KAIST

清華大學 Tsinghua University

UNIVERSITA DEGLI STUDI DI TORINO

A Tutorial on Hypergraph Neural Networks: An In-Depth and Step-by-Step Guide

Part 2. Input Features and Structures


Sunwoo Kim*


Soo Yong Lee*


Yue Gao


Alessia Antelmi


Mirko Polato


Kijung Shin

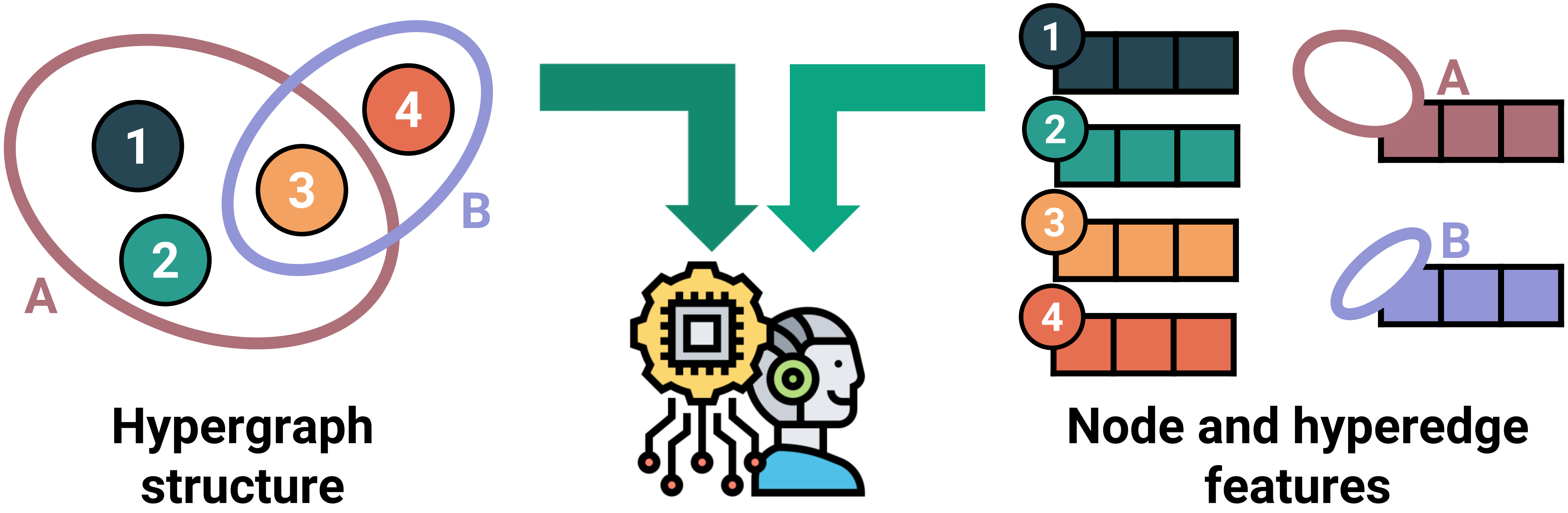
Part 2. Inputs

The slides are available at https://sites.google.com/view/hnn-tutorial

SCAN ME

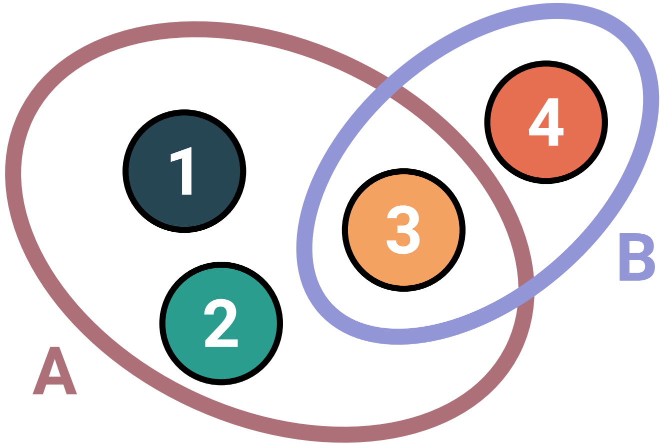
Hypergraph Neural Network Inputs

- The input quality can be critical for effective application of HNNs.
  - The inputs typically include **hypergraph structure** and (node and/or hyperedge) **feature vectors**.

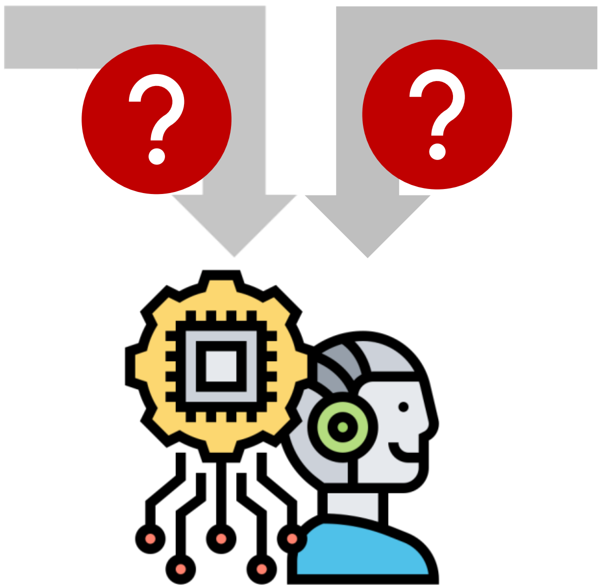


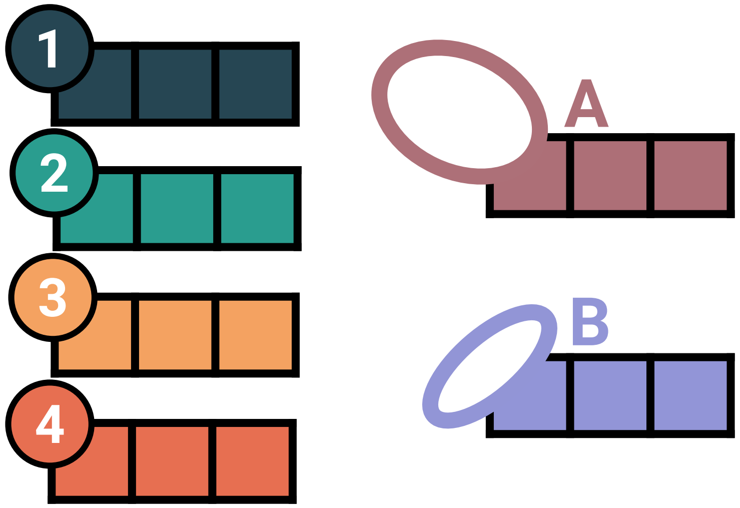
Hypergraph Neural Network Inputs (cont.)

- **Key questions** about the HNN inputs include:
  - Q1) How are hypergraph structures expressed?
  - Q2) What input features are typically used?



Hypergraph structure

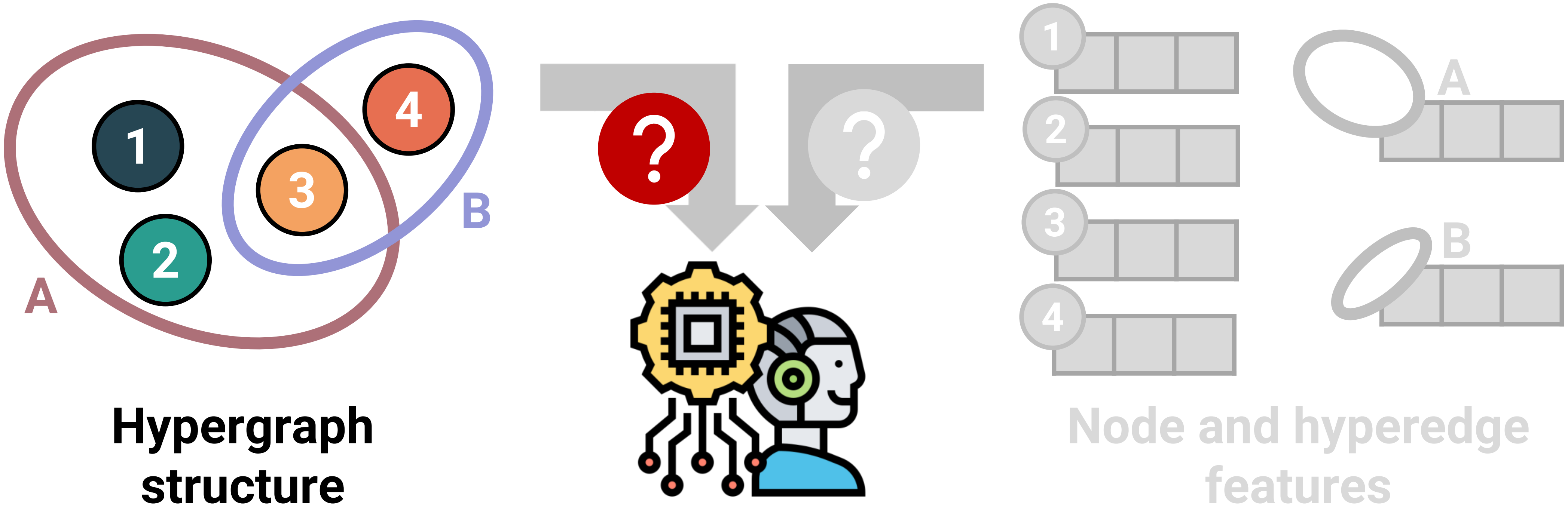




Node and hyperedge features

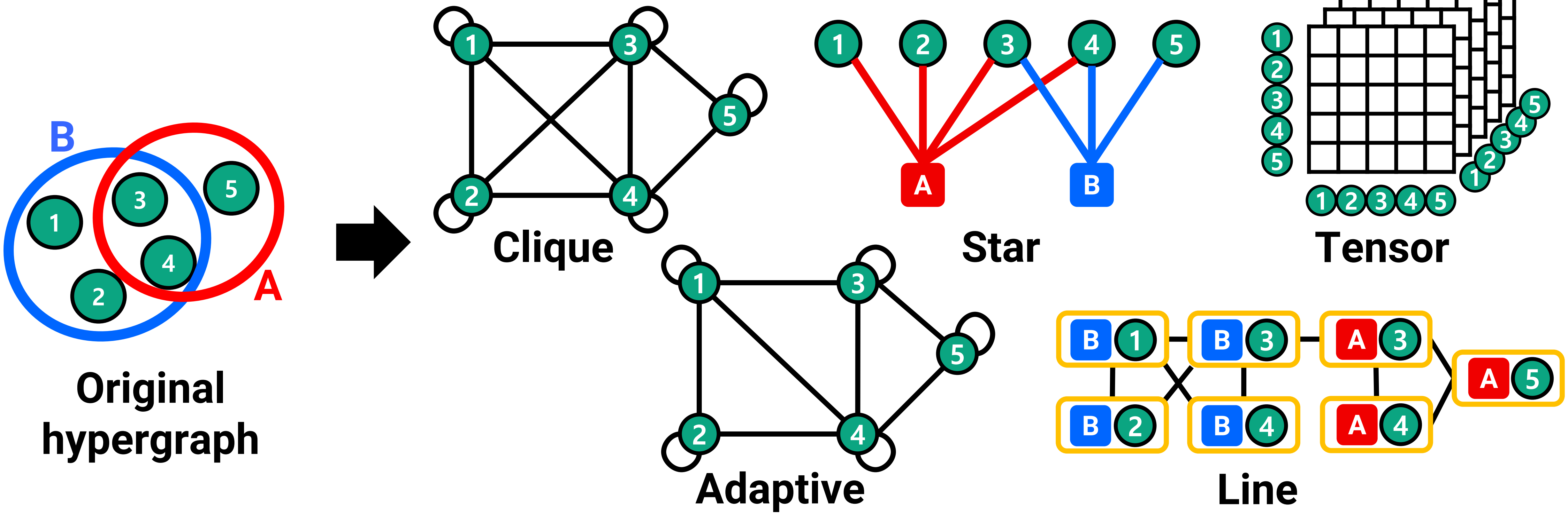
Hypergraph Neural Network Inputs (cont.)

- **Key questions** about the HNN inputs include:
  - **Q1**) How are hypergraph structures expressed?
  - **Q2**) What input features vector are typically used?



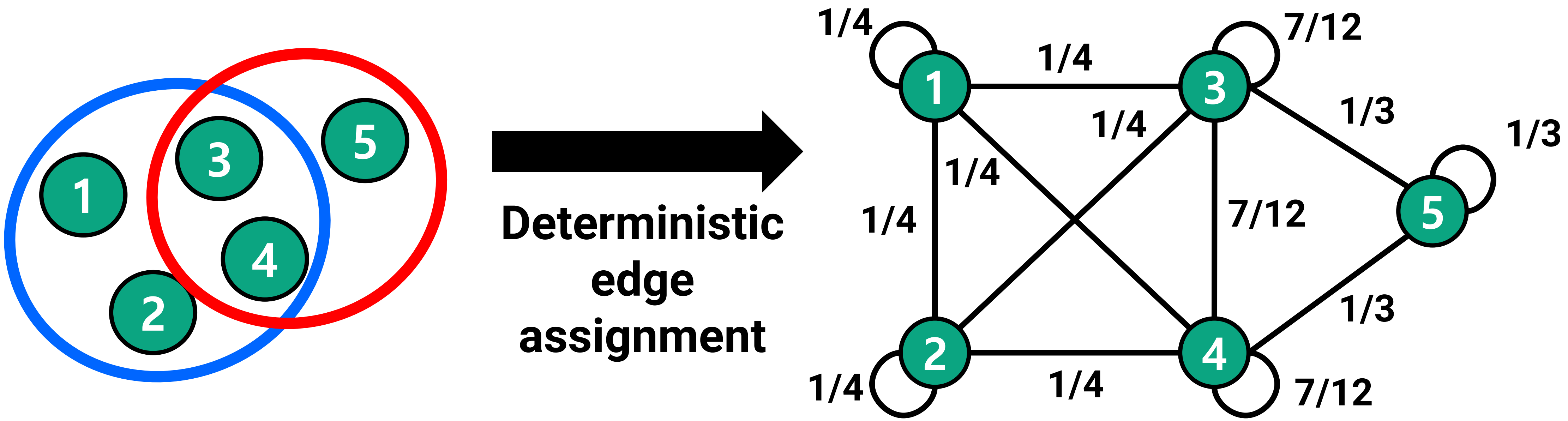
Q1) Expressing Hypergraph Structures

- We introduce five different ways to express hypergraph structures:
  - Clique-, adaptive-, star-, line-, and tensor-expansions.



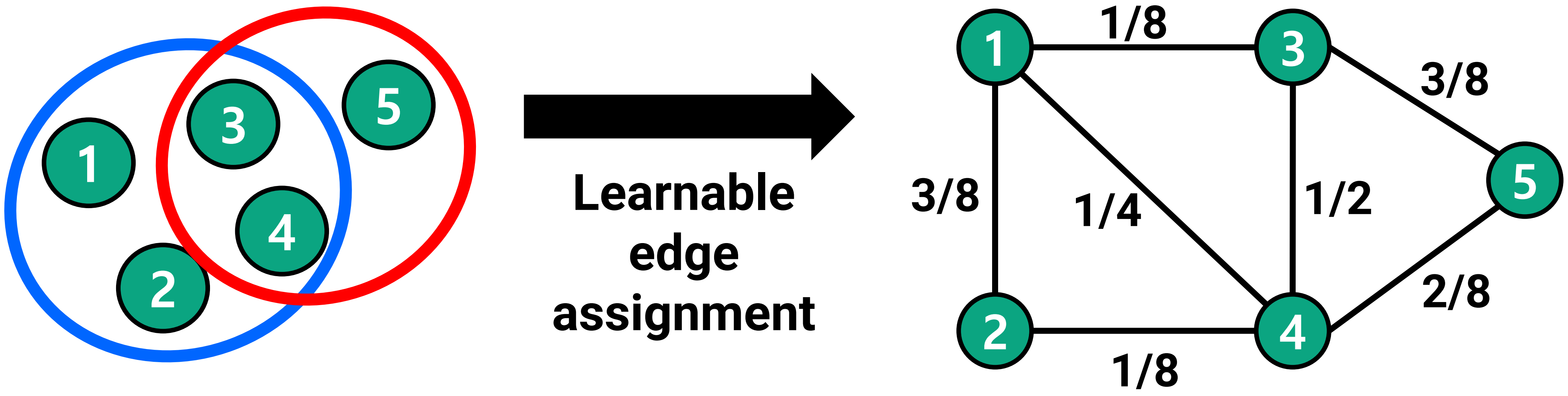
Q1) Expressing Hypergraph Structures: Clique

- **Clique expansion** of a hypergraph is a homogeneous, pair-wise graph.
  - It transforms a hyperedge into a clique of pairwise edges.
  - Feng et al. (2019) further weigh edges with learnable parameters.



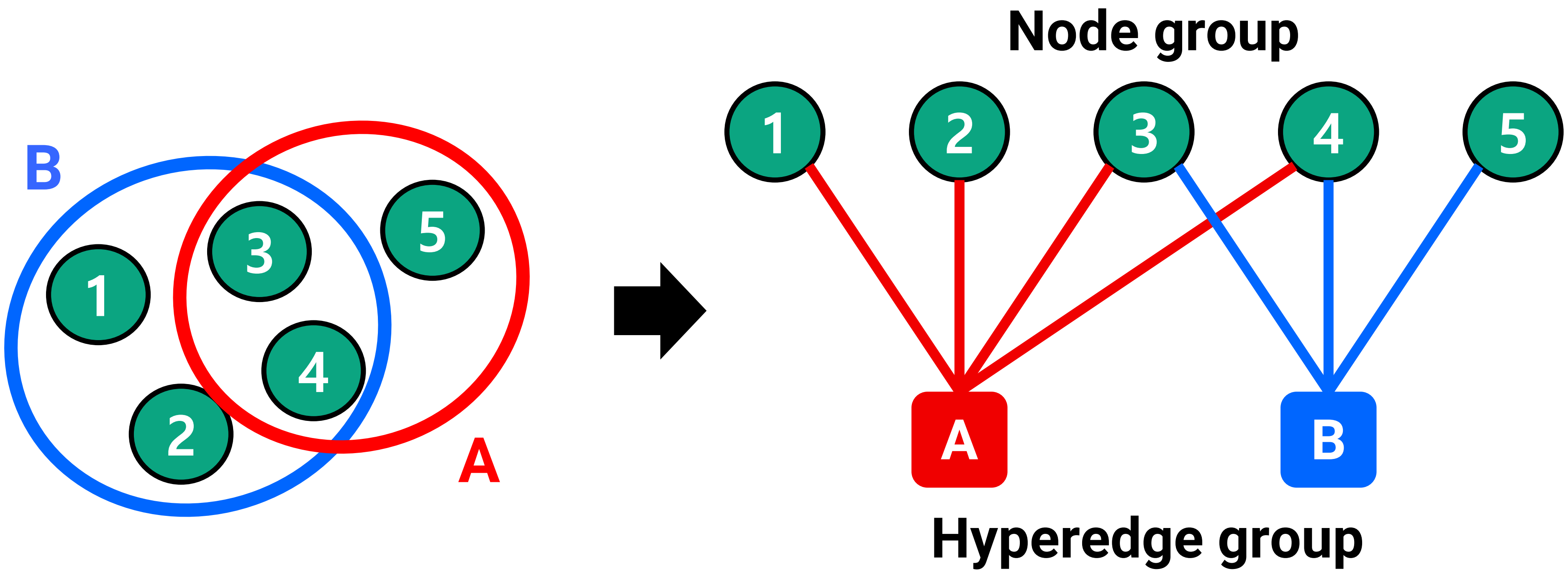
Q1) Expressing Hypergraph Structures: Adaptive

- **Adaptive expansion** of a hypergraph is a homogeneous, pair-wise graph.
  - It transforms a hyperedge into pairwise edge(s) via learnable rules.
  - Qian et al. (2023) assign and weigh edges based on node features.



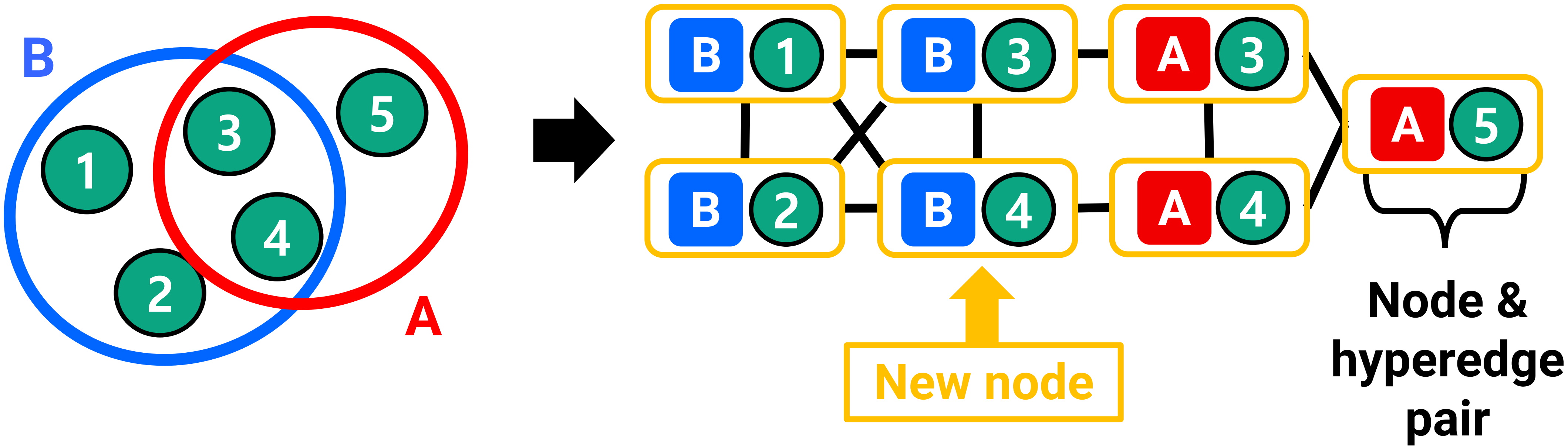
Q1) Expressing Hypergraph Structures: Star

- **Star expansion** of a hypergraph is a bipartite, pair-wise graph.
  - It transforms a hyperedge into a new node.
  - It joins a hyperedge (i.e., new node) and its nodes via pair-wise edges.



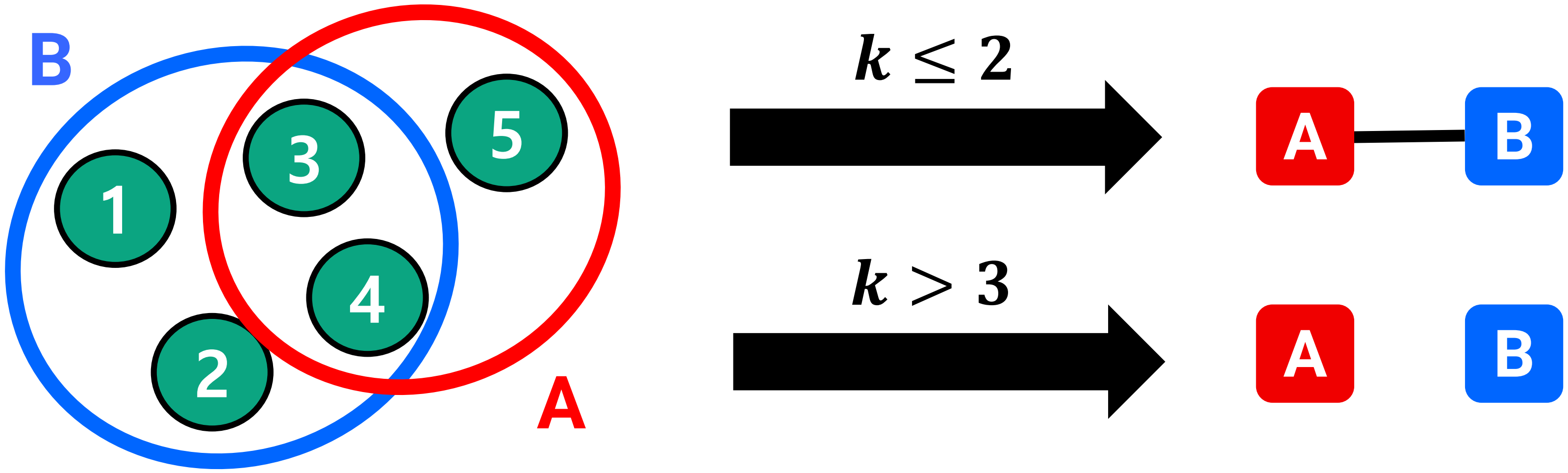
Q1) Expressing Hypergraph Structures: Line

- **Line expansion** of a hypergraph is a homogeneous, pair-wise graph.
    - It transforms a *pair* of a hyperedge and its node into a new node.
    - Pair-wise edges join new nodes sharing a hyperedge or node.



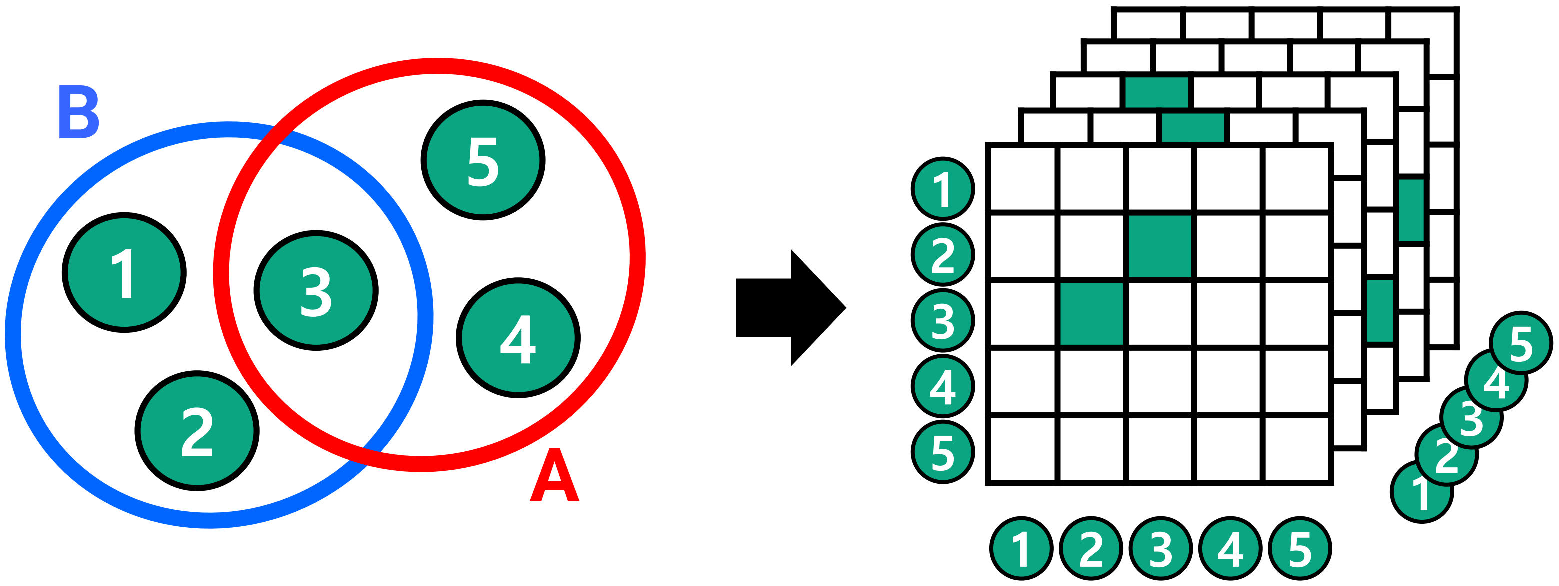
Q1) Expressing Hypergraph Structures: Line

- Note that line expansion is different from a (k-)line graph
- **k-Line graph** from a hypergraph is a homogeneous, pair-wise graph.
  - It transforms a hyperedge into a new node.
  - It joins hyperedge (i.e., new node) pairs with at least k common nodes



Q1) Expressing Hypergraph Structures: Tensor

- A hypergraph can be expressed with a **binary tensor $\mathcal{A}$**.
  - Tensor order expresses the maximum hyperedge size, and each tensor dimension corresponds to node index.



B

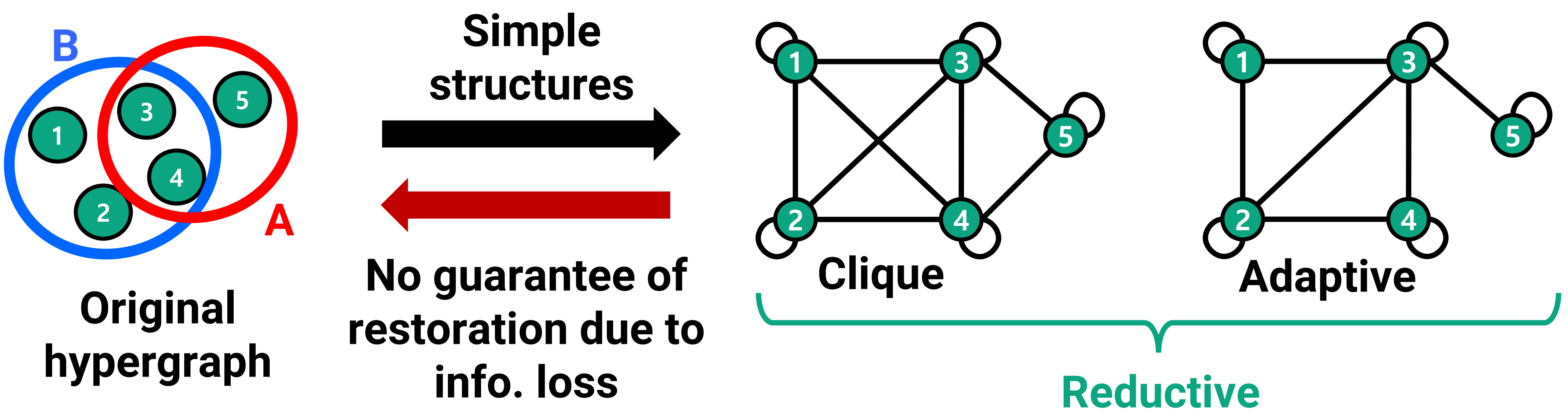
$= \mathcal{A}_{1,2,3} = \mathcal{A}_{1,3,2} = \mathcal{A}_{2,1,3}$
$= \mathcal{A}_{2,3,1} = \mathcal{A}_{3,1,2} = \mathcal{A}_{3,2,1}$

A

$= \mathcal{A}_{3,4,5} = \mathcal{A}_{3,5,4} = \mathcal{A}_{4,3,5}$
$= \mathcal{A}_{4,5,3} = \mathcal{A}_{5,3,4} = \mathcal{A}_{5,4,3}$

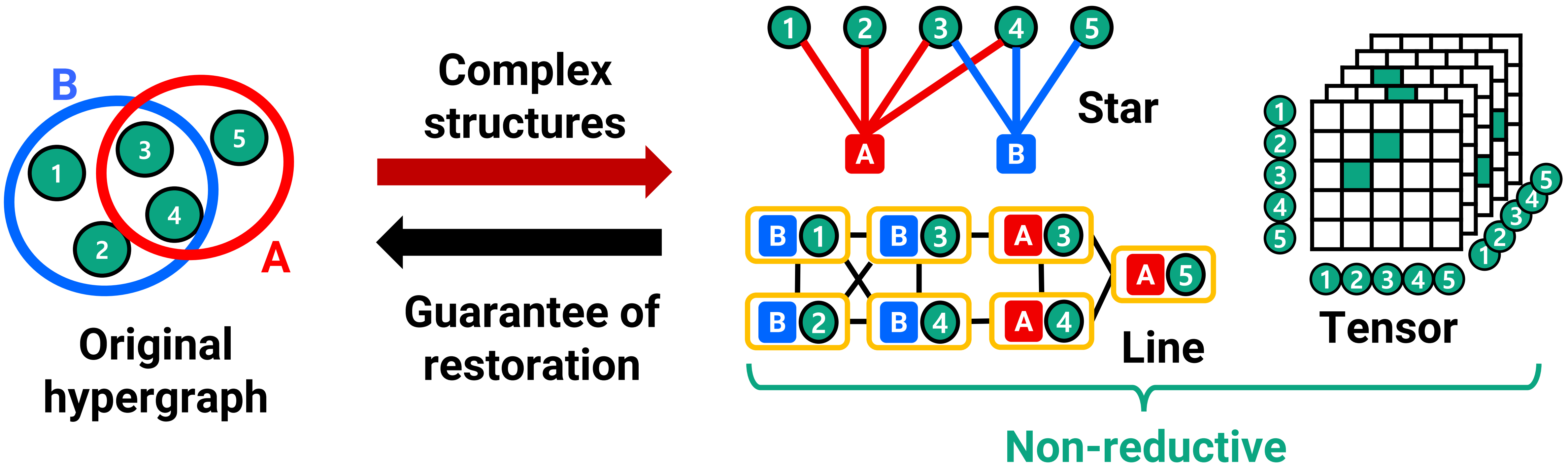
Q1) Expressing Hypergraph Structures (cont.)

- The discussed approaches are either **reductive** or **non-reductive**.
  - Reductive approaches yield simple and straight-forward graph structures.
  - However, it may incur information loss after transformation.



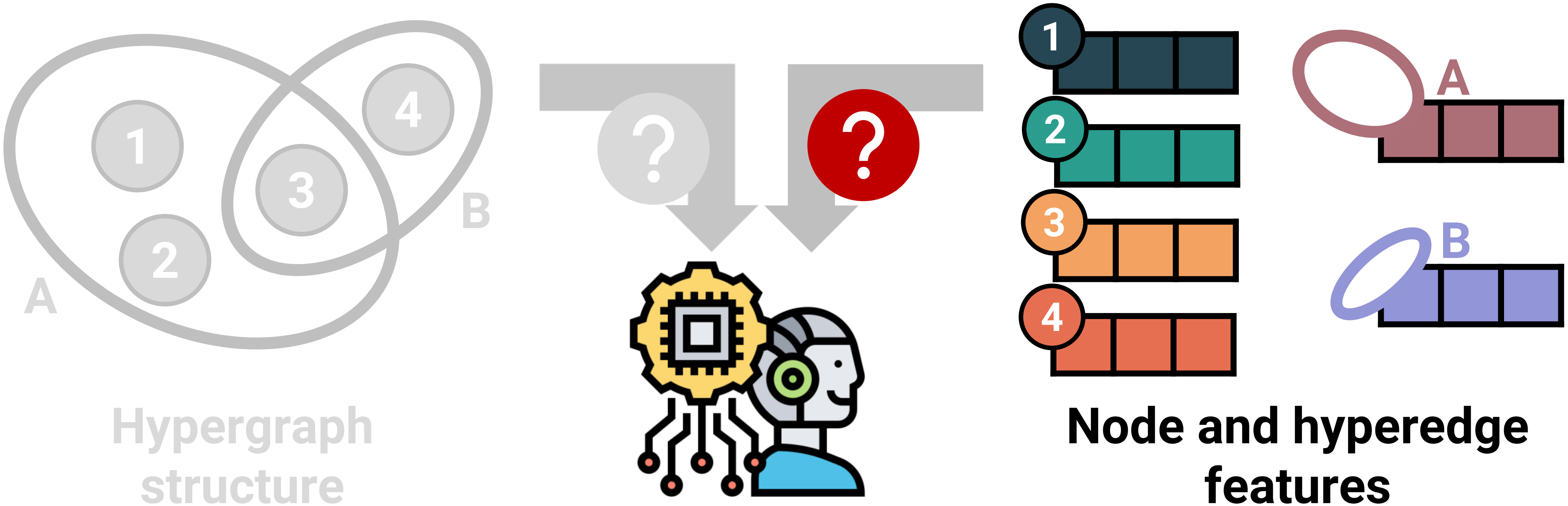
Q1) Expressing Hypergraph Structures (cont.)

- The discussed approaches are either **reductive** or **non-reductive**.
  - Non-reductive transformation incur no information loss.
  - However, they are often more complex and, thus, difficult to handle.



Hypergraph Neural Network Inputs (cont.)

- **Key questions** about the HNN inputs include:
  - Q1) How are hypergraph structures expressed?
  - **Q2**) What input features vector are typically used?

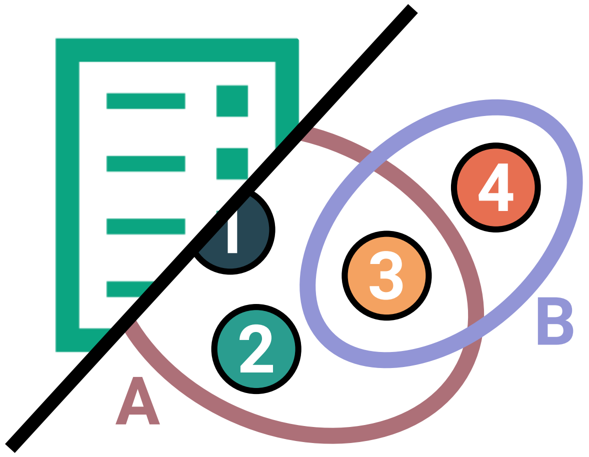


Hypergraph structure

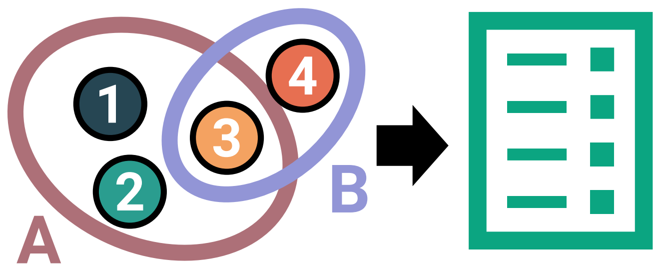
Node and hyperedge features

Q2) Node and Hyperedge Features

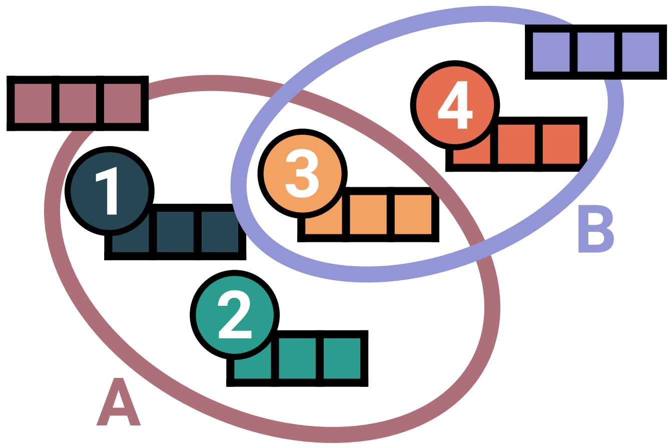
- Typically, **input feature types** for HNNs include:
  - external, structural, and identity features.



External features



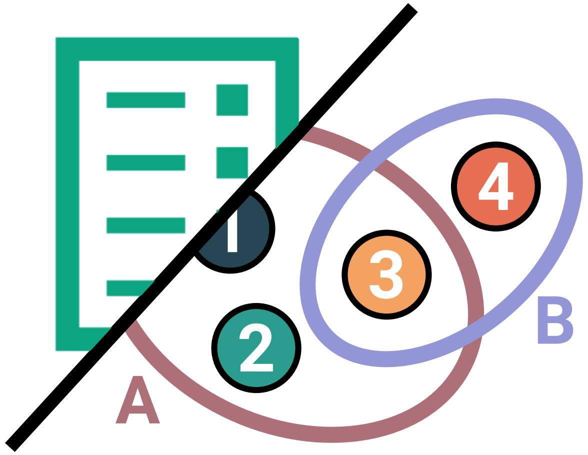
Structural features

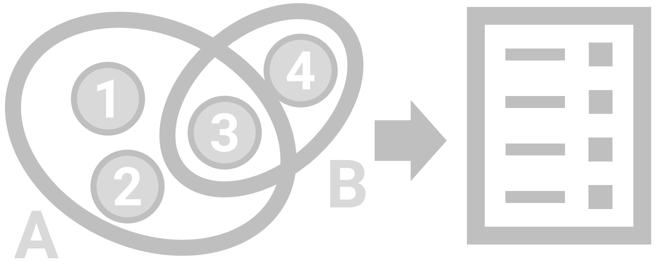


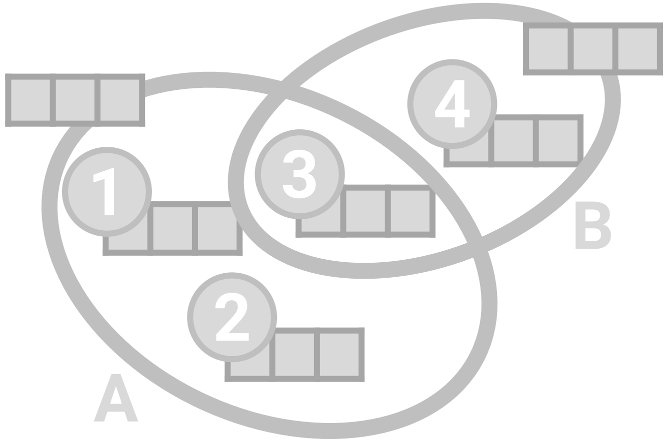
Identity features

Q2) Node and Hyperedge Features: External

- **External features** are information about a hypergraph that is not directly obtained from its structure,
  - encouraging HNNs to capture info. beyond those reflected in structure.

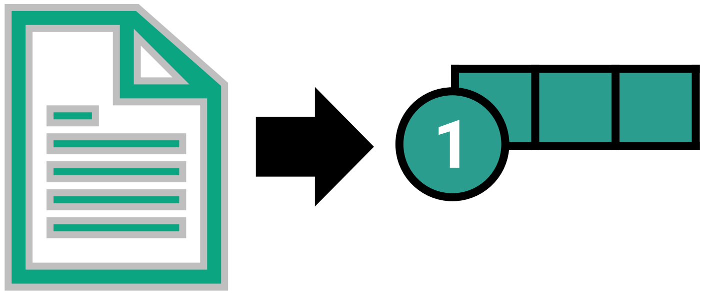




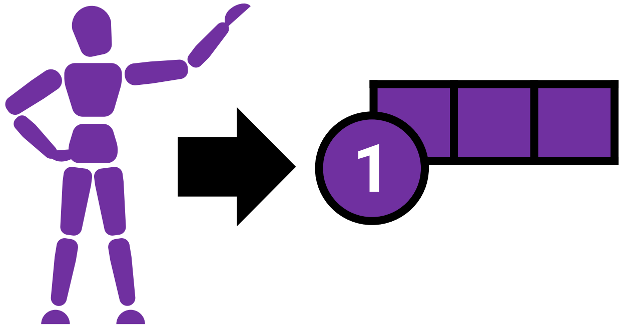


Q2) Node and Hyperedge Features: External (cont.)

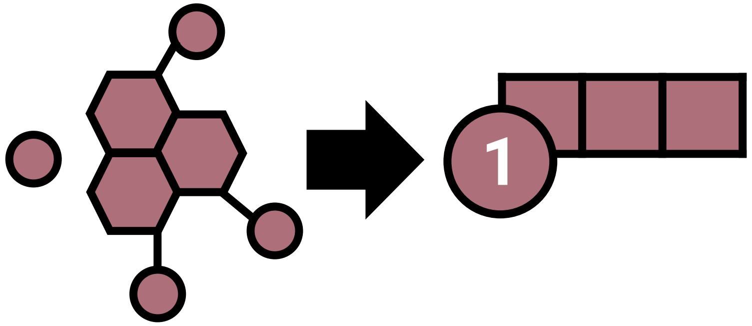
- **External node features** in popular benchmark datasets include:
  - 1) bag-of-words vectors of article nodes, 2) visual object embeddings of image nodes, 3) counts of atoms within molecule nodes, etc.



Bag-of-words



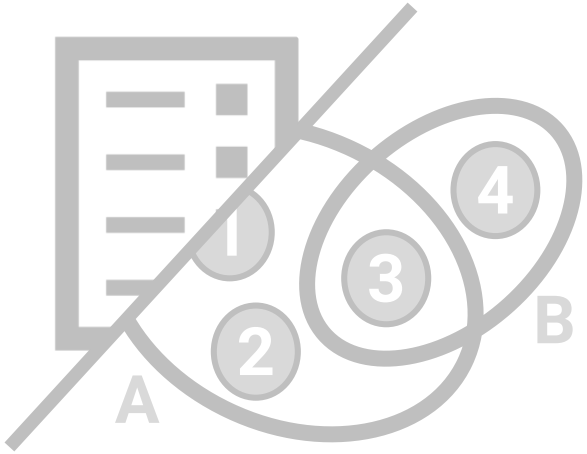
Visual object embedding



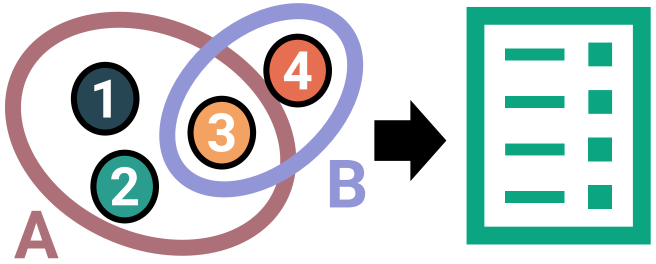
Count of atoms

Q2) Node and Hyperedge Features: Structural

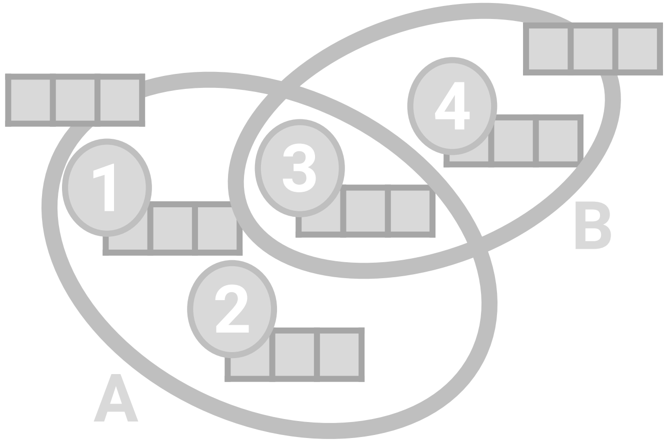
- **Structural features** are derived from the input hypergraph structure,
  - typically capturing structural proximity or similarity among nodes.
  - The structural features are either ***local*** or ***global***.



External features



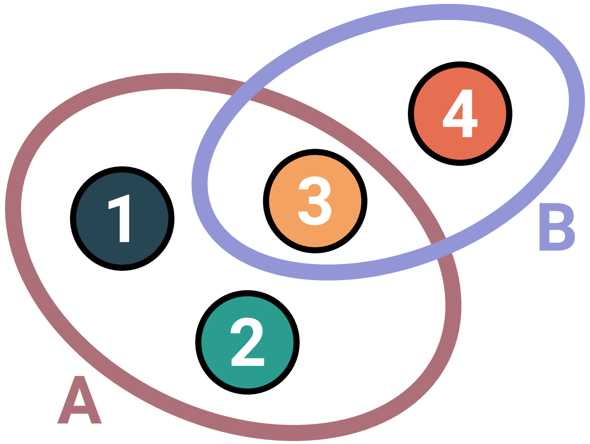
Structural features



Identity features

Q2) Node and Hyperedge Features: Structural (cont.)

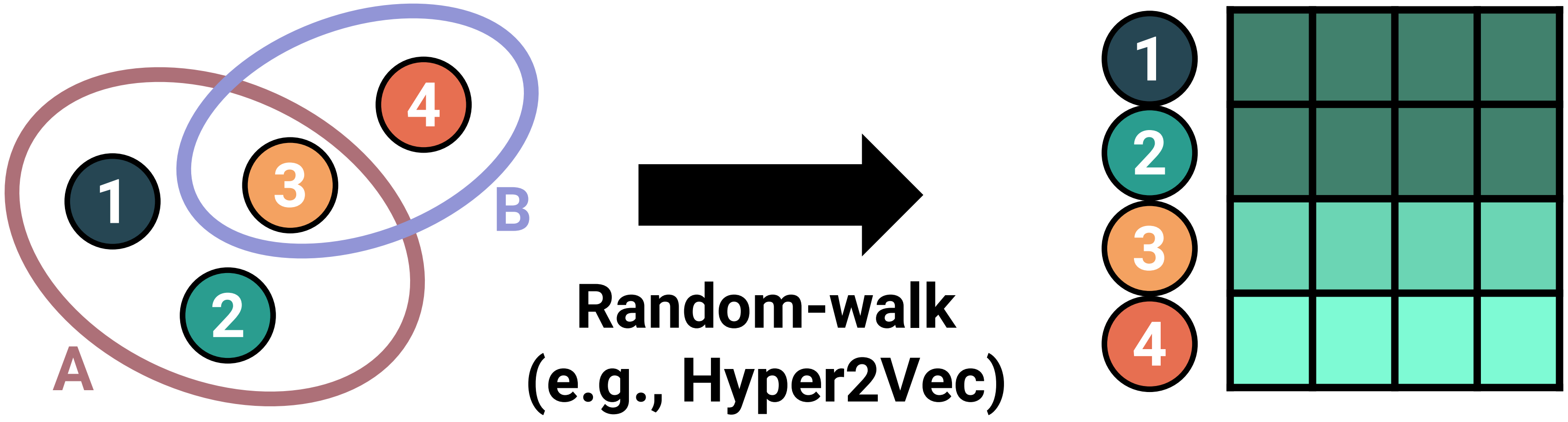
- For **local structural features**, Liu et al. (2024) leverage the incidence matrix as part of the input features,
  - capturing local neighborhood information for each node.



	A	B
1	1	0
2	1	0
3	1	1
4	0	1

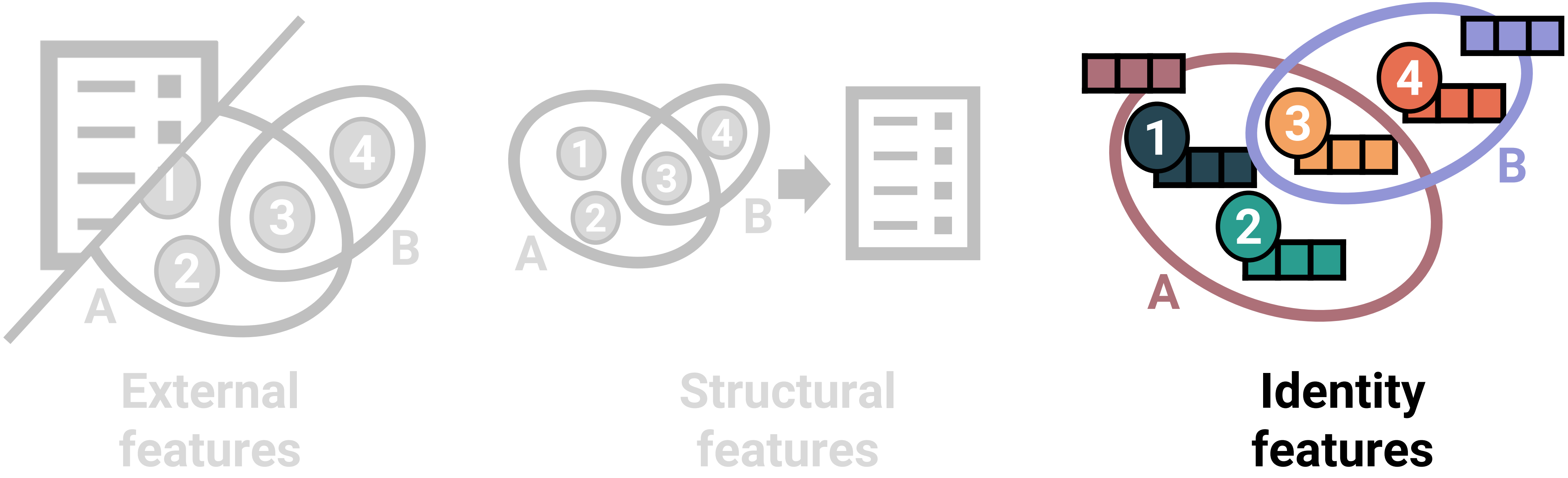
Q2) Node and Hyperedge Features: Structural (cont.)

- For **global structural features**, Zhang et al. (2020) leverage random-walk-based node features,
  - capturing proximity of each node to all other nodes.



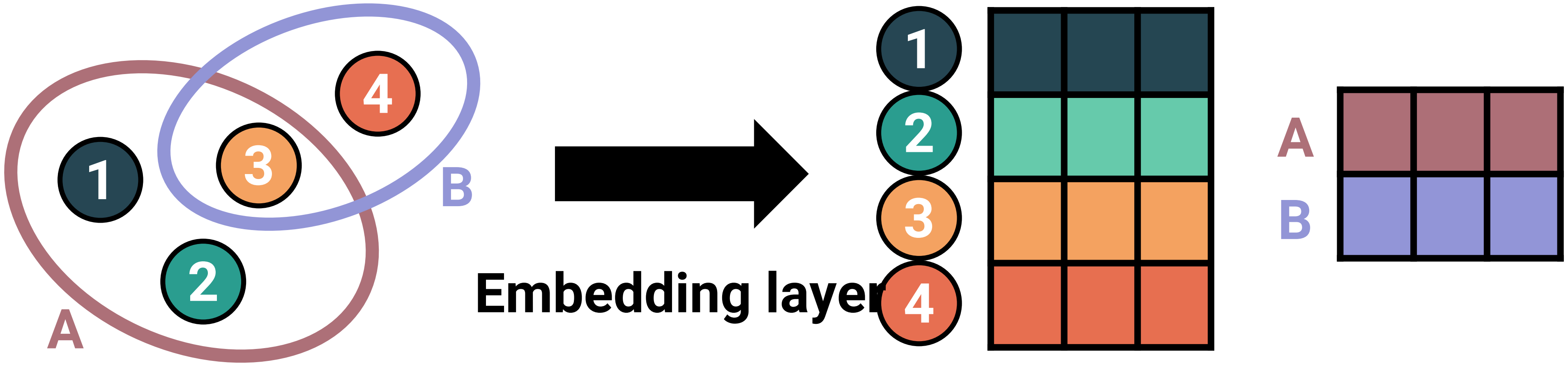
Q2) Node and Hyperedge Features: Identity

- **Identity features** are indicator vectors uniquely assigned to each node and hyperedge,
  - encouraging HNNs to learn distinct embeddings for each node.



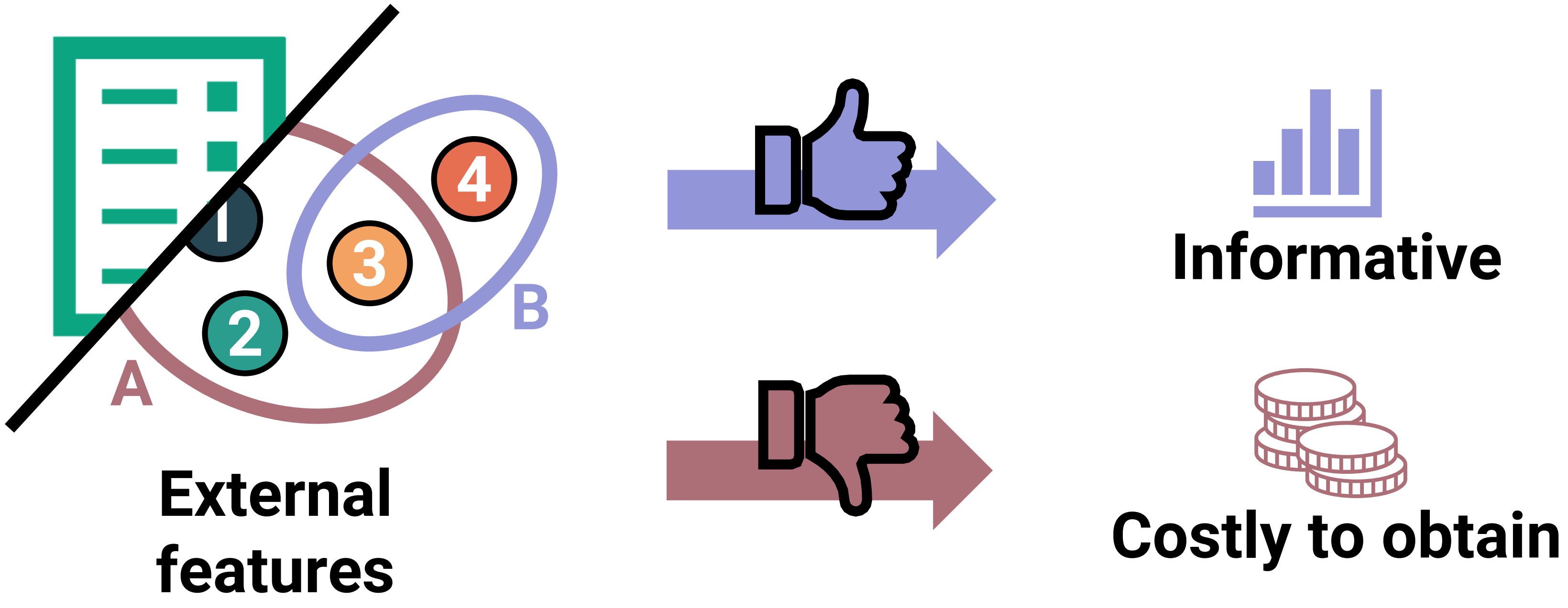
Q2) Node and Hyperedge Features: Identity (cont.)

- Ji et al. (2020) generated **identity features** via an embedding layer,
  - such that each node and hyperedge has a learnable random feature vector.



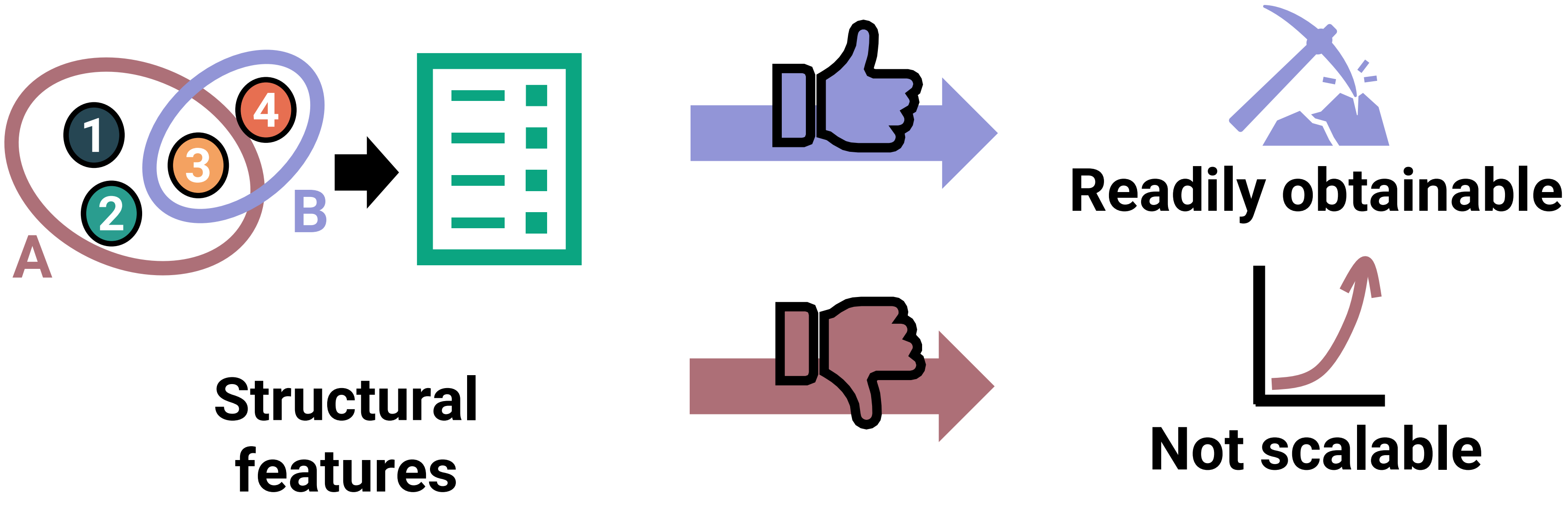
Q2) Node and Hyperedge Features (cont.)

- External, structural, and identity features have **pros and cons**.
  - **External features** can be highly informative, encouraging HNNs to be more effective. However, obtaining external features can be costly.



Q2) Node and Hyperedge Features (cont.)

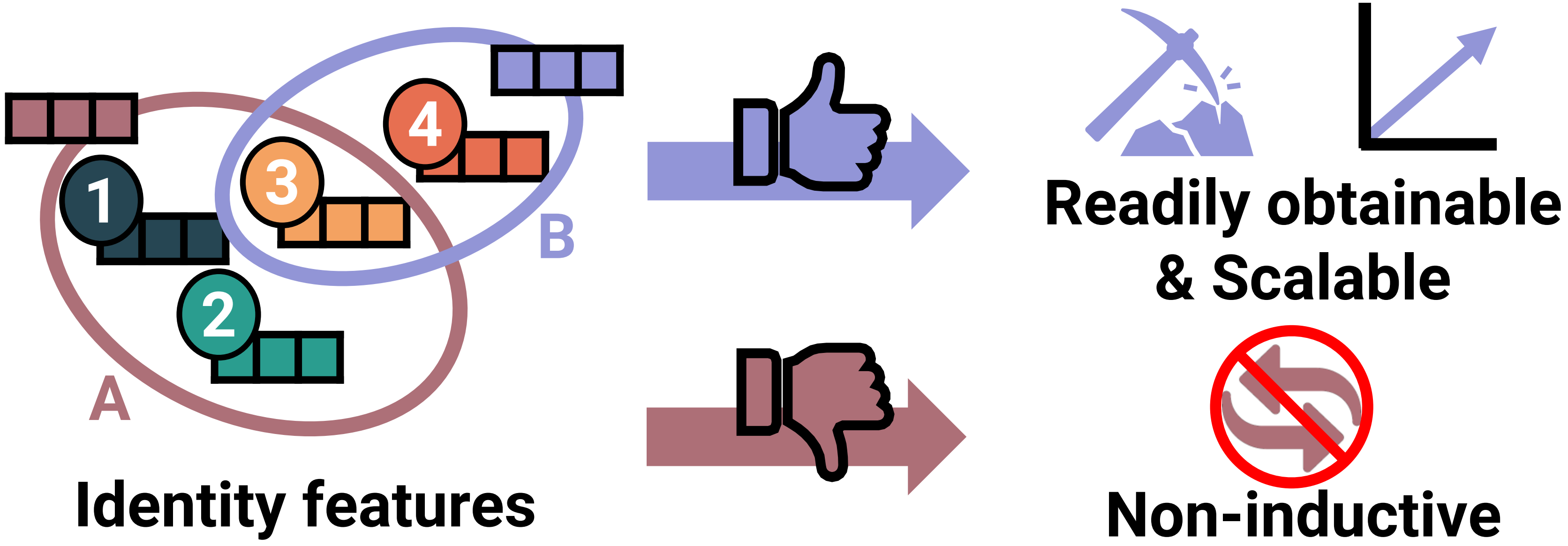
- External, structural, and identity features have **pros and cons**.
  - **Structural features** are readily obtainable from hypergraph structure. However, they may not be scalable for large hypergraphs.



Q2) Node and Hyperedge Features (cont.)

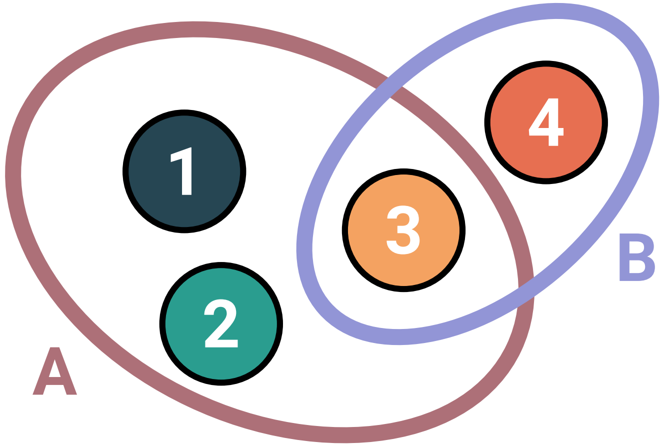
- External, structural, and identity features have **pros and cons**.
  - **Identity features** are readily obtainable and scalable.

    However, they may not be applicable to inductive settings where test hypergraphs are different from training hypergraphs.

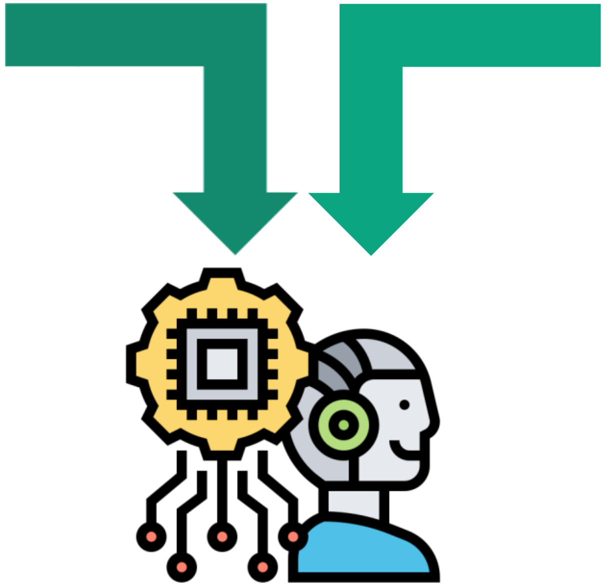


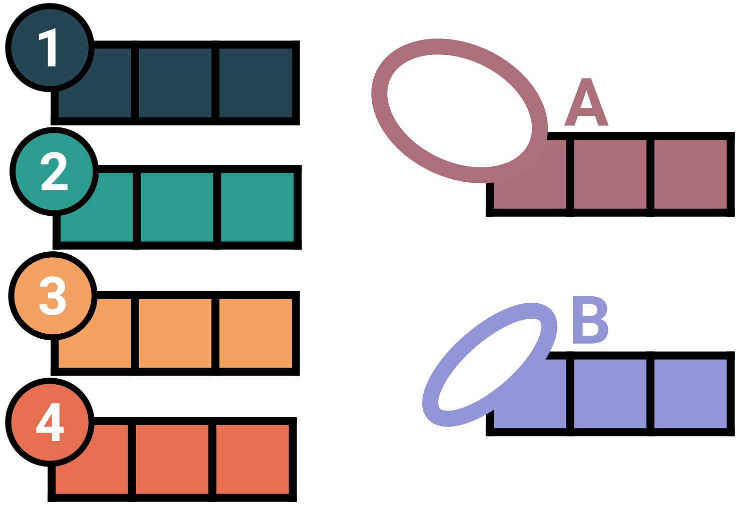
Part 2 Summary

- The input quality can be critical for effective application of HNNs.
  - The inputs typically include **hypergraph structure** and (node and/or hyperedge) **feature vectors**.



Hypergraph structure

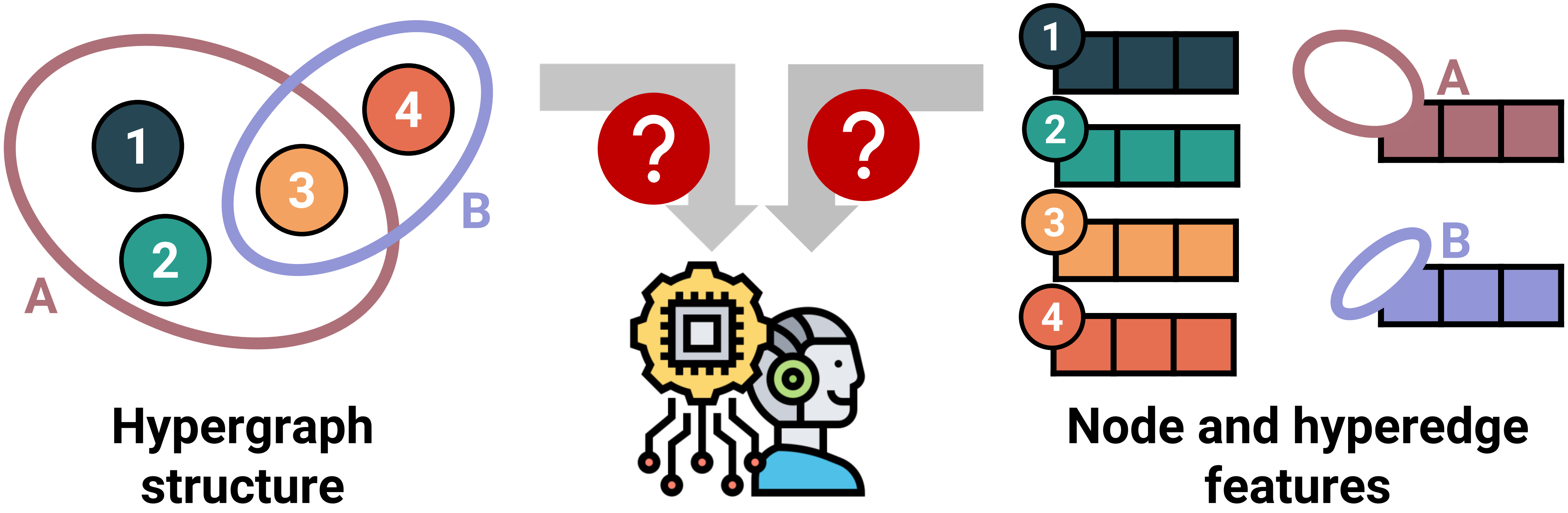




Node and hyperedge features

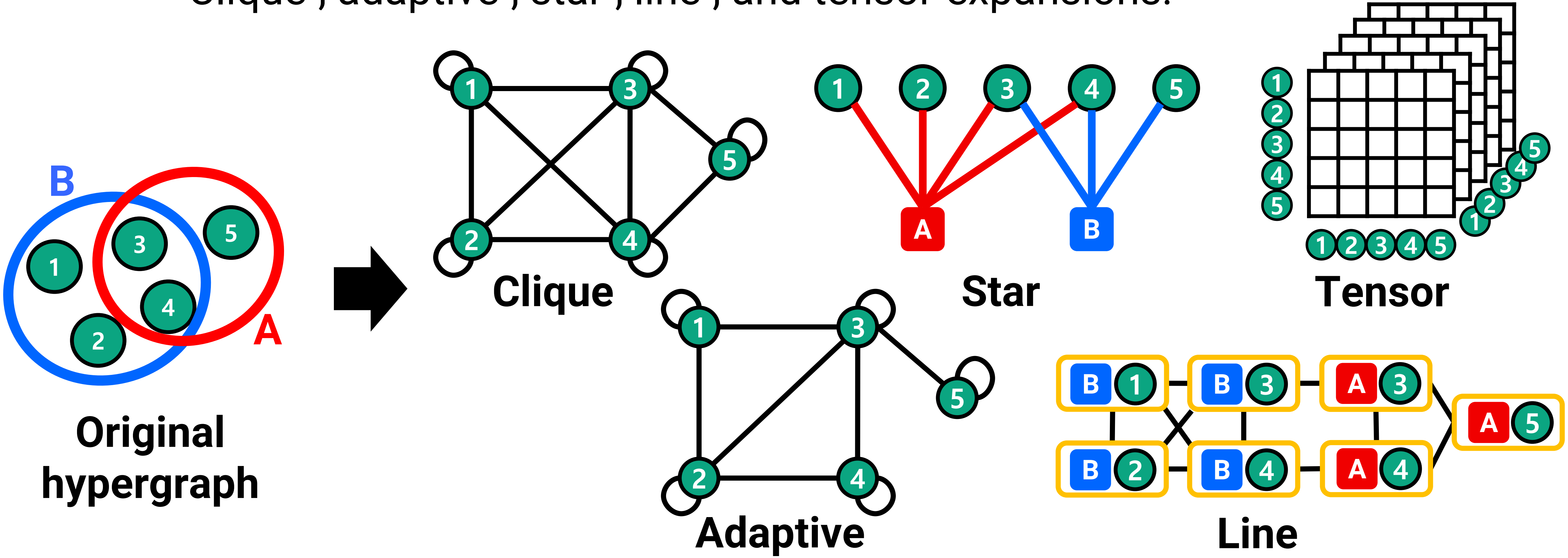
Part 2 Summary (cont.)

- **Key questions** about the HNN inputs include:
  - **Q1**) How are hypergraph structures expressed?
  - **Q2**) What input features vector are typically used?



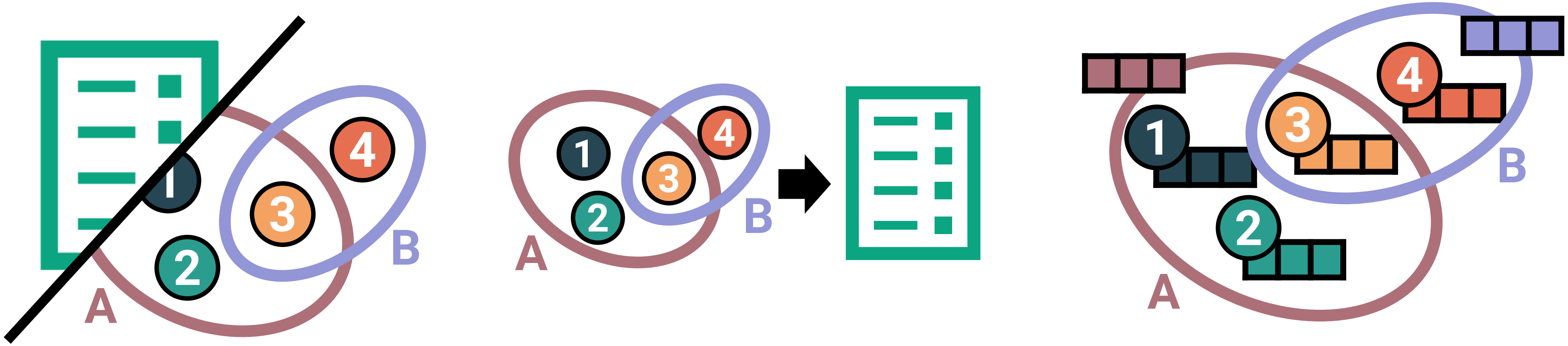
Part 2 Summary (cont.)

- We introduce five different ways to express hypergraph structures:
  - Clique-, adaptive-, star-, line-, and tensor-expansions.



Part 2 Summary (cont.)

- Typically, input features for HNNs include:
  - external, structural, and identity features.



External features

Structural features

Identity features