Machine Learning with Graphs (MLG)

Node Classification

The era before Graph Representation Learning (GRL)

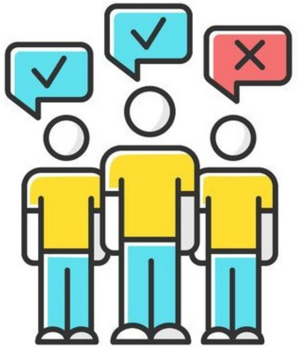
Cheng-Te Li (李政德)

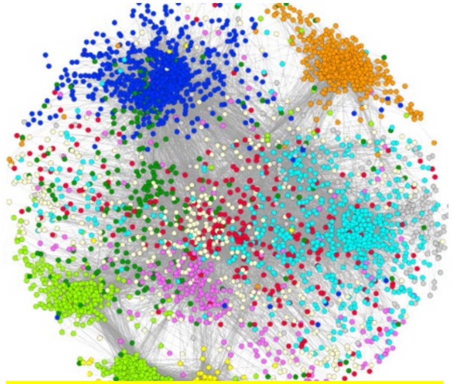
Institute of Data Science

National Cheng Kung University

chengte@mail.ncku.edu.tw

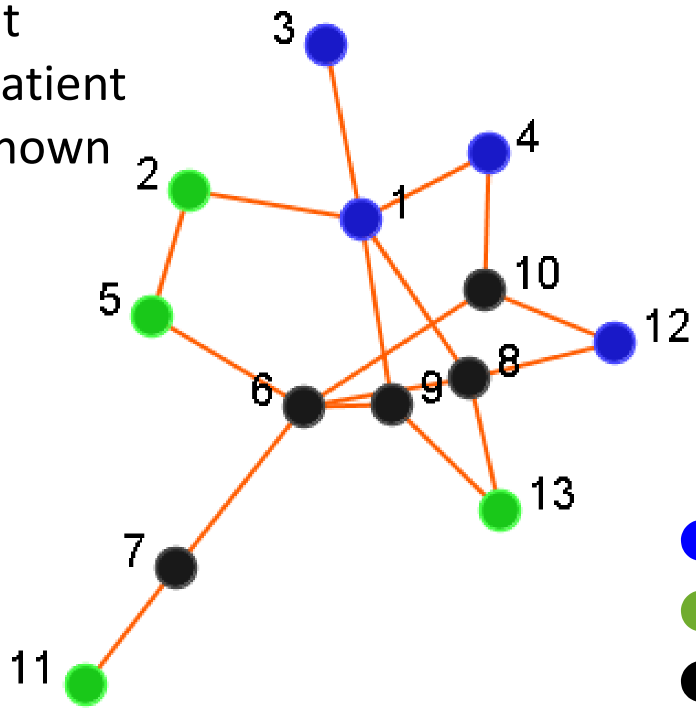
Node Classification





COVID-19 Test | Public Opinion | Online Advertising | Protein Analysis

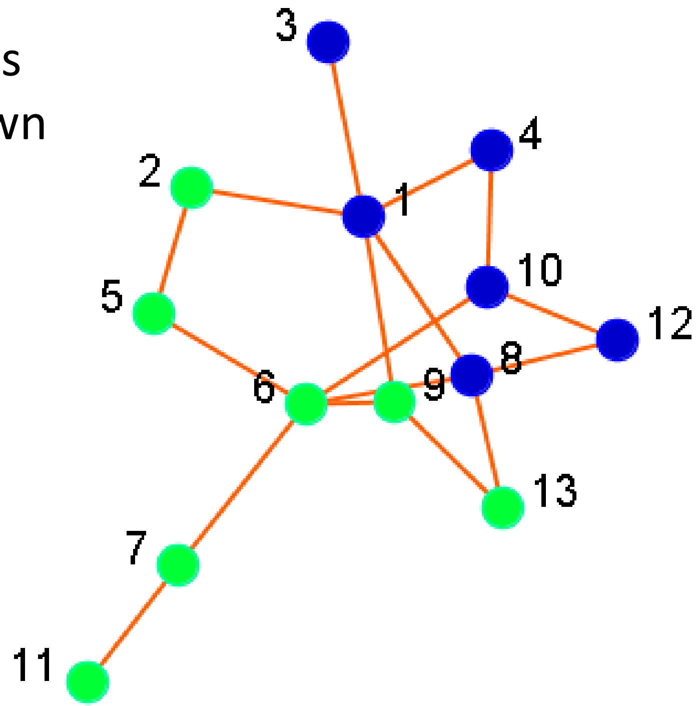
●: Patient
●: Non-Patient
●: ? Unknown



●: Gaming
●: Cosmetics
●: ? Unknown

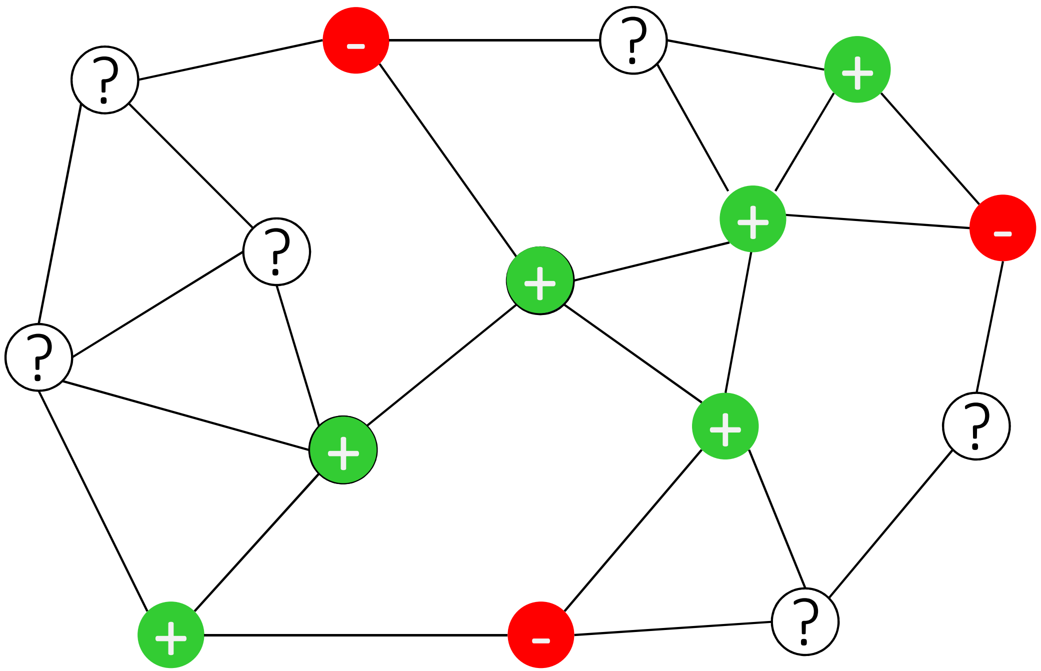
Prediction

●: Blue Party
●: Green Party
●: ? Unknown



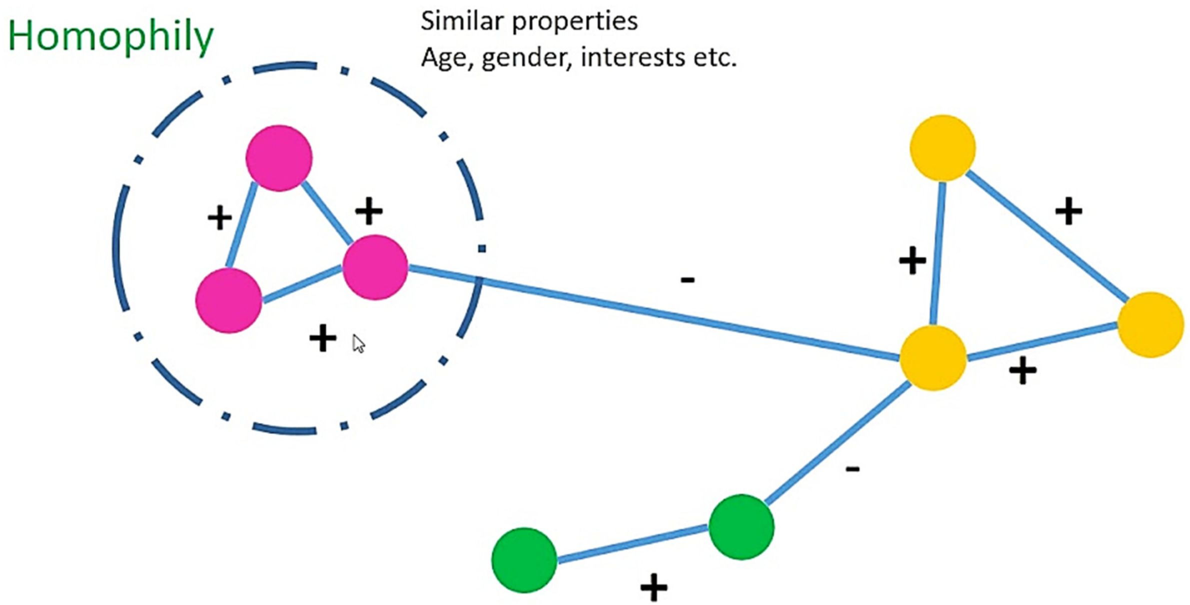
Node Classification Problem

- Some of the nodes in a network are labeled, the goal is to produce the labels of the rest nodes
  - Each node may have some attributes (features)
  - E.g. gender, income, hometown, interests, favorites, etc



Homophily 物以類聚

- Connected nodes tend to have the same label
  - People with similar characteristics tend to befriend each other
- Examples
  - Friends sharing common interests/preferences
  - Webpages hyperlinked to each other have the same topic
  - Papers cited with one another belong to the same area
  - Proteins frequently interacted possess the same function



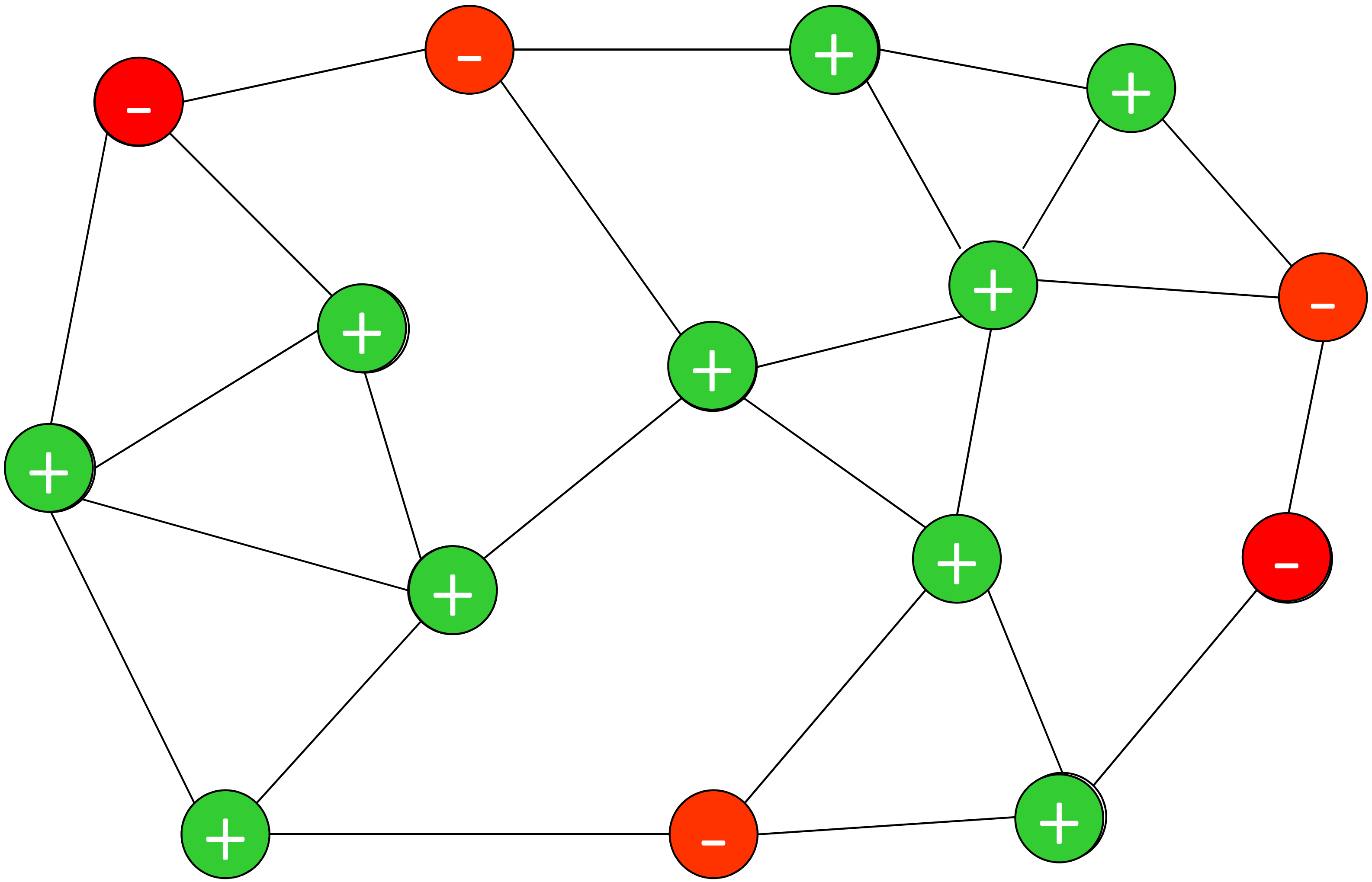
Relational Neighbor (RN) Classifier

- Estimate $P(c|v)$, the label-membership probability of node v having label c, as the weighted proportion of nodes in N_v that belong to label c

$$P(c|v) = \frac{1}{Z} \sum_{\{u \in N_v | label(u)=c\}} w(v,u) \qquad Z = \sum_{u \in N_v} w(v,u)$$

  - N_v: the set of neighboring nodes of node v
  - $w(v,u)$: the edge weight between nodes v and u
  - Nodes in N_v that are not of the same label as e are ignored
  - If N_v is empty or has no nodes with labels → use global $P(c)$
- Make the classification based on $\mathrm{argmax}_{c \in C} P(c|v)$

Example of RN Classifier



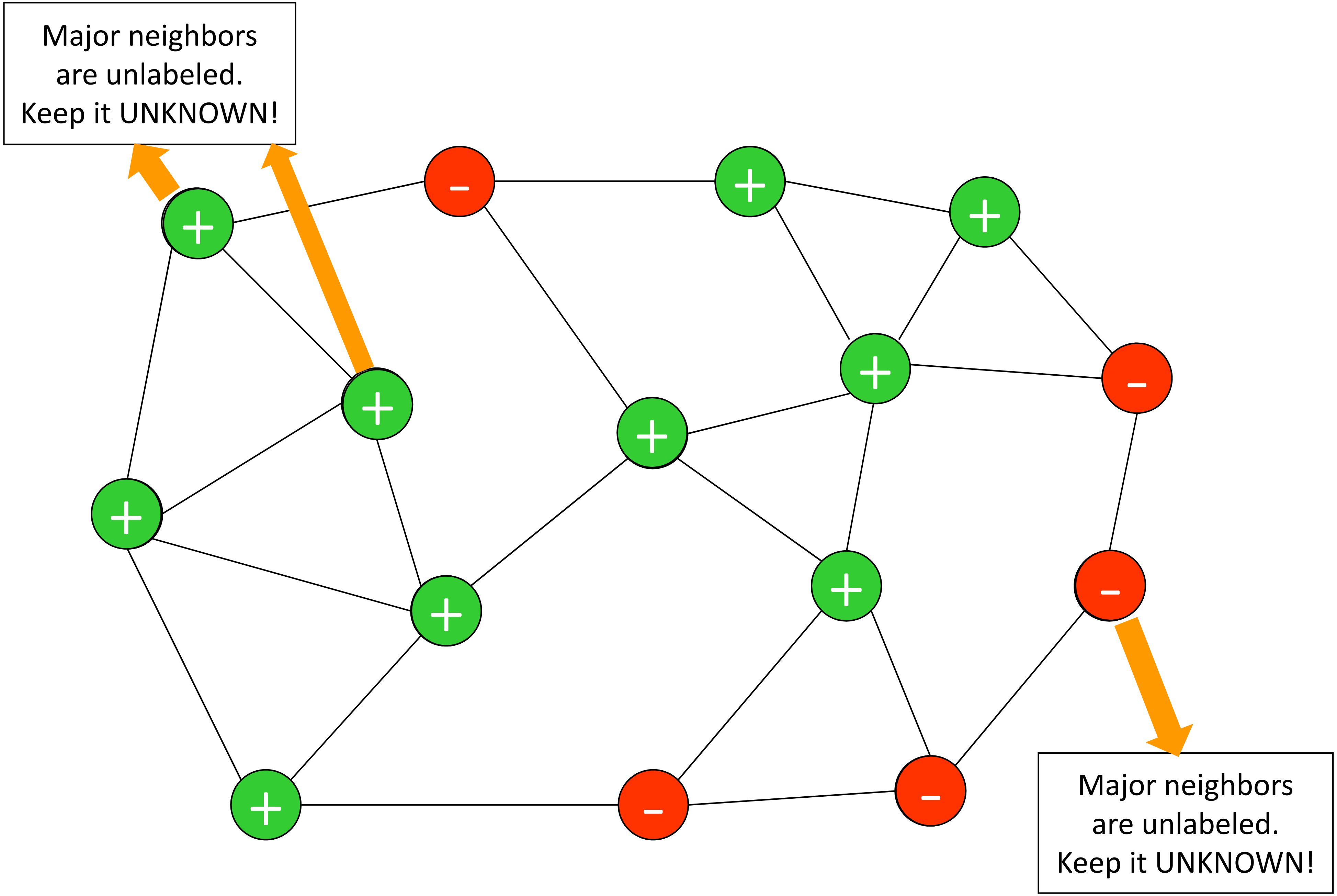
Problems of RN Classifier

- When surrounding by non-labeled nodes, assign class labels based on the prior
- Problem 1: If the labeled nodes are very sparse, the prediction will be dominated by label prior
- Problem 2: The order of the label classification could affect the results

Iterative Relational Neighbor Classifier

- Iteratively classify nodes using RN in its inner loop
- Unlabeled nodes that just get labels will affect the classification of the remaining labeled nodes
- At iteration i: $(i, j > 0)$
  - $RN(i)$ uses the labels derived by $RN(j)$, $j < i$, to estimate the probability $P(c|v)$ of currently unlabeled nodes
- Introduce the UNKNOWN tag
  - If the majority of the neighbors of node v are unlabeled
  - Delay the classification for nodes with the unknown tag until node v's majority of neighbors are labeled
- Stop when no unknown nodes are left
  or when no nodes can be classified

Example of Iterative RN Classifier



Weighted-Vote RN (wvRN) Classifier

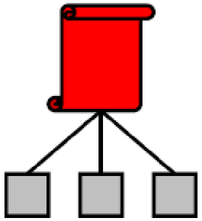
- wvRN estimates $P(c|v)$ as the weighed mean of the label-membership probabilities of nodes in N_v
  - Use RN to initialize $P(c|v)$
  - If v or u has no labeled neighbors, use the prior probabilities observed in the training data
  - Update $P(c|v)$ until convergence

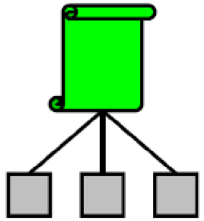
$$P(c|v) = \frac{1}{Z} \sum_{\boxed{u \in N_v}} w(v,u) \times \boxed{P(c|u)}$$

$$Z = \sum\nolimits_{u \in N_v} w(v,u)$$

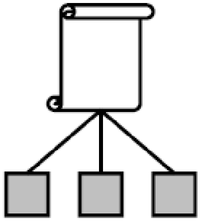
Attribute-only Node Classification

a1	a2	a3	L
0	1	0	R



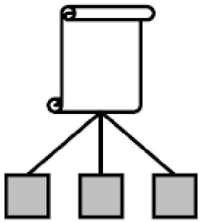


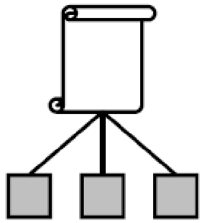
a1	a2	a3	L
1	1	0	G



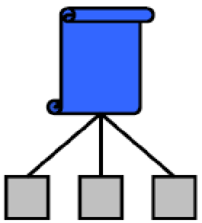
a1	a2	a3	L
1	1	0	?

a1	a2	a3	L
1	0	0	?





a1	a2	a3	L
1	0	1	?



a1	a2	a3	L
1	1	1	B

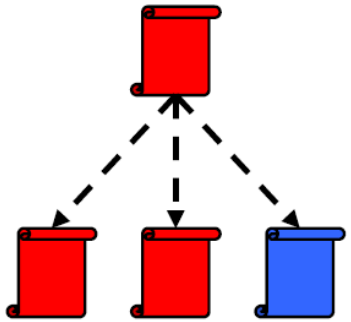
Attribute-only Node Classification

a1	a2	a3	L
1	0	0	R
1	1	0	R
0	1	1	B
0	0	1	B
0	0	1	G
0	0	0	G
0	1	1	?
1	0	1	?
0	0	0	?
0	0	1	?

Learn a classifier, such as Naïve Bayes, k-NN, Logistic Regression, etc

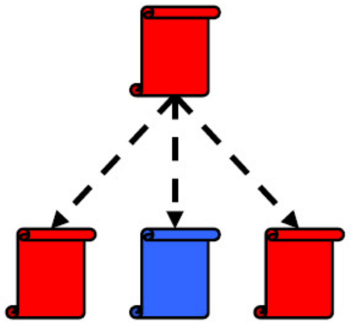
Use the classifier to predict these

Problem on Link-based Node Classification



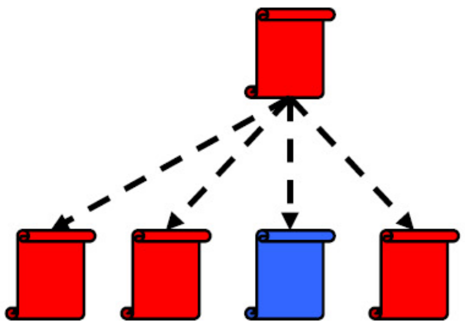
a1	a2	a3	N1	N2	N3	L
0	1	0	R	R	B	R

How do we order the neighbors?



a1	a2	a3	N1	N2	N3	L
0	1	0	R	B	R	R

What if different nodes have different number of neighbors?

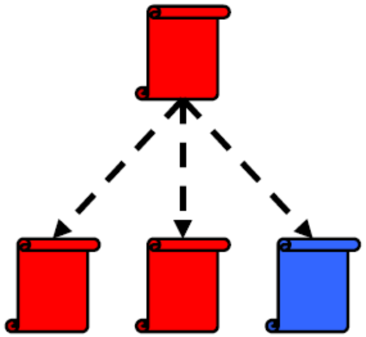


a1	a2	a3	N1	N2	N3	N4	L
0	1	0	R	R	B	R	R

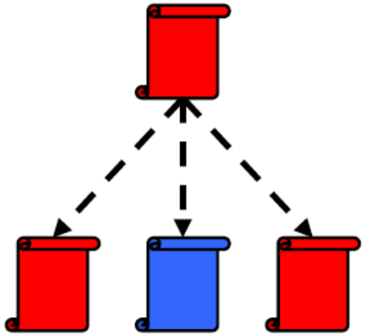
Information Aggregation

- Aggregate a set of attributes into a fixed length representation
  - Count
  - Proportion
  - Mode (Majority)
  - Exist (Binary)
  - Mean

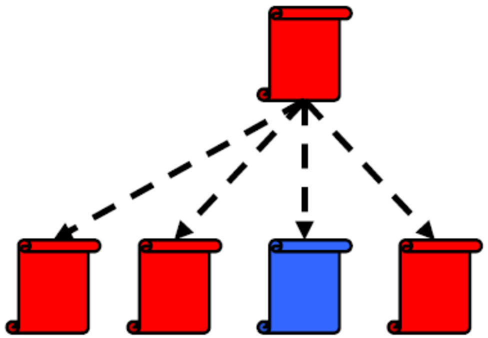
Aggregation: Count



a1	a2	a3	CR	CB	CG	L
0	1	0	2	1	0	R

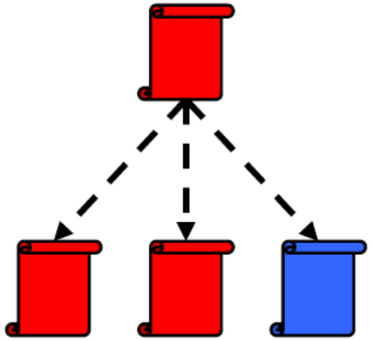


a1	a2	a3	CR	CB	CG	L
0	1	0	2	1	0	R

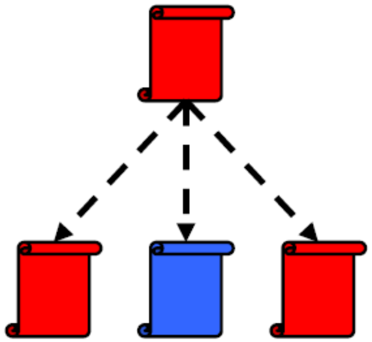


a1	a2	a3	CR	CB	CG	L
0	1	0	3	1	0	R

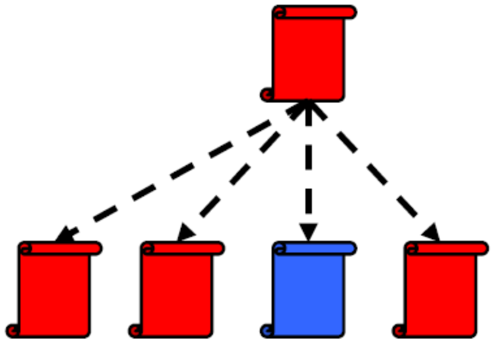
Aggregation: Proportion



a1	a2	a3	PR	PB	PG	L
0	1	0	0.67	0.33	0	R

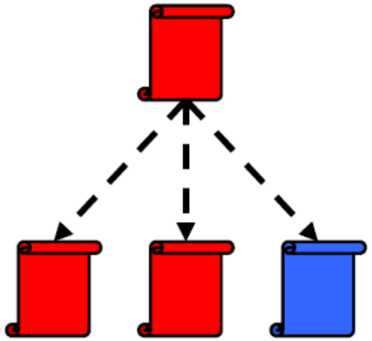


a1	a2	a3	PR	PB	PG	L
0	1	0	0.67	0.33	0	R

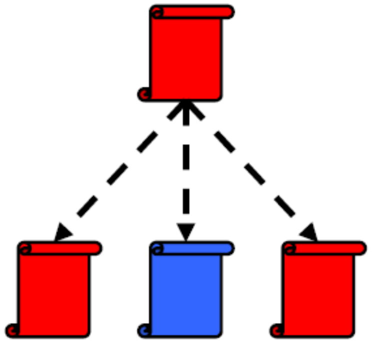


a1	a2	a3	PR	PB	PG	L
0	1	0	0.75	0.25	0	R

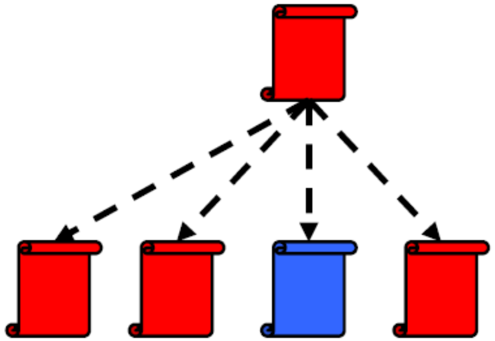
Aggregation: Exist



a1	a2	a3	ER	EB	EG	L
0	1	0	1	1	0	R



a1	a2	a3	ER	EB	EG	L
0	1	0	1	1	0	R



a1	a2	a3	ER	EB	EG	L
0	1	0	1	1	0	R

Iterative Classification Algorithm (ICA)

Bootstrap

1) Convert each node i to a feature vector $\mathbf{v}_i$
   - Various #neighbors → aggregation
   - E.g., mode, binary, count, proportion
2) Use Local Classifier $f(\mathbf{v}_i)$ to obtain its label y_i
   - e.g., SVM, LR, RF, XGBoost

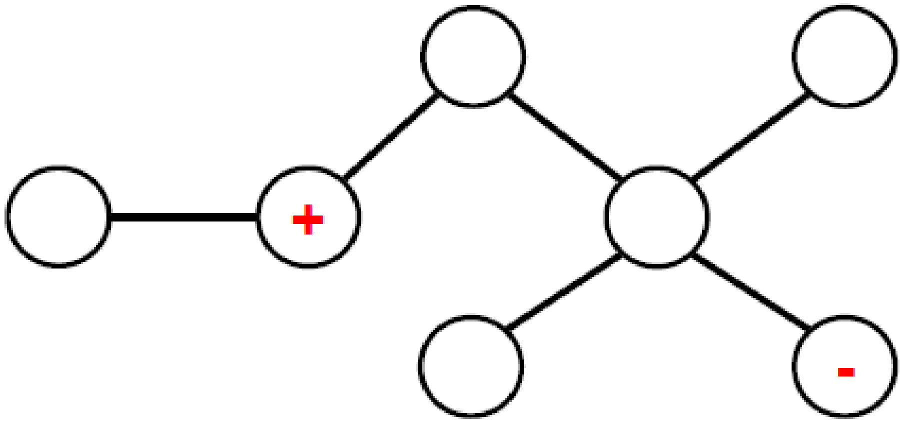
Iterative Classification

3) Repeat for each node i
   - Reconstruct feature vector $\mathbf{v}_i$ using current labels
   - Update label to $f(\mathbf{v}_i)$ based on prediction results

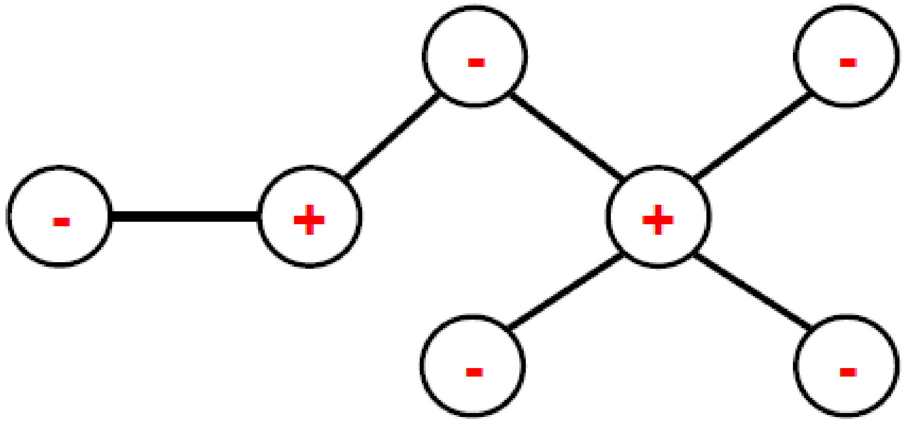
- Until labels are stabilized or max # iterations

Challenges on Node Labels

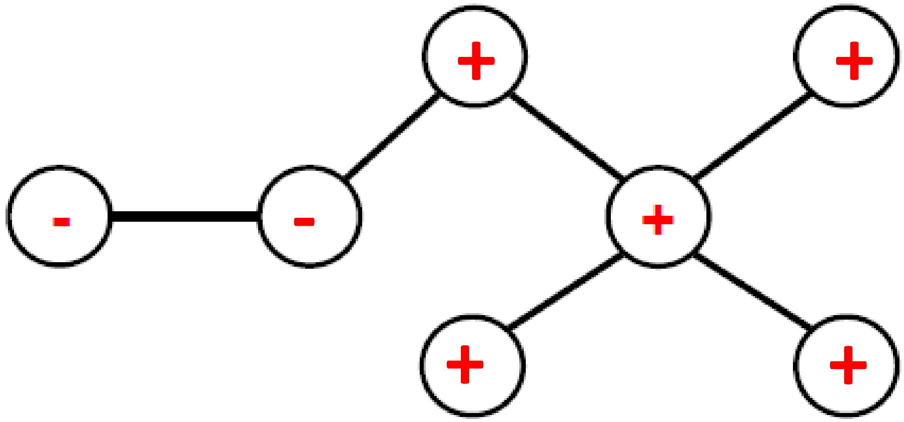
Sparse Labeling



Non-Homophily



Homophily

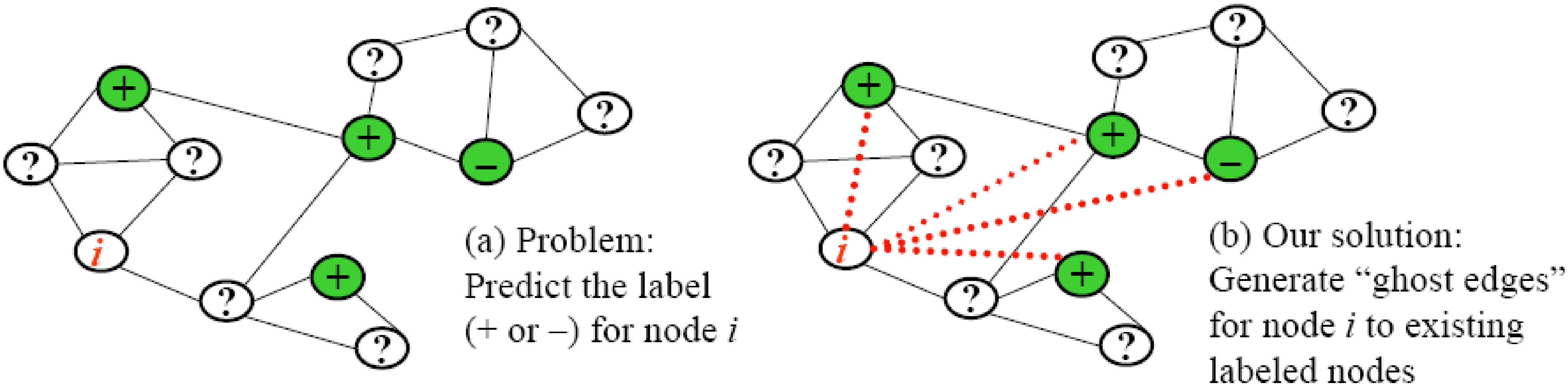


Solution: Ghost Edges

- To address the following problems:
  - Label Sparsity: # of unknown neighbors might be large
  - Link Sparsity: # of nodes are large, but loosely connected
  - Non-homophily: local positive correlation may not hold
- Solution: Adding Ghost Edges
  - Exploit information obtained from a node's non-neighbor nodes by adding ghost edges
  - Use learning methods to determine the effect of the other connected nodes

Ghost Edges

- Allow the information from labeled nodes to affect the classification of unlabeled nodes
- Create a single ghost edge between every <labeled, unlabeled> pair of nodes in our graph



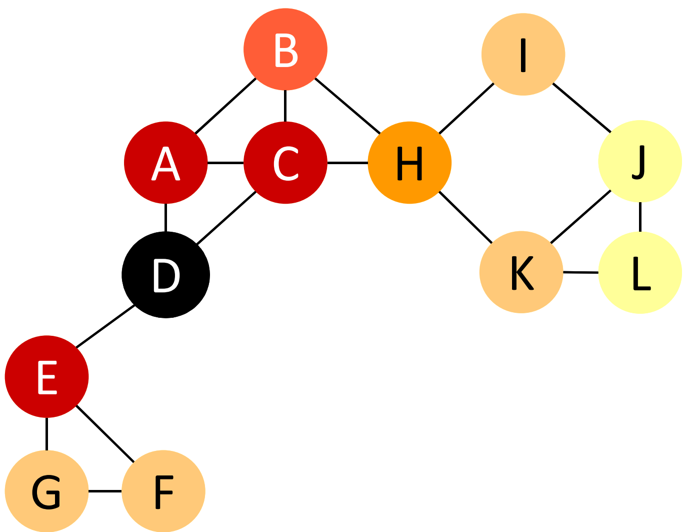
Label Sparsity: We have plenty of neighbors, but too few of them are labeled
Link Sparsity: There are plenty of labeled nodes, but we don't link to enough of them

Weighting Ghost Edges

- Ghost edges increase the number of labeled neighbors per node
- Ghost edge weights should correspond with correlation between node labels
- **Assumption**: Correlation is higher between labels of nodes that are "closer" to each other
  - Each node's influence is NOT equal
- Assign a **weight** to each ghost edge based on proximity

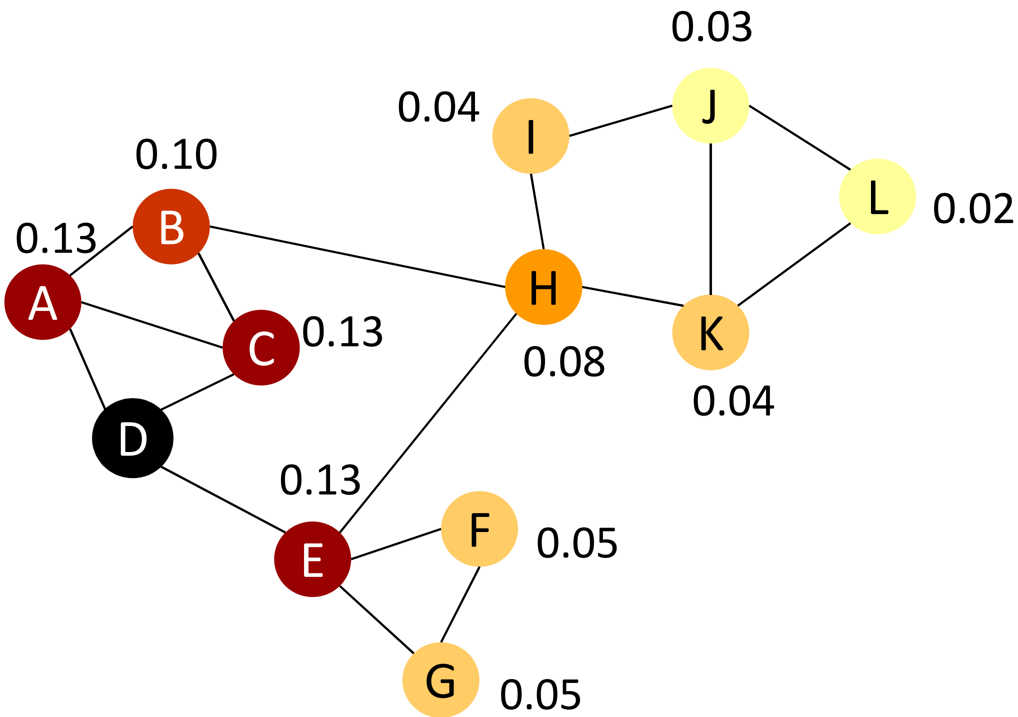
Measure Node Proximity by
Random Walk with Restart (RWR)

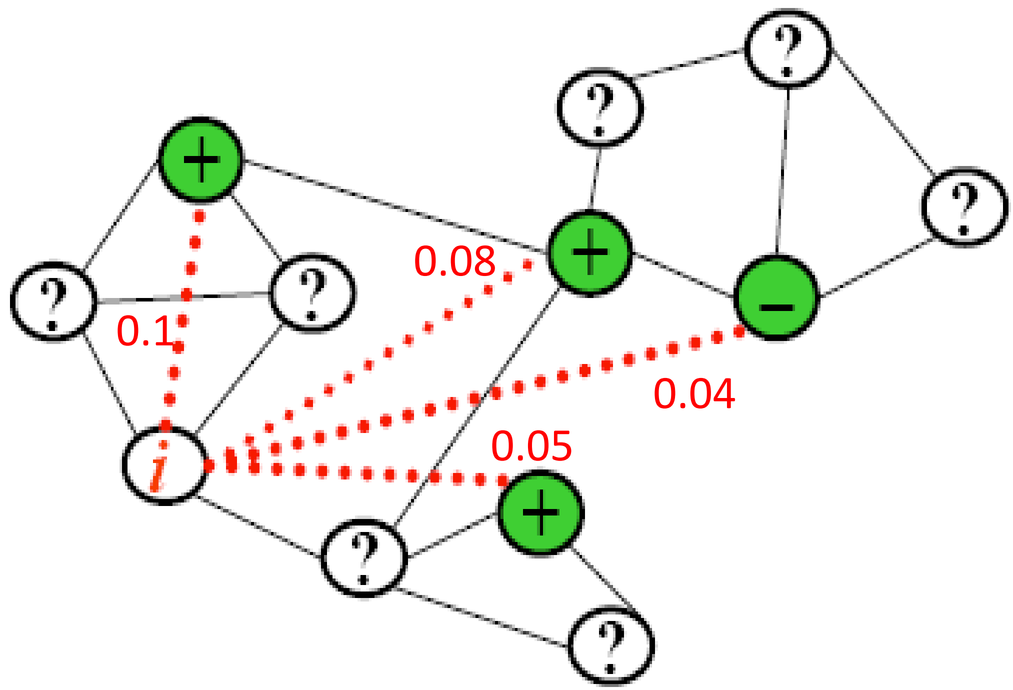
$$\underbrace{\underset{nx1}{R}}_{\text{Score Vector}} = \alpha \underbrace{\underset{nxn}{\widetilde{W}}}_{\text{Adjacency Matrix}} \underset{nx1}{R} + \underbrace{(1-\alpha)}_{\text{Fly-out Probability}} \underbrace{\underset{nx1}{E}}_{\text{Starting Vector}}$$



$$\begin{pmatrix} \text{[illegible]} \\ \text{[illegible]} \\ \text{[illegible]} \\ \text{[illegible]} \\ \text{[illegible]} \\ \text{[illegible]} \\ \text{[illegible]} \\ \text{[illegible]} \\ \text{[illegible]} \\ \text{[illegible]} \\ \text{[illegible]} \\ \text{[illegible]} \end{pmatrix} = 0.9 \times \begin{pmatrix} 0 & 1/3 & 1/3 & 1/3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1/3 & 0 & 1/3 & 0 & 0 & 0 & 0 & 1/4 & 0 & 0 & 0 & 0 \\ 1/3 & 1/3 & 0 & 1/3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1/3 & 0 & 1/3 & 0 & 1/4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/3 & 0 & 1/2 & 1/2 & 1/4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/4 & 0 & 1/2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/4 & 1/2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/3 & 0 & 0 & 1/4 & 0 & 0 & 0 & 1/2 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/4 & 0 & 1/3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 0 & 1/3 & 1/2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/4 & 0 & 1/3 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/3 & 1/3 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0.13 \\ 0.10 \\ 0.13 \\ 0.22 \\ 0.13 \\ 0.05 \\ 0.05 \\ 0.08 \\ 0.04 \\ 0.03 \\ 0.04 \\ 0.02 \end{pmatrix} + 0.1 \times \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

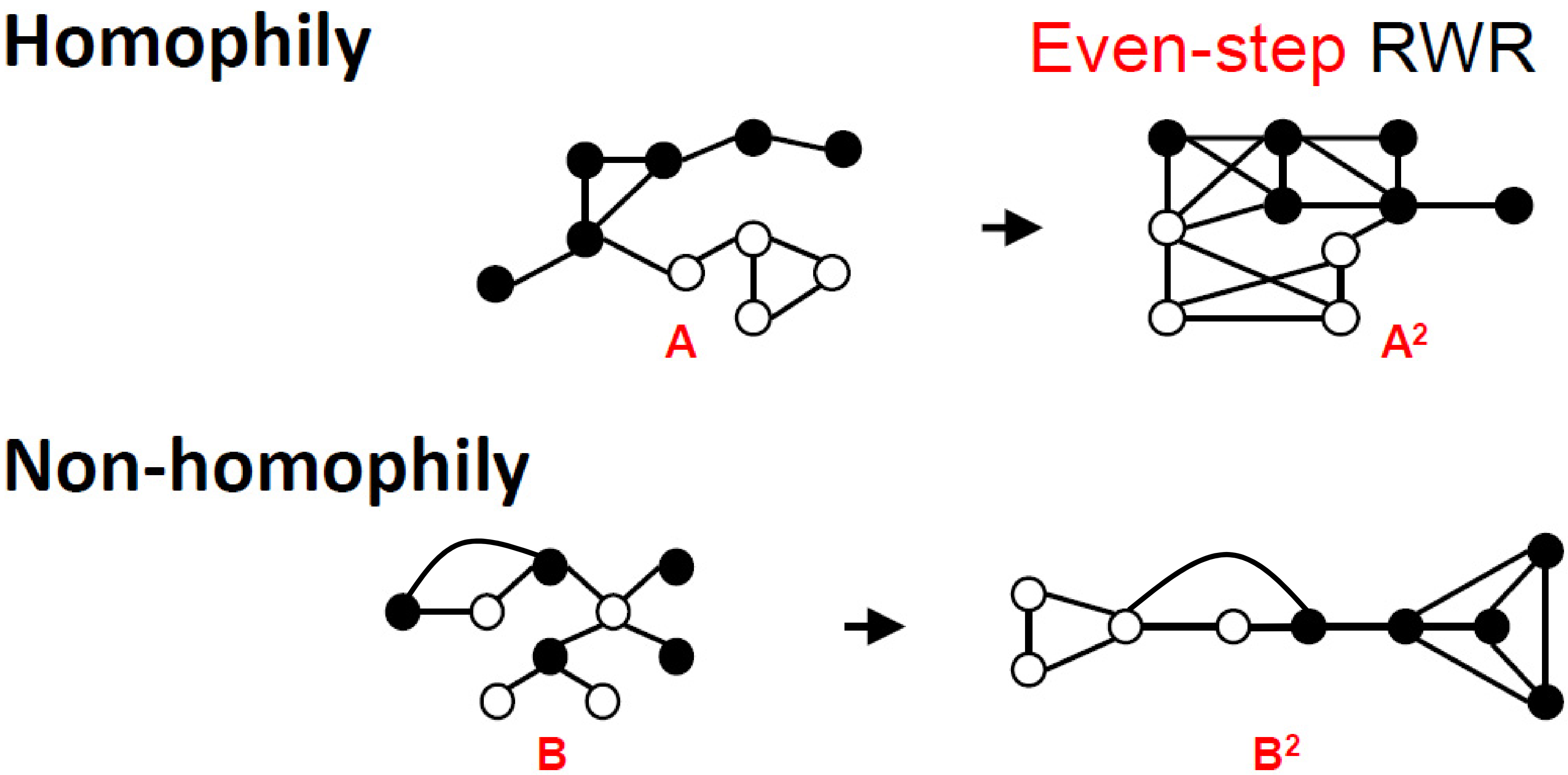
From Proximity to Ghost Edge Weights





How to Deal with non-Homophily?

- Even-step Random Walk with Restart



$$R = c\widetilde{W}^2 R + (1 - c)E$$

Two Ghost-Edge Classifiers

- **GhostEdgeNL** (non-learning)
  - Ignore observed edges
  - Create ghost edges from unlabeled to labeled nodes
  - Take weighted vote of ghost edge neighbors

→ Apply wvRN

- **GhostEdgeL** (learning)
  - Uses labeled nodes to learn label-dependencies separately across for observed edges and ghost edges
  - Bin ghost edges by proximity scores and learn dependencies separately for each bin (e.g., <0.1, 0.1~0.2, 0.2~0.4, >0.4)
  - Features
    - Count of neighbors of each class observed edges (2 features)
    - Count of neighbors of each class across ghost edges for each bin ("2 x number of bins" features)

→ Apply ML methods, e.g., SVM, LR, RF, XGBoost

GhostEdgeL

Representation of instances for learning

	Observed Edges		Ghost Edges								Class Label
	C+	C-	C+ (<0.1)	C- (<0.1)	C+ (0.1~0.2)	C- (0.1~0.2)	C+ (0.2~0.4)	C- (0.2~0.4)	C+ (>0.4)	C- (>0.4)	
v_1											
v_2											
v_3											
…											
v_n											

Short Summary

- Unsupervised Relational Neighbor-based
  - Relational Neighbor (RN) Classifier
  - Weighted Vote RN (wvRN) Classifier

Homophily
Within Network

- Supervised Learning-based
  - Link-based Node Classification
  - Iterative Classification Algorithm (ICA)

Cross/Within Network

- Random Walk-based
  - GhostEdge Algorithm

Flexible
Non-homophily & Very Sparse