ICDM 2022: Risk Commodities Detection on Large-Scale E-Commence Graphs

Team: **SYSU-GEAR**

Rank: 8th (Session I) 3rd (Session II)

- Jintang Li: Sun Yat-sen University
- Yike Xu: South China University of Technology
- Junyuan Fang: City University of Hong Kong

中山大學 SUN YAT-SEN UNIVERSITY

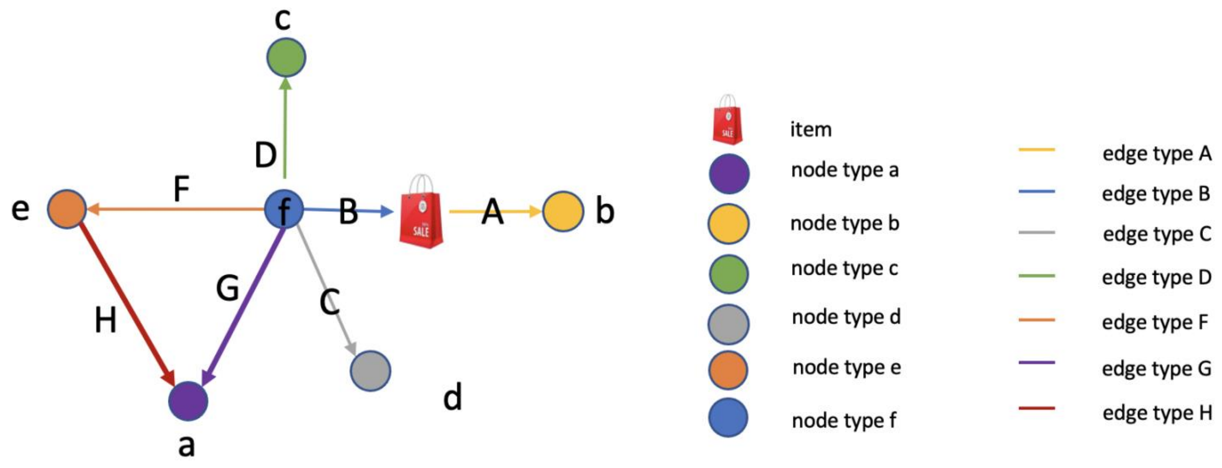
華南理工大學 South China University of Technology

香港城市大學 City University of Hong Kong

Background

Background

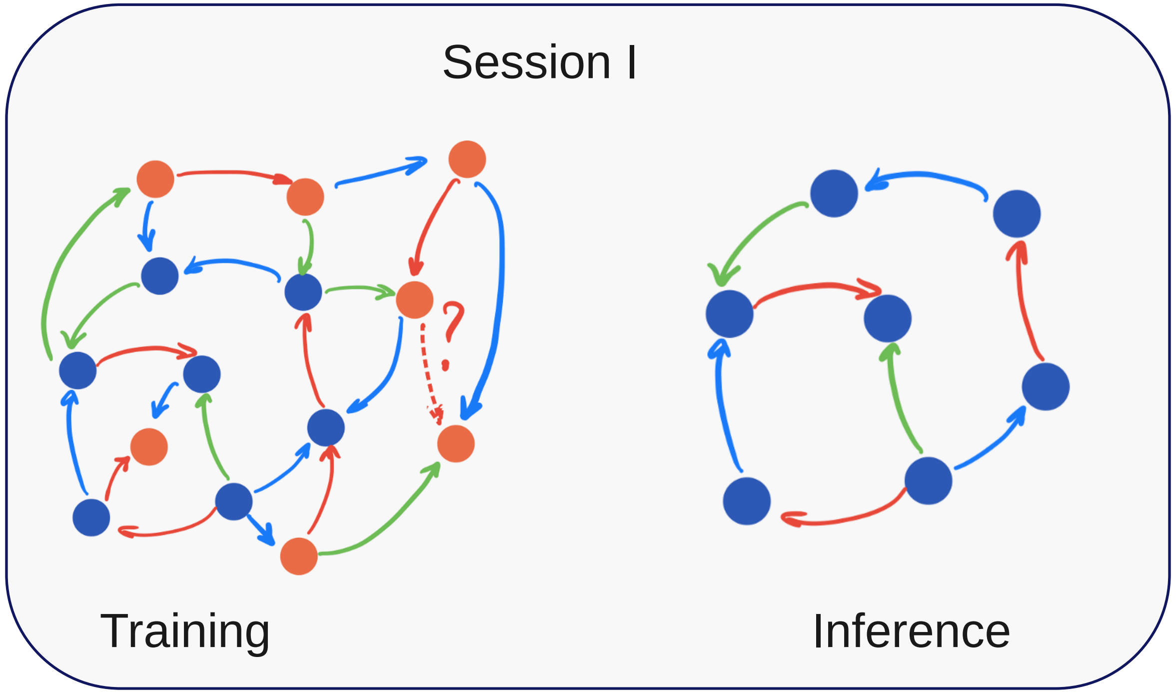
- Risk Commodities Detection on Large-Scale E-Commence Graphs
- Challenges (1/4):
  - Large-scale and heterogeneous:13M nodes, 157M edges, 7 node types and 7 edge types

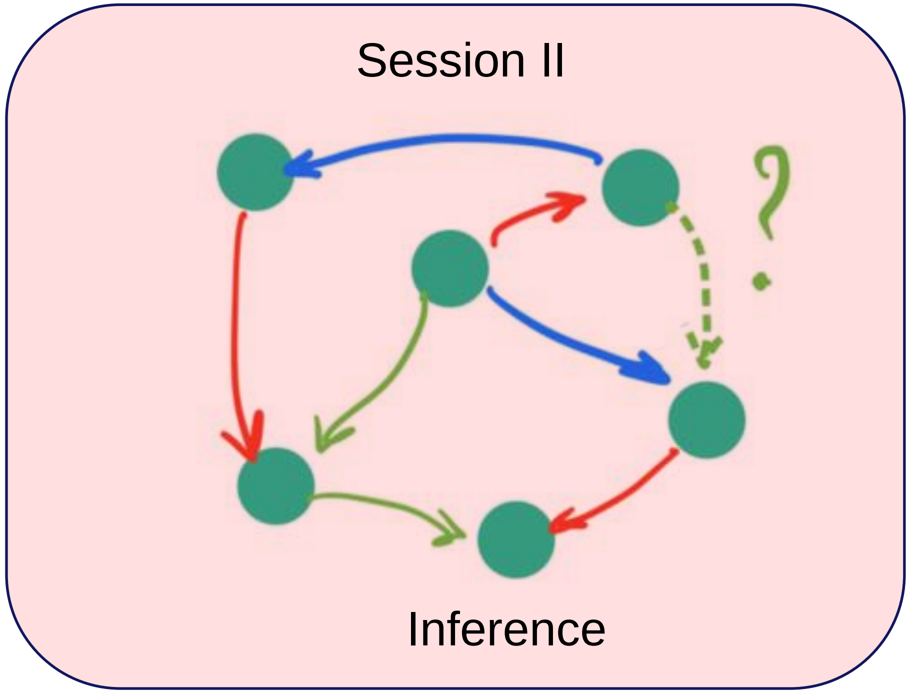


# Node type	# Edge type	# Node	# Edge
7	7	13,806,619	157,814,864

Background

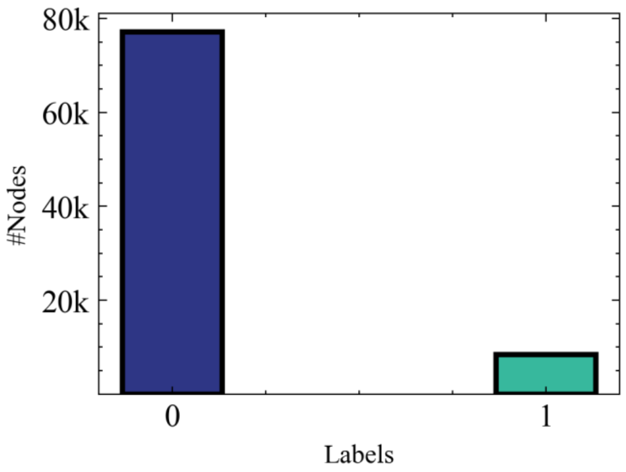
- Risk Commodities Detection on Large-Scale E-Commence Graphs
- Challenges (2/4):
  - Inductive and transferable capability: inference on another graph that is not visible during training (Session II)





Background

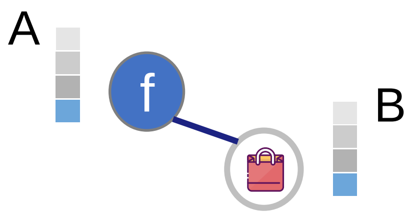
- Risk Commodities Detection on Large-Scale E-Commence Graphs
- Challenges (3/4):
  - Imbalance of node labels: positive (8k) and negative (77k) ≈ 1 :10



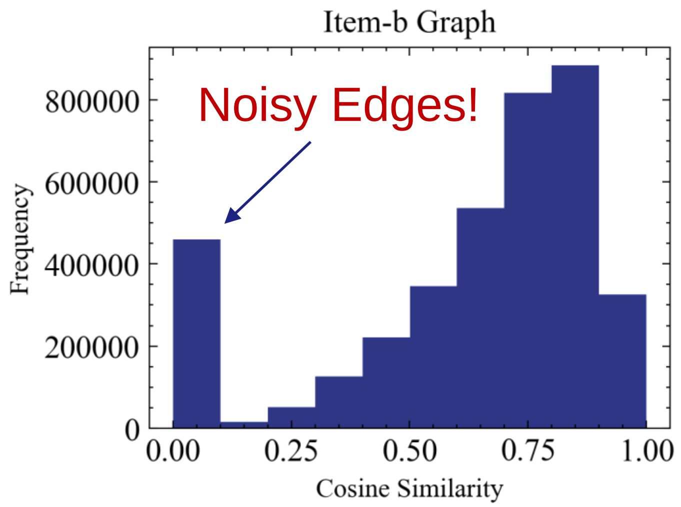
# Positive sample	# Negative sample
8,364	77,198

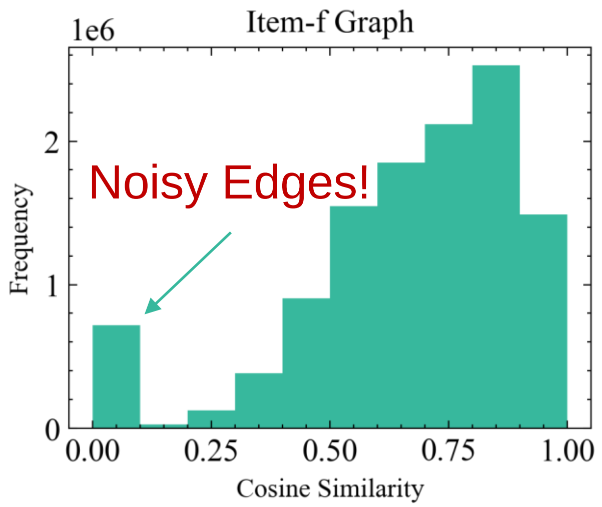
Background

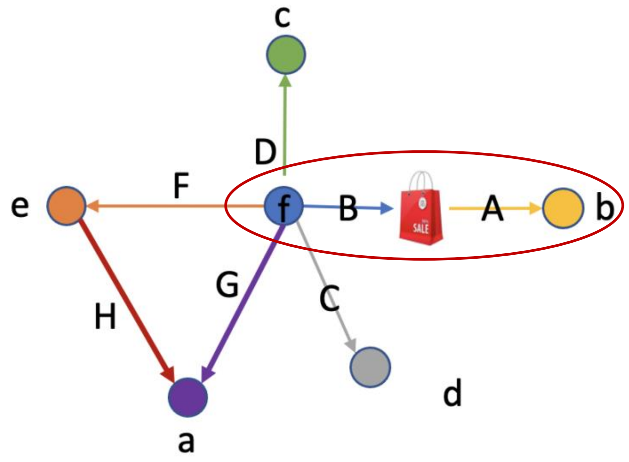
- Risk Commodities Detection on Large-Scale E-Commence Graphs
- Challenges (4/4):
  - Data noise on graph structure/node features



$$\text{cosine similarity} = S_C(A,B) := \cos(\theta) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

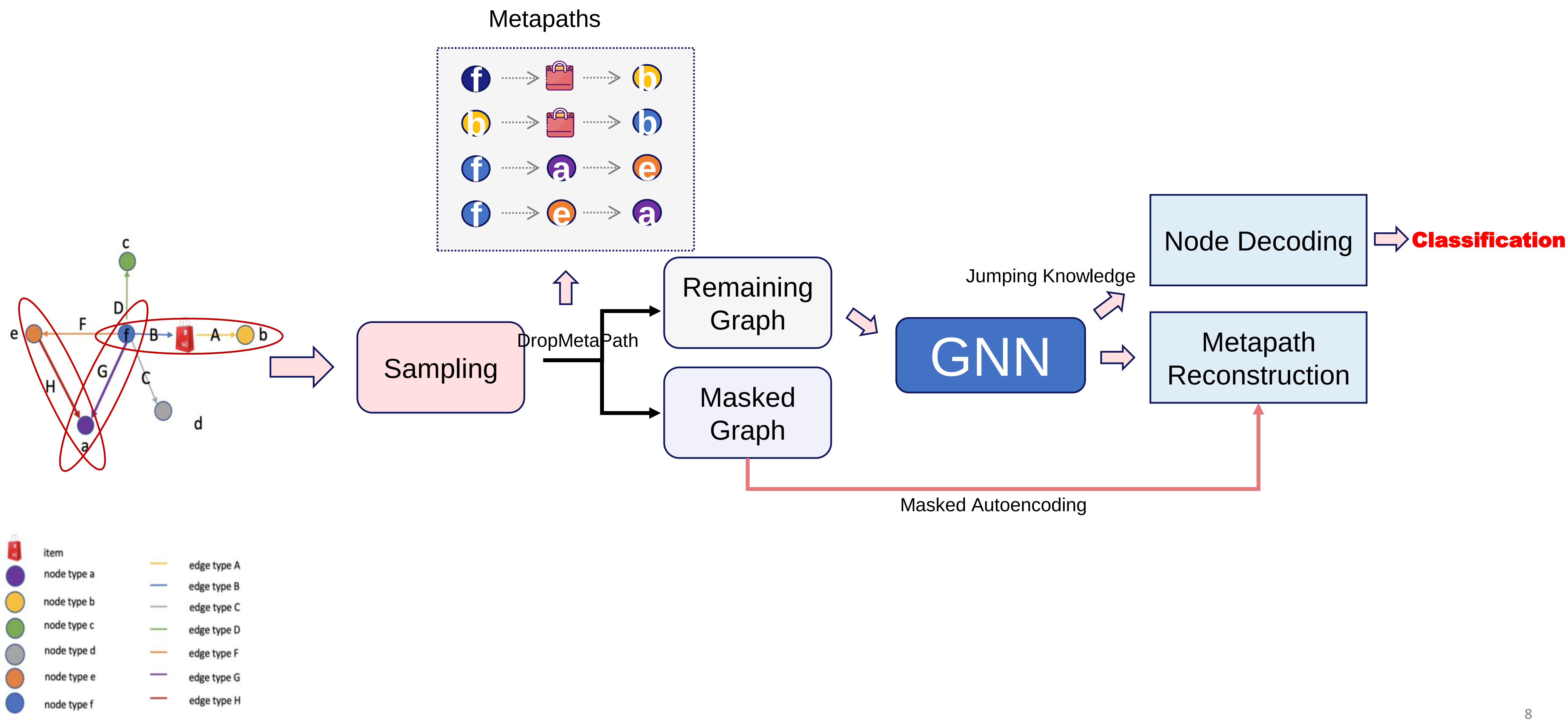






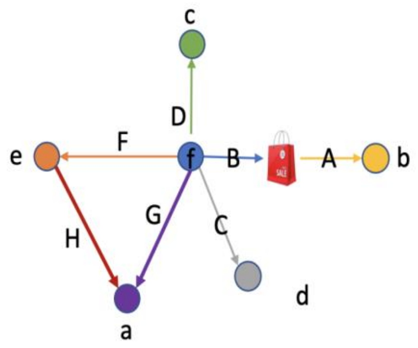
Method: HeteroGNN (Overall Framework)

Overall Framework: HeteroGNN



SYSU-GEAR: Jintang Li, Yike Xu, Junyuan Fang

Overall Framework: HeteroGNN

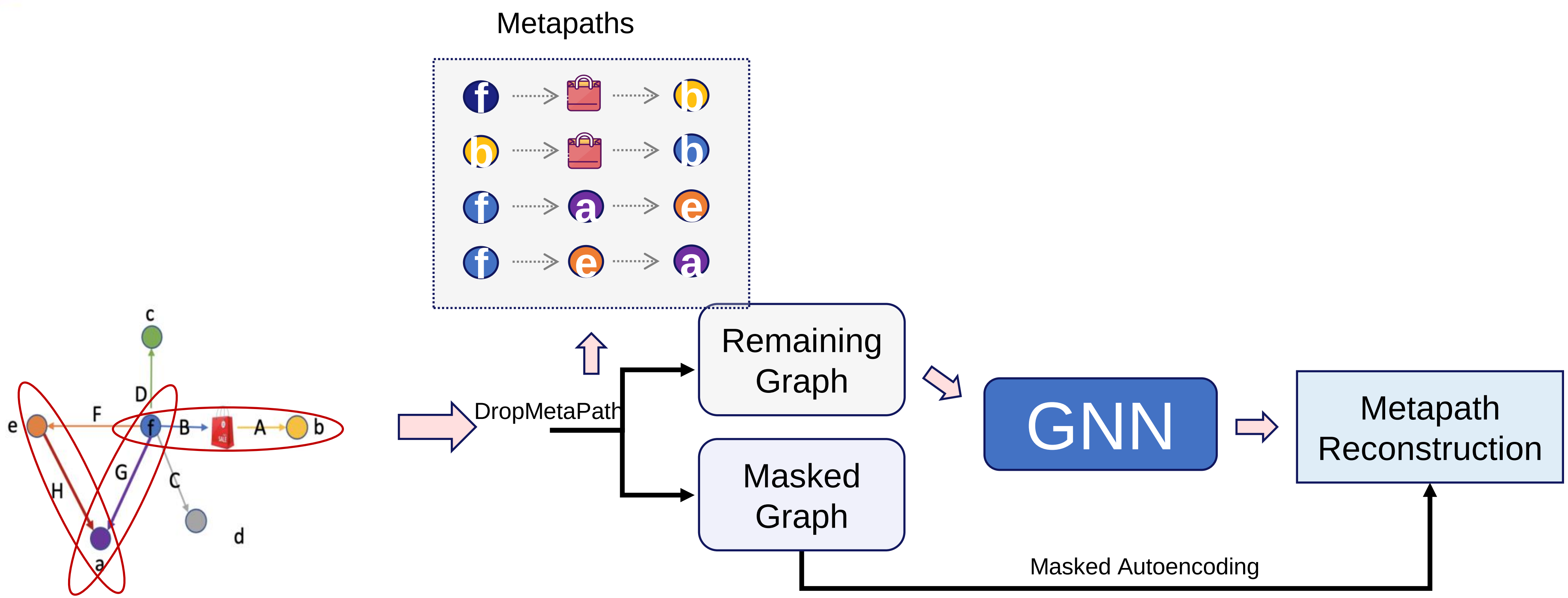


Challenge 1 (scalability) &
Challenges 2 (inductive capability): Neighborhood sampling

Sampling

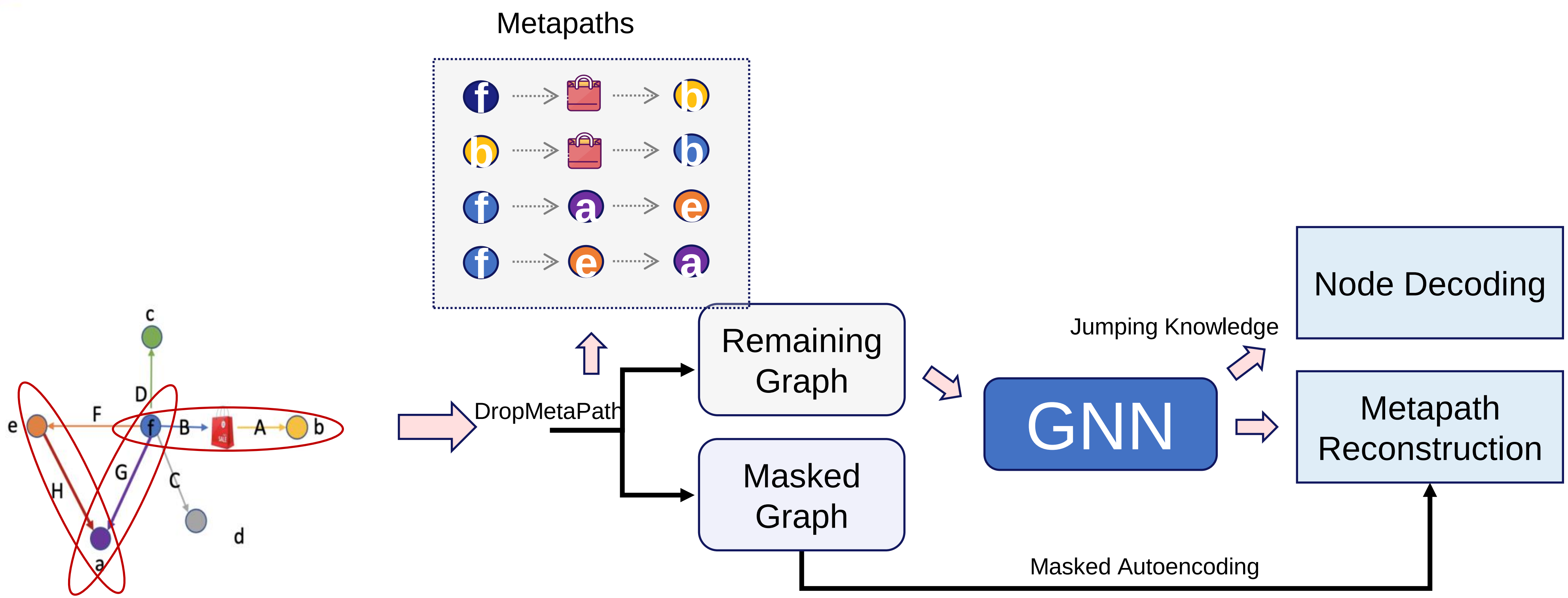
Challenge 3 (Imbalance): resampling

Overall Framework: HeteroGNN



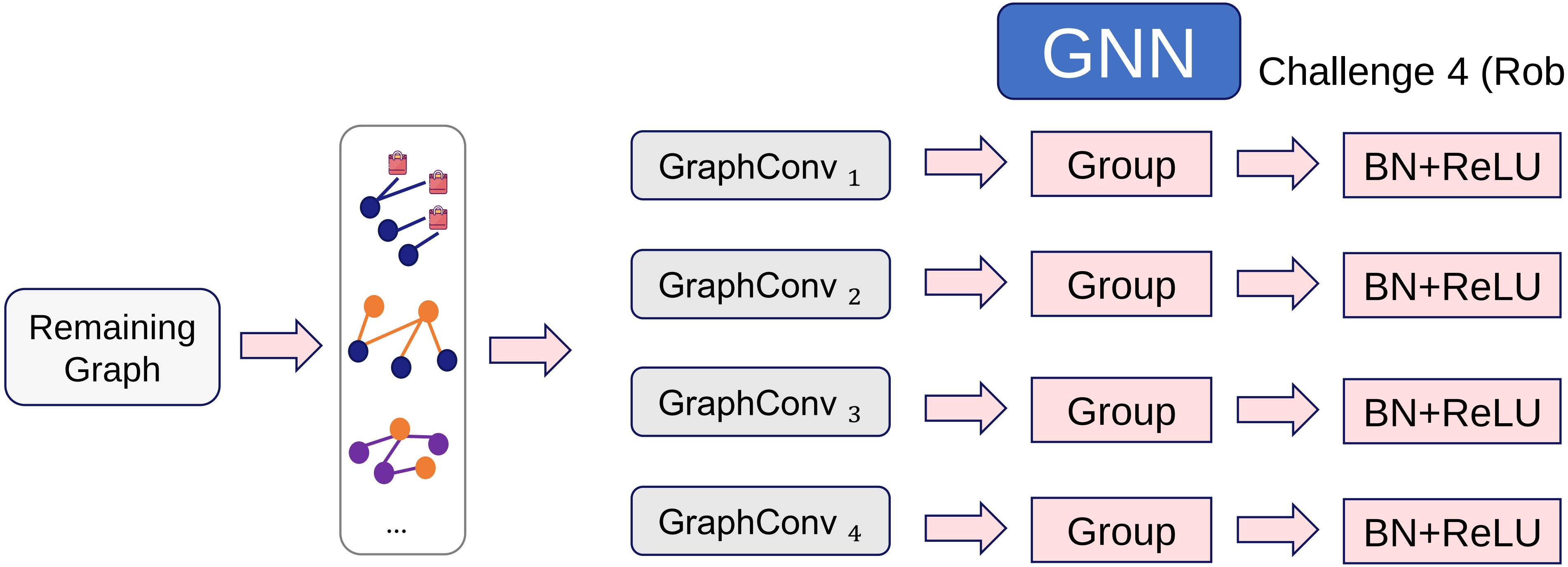
Challenge 4 (Robustness): DropMetaPath, Masked Autoencoding

Overall Framework: HeteroGNN



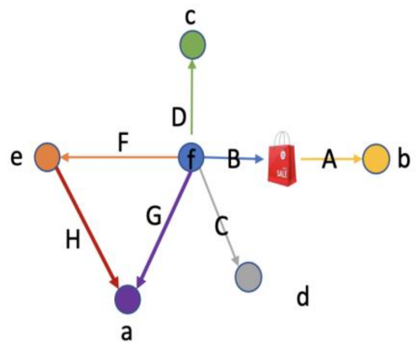
Challenge 4 (Robustness): DropMetaPath, Masked Autoencoding, Jumping Knowledge

Overall Framework: HeteroGNN



Challenge 4 (Robustness): BN

Challenge 1 (Heterogenous): Heterogenous GraphConv

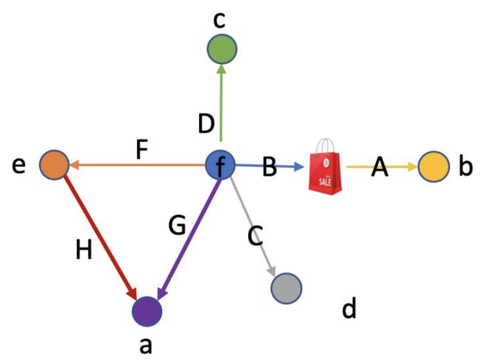


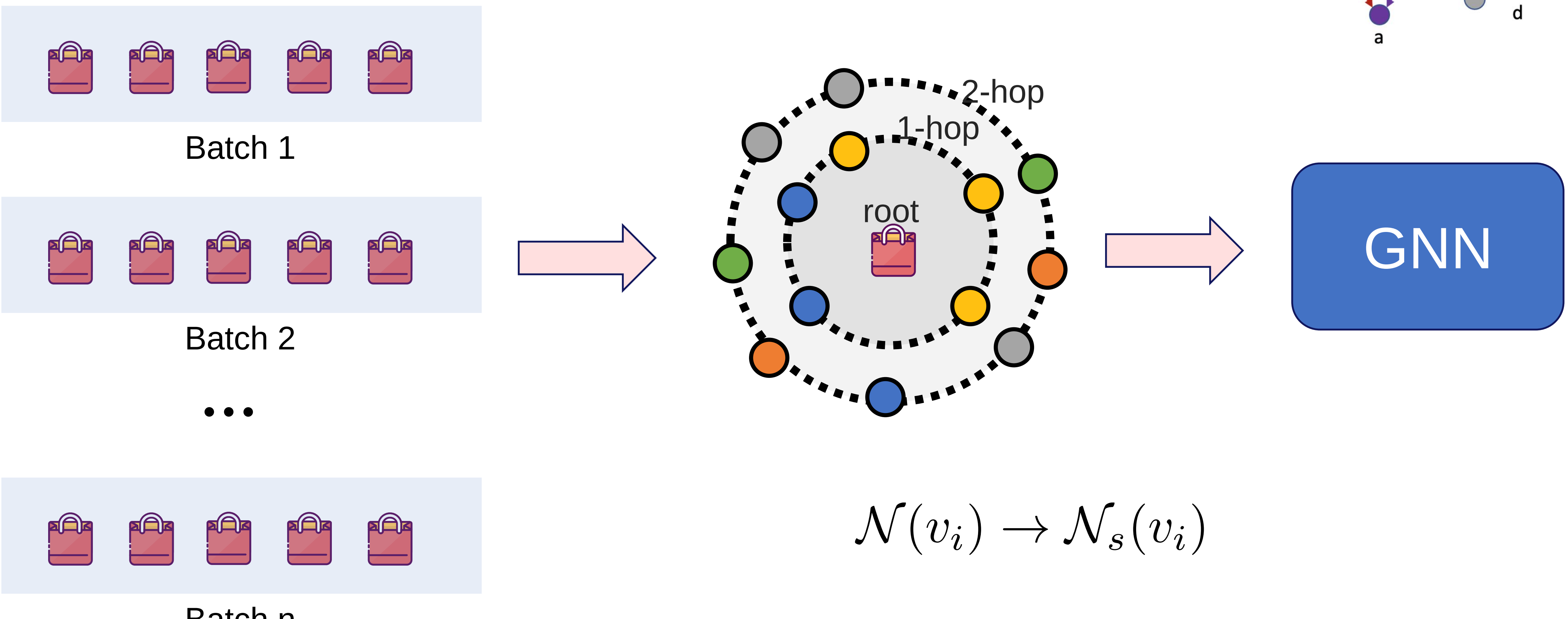
Modeling Relational Data with Graph Convolutional Networks. ESWC 18

Method: HeteroGNN (Details)

Method: HeteroGNN

- Multi-hop neighborhood sampling





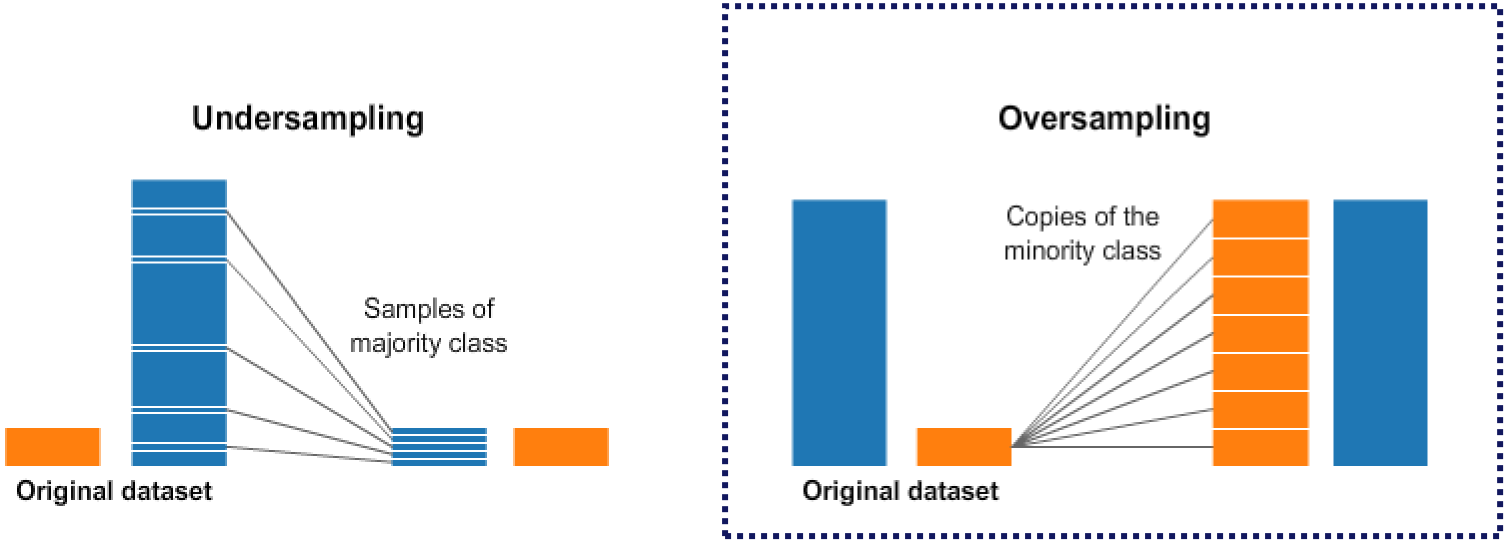
$$\mathcal{N}(v_i) \rightarrow \mathcal{N}_s(v_i)$$

Inductive Representation Learning on Large Graphs. NeurIPS 17.

Method: HeteroGNN

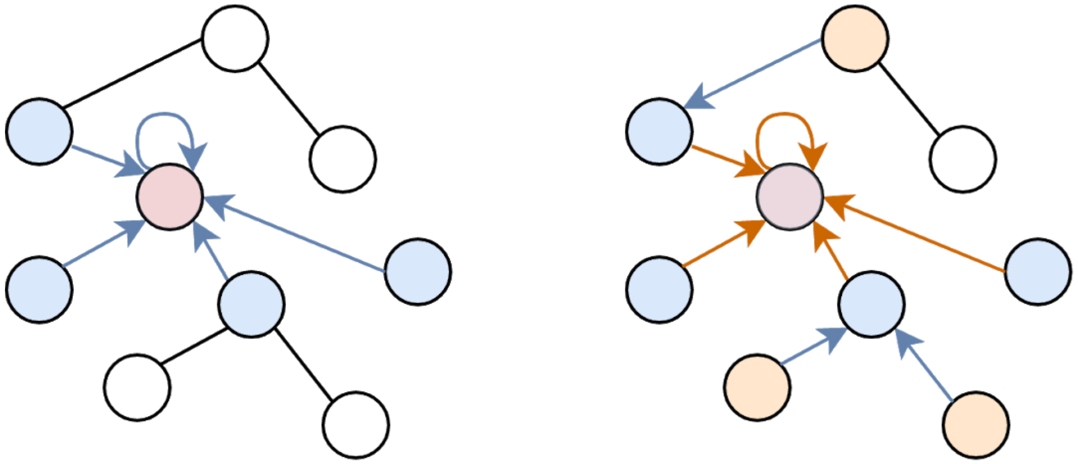
Resampling

- Most data falls into major class (0) while a few data falls into the minority class (1)
- Over-sampling technique is used to produce equally distributed data fall into each class



Method: HeteroGNN

- Stacking graph convolutional layers
  - Performing multiple updates increases the “receptive field” of each node.



Layer 1 Layer 2

concatenate self embedding and neighbor embedding

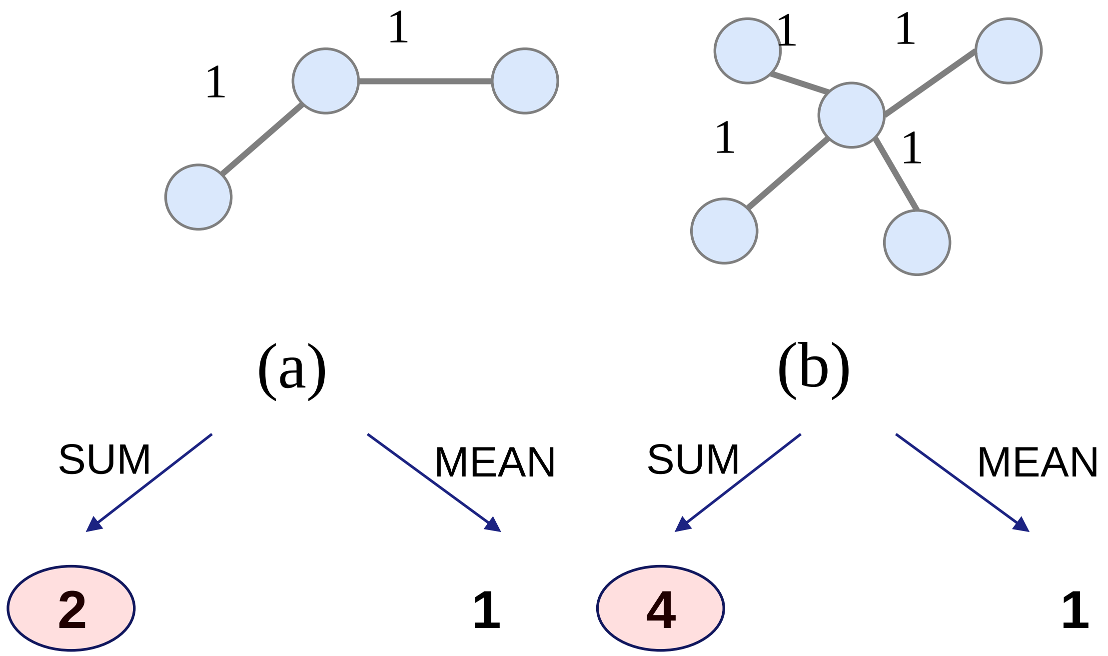
$$\mathbf{h}_v^k = \sigma\left(\left[\mathbf{W}_k \cdot \boxed{\mathrm{AGG}\left(\{\mathbf{h}_u^{k-1}, \forall u \in N(v)\}\right)}, \mathbf{B}_k \mathbf{h}_v^{k-1}\right]\right)$$

generalized aggregation

https://spindro.github.io/post/gnn/

Method: HeteroGNN

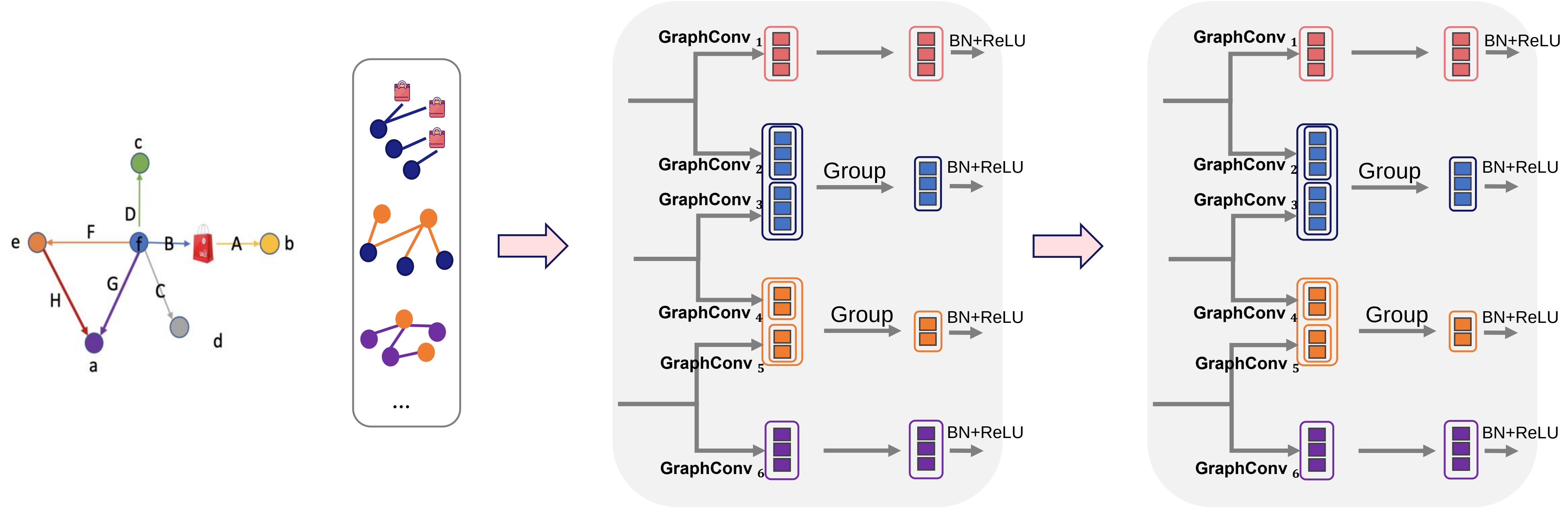
- Stacking graph convolutional layers
  - Performing multiple updates increases the “receptive field” of each node.
  - Use SUM aggregation instead of MEAN aggregation to increase expressiveness



AGG = SUM is better!

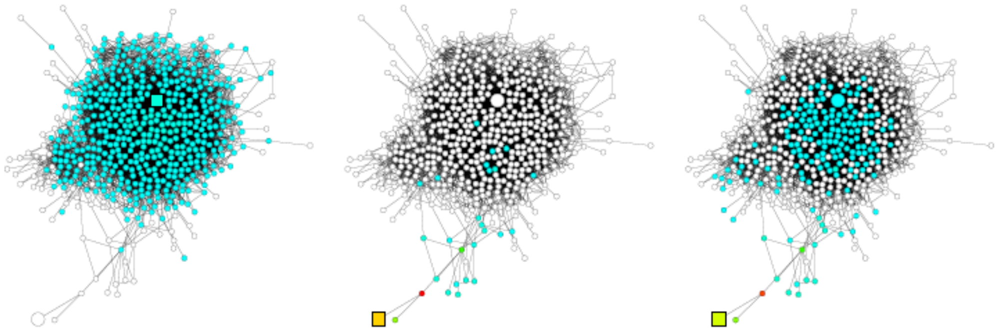
Method: HeteroGNN

- Stacking graph convolutional layers
  - Performing multiple updates increases the "receptive field" of each node.
  - Use SUM aggregation instead of MEAN aggregation to increase expressiveness
  - Heterogenous GraphConv: Each graph view corresponds to an edge type

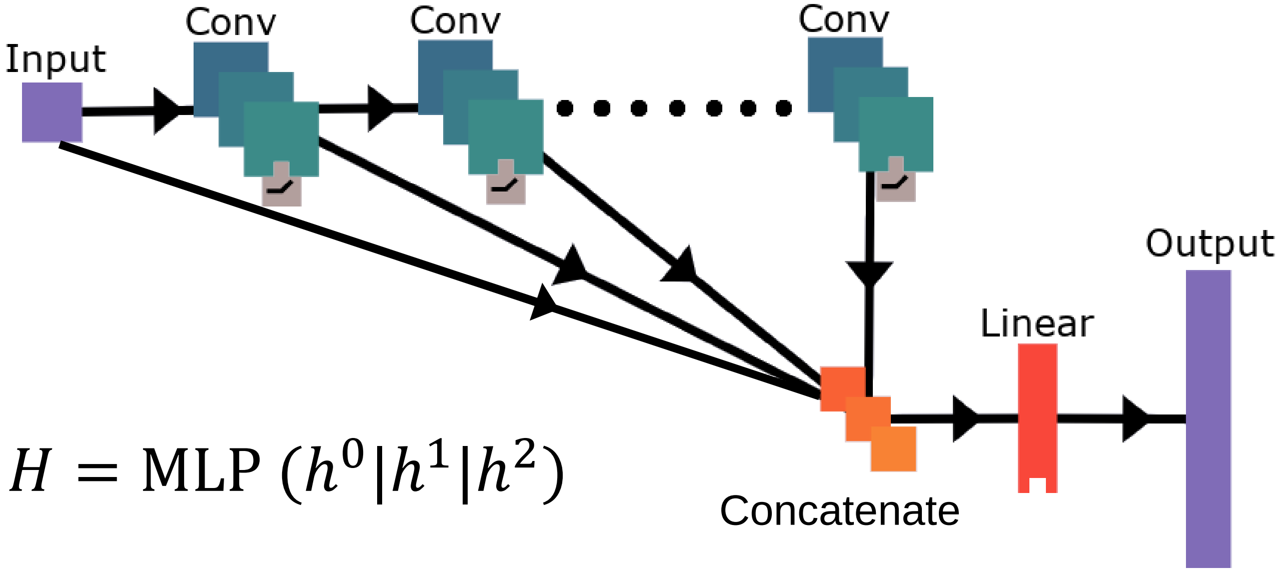


Method: HeteroGNN

- Jumping Knowledge
  - Different subgraph structures result in very different neighborhood sizes
  - Jumping Knowledge enables adaptive, structure-aware representations
  - Such representations are particularly interesting for representation learning on large complex graphs with diverse subgraph structures



(a) 4 steps at core (b) 4 steps at tree (c) 5 steps at tree



$$H = \mathrm{MLP}\ (h^0|h^1|h^2)$$

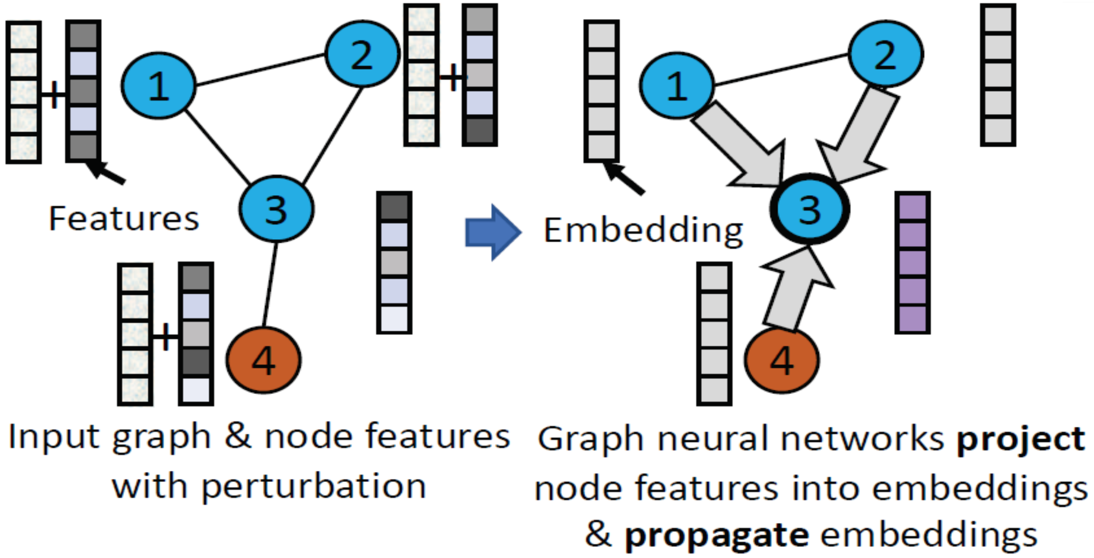
Representation Learning on Graphs with Jumping Knowledge Networks. ICML 18

Method: HeteroGNN

Graph Adversarial Training

- Adversarial Training (AT) is a dynamic regularization technique, performing perturbations on input features to improve model robustness and generalization
- AT on graphs can propagate the applied perturbations to its local neighborhoods for each node

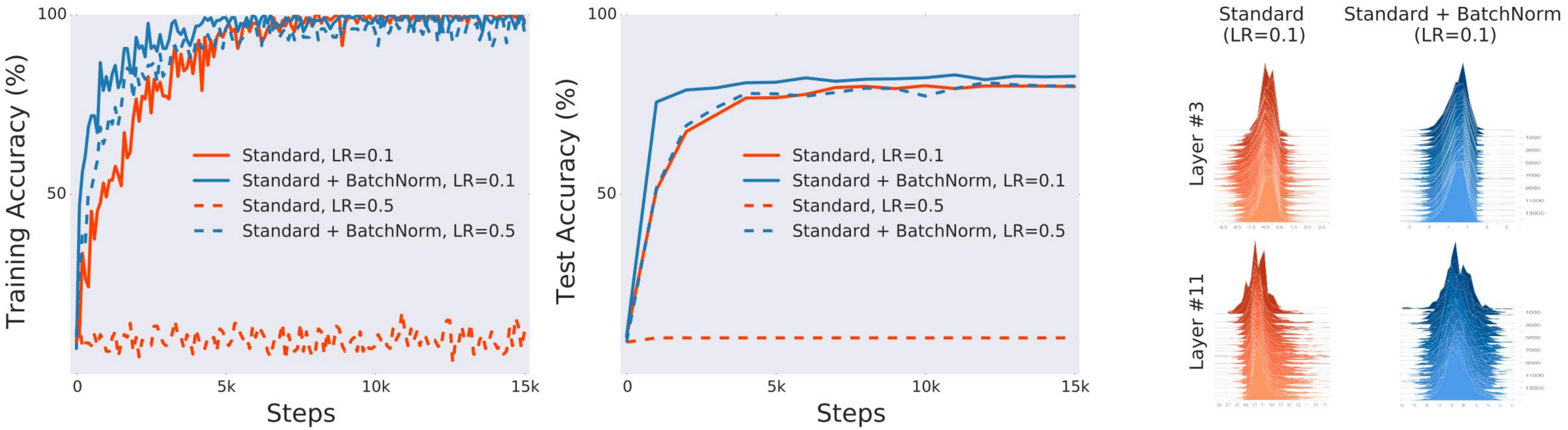
$$x' = \mathrm{x} + \delta, \delta \sim \mathrm{U}(-0.5, 0.5)$$



Graph Adversarial Training: Dynamically Regularizing Based on Graph Structure. TKDE 19

Method: HeteroGNN

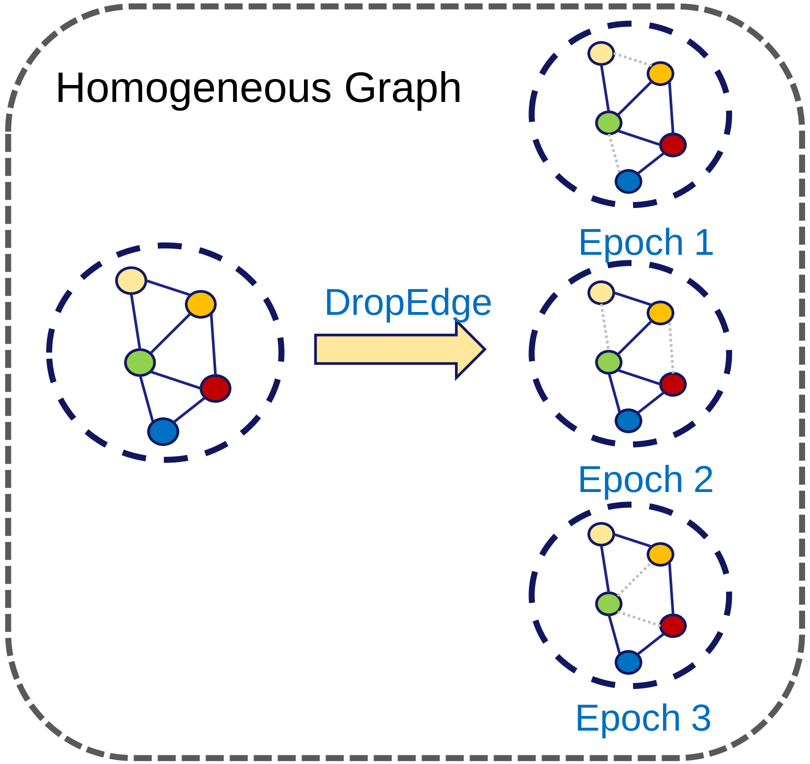
- Batch Normalization
  - BN can benefit the convergence/training speed of neural network models
  - The input distributions would to be much less pronounced between layers

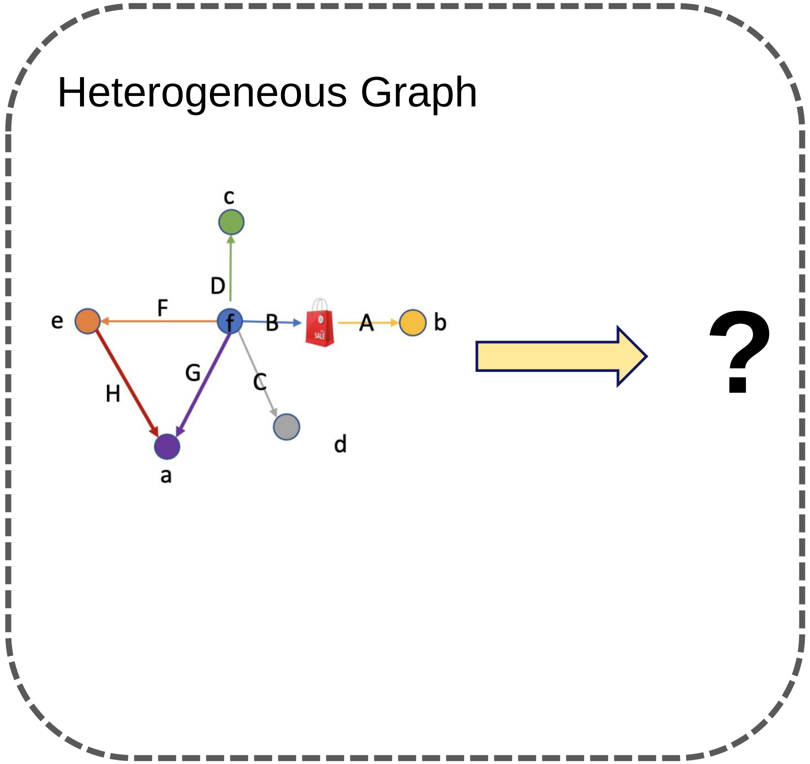


How Does Batch Normalization Help Optimization? NeurIPS 18

Method: HeteroGNN

- DropMetapath
  - DropEdge: a regularization trick which can implicitly reduce the inherent noise effect by randomly drop edges and nodes during training

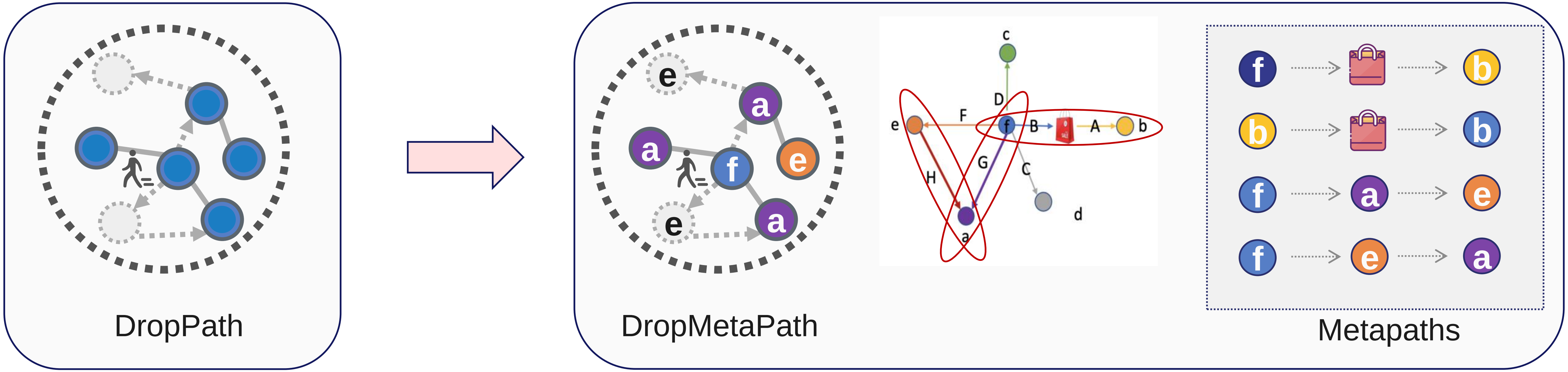




DropEdge: Towards Deep Graph Convolutional Networks on Node Classification. ICLR 20

Method: HeteroGNN

- DropMetapath
  - DropPath: a structured dropout for graph neural networks
  - DropMetaPath: an extension of DropPath on heterogeneous graphs



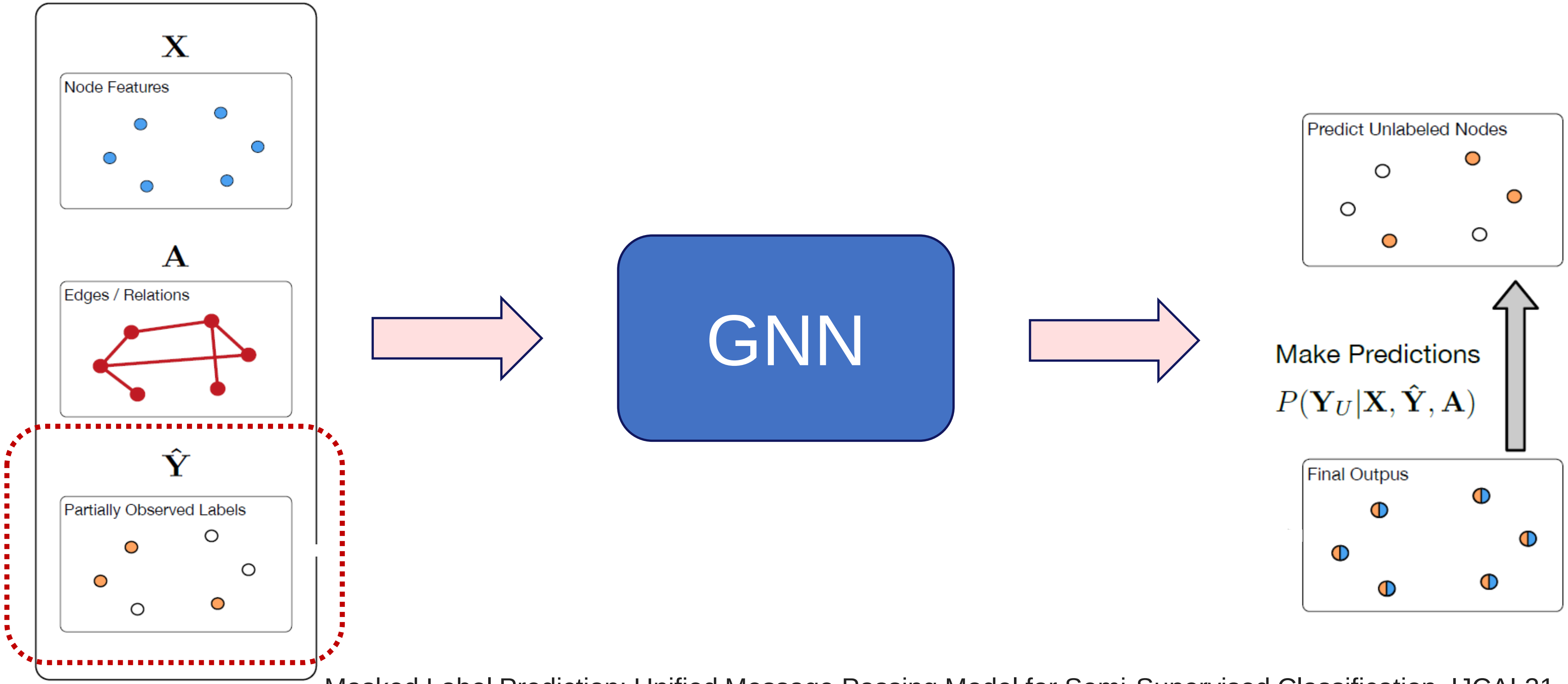
MaskGAE: Masked Graph Modeling Meets Graph Autoencoders. arXiv 22

Method: HeteroGNN

Session I

Masked Label Propagation

- UniMP: Unified Message Passing that combines the feature and label propagation together

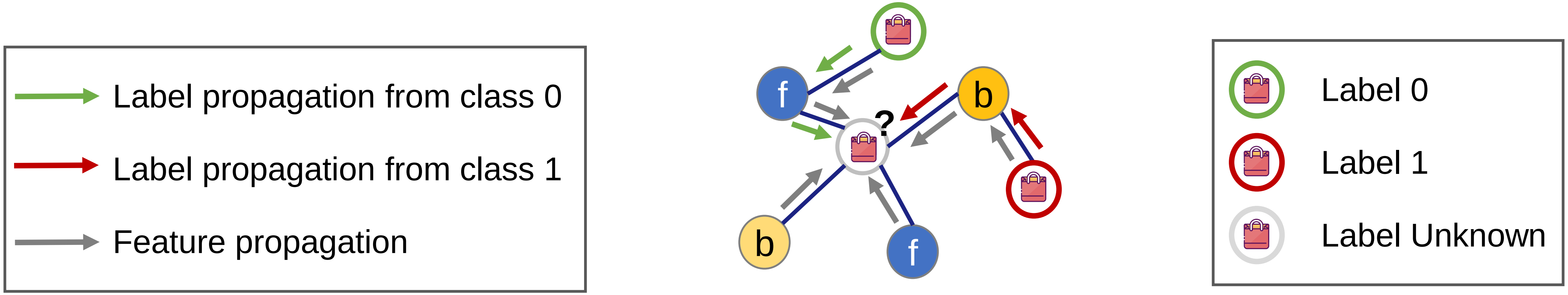


Masked Label Prediction: Unified Message Passing Model for Semi-Supervised Classification. IJCAI 21

Method: HeteroGNN

Session I

- Masked Label Propagation
  - UniMP: Unified Message Passing that combines the feature and label propagation together



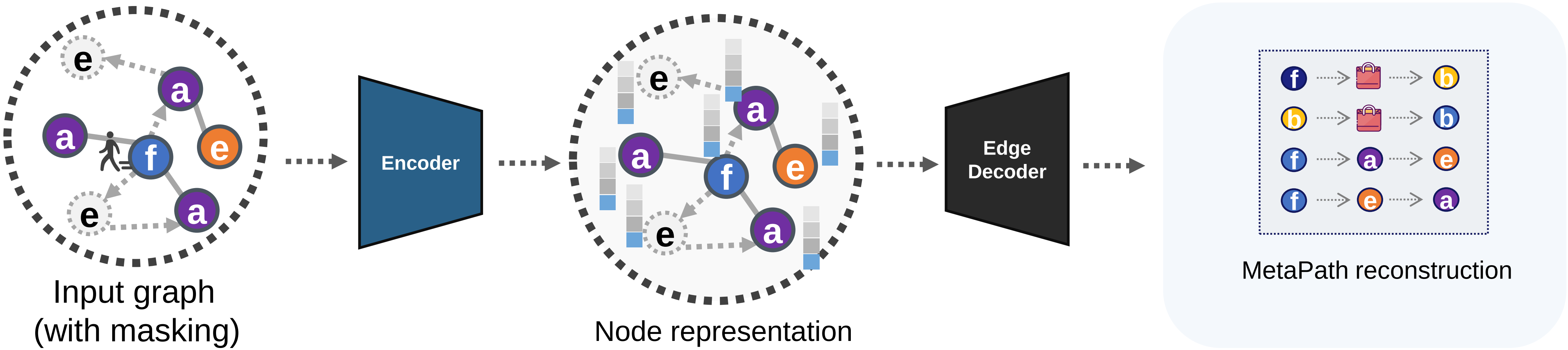
Masked Label Prediction: Unified Message Passing Model for Semi-Supervised Classification. IJCAI 21

Method: HeteroGNN

Session II

Masked Autoencoding

- DropMetaPath has provided self-supervised signals (metapaths)
- An additional edge decoder to learn to reconstruct masked metapaths



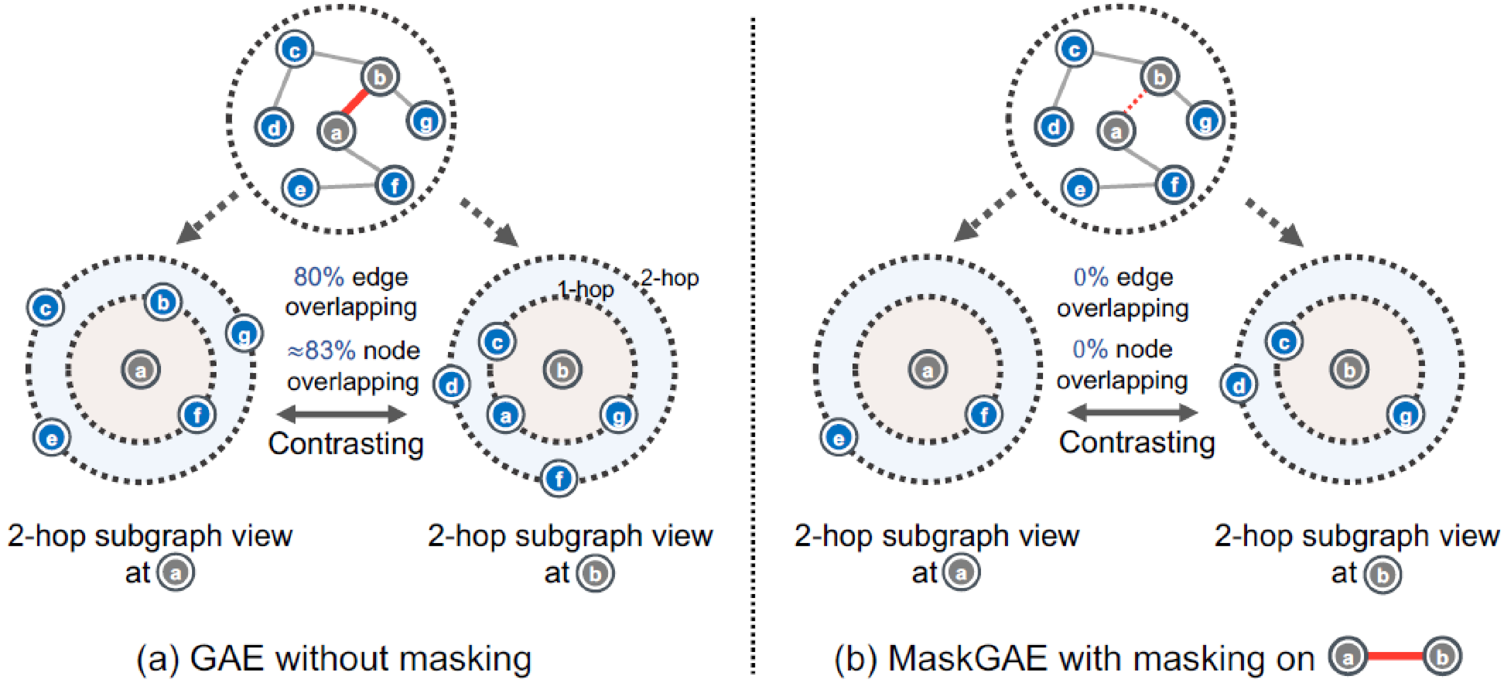
MaskGAE: Masked Graph Modeling Meets Graph Autoencoders. arXiv 22

Method: HeteroGNN

Session II

Masked Autoencoding

- Masking on the positive edge helps the contrastive scheme, as it significantly reduces the redundancy of two paired subgraph views.



MaskGAE: Masked Graph Modeling Meets Graph Autoencoders. arXiv 22

Experimental Results

Method: HeteroGNN

Session I

No.	Model	Val AP	Test AP
0	Baseline: RGCN	95.73	91.15
1	Base model: HeteroGNN (sum)	95.85	92.19
2	1+resampling	95.77	92.92
3	2+Batch Normalization	96.14	93.08
4	3+Graph Adversarial Training	96.56	93.16
5	4+Jumping Knowledge	96.35	93.22
6	5+DropMetapath	96.42	93.31
7	6+Masked Label Propagation	97.09	93.69

Method: HeteroGNN

Session II

No.	Model	Val AP	Test AP
0	**Session I (7)**	97.09	89.84
1	0 - Masked Label Propagation	96.42	90.52
2	1+Hidden size (256->512)	96.54	91.13
3	2+Node decoder (1->2)	96.68	91.40
4	3+Masked Autoencoding	96.65	91.76
5	4+sampling size (150->100)	96.92	91.83

Future Direction: Robust Aggregation Function

Robust Aggregation Function

What makes a robust GNN?

- **Definition:** **Breakdown point** $\epsilon(f, \mathcal{N})$ is defined as the smallest contamination fraction under which the aggregation function f breaks down:

$$\min_{m \in \mathbb{N}} \left\{ \frac{m}{|\mathcal{N}_v| + m} : \sup_{\tilde{\mathcal{N}}_v : |\tilde{\mathcal{N}}_v| = m} \left| f\left(\mathcal{N}_v \cup \widetilde{\mathcal{N}}_v\right) - f(\mathcal{N}_v) \right| = \infty \right\}$$

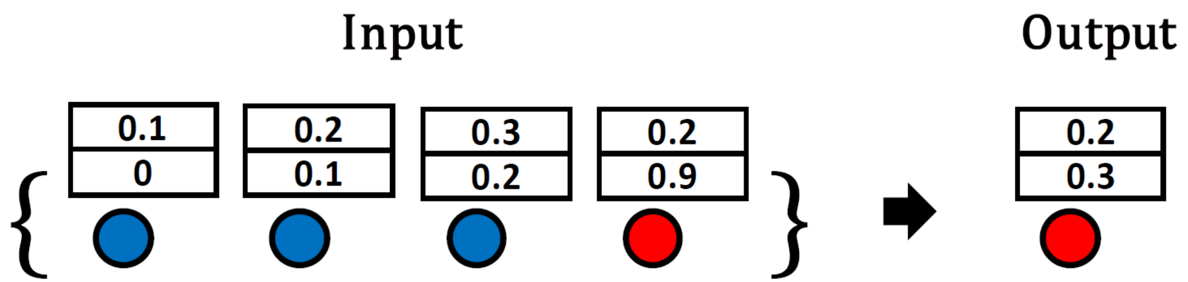
$\mathcal{N}$ input nodes set
$\tilde{\mathcal{N}}$ perturbation set

- The smallest value of breakdown point is $1/(|\mathcal{N}_v| + 1)$
- The value of the breakdown point can not exceed $1/2$

Understanding Structural Vulnerability in Graph Convolutional Networks. IJCAI 2021

Robust Aggregation Function

- The **non-robust aggregation function** leads to structural vulnerability of GNNs



Benign Vicious

$$a_v = \sum_{u \in \mathcal{N}_v} w_{uv} h_u + \sum_{\tilde{u} \in \widetilde{\mathcal{N}}_v} w_{\tilde{u}v} h_{\tilde{u}}$$

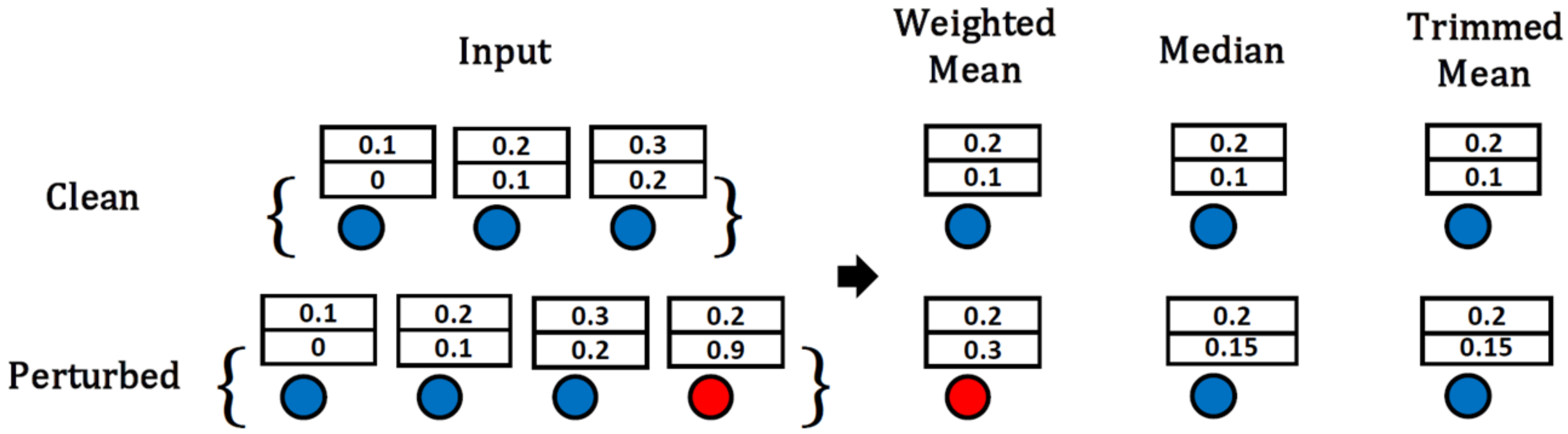
breakdown point = $\frac{1}{|\mathcal{N}_v| + 1}$

- The output of GNNs can be easily changed by injecting a node with extreme value as feature

Understanding Structural Vulnerability in Graph Convolutional Networks. IJCAI 2021

Robust Aggregation Function

Robust aggregation functions



Median

$$a_v = \begin{cases} \left(h_{n/2} + h_{(n/2)+1}\right)/2 & n \text{ is even} \\ h_{(n+1)/2} & n \text{ is odd} \end{cases}$$

breakdown point = 1/2

Maximum

Trimmed mean

breakdown point = $\frac{\alpha n + 1}{n}$

$$a_v = \frac{1}{n - 2\lfloor n\alpha \rfloor} \sum_{u=s}^{t} h_u, s = \lfloor n\alpha \rfloor + 1, t = n - \lfloor n\alpha \rfloor$$

Understanding Structural Vulnerability in Graph Convolutional Networks. IJCAI 2021

Reference

- ESWC 18, Modeling Relational Data with Graph Convolutional Networks
- WWW 19, Heterogeneous Graph Attention Network
- AAAI 19, Weisfeiler and Leman Go Neural: Higher-order Graph Neural Networks
- NeurIPS 17, Inductive Representation Learning on Large Graphs
- NeurIPS 18, How Does Batch Normalization Help Optimization?
- ICLR 15, Explaining and Harnessing Adversarial Examples
- ICML 18, Representation Learning on Graphs with Jumping Knowledge Networks
- TKDE 19, Graph Adversarial Training: Dynamically Regularizing Based on Graph Structure
- ICLR 20, DropEdge: Towards Deep Graph Convolutional Networks on Node Classification
- arXiv 22, MaskGAE: Masked Graph Modeling Meets Graph Autoencoders
- IJCAI 21, Masked Label Prediction: Unified Message Passing Model for Semi-Supervised Classification
- NeurIPS 16, Variational Graph Auto-Encoders
- IJCAI 21, Understanding Structural Vulnerability in Graph Convolutional Networks

Thank you!

https://github.com/EdisonLeeeee/ICDM2022_competition_3rd_place_solution (github.com)

Contact me:

lijt55@mail2.sysu.edu.cn

中山大學 SUN YAT-SEN UNIVERSITY

華南理工大學 South China University of Technology

CityU 香港城市大學 City University of Hong Kong

Q & A

https://github.com/EdisonLeeeee/ICDM2022_competition_3rd_place_solution (github.com)

Contact me:

lijt55@mail2.sysu.edu.cn

中山大學 SUN YAT-SEN UNIVERSITY

華南理工大學 South China University of Technology

CityU 香港城市大學 City University of Hong Kong