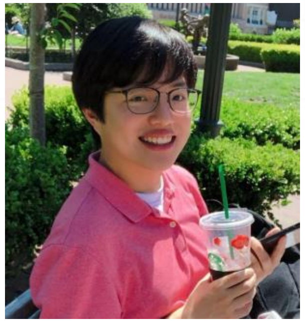
Are Weights Useful in Graph Summarization? – A Comparative Study

Shinhwan Kang   Kyuhan Lee   Kijung Shin

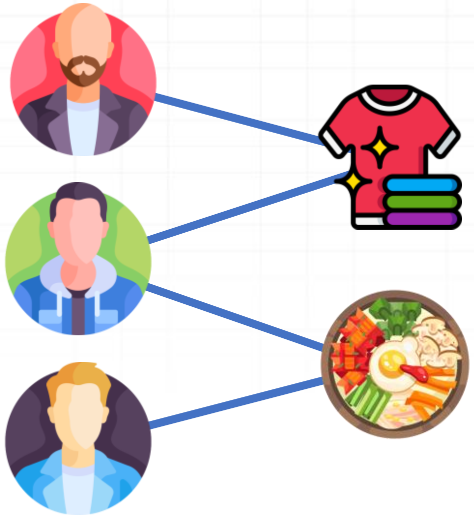
KAIST AI



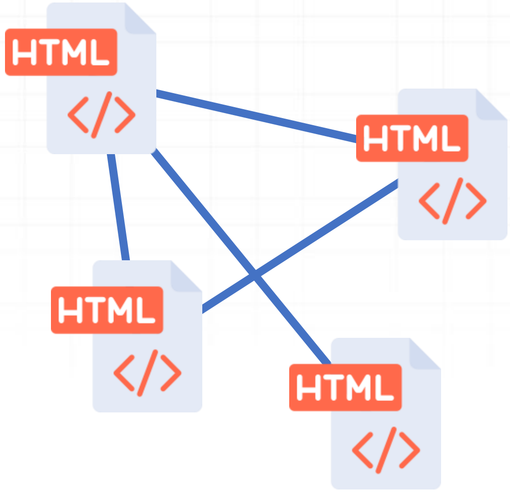
Graphs are everywhere!

- ***Graphs represent relationships*** such as
  - Friends in social networks
  - Purchase history
  - Hyperlinks between web pages

Social network



Purchase history



Web graph

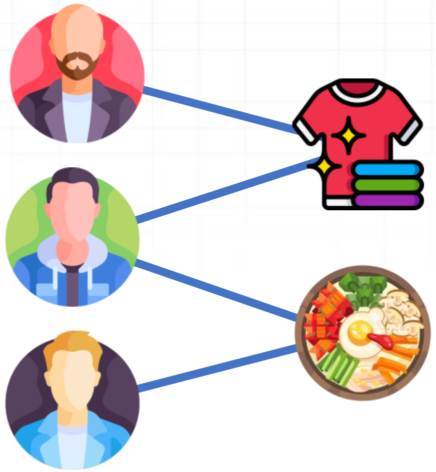
Graphs become large!

- Graphs ***grow rapidly*** at an unprecedented pace

25 million (2013)

157.4 million (2022)

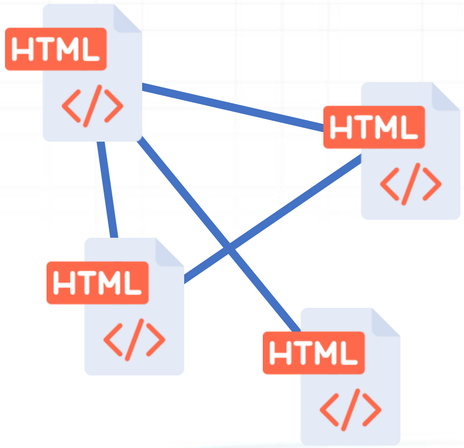
Users



0.6 billion (2013)

1.9 billion (2022)

Web pages



Graphs become large!

- Graphs ***grow rapidly*** at an unprecedented pace

25 million (2013)

⇩

157.4 million (2022)

Users

0.6 billion (2013)

⇩

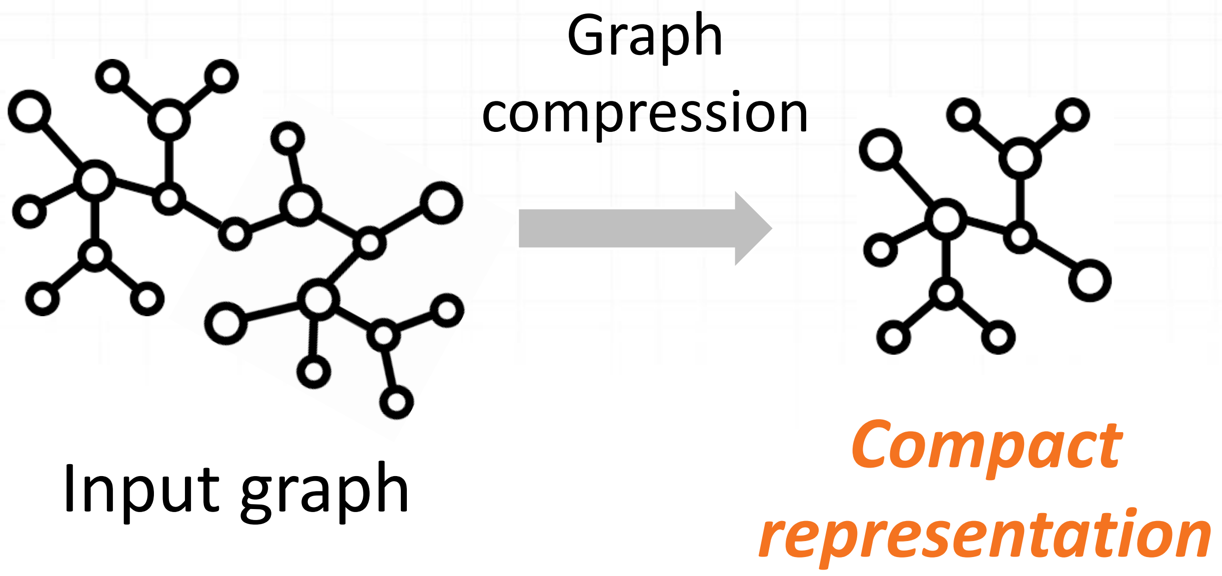
1.9 billion (2022)

Web pages

How do we efficiently utilize such large graphs?

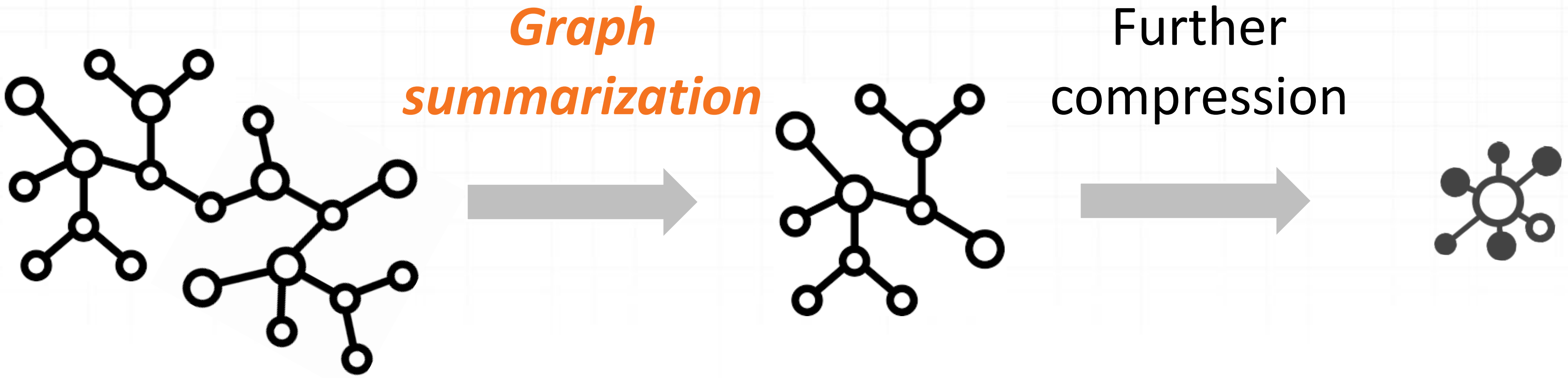
Graph compression

- Useful for ***efficient utilization of large graphs***
- To find a ***compact representation*** that exactly or approximately describes an input graph



Graph summarization: overview

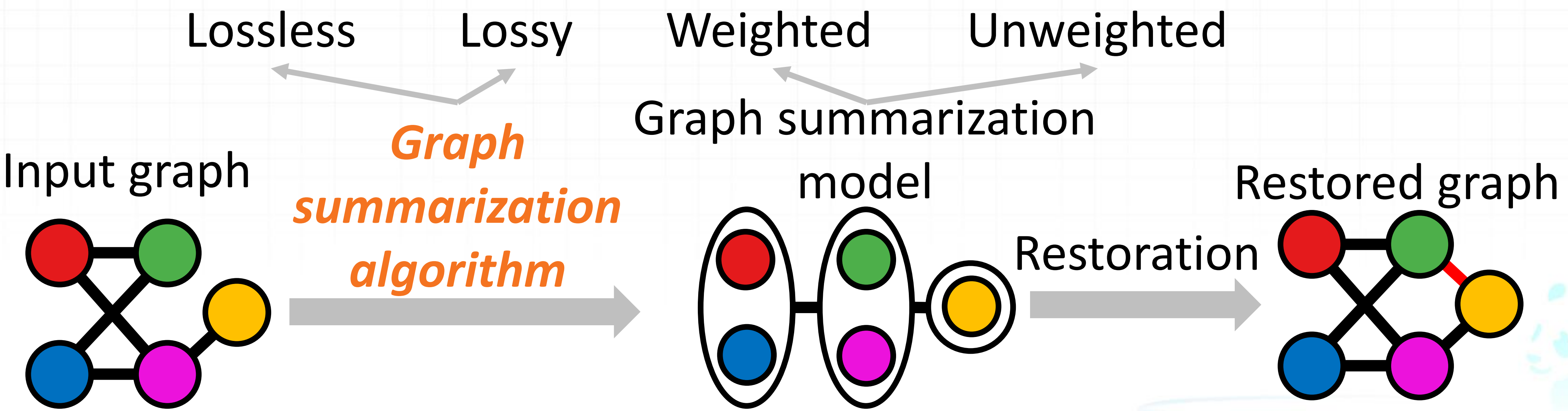
- Promising graph compression technique
- ***Summary graph*** is in the ***form of a graph***
  - Can process graph queries directly without restoration
  - Can apply other graph compression techniques



Input graph | ***Summary graph*** | Compressed graph

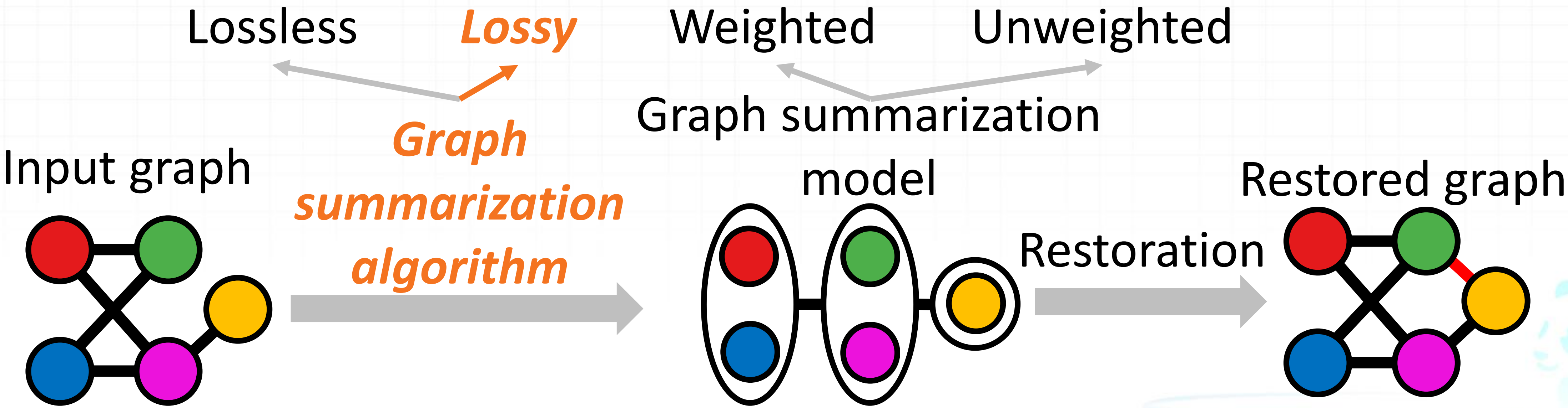
Graph summarization: overview

- We can categorize graph summarization methods ***depending on loss of information*** during summarization



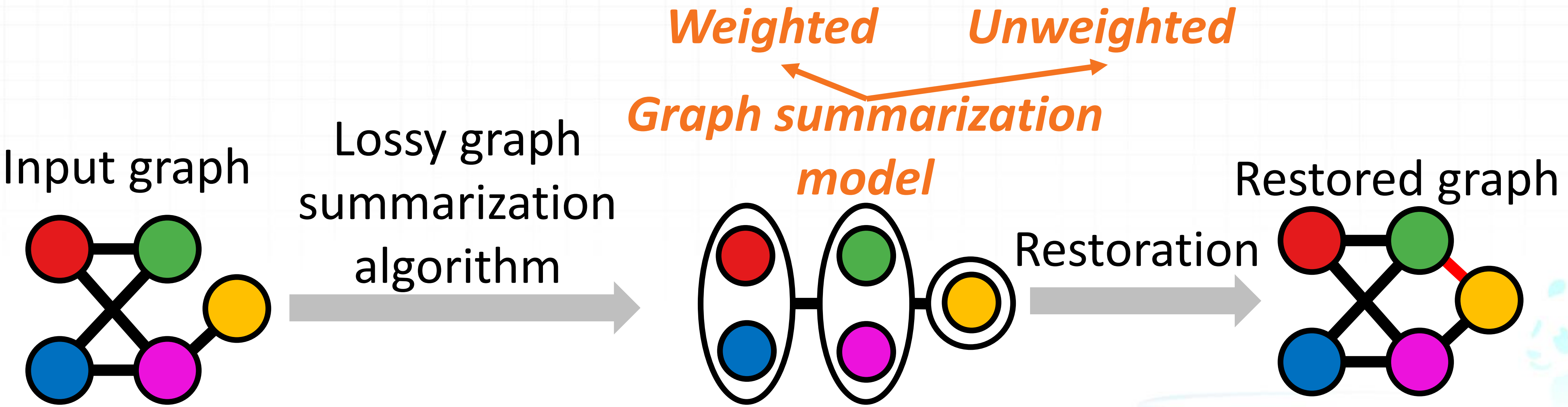
Graph summarization: overview

- We can categorize graph summarization methods depending on loss of information during summarization
- We *focus on lossy graph summarization*



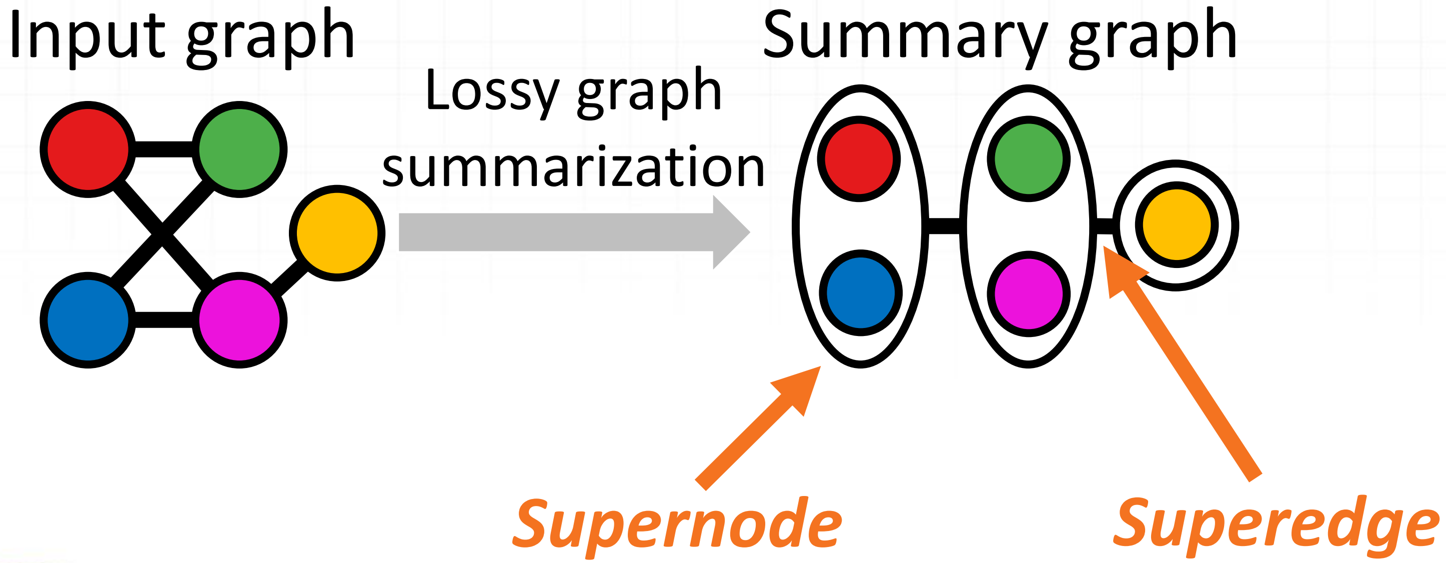
Graph summarization models

- We also can categorize graph summarization methods ***depending on summarization models***
- One of the two representative models ***allows edge weights in summary graphs***; the other does not



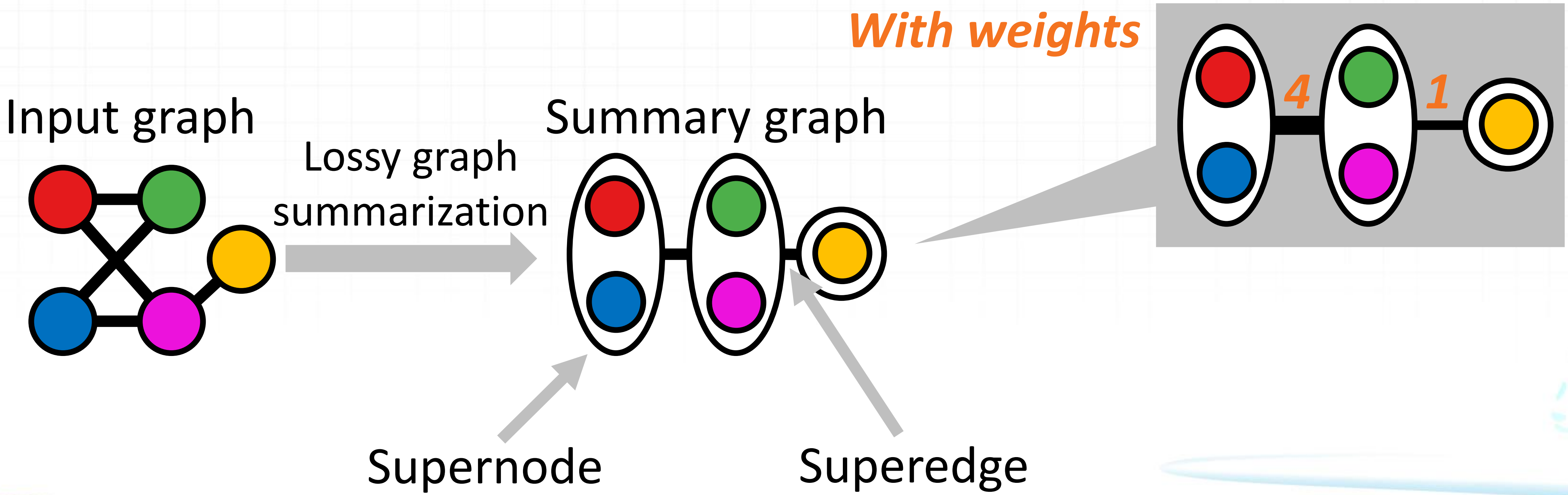
Weighted graph summarization model

- ***Supernode***: a node in a summary graph
- ***Superedge***: an edge in a summary graph



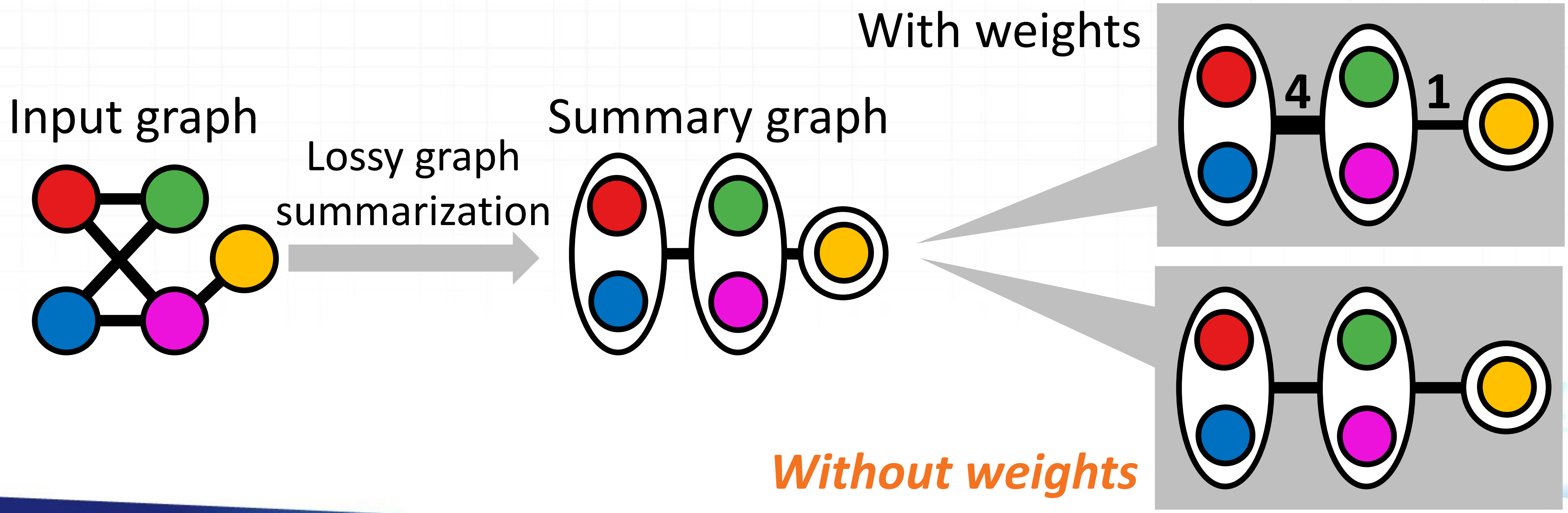
Weighted graph summarization model

- Summary graphs with edge weights ***contain information about the number of edges*** on each superedge



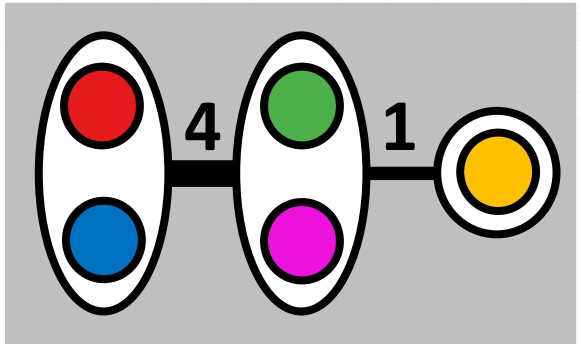
Unweighted graph summarization model

- Summary graphs without edge weights ***do not retain information about the number of edges*** on each superedge

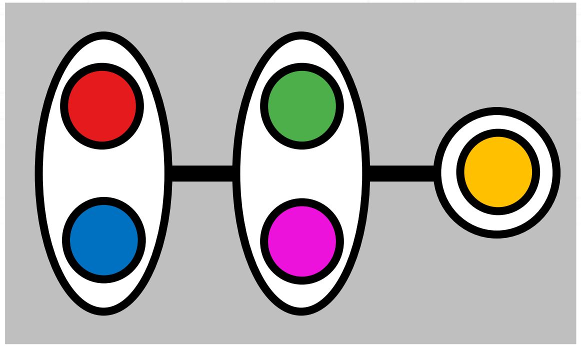


Which one is better between two models?

- There was ***no systematic comparison*** between two extensively-studied graph summarization models



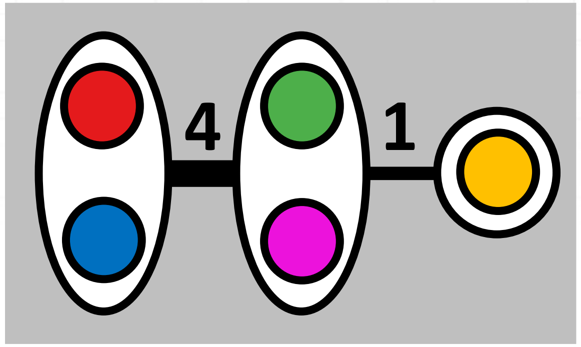
Weighted graph summarization model



Unweighted graph summarization model

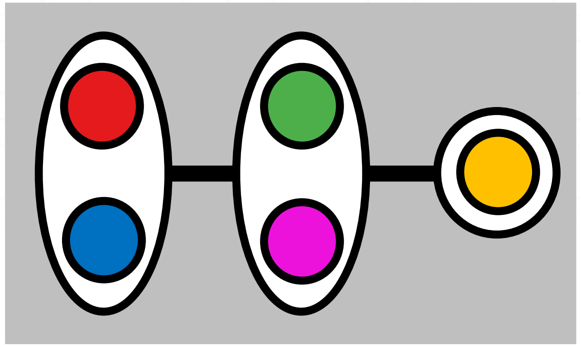
Which one is better between two models?

- There was no systematic comparison between two extensively-studied graph summarization models
- We conduct ***a systematic comparison*** in five aspects
  - For example, reconstruction and compression ratios
- To this end, we ***extend three algorithms to both models***



Weighted graph summarization model

Which one is better?



Unweighted graph summarization model

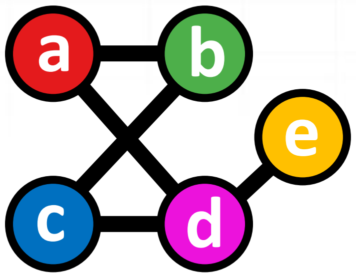
Road map

- Introduction
- ***Notations & Problem formulation <<***
- Considered algorithms
- Experiments & Theoretical analysis
- Conclusion

Notations: input graph

- Input graph $G = (V, E)$
  - Set of ***subnodes*** $V = \{1,2, \ldots, |V|\}$
  - Set of ***subedges*** $E \subseteq \binom{V}{2}$

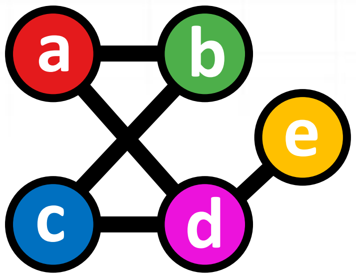
Input graph $G = (V, E)$



Notations: input graph

- Input graph $G = (V, E)$
  - Set of subnodes $V = \{1, 2, \ldots, |V|\}$
  - Set of subedges $E \subseteq \binom{V}{2}$
- ***Adjacency matrix A*** $\in \mathbb{R}^{|V| \times |V|}$ of G
  - $A_{\mathrm{ij}} = 1$ if $\{\mathrm{i}, \mathrm{j}\} \in E$ and $A_{\mathrm{ij}} = 0$ otherwise

Input graph $G = (V, E)$

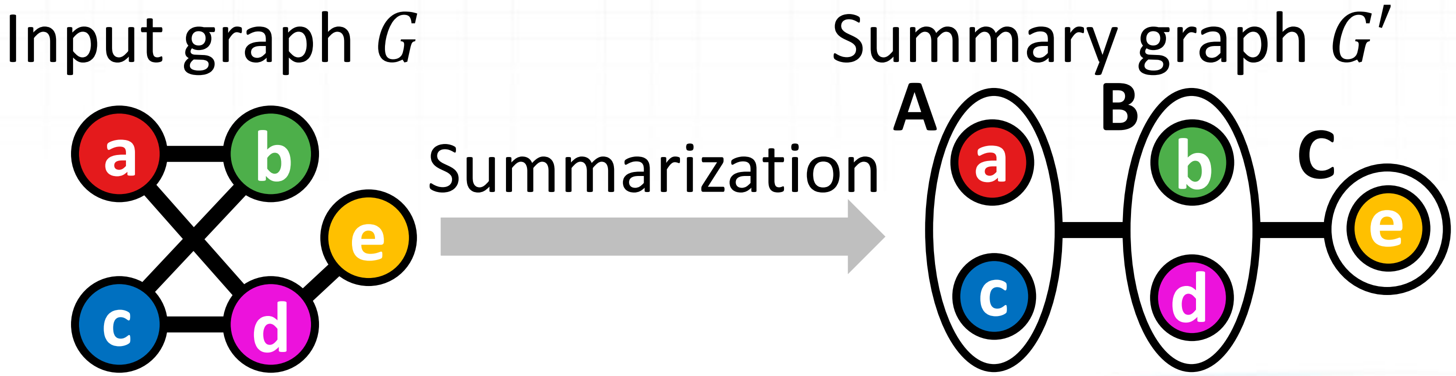


	a	b	c	d	e
a	0	1	0	1	0
b	1	0	1	0	0
c	0	1	0	1	0
d	1	0	1	0	1
e	0	0	0	1	0

←*Adjacency matrix A*

Notations: summary graph

- Summary graph $G' = (S, P)$ of $G = (V, E)$
  - Set of ***supernodes*** S is a partition of V
  - Set of ***superedges*** $P \subseteq \binom{S}{2}$



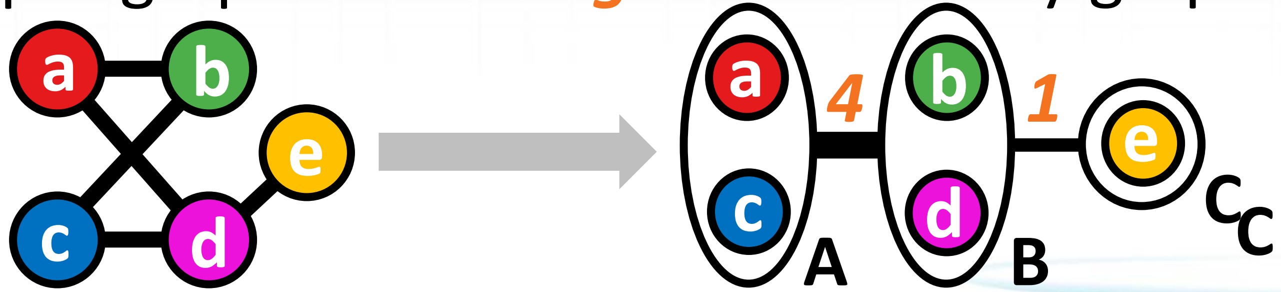
(Un)weighted summary graph

- Summary graph G' is either ***weighted*** or ***unweighted***

Superedge weight function

- Summary graph G' is either weighted or unweighted
- ***Weighted*** summary graph additionally has a ***superedge weight function ω***
  - ω takes each superedge and returns the its weights
    - (e.g.,) $\omega_{AB} = 4, \omega_{BC} = 1$

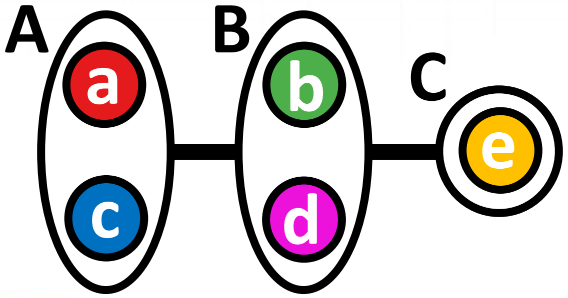
Input graph G ***Weighted*** summary graph



(Un)weighted reconstructed graph

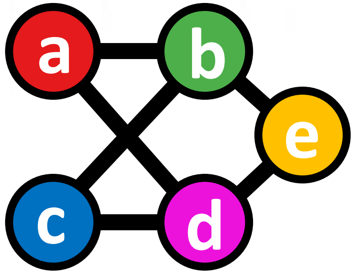
- ***Reconstructed*** graph $\hat{G}$ is obtained from a summary graph G'
- $\hat{A}$ denotes an ***adjacency matrix*** of a reconstructed graph $\hat{G}$

Summary graph G'



Reconstruct

Reconstructed graph $\hat{G}$



	a	b	c	d	e
a					
b					
c					
d					
e					

←***Reconstructed adjacency matrix*** $\hat{A}$

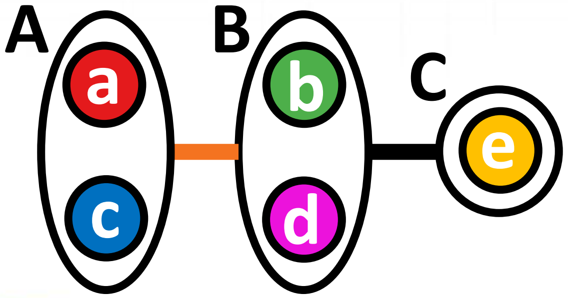
Reconstructed adjacency matrix

- With ***unweighted*** summary graphs, entries of $\hat{A}$ are defined as

$$\hat{A}_{ij} = \begin{cases} 1, & \text{if } i \neq j \text{ and } \{\boxed{S_i}, S_j\} \in P \\ 0, & \text{otherwise} \end{cases}$$

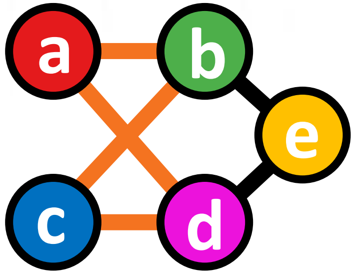
Supernode containing node i

Unweighted summary graph G'



Reconstruct

Reconstructed graph $\hat{G}$



	a	b	c	d	e
a	0	1	0	1	0
b	1	0	1	0	1
c	0	1	0	1	0
d	1	0	1	0	1
e	0	1	0	1	0

←Reconstructed adjacency matrix $\hat{A}$

Reconstructed adjacency matrix

- With ***unweighted*** summary graphs, entries of $\hat{A}$ are defined as

$$\hat{A}_{ij} = \begin{cases} 1, & \text{if } i \neq j \text{ and } \{\boxed{S_i}, S_j\} \in P \\ 0, & \text{otherwise} \end{cases}$$

Supernode containing node i

Unweighted summary graph G'

Reconstructed graph $\hat{G}$

	a	b	c	d	e
a	0	1	0	1	0
b	1	0	1	0	1
c	0	1	0	1	0
d	1	0	1	0	1
e	0	1	0	1	0

←Reconstructed adjacency matrix $\hat{A}$

Reconstructed adjacency matrix

- With ***weighted*** summary graphs, entries of $\hat{A}$ are defined as

$$\hat{A}_{ij} = \begin{cases} \dfrac{\omega_{S_iS_j}}{\boxed{\pi_{S_iS_j}}}, & \text{if } i \neq j \text{ and } \{S_i, S_j\} \in P \\ 0, & \text{otherwise} \end{cases}$$

π_{AB}: # of possible subedges between supernodes A and B

Weighted summary graph G'

Reconstructed graph $\hat{G}$

	a	b	c	d	e
a	0	1	0	1	0
b	1	0	1	0	0.5
c	0	1	0	1	0
d	1	0	1	0	0.5
e	0	0.5	0	0.5	0

←Reconstructed adjacency matrix $\hat{A}$

Reconstructed adjacency matrix

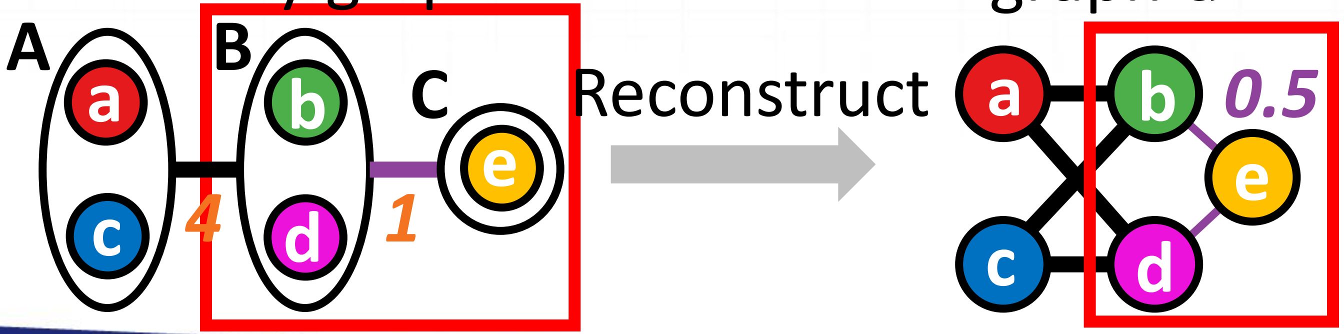
- With ***weighted*** summary graphs, entries of $\hat{A}$ are defined as

$$\hat{A}_{ij} = \begin{cases} \frac{\omega_{S_iS_j}}{\boxed{\pi_{S_iS_j}}}, & \text{if } i \neq j \text{ and } \{S_i, S_j\} \in P \\ 0, & \text{otherwise} \end{cases}$$

π_{AB}: # of possible subedges between supernodes A and B

Weighted summary graph G'

Reconstructed graph $\hat{G}$



	a	b	c	d	e
a	0	1	0	1	0
b	1	0	1	0	0.5
c	0	1	0	1	0
d	1	0	1	0	0.5
e	0	0.5	0	0.5	0

←Reconstructed adjacency matrix $\hat{A}$

Optimization problem formulation

- ***Given***: input graph G and a size budget k
- ***Find***: summary graph G'
  - G': (Un)weighted summary graph
- ***To minimize***: the L_p reconstruction error $||A - \hat{A}||_p$
- ***Subject to***: the size of summary graph G', $Size(G') \leq k$
  - (e.g.,) # of supernodes in G', # of bits to encode G'

Size of summary graph in bits

- The size of an unweighted summary graph in bits is defined as

$$\boxed{Size_{bits}(G') = 2|P|\, log_2|S| + |V|\, log_2\, |S|}$$

Size of summary graph in bits

- The size of an unweighted summary graph in bits is defined as

$$\boxed{Size_{bits}(G') = \mathbf{2|P|\, log_2|S|} + |V|\, log_2\,|S|}$$

- $\mathbf{2|P|\, log_2\, |S|}$ corresponds to $|P|$ ***superedges in bits***

Size of summary graph in bits

- The size of an unweighted summary graph in bits is defined as

$$\boxed{Size_{bits}(G') = \mathbf{2|P|\,log_2|S|} + |V|\,log_2|S|}$$

- $2|P|\log_2|S|$ corresponds to $|P|$ superedges in bits
- $\mathbf{|V|\,log_2|S|}$ corresponds to ***supernodes membership of $|V|$ subnodes in bits***

Size of summary graph in bits

- The size of a ***weighted*** summary graph in bits is defined as

$$\boxed{Size_{bits}(G') = 2|P|\, log_2|S| + |V|\, log_2\,|S| + \boldsymbol{|P|\, log_2\, \omega_{max}}}$$

- $2|P| \log_2 |S|$ corresponds to $|P|$ superedges in bits
- $|V| \log_2 |S|$ corresponds to supernodes membership of $|V|$ subnodes in bits
- For a ***weighted summary graph***, $\boldsymbol{|P|\, log_2\, \omega_{max}}$ corresponds to $|P|$ ***superedge weights in bits***

Road map

- Introduction
- Notations & Problem formulation
- ***Considered algorithms <<***
- Experiments & Theoretical analysis
- Conclusion

Optimization algorithms

- We introduce three ***optimization algorithms (W)*** for finding ***a weighted summary graph***
  - *k-Grass* (W) [1], *SSumM* (W) [2], *MoSSo-Lossy* (W) [3]

Extending optimization algorithms

- We introduce three optimization algorithms (W) for finding a weighted summary graph
  - *k-Grass* (W) [1], *SSumM* (W) [2], *MoSSo-Lossy* (W) [3]
- We ***extend*** each algorithm ***to provide an unweighted summary graph (U)*** by modifying their objective function

Algorithm
k-Grass (W)
SSumM (W)
MoSSo-Lossy (W)

Algorithm
k-Grass (U)
SSumM (U)
MoSSo-Lossy (U)

Extending optimization algorithms

- We introduce three optimization algorithms (W) for finding a weighted summary graph
  - *k-Grass* (W) [1], *SSumM* (W) [2], *MoSSo-Lossy* (W) [3]
- We extend each algorithm to provide an unweighted summary graph (U) by modifying their objective function

Algorithm
k-Grass (W)
SSumM (W)
MoSSo-Lossy (W)

Algorithm
k-Grass (U)
SSumM (U)
MoSSo-Lossy (U)

Inputs & output

- Given an input graph G, *k-Grass* ***produces a summary graph*** whose size is smaller than the size budget

Input: (1) ***input graph*** $\boldsymbol{G}$, and (2) ***size budget*** $\boldsymbol{k}$

Output: ***summary graph*** $\boldsymbol{G'}$

1. initialize G'
2. while $Size(G') > k$ do
3. merge a supernode pair $\{A, B\}$ whose merger increases $Loss()$ least
4. return G'

Initialization

- It ***first initializes the set of supernodes*** so that each subnode forms a singleton supernode

Input: (1) input graph G, and (2) size budget k
Output: summary graph G'
1. ***initialize*** G'
2. while $Size(G') > k$ do
3. merge a supernode pair $\{A, B\}$ whose merger increases $Loss()$ least
4. return G'

Merging

- After that, it ***greedily merges a supernode pair*** whose merger increases the objective function $\boldsymbol{Loss}()$ least

Input: (1) input graph G, and (2) size budget k
Output: summary graph G'
1. initialize G'
2. while $Size(G') > k$ do
3. ***merge a supernode pair*** $\{A, B\}$ whose merger increases $\boldsymbol{Loss}()$ least
4. return G'

Loss function of k-Grass

- Loss function of *k-Grass* ***(U, W)*** is the L_p reconstruction error

$$\left|\left|A - \hat{A}\right|\right|_p$$

where $\hat{A}$ is an adjacency matrix of a reconstructed graph

- Note that the adjacency matrix is ***reconstructed from an unweighted summary graph (U) or from a weighted summary graph (U)***

Termination

- If the size of summary graph $\boldsymbol{Size(G')}$ is smaller than the budget, it returns a summary graph

Input: (1) input graph G, and (2) size budget k
Output: summary graph G'
1. initialize G'
2. while $\boldsymbol{Size(G') > k}$ do
3. merge a supernode pair $\{A, B\}$ whose merger increases $Loss()$ least
4. return G'

Road map

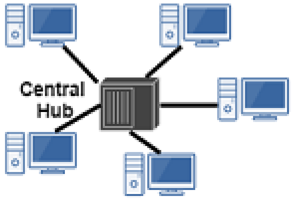
- Introduction
- Notations & Problem formulation
- Considered algorithms
- ***Experiments* & Theoretical analysis <<**
- Conclusion

Experiments: settings

- Datasets
  - 8 Real-world graphs (0.2M - 0.2B edges)

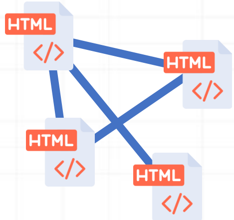
Social | Internet | Collaboration | Co-purchase | Hyperlinks | Email

LiveJournal



dblp

amazon



DBpedia

ENRON

- Considered graph summarization algorithms
  - {k-Grass, SSumM, MoSSo-Lossy} X {Weighted (W), Unweighted (U)}

Experiments: metrics

- ***Compression ratio***

$$\frac{Size_{bits}(G')}{2|E|\log_2|V|}$$

where $G = (V, E)$ is the input graph and G' is a summary graph

Experiments: metrics

- Compression ratio $$\frac{Size_{bits}(G')}{2|E|\log_2|V|}$$

where $G = (V, E)$ is the input graph and G' is a summary graph

- ***Quality: L_1/L_2 reconstruction error*** $||A - A'||_{p=\{1,2\}}$

Experiments: metrics

- Compression ratio $\dfrac{Size_{bits}(G')}{2|E|\log_2|V|}$

where $G = (V, E)$ is the input graph and G' is a summary graph

- Quality: L_1/L_2 reconstruction error $||A - A'||_{p=\{1,2\}}$
- ***Node importance: PageRank (PR)*** [4]
  - damping factor = 0.85

Experiments: metrics

- Compression ratio $$\frac{Size_{bits}(G')}{2|E|\log_2|V|}$$

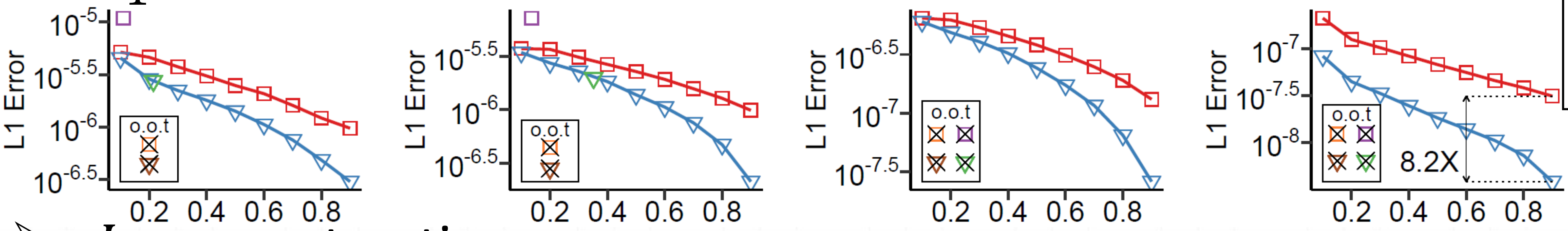
where $G = (V, E)$ is the input graph and G' is a summary graph

- Quality: L_1/L_2 reconstruction error $||A - A'||_{p=\{1,2\}}$
- Node importance: PageRank (PR) [4]
  - damping factor = 0.85
- ***Node proximity: Random Walk with Restart (RWR)*** [5]
  - damping factor = 0.95 with 100 randomly-sampled seeds

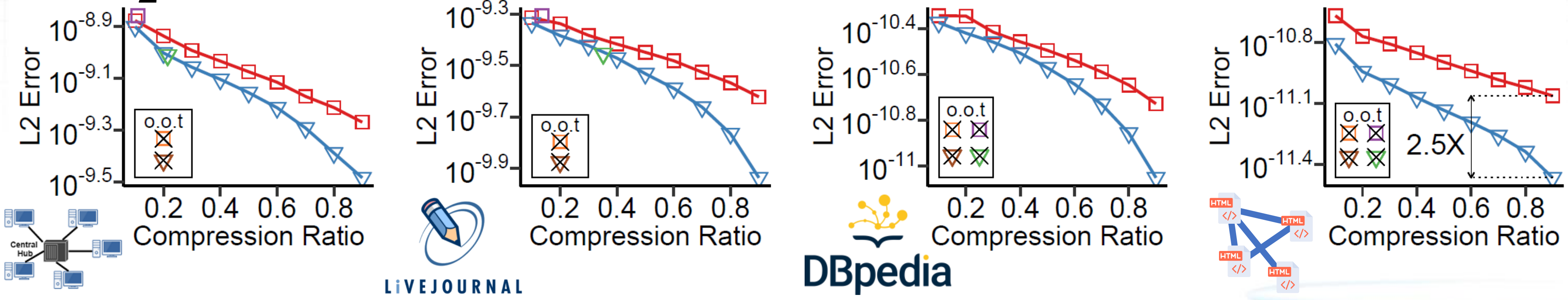
Experiments: quality of summary graphs

- Unweighted summary graphs ***described the input graph up to 8.2X more accurately*** than weighted ones

➢ L_1 reconstruction error

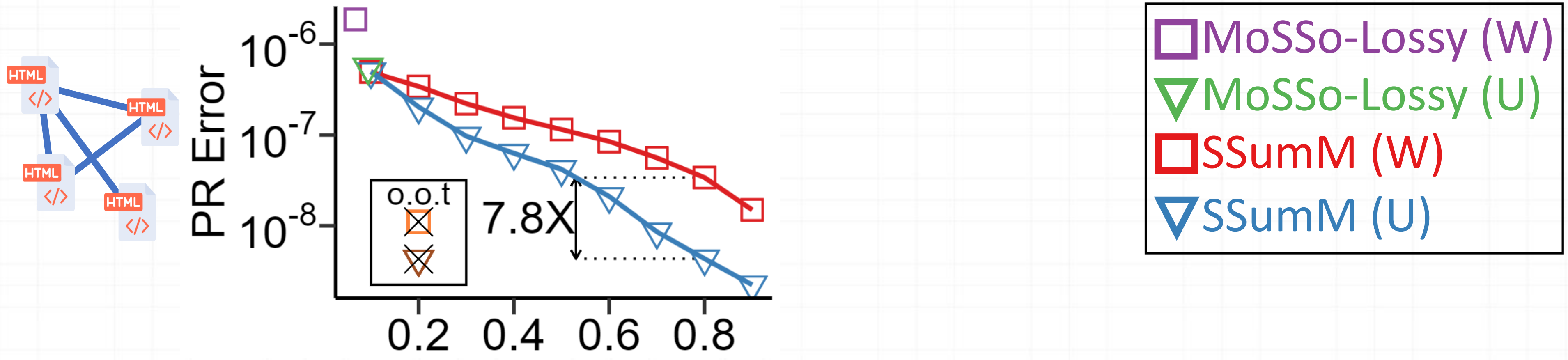


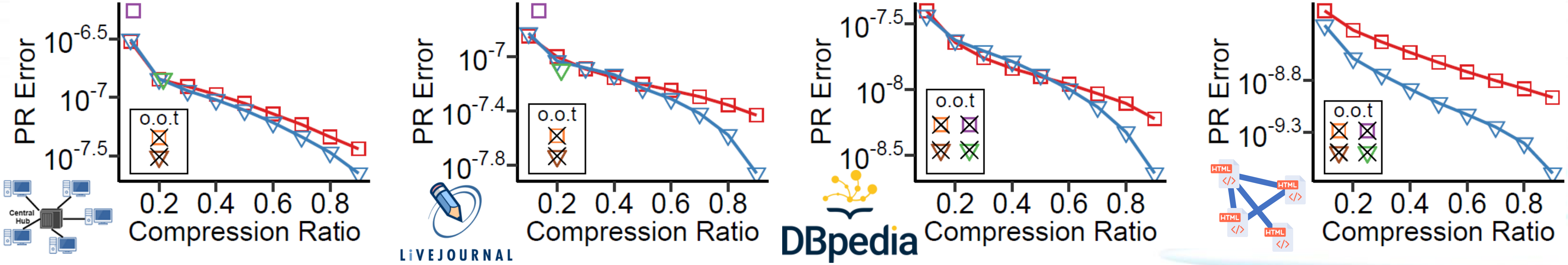
➢ L_2 reconstruction error



Experiments: maintain node importance

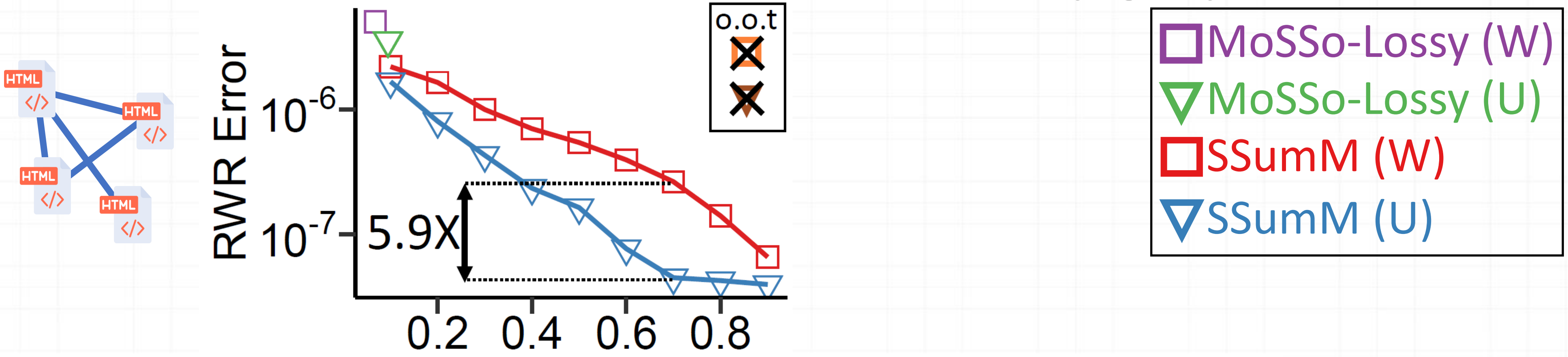
- Unweighted summary graphs ***maintained the node importance up to 7.8X more accurate*** than weighted ones

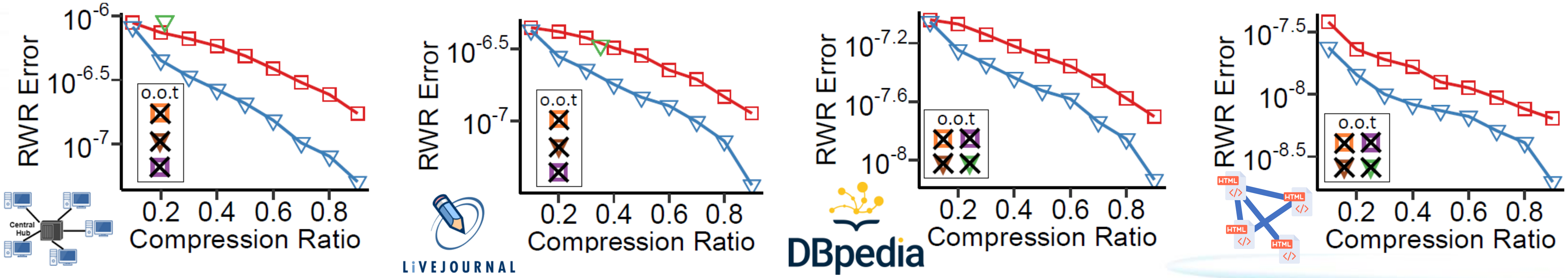




Experiments: preserve node proximity

- ***The answers from unweighted summary graphs*** are up to ***5.9X more accurate*** than from weighted summary graphs





Why can edge weights be harmful?

- Consider a graph G and its weighted summary (S, P, ω)
- Assume ω is not fixed but variable
- *Theorem 1.* When ***the $L1$ reconstruction error is minimized***, the weight of each superedge $\{A, B\} \in P$ is set so that the weights of all reconstructed subedges are either ***1 or 0, as if an unweighted summary graph is used***
- That is, edge weights do not contribute to accuracy while requiring additional space

Road map

- Introduction
- Notations & Problem formulation
- Considered algorithms
- Experiments & Theoretical analysis
- ***Conclusion <<***

Conclusion

- We ***conducted a systematic comparison*** between two graph summarization models with and without superedge weights
- We empirically ***revealed a surprising finding*** that removing superedge weights leads to significant improvements
- We ***developed a theoretical analysis*** to shed light on this counterintuitive observations (*Theorem 1*)

Github link: https://github.com/ShinhwanKang/PAKDD22-ComparativeStudy

References

[1] K. LeFevre and E. Terzi, “Grass: Graph structure summarization,” in SDM, 2010

[2] K. Lee, H. Jo, J. Ko, S. Lim, and K. Shin, “Ssumm: Sparse summarization of massive graphs,” in KDD, 2020

[3] Ko, Jihoon, Yunbum Kook, and Kijung Shin. “Incremental lossless graph summarization,” in KDD, 2020.

[4] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[5] H. Tong, C. Faloutsos, and J.-Y. Pan, “Random walk with restart: fast solutions and applications,” KAIS, vol. 14, no. 3, pp. 327–346, 2008.

PAKDD 2022 THE 26TH PACIFIC-ASIA CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING

KAIST AI
Kim Jaechul Graduate School

Are Weights Useful in Graph Summarization? – A Comparative Study

Shinhwan Kang   Kyuhan Lee   Kijung Shin

KAIST AI