \in \mathcal{R}^{1x1} \) is the summarised node embedding/binary prediction for a node \( i \)

Proposed Framework: Graph Group Discrimination (GGD)

Augmentation                 Corruption                                          Encoding                       Aggregation

Performance of Graph Group Discrimination (GGD)

---Case Study on Graph Self-supervised Learning

### Small-to-Medium scale Dataset

<table>
<thead>
<tr>
<th>Data</th>
<th>Method</th>
<th>Cora</th>
<th>CiteSeer</th>
<th>PubMed</th>
<th>Comp</th>
<th>Photo</th>
</tr>
</thead>
<tbody>
<tr>
<td>X, A, Y</td>
<td>GCN</td>
<td>81.5</td>
<td>70.3</td>
<td>79.0</td>
<td>76.3±0.5</td>
<td>87.3±1.0</td>
</tr>
<tr>
<td>X, A, Y</td>
<td>GAT</td>
<td>83.0±0.7</td>
<td>72.5±0.7</td>
<td>79.0±0.3</td>
<td>79.3±1.1</td>
<td>86.2±1.5</td>
</tr>
<tr>
<td>X, A, Y</td>
<td>SGC</td>
<td>81.0±0.0</td>
<td>71.9±0.1</td>
<td>78.9±0.0</td>
<td>74.4±0.1</td>
<td>86.4±0.0</td>
</tr>
<tr>
<td>X, A, Y</td>
<td>CG3</td>
<td>83.4±0.7</td>
<td>73.6±0.8</td>
<td>80.2±0.8</td>
<td>79.9±0.6</td>
<td>89.4±0.5</td>
</tr>
<tr>
<td>X, A</td>
<td>DGI</td>
<td>81.7±0.6</td>
<td>71.5±0.7</td>
<td>77.3±0.6</td>
<td>75.9±0.6</td>
<td>83.1±0.5</td>
</tr>
<tr>
<td>X, A</td>
<td>GMI</td>
<td>82.7±0.2</td>
<td>73.0±0.3</td>
<td>80.1±0.2</td>
<td>76.8±0.1</td>
<td>85.1±0.1</td>
</tr>
<tr>
<td>X, A</td>
<td>MVGRL</td>
<td>82.9±0.7</td>
<td>72.6±0.7</td>
<td>79.4±0.3</td>
<td>79.0±0.6</td>
<td>87.3±0.3</td>
</tr>
<tr>
<td>X, A</td>
<td>GRACE</td>
<td>80.0±0.4</td>
<td>71.7±0.6</td>
<td>79.5±1.1</td>
<td>71.8±0.4</td>
<td>81.8±1.0</td>
</tr>
<tr>
<td>X, A</td>
<td>BGRL</td>
<td>80.5±1.0</td>
<td>71.0±1.2</td>
<td>79.5±0.6</td>
<td>89.2±0.9</td>
<td>91.2±0.8</td>
</tr>
<tr>
<td>X, A</td>
<td>GBT</td>
<td>81.0±0.5</td>
<td>70.8±0.2</td>
<td>79.0±0.1</td>
<td>88.5±1.0</td>
<td>91.1±0.7</td>
</tr>
<tr>
<td>X, A</td>
<td>GGD</td>
<td>84.1±0.4</td>
<td>73.0±0.6</td>
<td>81.3±0.8</td>
<td>90.1±0.9</td>
<td>92.5±0.6</td>
</tr>
</tbody>
</table>

### Time Consumption Improvement (epoch per second)

<table>
<thead>
<tr>
<th>Method</th>
<th>Cora</th>
<th>CiteSeer</th>
<th>PubMed</th>
<th>Comp</th>
<th>Photo</th>
</tr>
</thead>
<tbody>
<tr>
<td>DGI</td>
<td>0.085</td>
<td>0.134</td>
<td>0.158</td>
<td>0.171</td>
<td>0.059</td>
</tr>
<tr>
<td>GMI</td>
<td>0.394</td>
<td>0.497</td>
<td>2.285</td>
<td>1.297</td>
<td>0.637</td>
</tr>
<tr>
<td>MVGRL</td>
<td>0.123</td>
<td>0.171</td>
<td>0.488</td>
<td>0.663</td>
<td>0.468</td>
</tr>
<tr>
<td>GRACE</td>
<td>0.056</td>
<td>0.092</td>
<td>0.893</td>
<td>0.546</td>
<td>0.203</td>
</tr>
<tr>
<td>BGRL</td>
<td>0.085</td>
<td>0.094</td>
<td>0.147</td>
<td>0.337</td>
<td>0.273</td>
</tr>
<tr>
<td>GBT</td>
<td>0.073</td>
<td>0.072</td>
<td>0.103</td>
<td>0.492</td>
<td>0.173</td>
</tr>
<tr>
<td>GGD</td>
<td>0.010</td>
<td>0.021</td>
<td>0.015</td>
<td>0.016</td>
<td>0.009</td>
</tr>
<tr>
<td>Improve</td>
<td>7.3-39.4</td>
<td>3.4-23.7</td>
<td>6.9-152.3</td>
<td>10.7-15.3</td>
<td>19.2-70.8</td>
</tr>
</tbody>
</table>

### Memory Consumption Improvement (MB)

<table>
<thead>
<tr>
<th>Method</th>
<th>Cora</th>
<th>CiteSeer</th>
<th>PubMed</th>
<th>Comp</th>
<th>Photo</th>
</tr>
</thead>
<tbody>
<tr>
<td>DGI</td>
<td>4,189</td>
<td>8,199</td>
<td>11,471</td>
<td>7,991</td>
<td>4,946</td>
</tr>
<tr>
<td>GMI</td>
<td>4,527</td>
<td>5,467</td>
<td>14,697</td>
<td>10,655</td>
<td>5,219</td>
</tr>
<tr>
<td>MVGRL</td>
<td>5,381</td>
<td>5,429</td>
<td>6,619</td>
<td>6,645</td>
<td>6,645</td>
</tr>
<tr>
<td>GRACE</td>
<td>1,913</td>
<td>2,043</td>
<td>12,597</td>
<td>8,129</td>
<td>4,881</td>
</tr>
<tr>
<td>BGRL</td>
<td>1,627</td>
<td>1,749</td>
<td>2,299</td>
<td>5,069</td>
<td>3,303</td>
</tr>
<tr>
<td>GBT</td>
<td>1,651</td>
<td>1,799</td>
<td>2,461</td>
<td>5,037</td>
<td>2,641</td>
</tr>
<tr>
<td>GGD</td>
<td>1,475</td>
<td>1,587</td>
<td>1,629</td>
<td>1,787</td>
<td>1,637</td>
</tr>
<tr>
<td>Improve</td>
<td>10.7-72.6%</td>
<td>11.8-80.6%</td>
<td>27.2-85.8%</td>
<td>64.5-83.2%</td>
<td>38.0-75.4%</td>
</tr>
</tbody>
</table>

Performance of Graph Group Discrimination (GGD)

Using only 0.18 seconds and 69.8% less memory to reach SOTA.

10783 faster than existing methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>Valid</th>
<th>Test</th>
<th>Memory</th>
<th>Time</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised GCN</td>
<td>73.0±0.2</td>
<td>71.7±0.3</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MLP</td>
<td>57.7±0.4</td>
<td>55.5±0.2</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Node2vec</td>
<td>71.3±0.1</td>
<td>70.1±0.1</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DGI</td>
<td>71.3±0.1</td>
<td>70.3±0.2</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GRACE(10k epos)</td>
<td>72.6±0.2</td>
<td>71.5±0.1</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BGRL(10k epos)</td>
<td>72.5±0.1</td>
<td>71.6±0.1</td>
<td>OOM (Full-graph)</td>
<td>/</td>
<td>/</td>
</tr>
<tr>
<td>GBT(300 epos)</td>
<td>71.0±0.1</td>
<td>70.1±0.2</td>
<td>14,959MB</td>
<td>6.47</td>
<td>1,941.00</td>
</tr>
<tr>
<td>GGD(1 epo)</td>
<td>72.7±0.3</td>
<td>71.6±0.5</td>
<td>4,513MB</td>
<td>69.8%</td>
<td>0.18</td>
</tr>
</tbody>
</table>

Fast convergence → converge with only 1 epoch

---

Outline for Graph Data Exploitation

- Overview of Graph Data Exploitation
- Techniques with Case Studies:
  - Graph Self-supervised Learning
  - Graph Semi-supervised Learning
  - Graph Active Learning
  - Graph Transfer Learning
Background of Graph Semi-supervised Learning

- Graph Semi-supervised Learning: **Only limited labels are provided**
  - Core idea from DC-GML view:
    - Learn to fully leverage/exploit the unlabeled part and collaborate with the labeled part
  - Methodology: **Regularization & Pseudo Labelling**
## Category of Graph Semi-supervised Learning

- **Category from DC-GML view**

<table>
<thead>
<tr>
<th>Graph Semi-supervised Learning</th>
<th>Techniques</th>
<th>Categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zhu et al. [223]</td>
<td>Graph Laplacian regularization</td>
<td>Regularization-based</td>
</tr>
<tr>
<td>Zhou et al. [216]</td>
<td>Graph Laplacian regularization</td>
<td>Regularization-based</td>
</tr>
<tr>
<td>Zhou et al. [217]</td>
<td>Local smoothness under homophily</td>
<td>Regularization-based</td>
</tr>
<tr>
<td>Li et al. [82]</td>
<td>Self-training with training set extension</td>
<td>Pseudo-labelling</td>
</tr>
<tr>
<td>GRAND [39]</td>
<td>L2 distance-based consistency</td>
<td>Regularization-based</td>
</tr>
<tr>
<td>M3S [148]</td>
<td>Clustering-based pseudo label generation</td>
<td>Pseudo-labelling</td>
</tr>
<tr>
<td>SimP-GCN [68]</td>
<td>Feature-level similarity in pairwise distance</td>
<td>Regularization-based</td>
</tr>
<tr>
<td>GCN-LPA [166]</td>
<td>Edge weights with graph structure regularization</td>
<td>Regularization-based</td>
</tr>
<tr>
<td>CG³ [162]</td>
<td>Self-supervised objective based regularization</td>
<td>Regularization-based</td>
</tr>
<tr>
<td>GCPN [163]</td>
<td>Contrastive and possion learning based regularization</td>
<td>Regularization-based</td>
</tr>
<tr>
<td>CycProp [88]</td>
<td>High-quality contextual node selection</td>
<td>Pseudo-labelling</td>
</tr>
</tbody>
</table>
Outline for Graph Data Exploitation

- Overview of Graph Data Exploitation
- Techniques with Case Studies:
  - Graph Self-supervised Learning
  - Graph Semi-supervised Learning
  - Graph Active Learning
  - Graph Transfer Learning
Background of Graph Active Learning

Given the fixed cost (e.g., human labour and expert knowledge) for label, how can we fully make the best use of such labelling budget?

- **Graph Active Learning**: dynamically select the samples to label during the training procedure

  In the practical active learning process, the nodes to label are selected automatically by the models following several **selection criteria**.

- **Category from DC-GML view**:
  - Rule-based
  - Reinforcement learning-based,
  - Influence function-based,
  - Other hybrid methods
### Table 7. Summary of methods in graph active learning.

<table>
<thead>
<tr>
<th>Graph Active Learning</th>
<th>Techniques</th>
<th>Categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGE [13]</td>
<td>Information entropy, density, and centrality rules</td>
<td>Rule-based</td>
</tr>
<tr>
<td>ANRMB [42]</td>
<td>Multi-armed bandit mechanism</td>
<td>Rule-based</td>
</tr>
<tr>
<td>ActiveHNE [24]</td>
<td>Multi-armed bandit mechanism on heterogeneous graphs</td>
<td>Rule-based</td>
</tr>
<tr>
<td>FeatProp [183]</td>
<td>Closest cluster center based labelling</td>
<td>Clustering-based</td>
</tr>
<tr>
<td>ATNE [65]</td>
<td>Active transfer learning based node selection</td>
<td>Rule-based</td>
</tr>
<tr>
<td>ASGN [50]</td>
<td>Sample diversity based node selection</td>
<td>Rule-based</td>
</tr>
<tr>
<td>GPA [54]</td>
<td>GCN-based policy network</td>
<td>RL-based</td>
</tr>
<tr>
<td>MetAL [103]</td>
<td>Meta-gradients estimation</td>
<td>Meta Learning-based</td>
</tr>
<tr>
<td>SEAL [85]</td>
<td>Adversarial learning with divergence value</td>
<td>Adversarial-based</td>
</tr>
<tr>
<td>GRAIN [205]</td>
<td>Diversified influence maximization objective</td>
<td>Influence-based</td>
</tr>
<tr>
<td>RIM [204]</td>
<td>Label reliability based influence score scaling</td>
<td>Influence-based</td>
</tr>
<tr>
<td>Attent [219]</td>
<td>Active graph alignment</td>
<td>Influence-based</td>
</tr>
<tr>
<td>ALG [202]</td>
<td>Clustering-based density &amp; Attention-based score</td>
<td>Metric-based</td>
</tr>
<tr>
<td>ALLIE [27]</td>
<td>Integrated graph coarsening and focal loss</td>
<td>RL-based</td>
</tr>
<tr>
<td>BIGENE [207]</td>
<td>Q-value decomposition with batch sampling selection</td>
<td>RL-based</td>
</tr>
<tr>
<td>IGP [203]</td>
<td>Information gain propagation for soft labelling</td>
<td>Influence-based</td>
</tr>
<tr>
<td>JuryGCN [75]</td>
<td>Jackknife uncertainty estimation</td>
<td>Influence-based</td>
</tr>
</tbody>
</table>
Outline for Graph Data Exploitation

- Overview of Graph Data Exploitation

- Techniques with Case Studies:
  - Graph Self-supervised Learning
  - Graph Semi-supervised Learning
  - Graph Active Learning
  - Graph Transfer Learning
Graph Transfer Learning

- Graph data distribution shift between the training and test graph data widely exits.
- Shifts might encompass attributes like node features, graph structures, and label distributions.

**According to whether label spaces of graphs is changed or not, the category**

**a) Close-set shift**: label space unchanged  
**b) Open-set shift**: new label classes emerge

---

**Fig. 9. Illustration of graph transfer learning in graph data-centric close-set shift and open-set shift.**
**Category from DC-GML view**

Table 8. Summary of methods in graph transfer learning.

<table>
<thead>
<tr>
<th>Graph Transfer Learning</th>
<th>Techniques</th>
<th>Categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>DANE [206]</td>
<td>Adversarial learning regularization</td>
<td>Close-set shift</td>
</tr>
<tr>
<td>UDA-GCN [181]</td>
<td>Adversarial learning with dual-GNN</td>
<td>Close-set shift</td>
</tr>
<tr>
<td>ACDNE [142]</td>
<td>Node affinity &amp; topological proximity preservation</td>
<td>Close-set shift</td>
</tr>
<tr>
<td>OpenWGL [182]</td>
<td>Variational graph autoencoder</td>
<td>Open-set shift</td>
</tr>
<tr>
<td>PGL [101]</td>
<td>Class space decomposition</td>
<td>Open-set shift</td>
</tr>
<tr>
<td>SRGNN [221]</td>
<td>Central moment discrepancy (CMD) measurement</td>
<td>Close-set shift</td>
</tr>
<tr>
<td>SOGA [104]</td>
<td>Mutual information maximization</td>
<td>Close-set shift</td>
</tr>
<tr>
<td>DGDA [14]</td>
<td>Domain and semantic separation</td>
<td>Close-set shift</td>
</tr>
<tr>
<td>SRNC [222]</td>
<td>Unified domain adaptation GNN</td>
<td>Close-set/Open-set shifts</td>
</tr>
</tbody>
</table>
Part 4: Frontiers of
Graph Data-centric MLOps
Outline for Graph Data-centric MLOps

- Overview of Graph Data-centric MLOps

- Techniques:
  - Graph Data Crowdsourcing and Synthesis
  - Graph Data Understanding, Visualization, and Valuation
  - Graph Data Privacy and Security
  - Graph MLOps

- Case Study in Graph MLOps:
  - 【NeurIPS-2023】“GNNEvaluator: Evaluating GNN Performance On Unseen Graphs Without Labels”
## Outline for Graph Data-centric MLOps

<table>
<thead>
<tr>
<th>Phases</th>
<th>Goals</th>
<th>Methods &amp; Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Graph Data Collection</strong></td>
<td><strong>Graph Data Crowdsourcing</strong></td>
<td>Amazon Mechanical Turk [4], Tang et al. [152], Cao et al. [15]</td>
</tr>
<tr>
<td></td>
<td><strong>Graph Data Synthesis</strong></td>
<td>SBMs [145], Koller et al. [78], Ying et al. [188], Unsupervised methods [106, 120], Semi-supervised methods [38, 111, 132, 155]</td>
</tr>
<tr>
<td><strong>Graph Data Exploration</strong></td>
<td><strong>Graph Data Understanding &amp; Visualization</strong></td>
<td>NetworkX [31], igraph [60], Neo4j [107]</td>
</tr>
<tr>
<td></td>
<td><strong>Graph Data Valuation</strong></td>
<td>GraphSVX [37]</td>
</tr>
<tr>
<td><strong>Graph Data Maintenance</strong></td>
<td><strong>Graph Data Privacy</strong></td>
<td>TrustworthyGNN [200], Zhang et al. [197], Liu et al. [92], Yu et al. [192], Mulle et al. [105], PGAS [198], Federatedscope-GNN [174], Tan et al. [151]</td>
</tr>
<tr>
<td></td>
<td><strong>Graph Data Security</strong></td>
<td>Sandhu et al. [135], Abidi et al. [1], Li et al. [87]</td>
</tr>
<tr>
<td><strong>Graph MLOps</strong></td>
<td></td>
<td>Kubeflow [81], Amazon SageMaker [6], Amazon Neptune [179]</td>
</tr>
<tr>
<td>Phases</td>
<td>Goals</td>
<td>Methods &amp; Tools</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------------------------------------</td>
<td>-----------------------------------------------------</td>
</tr>
<tr>
<td><strong>Graph Data Collection</strong></td>
<td><strong>Graph Data Crowdsourcing</strong></td>
<td>Amazon Mechanical Turk [4], Tang et al. [152], Cao et al. [15]</td>
</tr>
<tr>
<td></td>
<td><strong>Graph Data Synthesis</strong></td>
<td>SBMs [145], Koller et al. [78], Ying et al. [188], Unsupervised methods [106, 120], Semi-supervised methods [38, 111, 132, 155]</td>
</tr>
<tr>
<td><strong>Graph Data Exploration</strong></td>
<td><strong>Graph Data Understanding &amp; Visualization</strong></td>
<td>NetworkX [31], igraph [60], Neo4j [107]</td>
</tr>
<tr>
<td></td>
<td><strong>Graph Data Valuation</strong></td>
<td>GraphSVX [37]</td>
</tr>
<tr>
<td><strong>Graph Data Maintenance</strong></td>
<td><strong>Graph Data Privacy</strong></td>
<td>TrustworthyGNN [200], Zhang et al. [197], Liu et al. [92], Yu et al. [192], Mulle et al. [105], PGAS [198], Federatedscope-GNN [174], Tan et al. [151]</td>
</tr>
<tr>
<td></td>
<td><strong>Graph Data Security</strong></td>
<td>Sandhu et al. [135], Abidi et al. [1], Li et al. [87]</td>
</tr>
<tr>
<td><strong>Graph MLOps</strong></td>
<td></td>
<td>Kubeflow [81], Amazon SageMaker [6], Amazon Neptune [179]</td>
</tr>
</tbody>
</table>

*Key to practical deployment of GNNs --- GNN Evaluation*
Understanding and evaluating GNN models’ performance is a vital step for GNN model deployment and serving.

1) Seen test graph $G_{te}$ in the same distribution as the train graph $G_{tr}$

2) Known test graph labels for computing performance metric, e.g., Accuracy (ACC)

For instance,
in financial transaction networks:

- **GNN model designers**: expect their developed GNNs to excel in identifying newly emerging suspicious transactions
- **Users**: ensure how they could trust well-trained GNNs to know suspicious transactions within their own data

Background of GNN Model Evaluation

However, in real-world scenarios, the test graphs are typically "unseen & lacking annotations".

In real-world model evaluation of GNNs, we:

\( \times \) CAN NOT access the ground-truth labels of the test graph \( G_{te} \)

\( \times \) CAN NOT compute performance metric, e.g., Accuracy (ACC)

\( \times \) DO NOT know whether potential distribution shifts from the train graph \( G_{tr} \)

---Case Study on Graph MLOps

Given above scenarios, a natural question, i.e., “GNN model evaluation problem” arises:

**In the absence of labels in an unseen test graph, can we estimate the performance of a well-trained GNN model?**

---

**Case Study on Graph MLOps**

Definition of GNN Model Evaluation

**Definition of GNN Model Evaluation.** Given the observed training graph $S$, its well-trained model $\text{GNN}^*_S$, and an unlabeled unseen graph $T$ as inputs, the goal of GNN model evaluation aims to learn an accuracy estimation model $f_\phi(\cdot)$ parameterized by $\phi$ as:

$$\text{Acc}(T) = f_\phi(\text{GNN}^*_S, T),$$

where $f_\phi : (\text{GNN}^*_S, T) \rightarrow a$ and $a \in \mathbb{R}$ is a scalar denoting the overall node classification accuracy $\text{Acc}(T)$ for all unlabeled nodes of $T$. When the context is clear, we will use $f_\phi(T)$ for simplification.

To solve above problems,

We propose a two-stage GNN model evaluation framework with a “GNNEvaluator”

*Note that our principal goal is to estimate well-trained GNN models’ performance, rather than improve the generalization ability of new GNN models. In the whole evaluation process, the in-service GNN model is fixed.*

**GNNEvaluator: Evaluating GNN Performance On Unseen Graphs Without Labels**

--- Case Study on Graph MLOps

**Figure 1** Overall two-stage framework of the proposed GNN model evaluation with GNNEvaluator

- **Stage-1: DiscGraph set construction**
  - Incorporating training-test graph discrepancies into DiscGraph node attributes $x^i_{\text{disc}}$, structures $A^i_{\text{disc}}$, and accuracy labels $y^i_{\text{disc}}$

- **Stage-2: GNNEvaluator training and inference**
  - GNNEvaluator, train on DiscGraphs and output estimated ACC on the real-world test graph $T$
The performance of our proposed GNNEvaluator in evaluating well-trained GNNs' node classification accuracy under all test evaluation cases and models.

Table 1: Mean Absolute Error (MAE) performance of different GNN models across five random seeds. (GNNs are well-trained on the ACMv9 dataset and evaluated on the unseen and unlabeled Citationv2 and DBLPv8 datasets, i.e., A→C and A→D, respectively. Best results are in bold.)

<table>
<thead>
<tr>
<th>Methods</th>
<th>ACMv9→Citationv2</th>
<th>ACMv9→DBLPv8</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GCN</td>
<td>SAGE</td>
</tr>
<tr>
<td>ATC-MC-C [8]</td>
<td>2.41</td>
<td>5.74</td>
</tr>
<tr>
<td>ATC-NE [8]</td>
<td>3.97</td>
<td>8.02</td>
</tr>
<tr>
<td>ATC-NE-C [8]</td>
<td>4.44</td>
<td>6.99</td>
</tr>
<tr>
<td>Thres. (γ = 0.7) [6]</td>
<td>32.64</td>
<td>35.81</td>
</tr>
<tr>
<td>Thres. (γ = 0.8) [6]</td>
<td>26.30</td>
<td>29.60</td>
</tr>
<tr>
<td>Thres. (γ = 0.9) [6]</td>
<td>17.56</td>
<td>21.34</td>
</tr>
<tr>
<td><strong>GNNEvaluator (Ours)</strong></td>
<td><strong>4.85</strong></td>
<td><strong>4.11</strong></td>
</tr>
</tbody>
</table>

Table 2: Mean Absolute Error (MAE) performance of different GNN models across five random seeds. (GNNs are well-trained on the Citationv2 dataset and evaluated on the unseen and unlabeled ACMv9 and DBLPv8 datasets, i.e., C→A and C→D, respectively. Best results are in bold.)

<table>
<thead>
<tr>
<th>Methods</th>
<th>Citationv2→ACMv9</th>
<th>Citationv2→DBLPv8</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GCN</td>
<td>SAGE</td>
</tr>
<tr>
<td>ATC-MC [8]</td>
<td>9.50</td>
<td>13.40</td>
</tr>
<tr>
<td>ATC-MC-C [8]</td>
<td>6.93</td>
<td>11.75</td>
</tr>
<tr>
<td>ATC-NE [8]</td>
<td>8.86</td>
<td>13.04</td>
</tr>
<tr>
<td>ATC-NE-C [8]</td>
<td>7.73</td>
<td>13.94</td>
</tr>
<tr>
<td>Thres. (γ = 0.7) [6]</td>
<td>37.33</td>
<td>36.61</td>
</tr>
<tr>
<td>Thres. (γ = 0.8) [6]</td>
<td>29.62</td>
<td>28.95</td>
</tr>
<tr>
<td>Thres. (γ = 0.9) [6]</td>
<td>19.59</td>
<td>19.06</td>
</tr>
<tr>
<td><strong>GNNEvaluator (Ours)</strong></td>
<td><strong>5.45</strong></td>
<td><strong>8.53</strong></td>
</tr>
</tbody>
</table>

---

- Experiments on 3 real-world graph datasets in 6 cases potential domain shift, each evaluating 5 models:
- Consistent outstanding performance over all GNN models and cases!
Part 5: Future Directions & Conclusion
Promising Future Directions

- Exploration of complex and dynamic graph data


Promising Future Directions

- **General and automatic graph data improvement.**

Promising Future Directions

- Standardized graph data benchmarks
- Collaborative development of graph data and model
- Comprehensive graph data lifecycle management pipelines
Promising Future Directions

- Exploration of complex and dynamic graph data
- General and automatic graph data improvement
- Standardized graph data benchmarks
- Collaborative development of graph data and model
- Comprehensive graph data lifecycle management pipelines
Conclusion

Promising Data-centric Graph Machine Learning (DC-GML)

How To Enhance Graph Data Availability and Quality?

Graph Data Improvement

Graph MLOps

How To Learn From Graph Data With Limited-availability and Low-quality?

Graph Data Exploitation

How To Build Graph MLOps System: The Graph Data-centric View?

Graph Data Maintenance

Graph Data Exploration

Graph Data Collection
Three Core Research Questions

RQ1: How to enhance graph data availability and quality?

Enhanced Graph Data

RQ2: How to learn from graph data with limited-availability and low-quality?

Enhanced Graph Data ➔ Enhanced Graph ML Models

RQ3: How to build graph MLOps systems from the graph data-centric view?

Systematic & Comprehensive
Data-centric Graph Machine Learning (DC-GML)
Conclusion

Comprehensive Taxonomy

Graph Data Improvement

- Graph Structure Enhancement
  - Graph Structure Learning || Graph Sparsification || Graph Diffusion
- Graph Feature Enhancement
  - Graph Feature Completion || Graph Feature Denoising
- Graph Label Enhancement
  - Graph Pseudo-labelling || Graph Label Denoising || Graph Class-imbalanced Sampling
- Graph Size Enhancement
  - Graph Size Reduction || Graph Data Augmentation

Graph Data Exploitation

- Graph Self-supervised Learning
- Graph Semi-supervised Learning
- Graph Active Learning
- Graph Transfer Learning

Graph Data Collection

- Graph Data Crowdsourcing & Synthesis

Graph Data Exploration

- Graph Data Understanding, Visualization & Valuation

Graph Data Maintenance

- Graph Data Privacy & Security

Fig. 2. The framework and taxonomy of data-centric graph machine learning (DC-GML).
Conclusion

Extensive & Open Potentials of DC-GML

A. Standardized graph machine learning workflow

B. Enhanced graph data understanding

C. Better graph learning model performance

D. Wider graph data application range

... continual and broader applications in DC-GML...