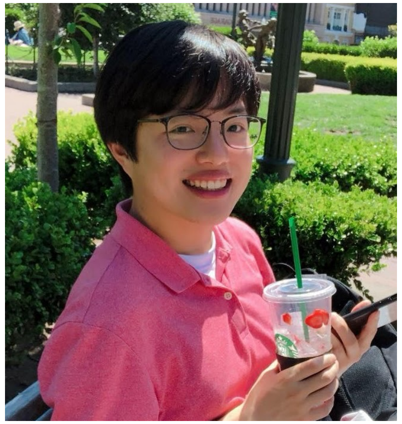
KAIST

The 37TH AAAI Conference (AAAI 2023)

I'm Me, We're Us, and I'm Us: Tri-directional Contrastive Learning on Hypergraphs

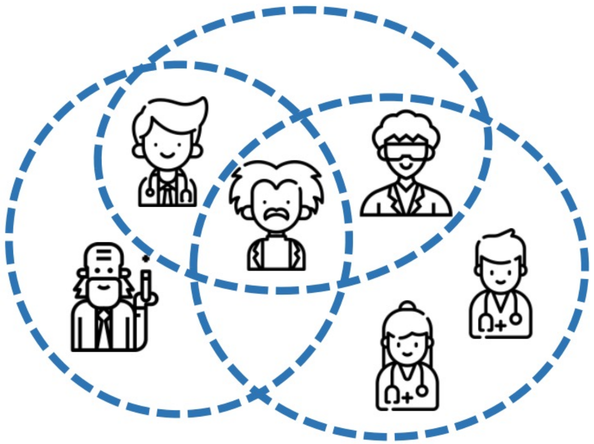
Dongjin Lee



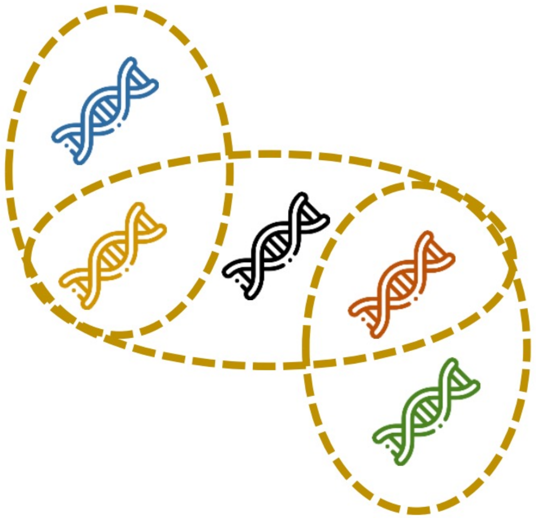
Kijung Shin

Hypergraphs are Everywhere

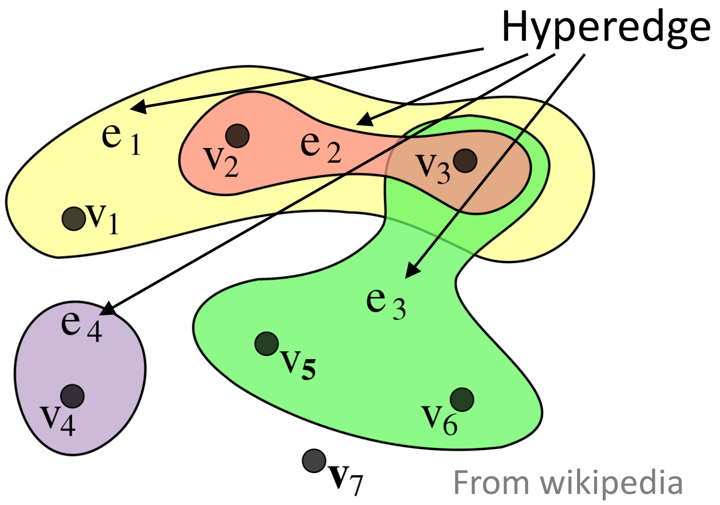
- Many real-world interactions are group-wise.
  - Ex) Collaborations of researchers, interactions of proteins, co-purchases of items.
- A hypergraph can represent such group-wise interactions naturally.
  - A hypergraph is a set of hyperedges that allow containing any number of nodes.



Collaborations of researchers



Joint interactions of proteins



Hypergraph

Machine Learning on Hypergraphs

- Hypergraph-based ML approaches show its effectiveness on various tasks.
  - Methods: HGNN [Feng et al. 2019], HNHN [Dong, Sawin, and Bengio 2020], AllSet [Chien et al. 2022], etc.
  - Tasks: classification [Feng et al. 2019], clustering [Benson, Gleich, and Leskovec 2016], outlier detection [Lee, Choe, and Shin 2022], etc.

- Most hypergraph neural networks (HNNs) are trained in a (semi-)supervised way.
  - High-quality data labeling is time, resource, and labor-intensive.
  - HNNs trained in a supervised way may overfit and fail to generalize.

- Self-supervised learning, which does not require labels, has become popular, and especially **contrastive learning (CL)** has achieved great success.

Machine Learning on Hypergraphs

- Hypergraph-based ML approaches show its effectiveness on various tasks.
  - Methods: HGNN [Feng et al. 2019], HNHN [Dong, Sawin, and Bengio 2020], AllSet [Chien et al. 2022], etc.
  - Tasks: classification [Feng et al. 2019], clustering [Benson, Gleich, and Leskovec 2016], outlier detection [Lee, Choe, and Shin 2022], etc.

- Most hypergraph neural networks (HNNs) are trained in a (semi-)supervised way.
  - High-quality data labeling is <u>time, resource, and labor-intensive</u>.
  - HNNs trained in a supervised way may <u>overfit and fail to generalize</u>.

- Self-supervised learning, which does not require labels, has become popular, and especially **<u>contrastive learning (CL)</u>** has achieved great success.

Machine Learning on Hypergraphs

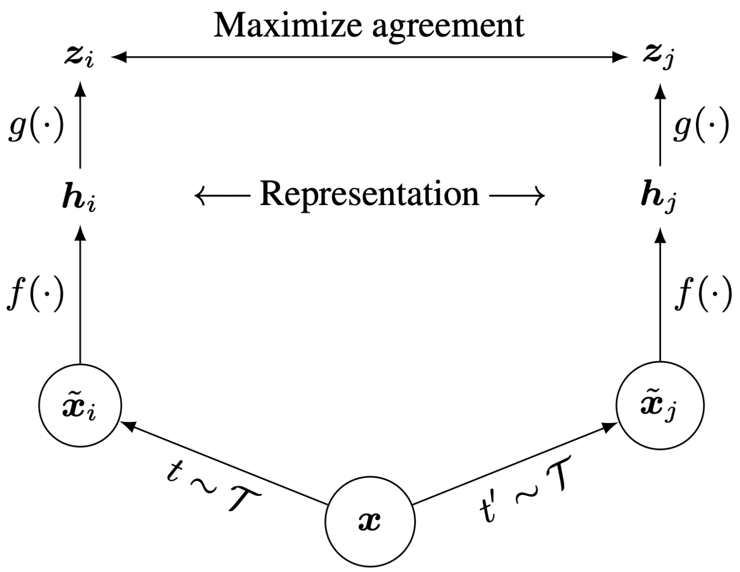
- Hypergraph-based ML approaches show its effectiveness on various tasks.
  - Methods: HGNN [Feng et al. 2019], HNHN [Dong, Sawin, and Bengio 2020], AllSet [Chien et al. 2022], etc.
  - Tasks: classification [Feng et al. 2019], clustering [Benson, Gleich, and Leskovec 2016], outlier detection [Lee, Choe, and Shin 2022], etc.

- Most hypergraph neural networks (HNNs) are trained in a (semi-)supervised way.
  - High-quality data labeling is <u>time, resource, and labor-intensive</u>.
  - HNNs trained in a supervised way may <u>overfit and fail to generalize</u>.

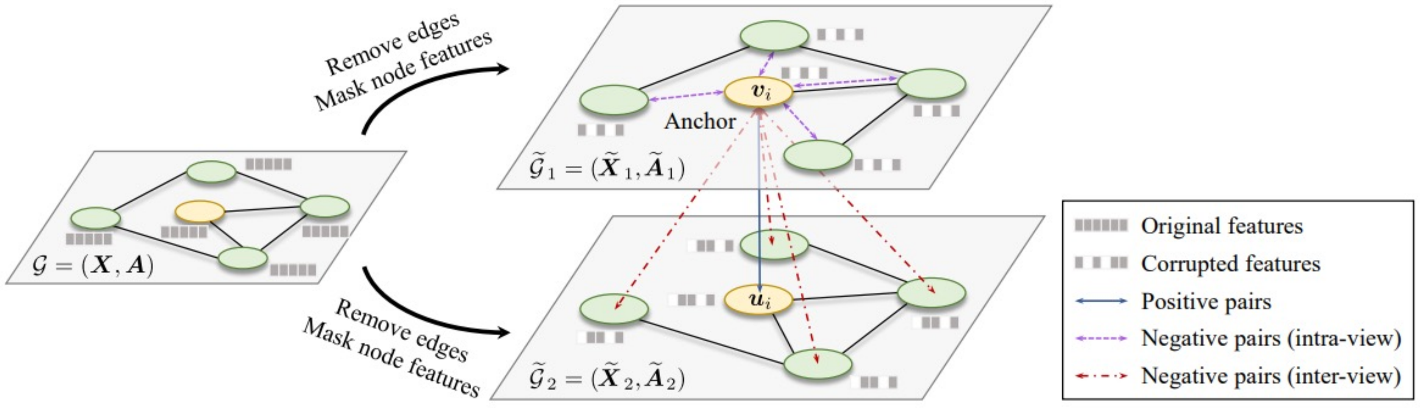
- Self-supervised learning, which does not require labels, has become popular, and especially **<u>contrastive learning (CL)</u>** has achieved great success.

Contrastive Learning Paradigm

- CL aims to **maximize the agreement** between differently augmented views of the same input via a contrastive loss [Chen et al. 2020].
- CL has demonstrated its effectiveness in computer vision, NLP, and graph domains.
- CL on hypergraphs remains largely underexplored.



SimCLR [Chen et al. 2020]



GRACE [Zhu et al. 2020]

In This Work

- In this work, we propose ***TriCL***, a novel hypergraph contrastive learning method, to answer the following questions.

Questions?

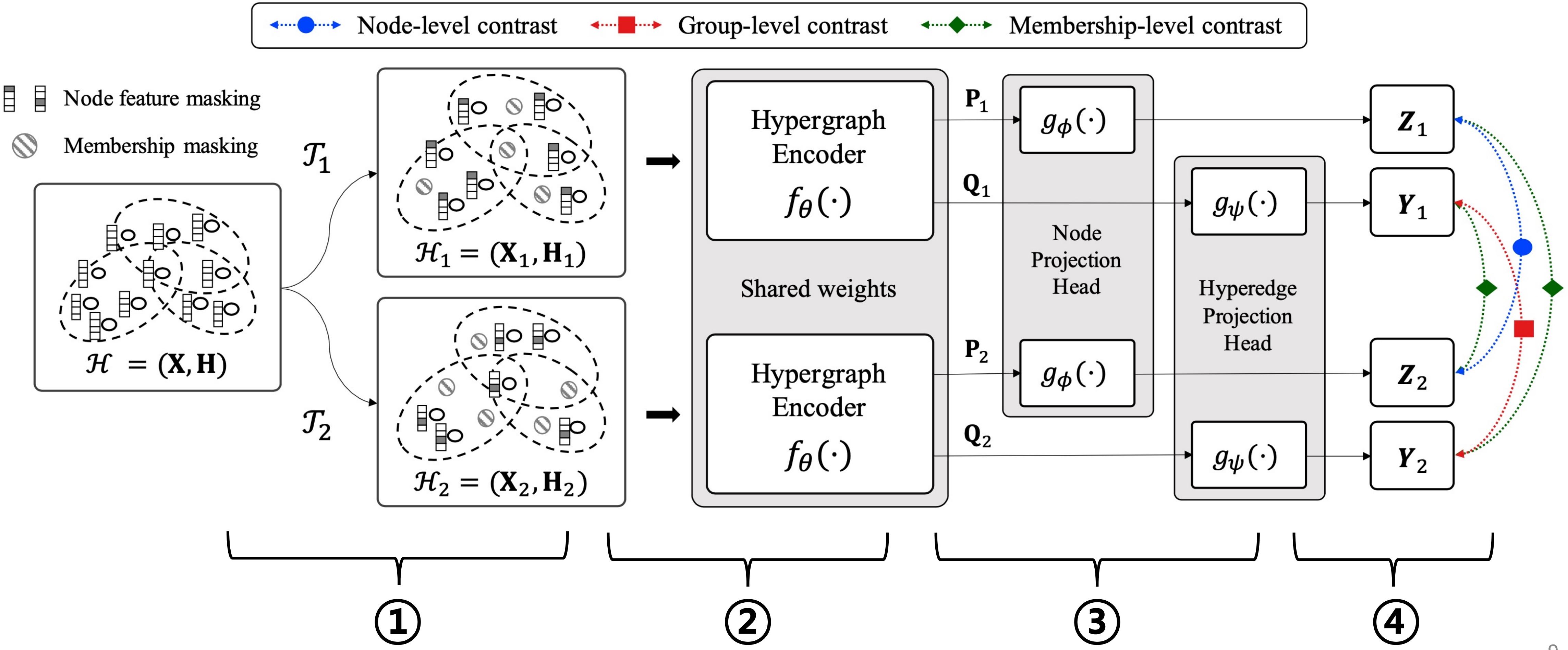
- (Q1) What to contrast?
- (Q2) How to augment a hypergraph?
- (Q3) How to select negative samples?

Road Map

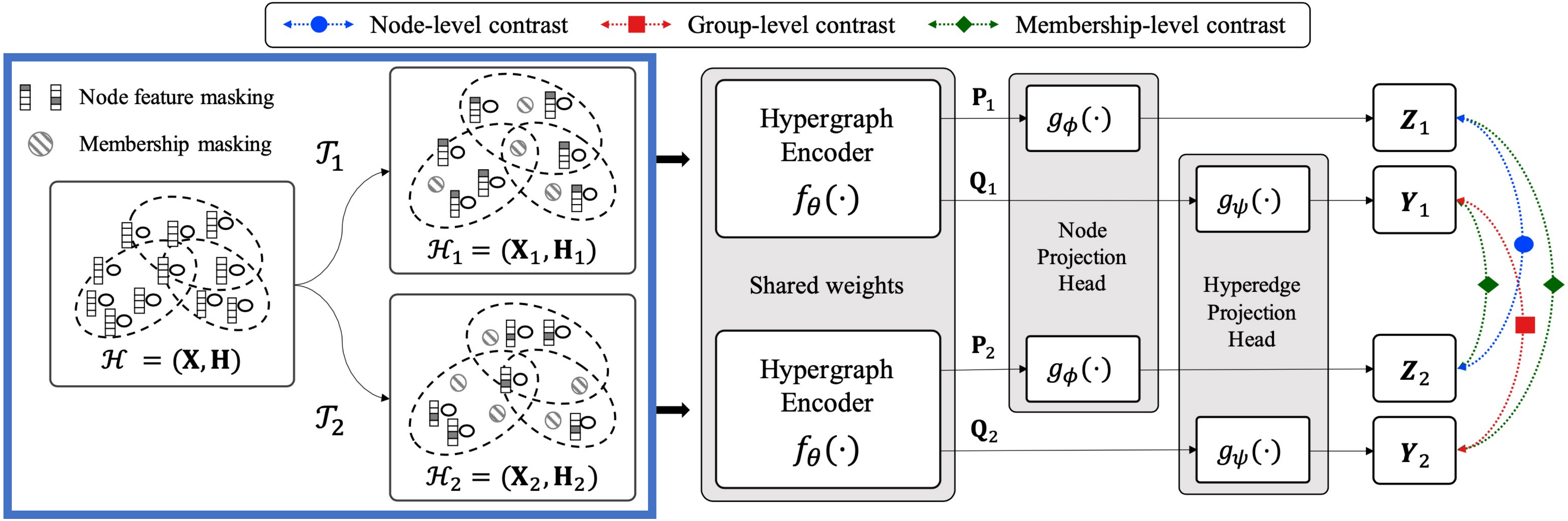
- Preamble
- **Proposed Method: TriCL**
- Experiments
- Conclusion

Proposed Approach: TriCL

- **Goal**: to train a hypergraph encoder in a contrastive manner



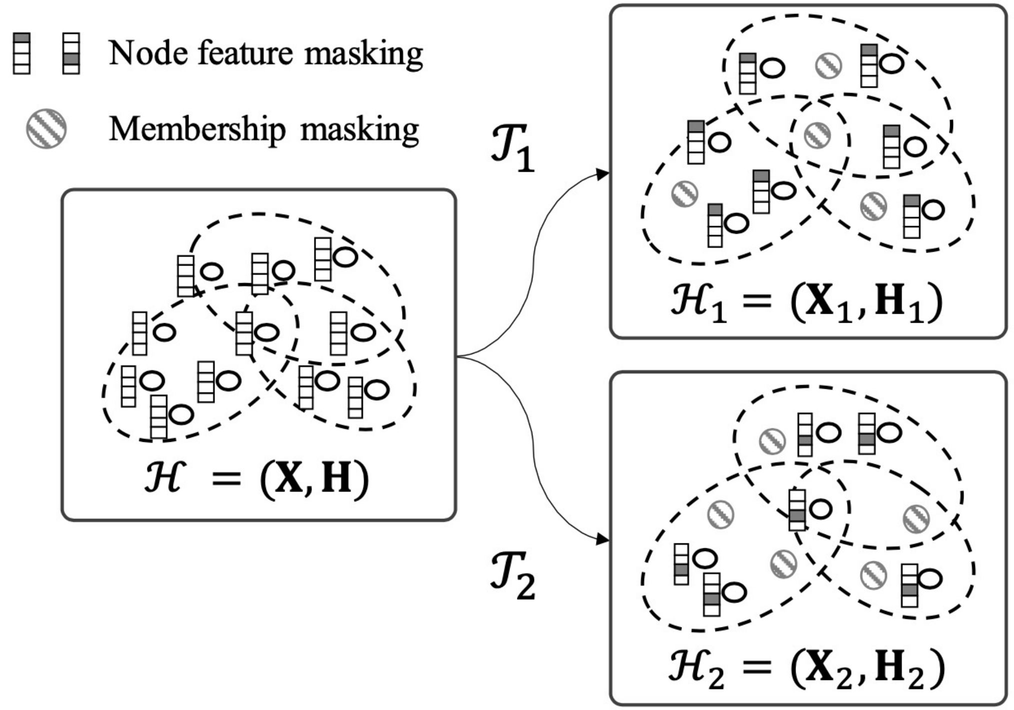
Proposed Approach: TriCL



① Hypergraph augmentation

Hypergraph Augmentation

- TriCL first <u>generates two alternative views</u> of the hypergraph $\mathcal{H}$, by applying stochastic hypergraph augmentation functions $\mathcal{T}_1$ and $\mathcal{T}_2$.
- Augmentation: ***node feature masking*** + ***membership masking***.



Node feature masking

Column-wise masking

$$\begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \\ \vdots \\ \\ \\ \\ \end{array} \begin{bmatrix} 0 & 0 & 0 & \cdots & 1 & 0 & 0 \\ 1 & 1 & 0 & \cdots & 0 & 0 & 1 \\ 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots \\ 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 & 1 \end{bmatrix}$$

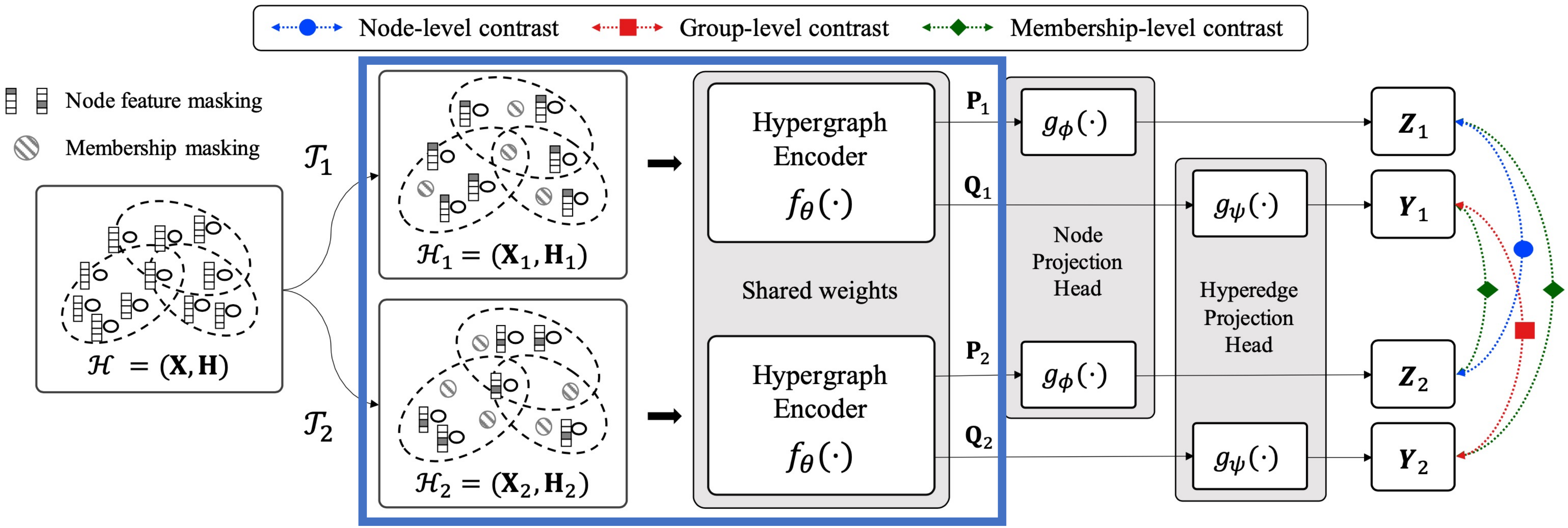
Corrupted feature matrix

Membership masking

$$\begin{array}{c|c} & \begin{array}{ccccccc} e_1 & e_2 & e_3 & \cdots & & & \end{array} \\ \begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \\ \vdots \\ \\ \\ \\ \end{array} & \begin{bmatrix} 0 & 0 & 0 & \cdots & 1 & 0 & 0 \\ 1 & 0 & 0 & \cdots & 0 & 0 & 1 \\ 1 & 0 & 1 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 1 & 1 & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots \\ 0 & 1 & 0 & \cdots & 0 & 1 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 1 \end{bmatrix} \end{array}$$

Corrupted incidence matrix

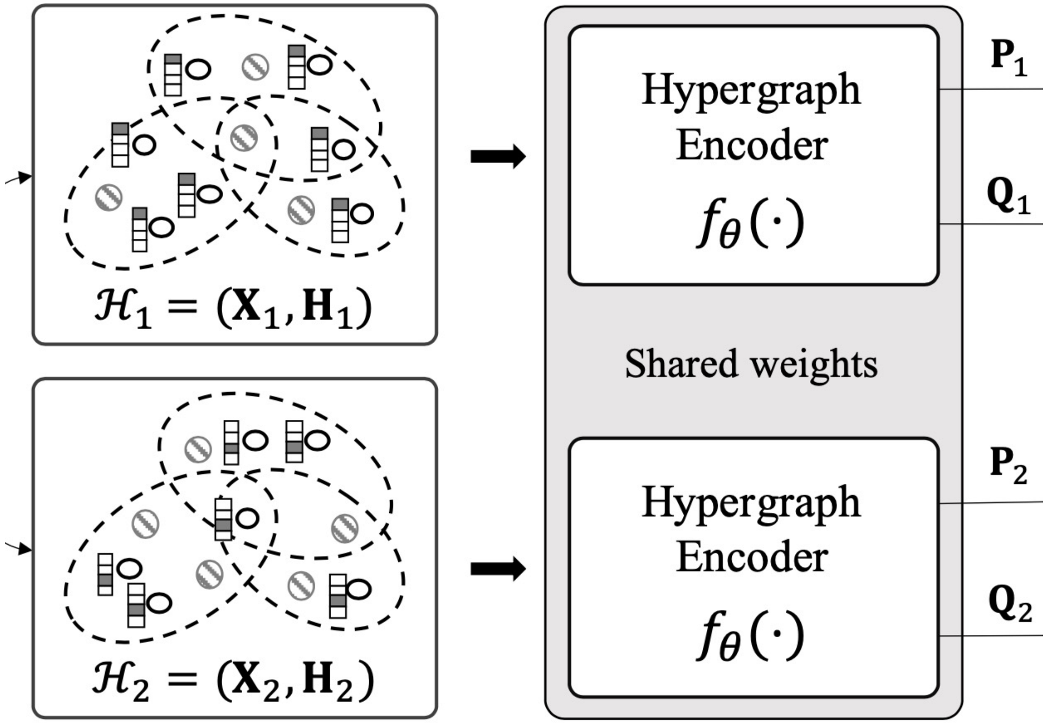
Proposed Approach: TriCL



② *Hypergraph encoder*

Hypergraph Encoder

- The encoder produces node and hyperedge representations, $\mathbf{P}$ and $\mathbf{Q}$, for two augmented views.
- TriCL uses a simple ***mean pooling layer*** as an encoder.

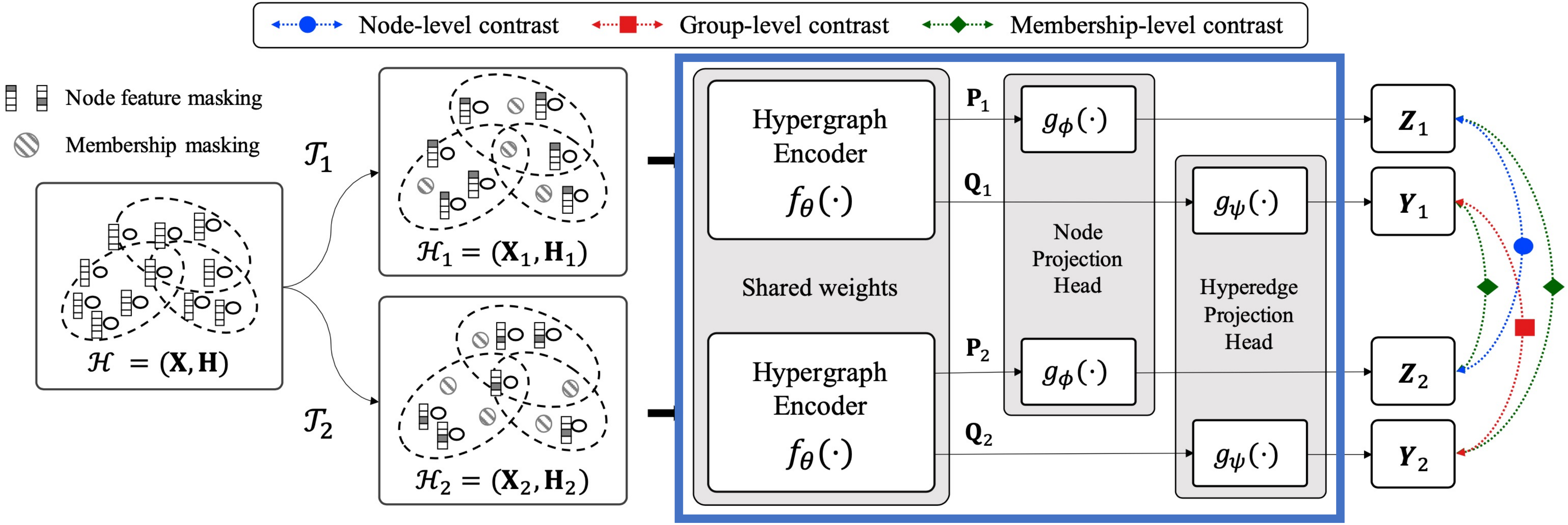


At layer k,

$$\mathbf{P}^{(k)} = \sigma\left(\mathbf{D}_V^{-1}\mathbf{H}\mathbf{W}\mathbf{Q}^{(k)}\mathbf{\Theta}_V^{(k)} + \mathbf{b}_V^{(k)}\right)$$

$$\mathbf{Q}^{(k)} = \sigma\left(\mathbf{D}_E^{-1}\mathbf{H}^T\mathbf{P}^{(k-1)}\mathbf{\Theta}_E^{(k)} + \mathbf{b}_E^{(k)}\right)$$

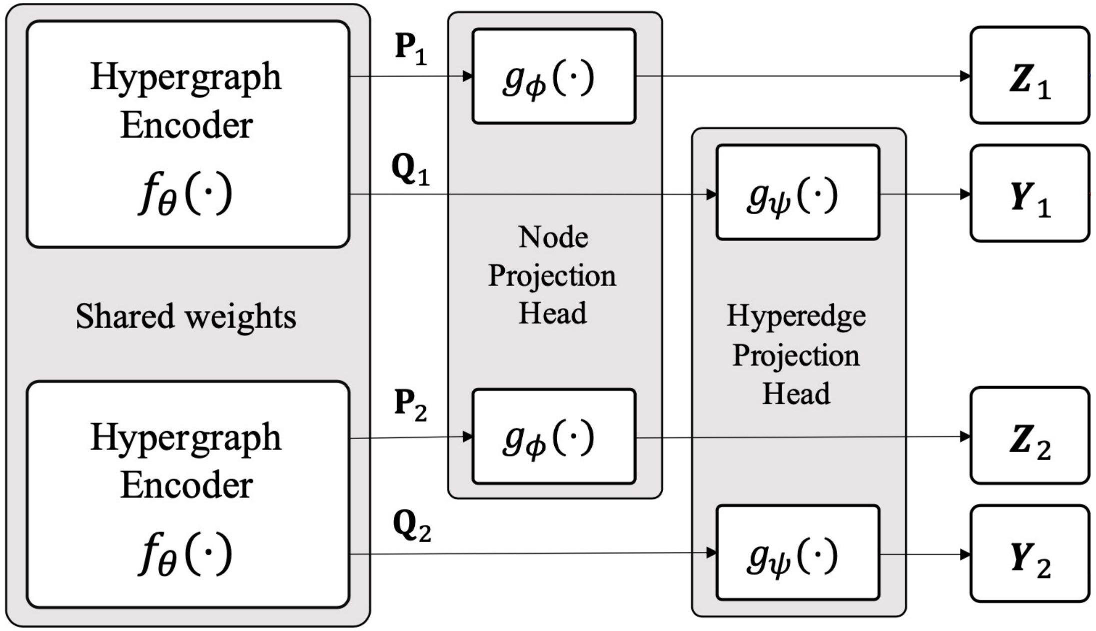
Proposed Approach: TriCL



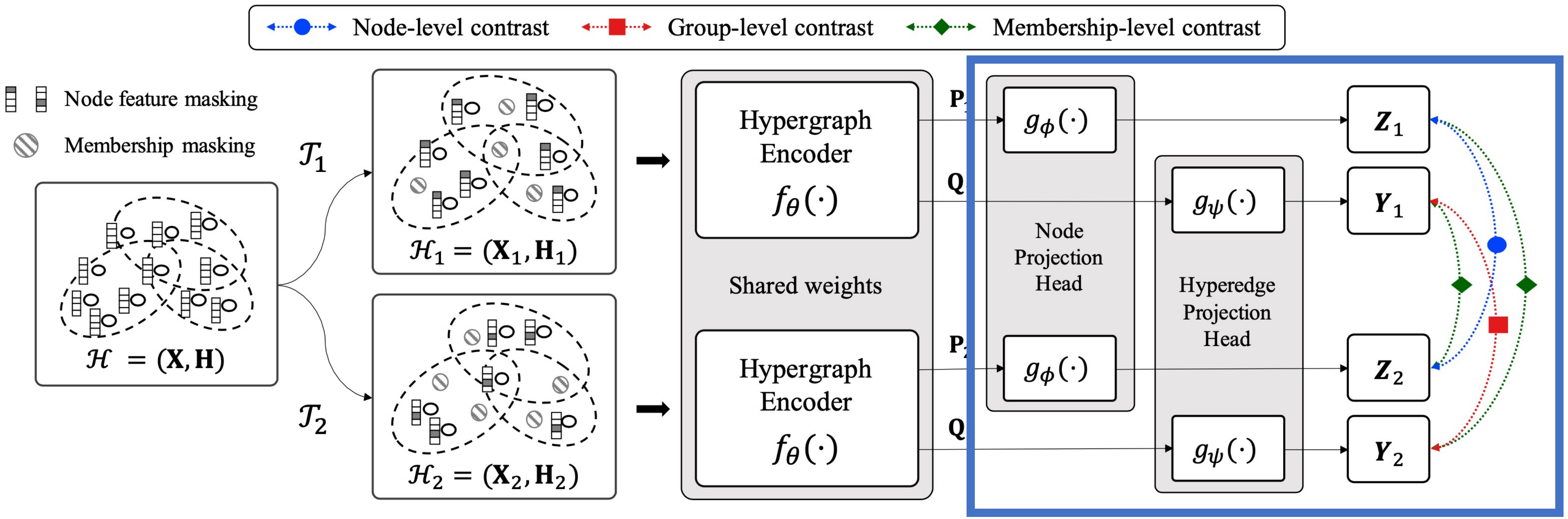
③ ***Projection head***

Projection Head

- SimCLR [Chen et al. 2020] empirically demonstrates that including projection head helps to improve the quality of representations.
- Projection head: ***two-layer MLP*** and ***ELU activation***.



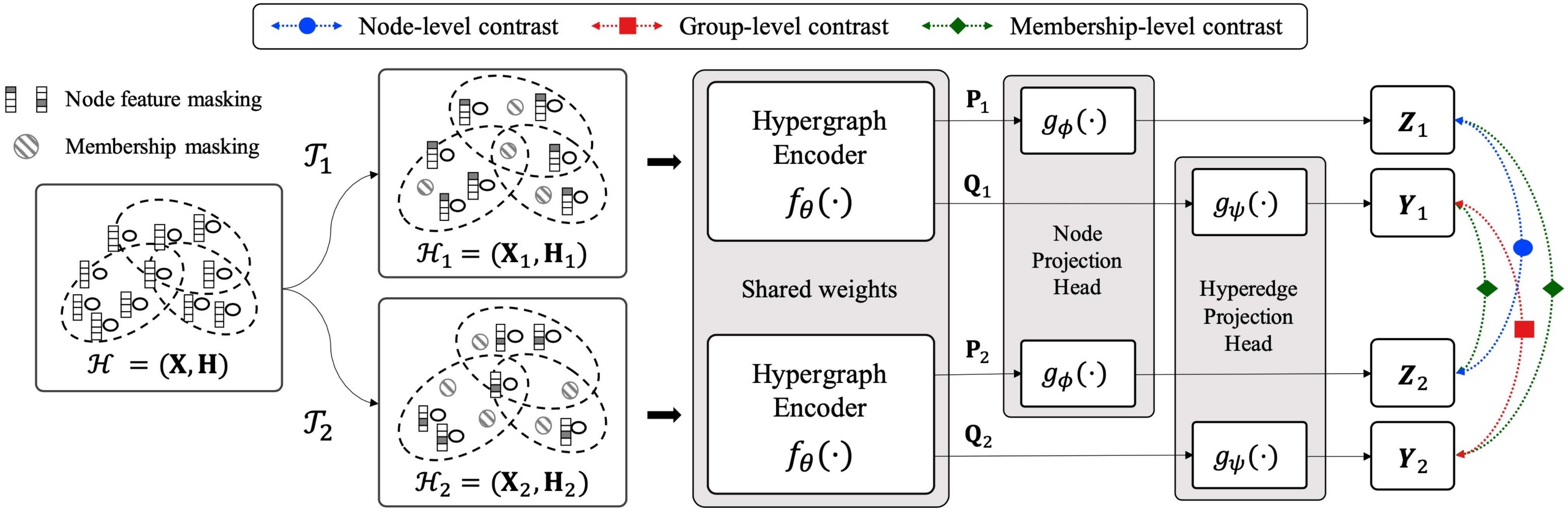
Proposed Approach: TriCL



④ *Tri-directional contrastive loss*

Tri-directional Contrastive Loss

- Three contrastive objectives:
    1. ***Node-level*** contrast: discriminate the same node from other nodes.
    2. ***Group-level*** contrast: discriminate the same hyperedge from other hyperedges.
    3. ***Membership-level*** contrast: discriminate a 'real' membership from a 'fake' one.

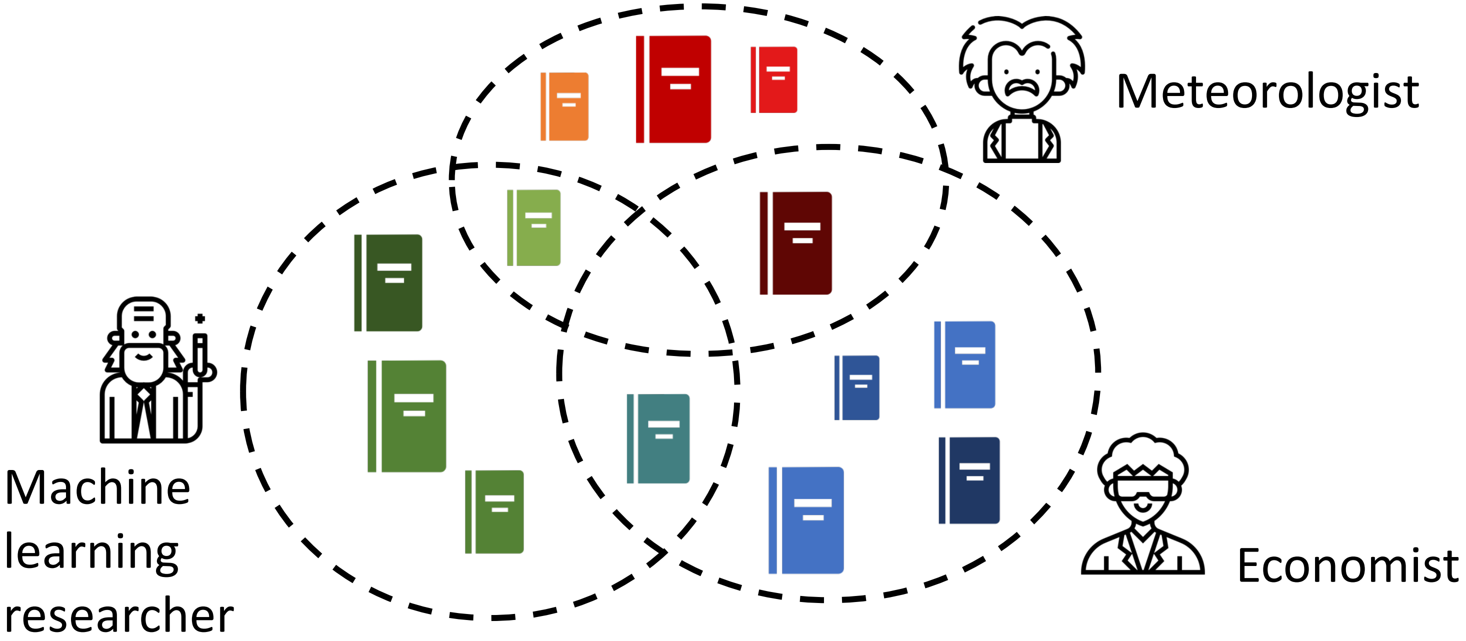


Motivating Examples

Q) How can the three forms of contrast be helpful for representation learning?

① In node classification, Information about a group of nodes could be helpful.

- Papers written by the same author are more likely to belong to the same field and cover similar topics (i.e., homophily exists in hypergraphs).

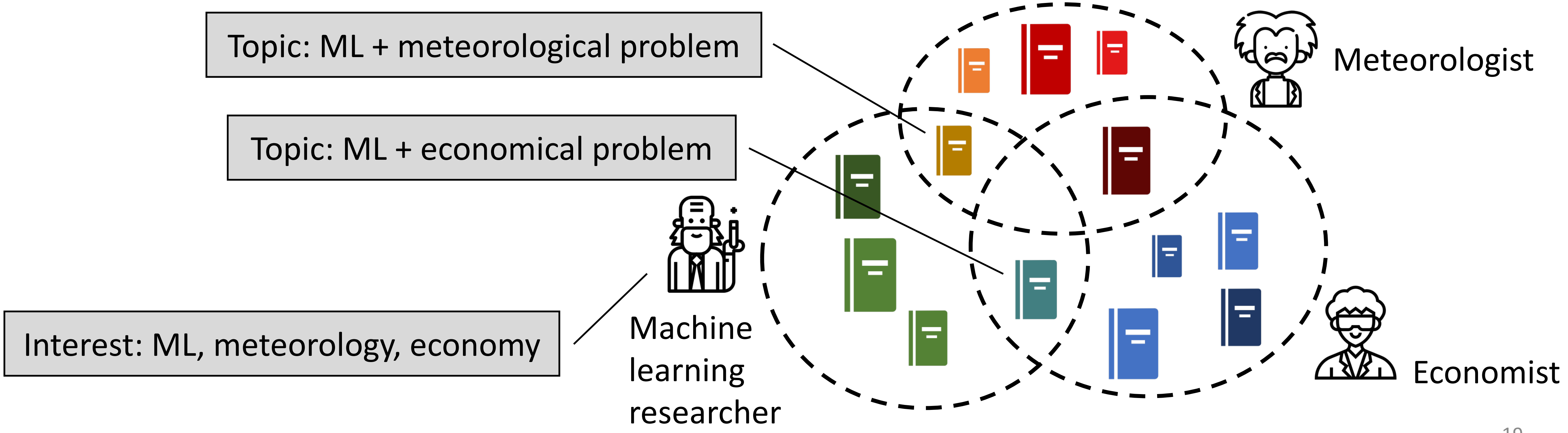


Motivating Examples

Q) How can the three forms of contrast be helpful for node representation learning?

① In node classification, Information about a group of nodes could be helpful.

② Leveraging node-hyperedge membership helps enrich the information of node and hyperedge.



Tri-directional Contrastive Loss

Node contrastive loss

- Discriminate the same node from other nodes.
- Embed *microscopic structural information* into embeddings.

For node i: $$\ell_n(\boldsymbol{z}_{1,i}, \boldsymbol{z}_{2,i}) = -\log \frac{e^{s(\boldsymbol{z}_{1,i}, \boldsymbol{z}_{2,i})/\tau_n}}{\sum_{k=1}^{|V|} e^{s(\boldsymbol{z}_{1,i}, \boldsymbol{z}_{2,k})/\tau_n}},$$

For all nodes: $$\mathcal{L}_n = \frac{1}{2|V|} \sum_{i=1}^{|V|} \{\ell_n(\boldsymbol{z}_{1,i}, \boldsymbol{z}_{2,i}) + \ell_n(\boldsymbol{z}_{2,i}, \boldsymbol{z}_{1,i})\}$$

(Proposed) Tri-directional contrastive loss:

$$\mathcal{L} = \mathcal{L}_n + \omega_g \mathcal{L}_g + \omega_m \mathcal{L}_m$$

Tri-directional Contrastive Loss

Group contrastive loss

- Discriminate the same hyperedge from other hyperedges.
- Embed *group-level structural information* into embeddings.

For hyperedge j: $$\ell_g(\boldsymbol{y}_{1,j}, \boldsymbol{y}_{2,j}) = -\log \frac{e^{s(\boldsymbol{y}_{1,j}, \boldsymbol{y}_{2,j})/\tau_g}}{\sum_{k=1}^{|E|} e^{s(\boldsymbol{y}_{1,j}, \boldsymbol{y}_{2,k})/\tau_g}},$$

For all hyperedges: $$\mathcal{L}_g = \frac{1}{2|E|} \sum_{j=1}^{|E|} \left\{ \ell_g(\boldsymbol{y}_{1,j}, \boldsymbol{y}_{2,j}) + \ell_g(\boldsymbol{y}_{2,j}, \boldsymbol{y}_{1,j}) \right\}$$

(Proposed) Tri-directional contrastive loss:

$$\mathcal{L} = \mathcal{L}_n + \omega_g \mathcal{L}_g + \omega_m \mathcal{L}_m$$

Tri-directional Contrastive Loss

Membership contrastive loss

- Discriminate a ‘real’ membership from a ‘fake’ one.
- Apply *<u>group-level constraints to nodes</u>*.

For membership (i, j):

$$\ell_m(\boldsymbol{z}_i, \boldsymbol{y}_j) = -\log \underbrace{\frac{e^{\mathcal{D}(\boldsymbol{z}_i, \boldsymbol{y}_j)/\tau_m}}{e^{\mathcal{D}(\boldsymbol{z}_i, \boldsymbol{y}_j)/\tau_m} + \sum_{k: i \notin k} e^{\mathcal{D}(\boldsymbol{z}_i, \boldsymbol{y}_k)/\tau_m}}}_{\text{when } \boldsymbol{z}_i \text{ is the anchor}} - \log \underbrace{\frac{e^{\mathcal{D}(\boldsymbol{z}_i, \boldsymbol{y}_j)/\tau_m}}{e^{\mathcal{D}(\boldsymbol{z}_i, \boldsymbol{y}_j)/\tau_m} + \sum_{k: k \notin j} e^{\mathcal{D}(\boldsymbol{z}_k, \boldsymbol{y}_j)/\tau_m}}}_{\text{when } \boldsymbol{y}_j \text{ is the anchor}},$$

For all memberships:

$$\mathcal{L}_m = \frac{1}{2K} \sum_{i=1}^{|V|} \sum_{j=1}^{|E|} \mathbb{1}_{[h_{ij}=1]} \{ \ell_m(\boldsymbol{z}_{1,i}, \boldsymbol{y}_{2,j}) + \ell_m(\boldsymbol{z}_{2,i}, \boldsymbol{y}_{1,j}) \}$$

(Proposed) Tri-directional contrastive loss:

$$\mathcal{L} = \mathcal{L}_n + \omega_g \mathcal{L}_g + \omega_m \mathcal{L}_m$$

Road Map

- Preamble
- Proposed Method: TriCL
- **Experiments**
- Conclusion

Datasets

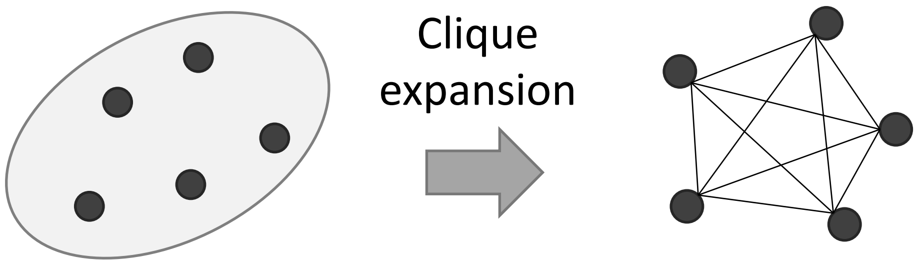
- 10 commonly used benchmark datasets
  - **Co-citation**: Cora, Citeseer, and Pubmed [Sen et al. 2008]
  - **Co-authorship**: Cora and DBLP [Rossi and Ahmed 2015]
  - **Vision and graphics**: NTU2012 [Chen et al. 2003] and ModelNet40 [Wu et al. 2015]
  - **UCI repository**: Zoo, 20Newsgroups, and Mushroom [Dua and Graff 2017]

	Cora-C	Citeseer	Pubmed	Cora-A	DBLP	Zoo	20News	Mushroom	NTU2012	ModelNet40
# Nodes	1,434	1,458	3,840	2,388	41,302	101	16,242	8,124	2,012	12,311
# Hyperedges	1,579	1,079	7,963	1,072	22,363	43	100	298	2,012	12,311
# Memberships	4,786	3,453	34,629	4,585	99,561	1,717	65,451	40,620	10,060	61,555
Avg. hyperedge size	3.03	3.20	4.35	4.28	4.45	39.93	654.51	136.31	5	5
Avg. node degree	3.34	2.37	9.02	1.92	2.41	17.00	4.03	5.00	5	5
Max. hyperedge size	5	26	171	43	202	93	2241	1808	5	5
Max. node degree	145	88	99	23	18	17	44	5	19	30
# Features	1,433	3,703	500	1,433	1,425	16	100	22	100	100
# Classes	7	6	3	7	6	7	4	2	67	40

Baselines

- 10 (semi-)supervised models:
  - MLP
  - Graph neural networks:
    - GCN [Kipf and Welling 2017]
    - GAT [Velickovic et al. 2018]
  - Hypergraph neural networks:
    - HGNN [Feng et al. 2019]
    - HyperConv [Bai, Zhang, and Torr 2021]
    - HNHN [Dong, Sawin, and Bengio 2020]
    - HyperGCN [Yadati et al. 2019]
    - HyperSAGE [Arya et al. 2020]
    - UniGCN [Huang and Yang 2021]
    - AllSetTransformer [Chien et al. 2022]
- 4 unsupervised models:
  - Network embedding method:
    - Node2vec [Grover and Leskovec 2016]
  - Graph CL methods:
    - DGI [Velickovic et al. 2018]
    - GRACE [Zhu et al. 2020]
  - Hypergraph CL method:
    - S^2-HHGR [Zhang et al. 2021]

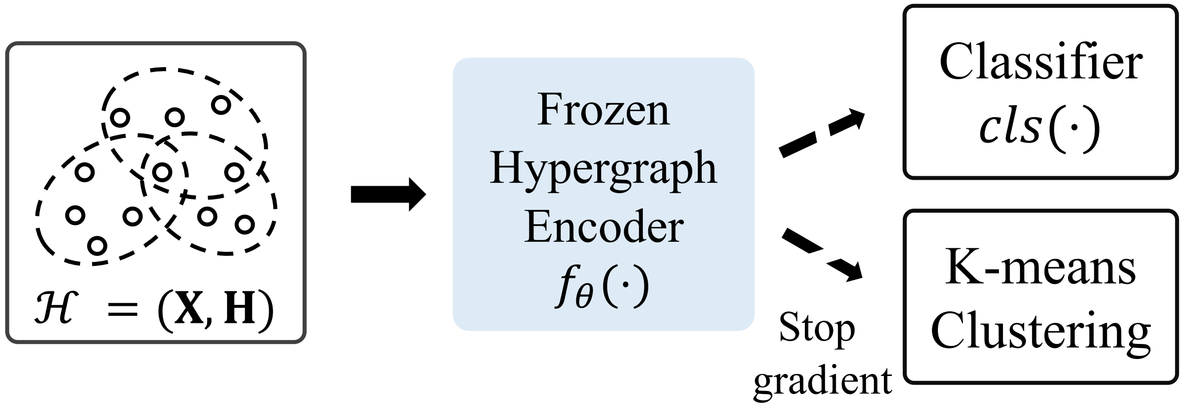
✓ For graph-based methods



Evaluation Protocol

Q) How to assess the quality of node representations learnt by TriCL?

- Task-1: Node classification
  - **Linear evaluation**: train the encoder unsupervised manner and then employ a ℓ_2-regularized logistic regression model on top of the frozen representations.
- Task-2: Clustering
  - **k-means clustering**: train the encoder unsupervised manner and then employ k-means clustering on top of the frozen representations.



Performance on Node Classification

- **<u>TriCL consistently outperforms its unsupervised baselines</u>** by significant margins.
- It also outperforms the models trained with label supervision.

: The best performance : The second-best performance A.R.: Average rank

	Method	Cora-C	Citeseer	Pubmed	Cora-A	DBLP	Zoo	20News	Mushroom	NTU2012	ModelNet40	A.R.↓
Supervised	MLP	60.32 ± 1.5	62.06 ± 2.3	76.27 ± 1.1	64.05 ± 1.4	81.18 ± 0.2	75.62 ± 9.5	79.19 ± 0.5	99.58 ± 0.3	65.17 ± 2.3	93.75 ± 0.6	12.5
	GCN⋆	77.11 ± 1.8	66.07 ± 2.4	82.63 ± 0.6	73.66 ± 1.3	87.58 ± 0.2	36.79 ± 9.6	OOM	92.47 ± 0.9	71.17 ± 2.4	91.67 ± 0.2	11.7
	GAT⋆	77.75 ± 2.1	67.62 ± 2.5	81.96 ± 0.7	74.52 ± 1.3	88.59 ± 0.1	36.48 ± 10.0	OOM	OOM	70.94 ± 2.6	91.43 ± 0.3	11
	HGNN	77.50 ± 1.8	66.16 ± 2.3	83.52 ± 0.7	74.38 ± 1.2	88.32 ± 0.3	78.58 ± 11.1	80.15 ± 0.3	98.59 ± 0.5	72.03 ± 2.4	92.23 ± 0.2	8.1
	HyperConv	76.19 ± 2.1	64.12 ± 2.6	83.42 ± 0.6	73.52 ± 1.0	88.83 ± 0.2	62.53 ± 14.5	79.83 ± 0.4	97.56 ± 0.6	72.62 ± 2.6	91.84 ± 0.1	9.8
	HNHN	76.21 ± 1.7	67.28 ± 2.2	80.97 ± 0.9	74.88 ± 1.6	86.71 ± 1.2	78.89 ± 10.2	79.51 ± 0.4	99.78 ± 0.1	71.45 ± 3.2	92.96 ± 0.2	8.9
	HyperGCN	64.11 ± 7.4	59.92 ± 9.6	78.40 ± 9.2	60.65 ± 9.2	76.59 ± 7.6	40.86 ± 2.1	77.31 ± 6.0	48.26 ± 0.3	46.05 ± 3.9	69.23 ± 2.8	15.1
	HyperSAGE	64.98 ± 5.3	52.43 ± 9.4	79.49 ± 8.7	64.59 ± 4.3	79.63 ± 8.6	40.86 ± 2.1	OOT	OOT	OOT	OOT	14.7
	UniGCN	77.91 ± 1.9	66.40 ± 1.9	<u>84.08 ± 0.7</u>	77.30 ± 1.4	90.31 ± 0.2	72.10 ± 12.1	**80.24 ± 0.4**	98.84 ± 0.5	73.27 ± 2.7	94.62 ± 0.2	5.9
	AllSet	76.21 ± 1.7	67.83 ± 1.8	82.85 ± 0.9	76.94 ± 1.3	90.07 ± 0.3	72.72 ± 11.8	79.90 ± 0.4	99.78 ± 0.1	75.09 ± 2.5	96.85 ± 0.2	6.2
Unsupervised	Node2vec⋆	70.99 ± 1.4	53.85 ± 1.9	78.75 ± 0.9	58.50 ± 2.1	72.09 ± 0.3	17.02 ± 4.1	63.35 ± 1.7	88.16 ± 0.8	67.72 ± 2.1	84.94 ± 0.4	15.6
	DGI⋆	78.17 ± 1.4	68.81 ± 1.8	80.83 ± 0.6	76.94 ± 1.1	88.00 ± 0.2	36.54 ± 9.7	OOM	OOM	72.01 ± 2.5	92.18 ± 0.2	9.3
	GRACE⋆	79.11 ± 1.7	68.65 ± 1.7	80.08 ± 0.7	76.59 ± 1.0	OOM	37.07 ± 9.3	OOM	OOM	70.51 ± 2.4	90.68 ± 0.3	10.4
	S^2-HHGR	78.08 ± 1.7	68.21 ± 1.8	82.13 ± 0.6	78.15 ± 1.1	88.69 ± 0.2	80.06 ± 11.1	79.75 ± 0.3	97.15 ± 0.5	73.95 ± 2.4	93.26 ± 0.2	6.8
	Random-Init	63.62 ± 3.1	60.44 ± 2.5	67.49 ± 2.2	66.27 ± 2.2	76.57 ± 0.6	78.43 ± 11.0	77.14 ± 0.6	97.40 ± 0.6	74.39 ± 2.6	96.29 ± 0.3	11.9
	TriCL-N	80.23 ± 1.2	70.28 ± 1.5	83.44 ± 0.6	81.94 ± 1.1	90.88 ± 0.1	79.94 ± 11.1	<u>80.18 ± 0.2</u>	99.76 ± 0.2	75.20 ± 2.6	97.01 ± 0.2	3.4
	TriCL-NG	<u>81.45 ± 1.2</u>	<u>71.38 ± 1.2</u>	83.68 ± 0.7	<u>82.00 ± 1.0</u>	<u>90.94 ± 0.1</u>	<u>80.19 ± 11.1</u>	<u>80.18 ± 0.2</u>	<u>99.81 ± 0.1</u>	**75.25 ± 2.5**	<u>97.02 ± 0.1</u>	<u>2</u>
	TriCL	**81.57 ± 1.1**	**72.02 ± 1.2**	**84.26 ± 0.6**	**82.15 ± 0.9**	**91.12 ± 0.1**	**80.25 ± 11.2**	80.14 ± 0.2	**99.83 ± 0.1**	<u>75.23 ± 2.4</u>	**97.08 ± 0.1**	**1.5**

Ablation Study

- The more types of contrast we use, the better the performance tends to be.
- ***Using all types of contrast*** ***achieves the best performance*** in most cases as they are complementarily reinforcing each other.

: The best performance : The second-best performance A.R.: Average rank

$\mathcal{L}_n$	$\mathcal{L}_g$	$\mathcal{L}_m$	Cora-C	Citeseer	Pubmed	Cora-A	DBLP	Zoo	20News	Mushroom	NTU2012	ModelNet40	A.R.↓
✓	-	-	80.23 ± 1.2	70.28 ± 1.5*	83.44 ± 0.6	81.94 ± 1.1	90.88 ± 0.1	79.94 ± 11.1	**80.18 ± 0.2**	99.76 ± 0.2	75.20 ± 2.6	97.01 ± 0.2	3.8
-	✓	-	79.69 ± 1.6	71.02 ± 1.3*	80.20 ± 1.3	78.98 ± 1.4	88.60 ± 0.2	79.31 ± 10.7	79.35 ± 0.4	99.13 ± 0.3	74.41 ± 2.6	96.66 ± 0.2	5.7
-	-	✓	76.76 ± 1.8	63.98 ± 2.0	79.86 ± 0.9	76.77 ± 1.1	63.95 ± 7.2	79.80 ± 11.0	79.27 ± 0.3	94.87 ± 0.7	73.11 ± 2.8	96.57 ± 0.2	6.9
✓	✓	-	<u>81.45 ± 1.2</u>	71.38 ± 1.4	83.68 ± 0.7	<u>82.00 ± 1.0</u>	<u>90.94 ± 0.1</u>	80.19 ± 11.1	**80.18 ± 0.2**	<u>99.81 ± 0.1</u>	**75.25 ± 2.5**	97.02 ± 0.1	<u>2.3</u>
✓	-	✓	80.49 ± 1.3	70.46 ± 1.5	<u>83.98 ± 0.7</u>	81.62 ± 1.0	90.75 ± 0.1	80.19 ± 11.1	80.15 ± 0.2	99.74 ± 0.2	75.12 ± 2.5	<u>97.03 ± 0.1</u>	3.6
-	✓	✓	80.80 ± 1.1	<u>71.73 ± 1.4</u>	82.81 ± 0.7	80.24 ± 1.0	90.17 ± 0.1	<u>80.20 ± 11.1</u>	79.29 ± 0.2	99.82 ± 0.1	73.76 ± 2.5	96.74 ± 0.1	4.1
✓	✓	✓	**81.57 ± 1.1**	**72.02 ± 1.4**	**84.26 ± 0.6**	**82.15 ± 0.9**	**91.12 ± 0.1**	**80.25 ± 11.2**	80.14 ± 0.2	**99.83 ± 0.1**	<u>75.23 ± 2.4</u>	**97.08 ± 0.1**	**1.4**

Robustness to the Number of Negatives

- TriCL-Subsampling (k) uses randomly subsampled k negatives across the hypergraph to construct the node- and group-level contrastive loss.
- ***TriCL is very robust*** to the number of negatives.
- Even if only ***two negatives*** are used, the performance degradation is ***<1%***.

A.P.D: Average performance degradation

Method	Cora-C	Citeseer	Pubmed	Cora-A	DBLP	Zoo	20News	Mushroom	NTU2012	ModelNet40	A.P.D.
S^2-HHGR all negatives	78.08 ± 1.7	68.21 ± 1.8	82.13 ± 0.6	78.15 ± 1.1	88.69 ± 0.2	80.06 ± 11.1	79.75 ± 0.3	97.15 ± 0.5	73.95 ± 2.4	93.26 ± 0.2	-
TriCL-Subsampling ($k = 2$)	80.62 ± 1.3	71.95 ± 1.3	83.22 ± 0.7	81.25 ± 1.0	90.66 ± 0.2	80.10 ± 11.1	80.03 ± 0.2	99.82 ± 0.1	74.95 ± 2.6	97.02 ± 0.1	0.49%
TriCL-Subsampling ($k = 4$)	81.15 ± 1.2	$\mathbf{72.24 \pm 1.2}$	83.91 ± 0.7	81.85 ± 0.9	90.83 ± 0.1	80.16 ± 11.3	80.08 ± 0.2	$\mathbf{99.84 \pm 0.1}$	75.02 ± 2.6	97.05 ± 0.1	0.18%
TriCL-Subsampling ($k = 8$)	81.32 ± 1.2	72.04 ± 1.3	83.88 ± 0.7	82.05 ± 0.9	90.93 ± 0.1	80.14 ± 11.2	80.12 ± 0.2	99.84 ± 0.1	75.09 ± 2.5	97.05 ± 0.1	0.14%
TriCL-Subsampling ($k = 16$)	81.49 ± 1.1	72.02 ± 1.2	84.23 ± 0.7	82.10 ± 0.9	90.97 ± 0.1	80.10 ± 11.1	80.13 ± 0.2	99.84 ± 0.1	75.16 ± 2.5	97.07 ± 0.1	0.06%
TriCL all negatives	$\mathbf{81.57 \pm 1.1}$	72.02 ± 1.2	$\mathbf{84.26 \pm 0.6}$	$\mathbf{82.15 \pm 0.9}$	$\mathbf{91.12 \pm 0.1}$	$\mathbf{80.25 \pm 11.2}$	$\mathbf{80.14 \pm 0.2}$	99.83 ± 0.1	$\mathbf{75.23 \pm 2.4}$	$\mathbf{97.08 \pm 0.1}$	-

Performance on Clustering

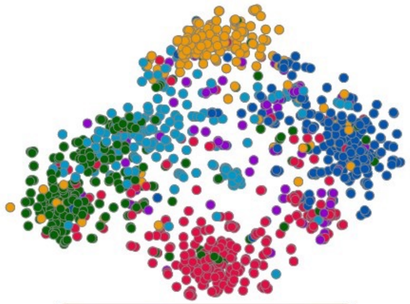
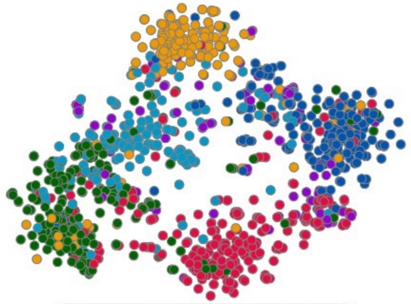
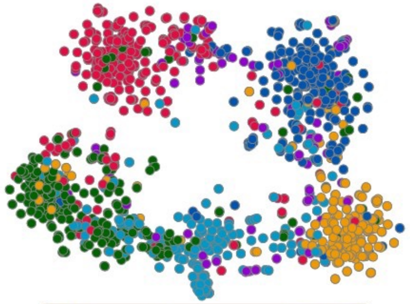
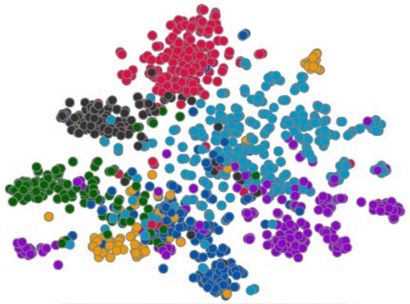
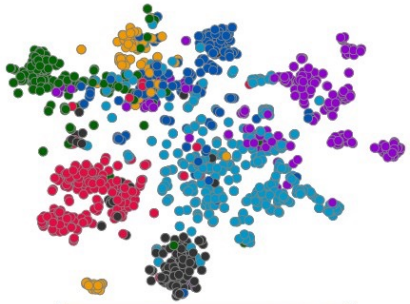
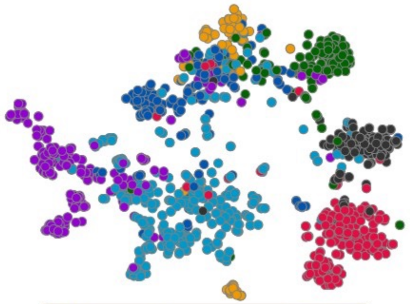
- TriCL achieves strong clustering performance in terms of NMI and F1.
- Why? The node embeddings learned by **TriCL simultaneously preserve local and community structural information** by fully utilizing group-level contrast.

A.R.: Average rank

Method	Cora-C		Citeseer		Pubmed		Cora-A		DBLP		Zoo		20News		Mushroom		NTU2012		ModelNet40		A.R.↓
	NMI↑	F1↑	NMI↑	F1↑	NMI↑	F1↑	NMI↑	F1↑	NMI↑	F1↑	NMI↑	F1↑	NMI↑	F1↑	NMI↑	F1↑	NMI↑	F1↑	NMI↑	F1↑	
features	20.0	28.8	21.5	36.1	19.5	**53.4**	17.2	29.2	37.0	47.3	78.3	77.3	15.7	41.1	**36.6**	**72.4**	81.7	69.0	90.6	86.5	3.8
Node2vec*	39.1	44.5	24.5	38.5	23.1	40.1	16.0	34.1	32.4	37.8	11.5	41.6	8.7	26.6	1.6	44.0	78.3	57.7	72.9	53.1	5.0
DGI*	**54.8**	60.1	40.1	51.7	**30.4**	53.0	45.2	52.5	58.0	57.7	13.0	13.8	OOM		OOM		79.6	61.7	85.0	73.7	3.1
GRACE*	44.4	45.6	33.3	45.7	16.7	41.9	37.9	43.3	16.7	41.9	7.3	29.4	OOM		OOM		74.6	47.5	79.4	59.9	4.9
S^2-HHGR	51.0	56.8	41.1	53.1	27.7	53.2	45.4	52.3	60.3	62.7	90.9	**91.1**	**39.0**	**58.7**	18.6	60.6	82.7	71.2	91.0	90.6	2.1
TriCL	54.5	**60.6**	**44.1**	**57.4**	30.0	51.7	**49.8**	**56.7**	**63.1**	**63.0**	**91.2**	89.3	35.6	54.2	3.8	65.1	**83.2**	**71.5**	**95.7**	**94.7**	**1.6**

Qualitative Analysis

- ***<u>TriCL gives more visually and numerically</u>*** (based on t-SNE plots and the Silhouette score) ***<u>distinguishable clusters</u>*** than its two variants.

	TriCL-N	TriCL-NG	TriCL
Citeseer	 Silhouette score: 0.166	 Silhouette score: 0.175	 **Silhouette score: 0.194**
Cora-C	 Silhouette score: 0.167	 Silhouette score: 0.215	 **Silhouette score: 0.244**

Road Map

- Preamble
- Proposed Method: TriCL
- Experiments
- **Conclusion**

Conclusion

- We proposed ***TriCL***, a novel hypergraph contrastive learning method.
  - (Q1) What to contrast?
    - (A) ***Tri-directional contrast***: node, group, and membership contrast
  - (Q2) How to augment a hypergraph?
    - (A) ***Node feature masking*** and ***membership masking***
  - (Q3) How to select negative samples?
    - (A) ***Uniform random sampling***

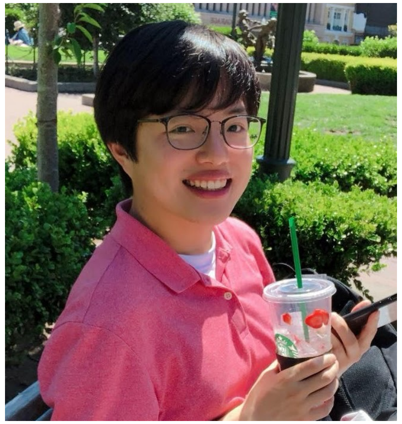
Code and Data:
https://github.com/wooner49/TriCL

SCAN ME

The 37TH AAAI Conference (AAAI 2023)

I’m Me, We’re Us, I’m Us: Tri-directional Contrastive Learning on Hypergraphs

Dongjin Lee



Kijung Shin