CIKM 2022 Research Paper

LTE4G: Long-Tail Experts for Graph Neural Networks

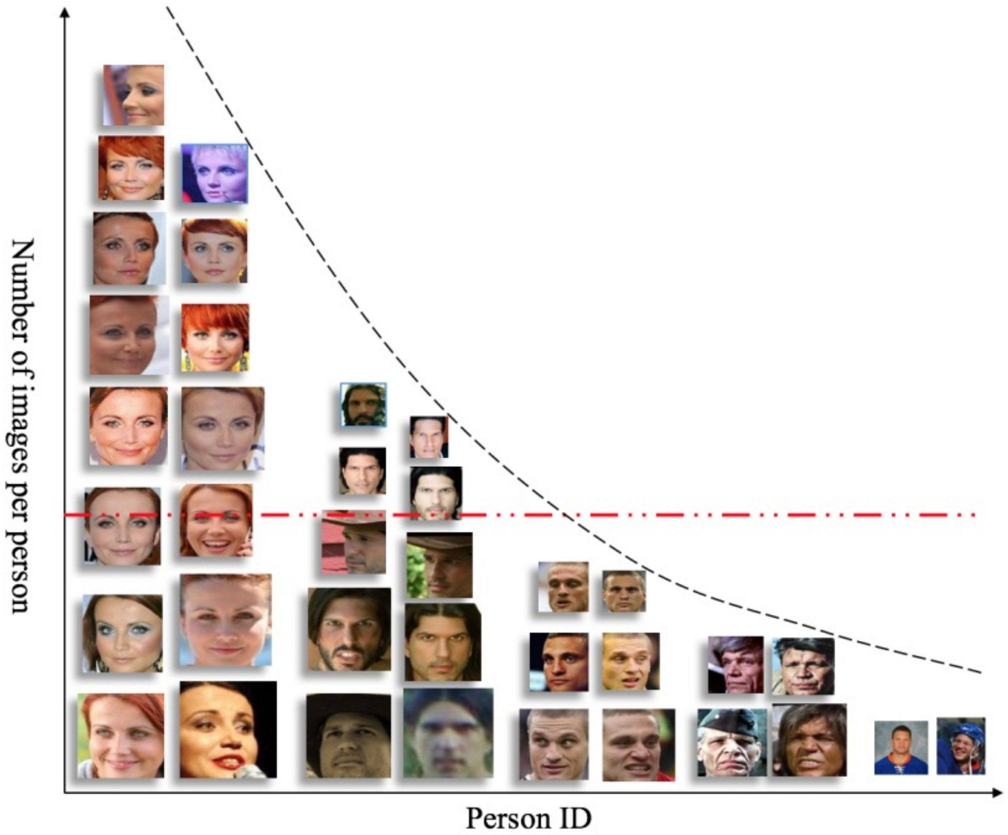
Sukwon Yun, Kibum Kim, Kanghoon Yoon, Chanyoung Park

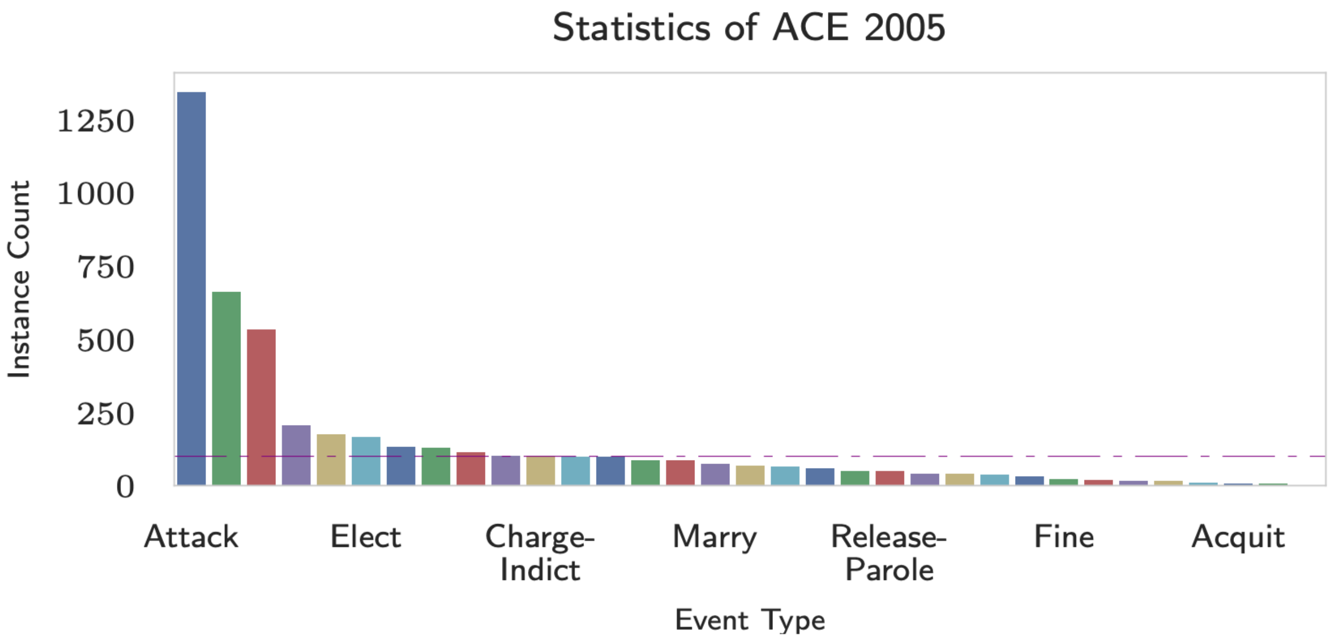
Korea Advanced Institute of Science and Technology (KAIST)

DSAIL Data Science & Artificial Intelligence Lab.

TABLE OF CONTENTS

BACKGROUND: LONG-TAIL PROBLEM





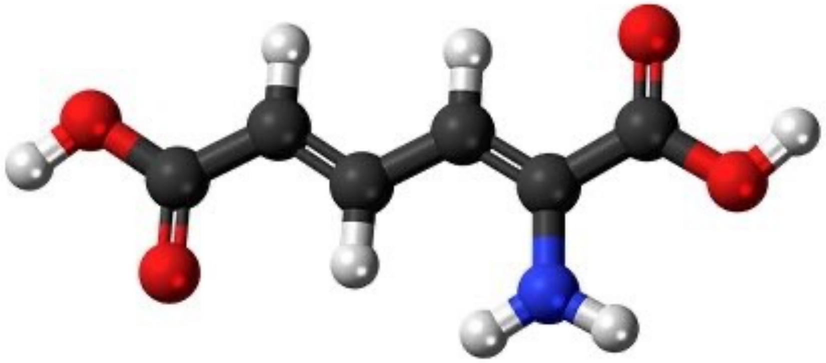
Compared to Head cases, Tail cases are under-represented and thus fail to generalize!

Zhang, Xiao, et al. "Range loss for deep face recognition with long-tailed training data." *Proceedings of the IEEE International Conference on Computer Vision*. 2017.
Liu, Jian, et al. "Event detection via gated multilingual attention mechanism." *Proceedings of the AAAI conference on artificial intelligence*. Vol. 32. No. 1. 2018.

BACKGROUND: GRAPH NEURAL NETWORKS

- **Networks are everywhere!**
  - **Graphs are natural way to model such networks.**

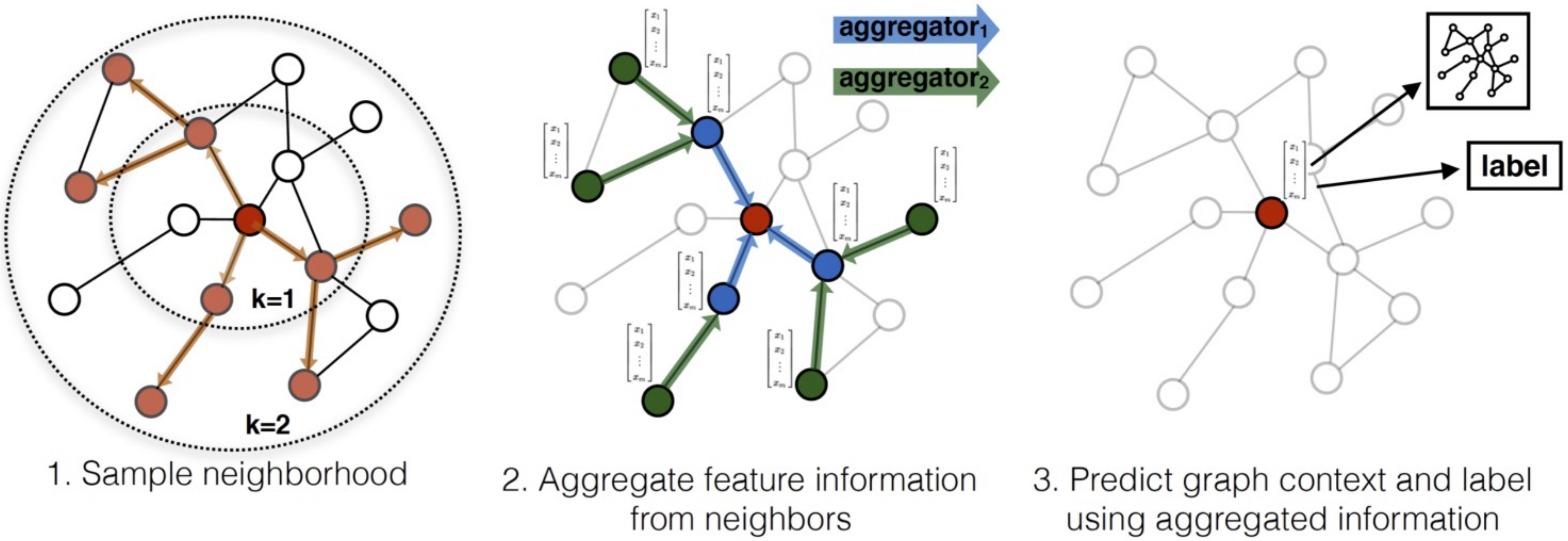
Social Networks



Molecules

BACKGROUND: GRAPH NEURAL NETWORKS

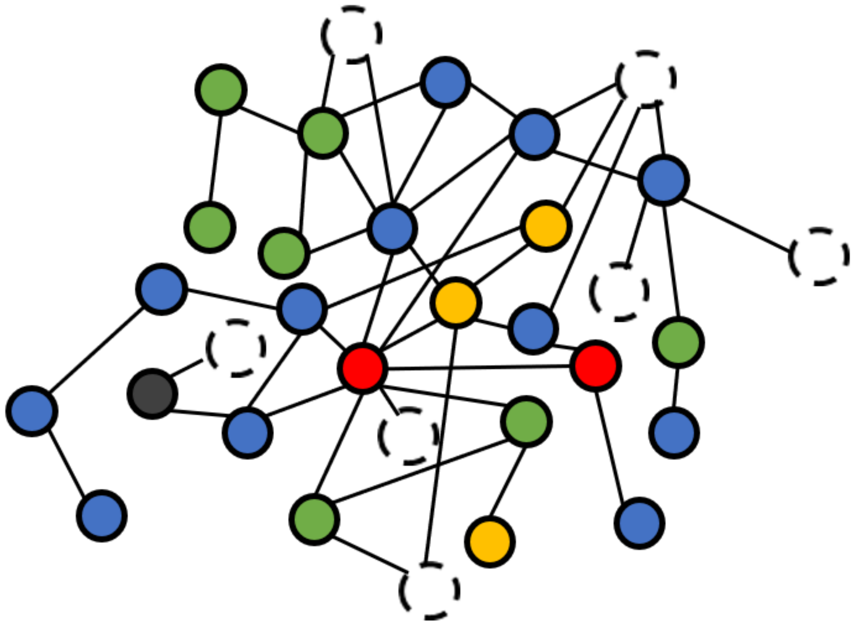
- **Graph Neural Networks (GNNs)**
  - **Compress a set of vectors into a single vector (i.e., node representation)**
  - **Message and Aggregation Scheme**



Hamilton, Will, Zhitao Ying, and Jure Leskovec. "Inductive representation learning on large graphs." *Advances in neural information processing systems* 30 (2017).

BACKGROUND: LONG-TAIL PROBLEM ON GRAPH NEURAL NETWORKS

- **In graph, long-tail problem lies on class and degree perspectives!**

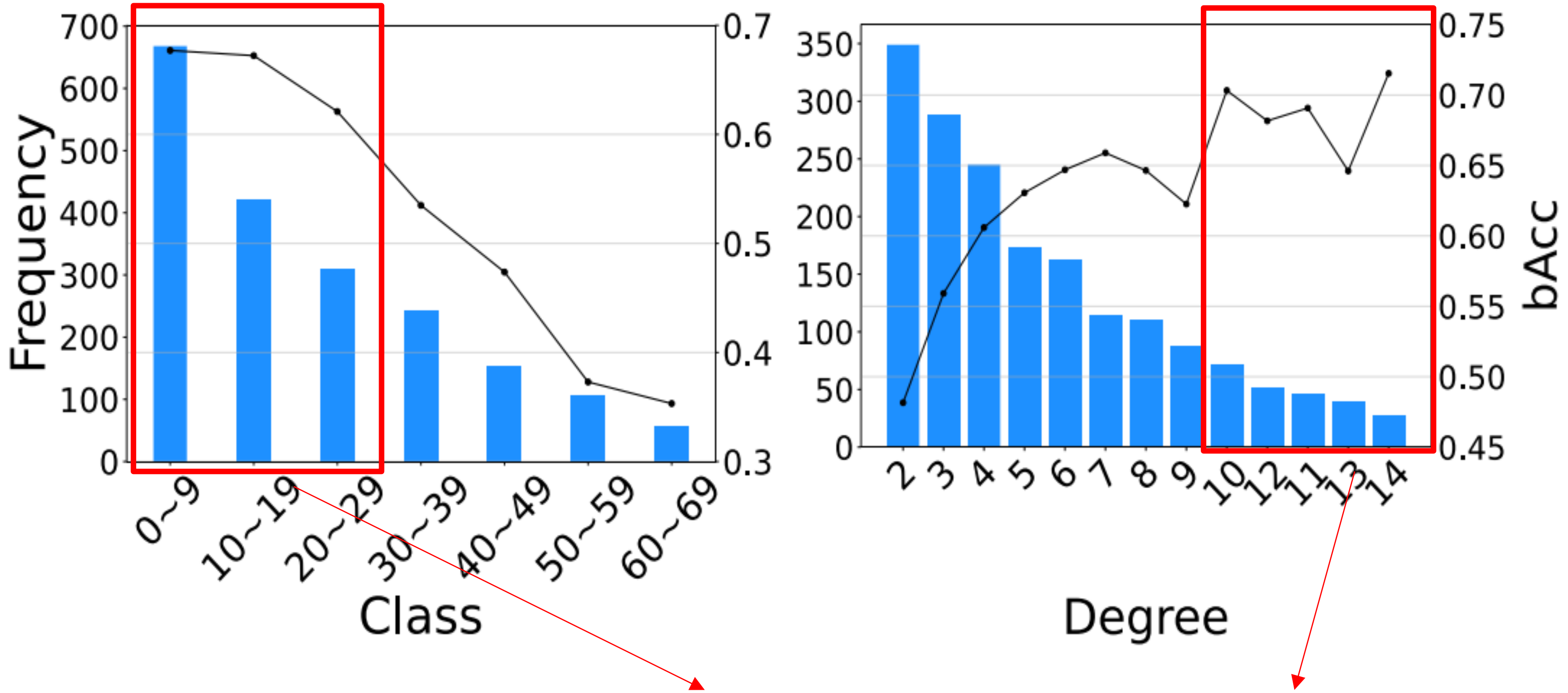


Class Cardinality

Degree Rank

BACKGROUND: LONG-TAIL PROBLEM ON GRAPH NEURAL NETWORKS

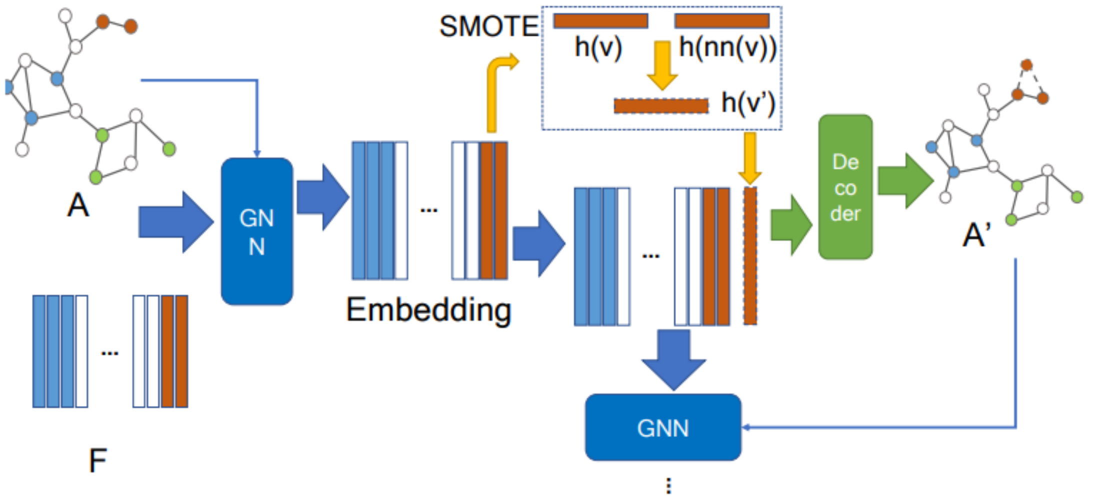
- **In graph, long-tail problem lies on class and degree perspectives!**

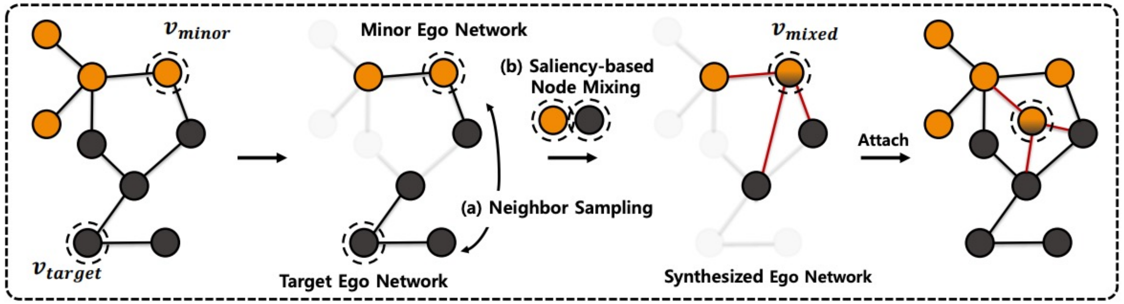


GNNs tend to be biased towards Head Class node and Head Degree node!

BACKGROUND: LONG-TAIL PROBLEM ON GRAPH NEURAL NETWORKS

- **Class Perspective**
  - **GraphSMOTE [1], GraphENS [2] – Oversamples Tail Class nodes**





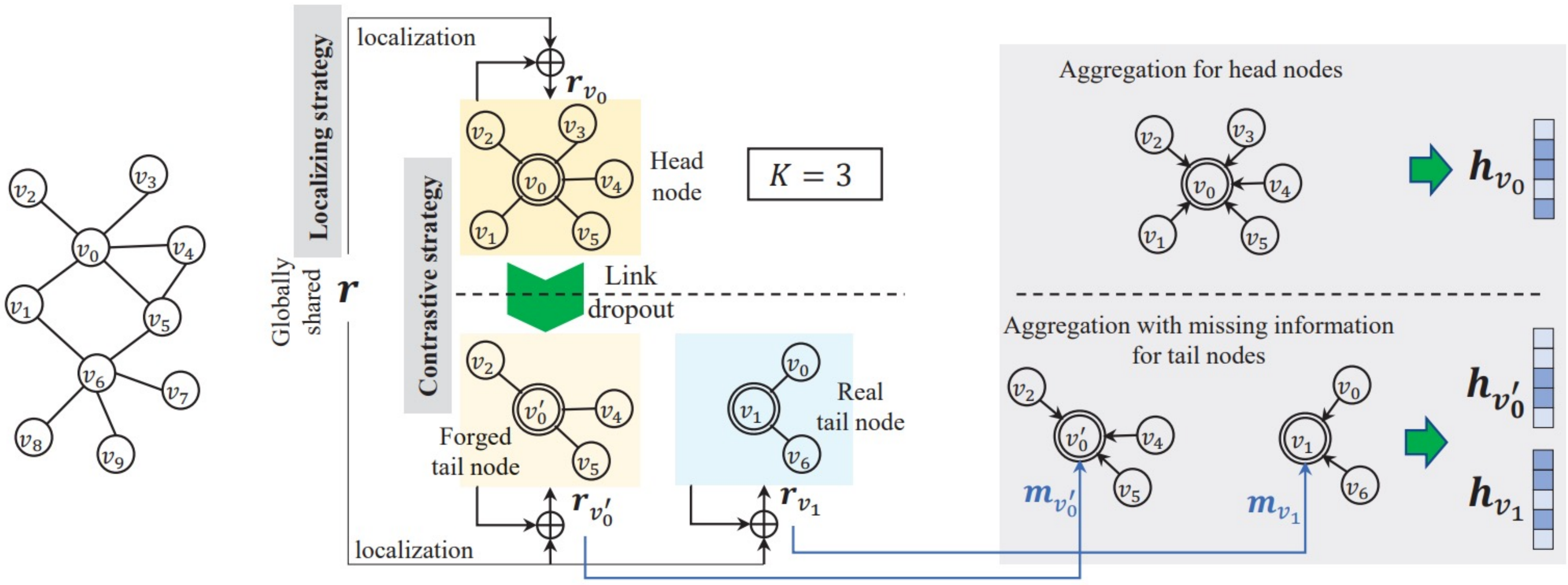
However, the degree long-tailedness has not been considered!

[1] Zhao, Tianxiang, Xiang Zhang, and Suhang Wang. "Graphsmote: Imbalanced node classification on graphs with graph neural networks." *Proceedings of the 14th ACM international conference on web search and data mining*. 2021.

[2] Park, Joonhyung, Jaeyun Song, and Eunho Yang. "GraphENS: Neighbor-Aware Ego Network Synthesis for Class-Imbalanced Node Classification." *International Conference on Learning Representations*. 2021.

BACKGROUND: LONG-TAIL PROBLEM ON GRAPH NEURAL NETWORKS

- **Degree Perspective**
  - **Tail-GNN [1] – Enhance representation of Tail Degree nodes via Head Degree nodes**



(a) Toy network (b) Neighborhood translation for missing information prediction (c) Neighbors aggregation with missing information

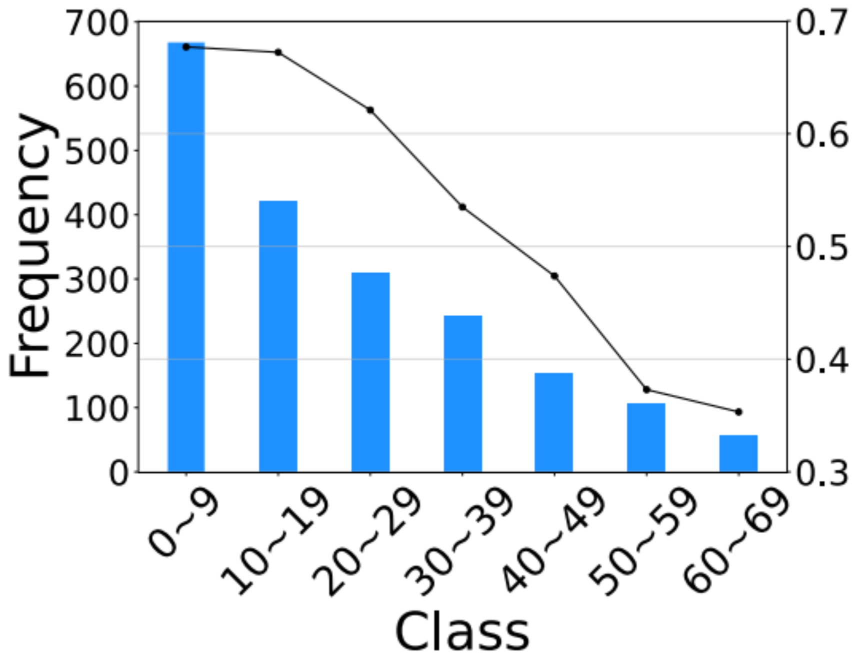
However, the class long-tailedness has not been considered!

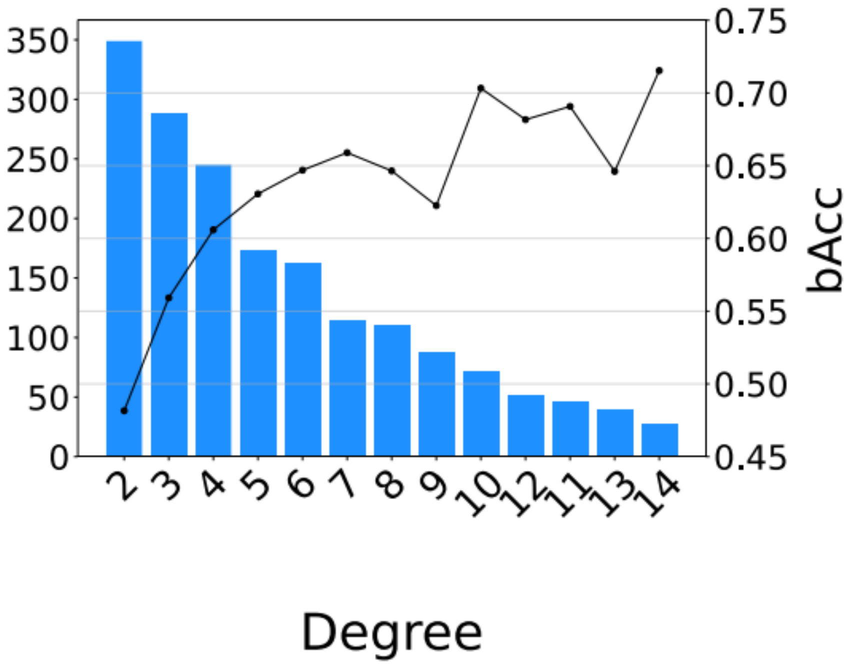
DSAIL@KAIST

[1] Liu, Zemin, Trung-Kien Nguyen, and Yuan Fang. "Tail-gnn: Tail-node graph neural networks." *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2021.

MOTIVATION: CAN WE JOINTLY ADDRESS BOTH LONG-TAILEDNESS?

- **Existing works only consider either class- or degree-longtailedness**
  - ***Can we view the both class- and degree- long-tailedness jointly?***





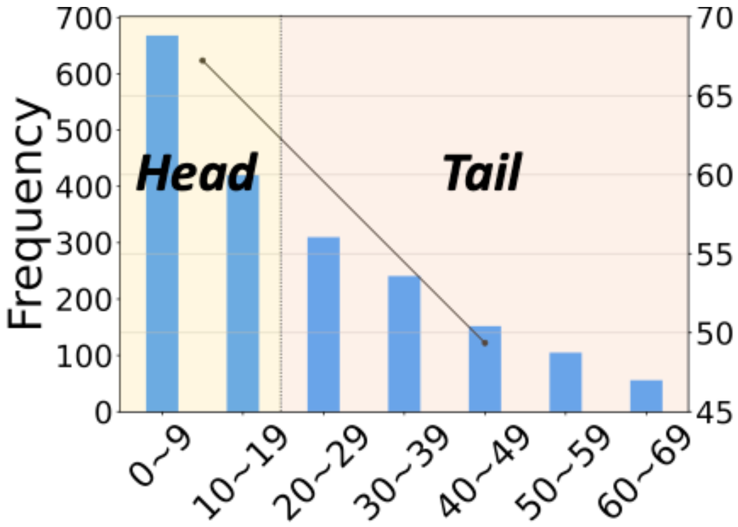
Balanced Accuracy (bAcc) = $\dfrac{(\text{True Positive Rate} + \text{True Negative Rate})}{2}$

Predict

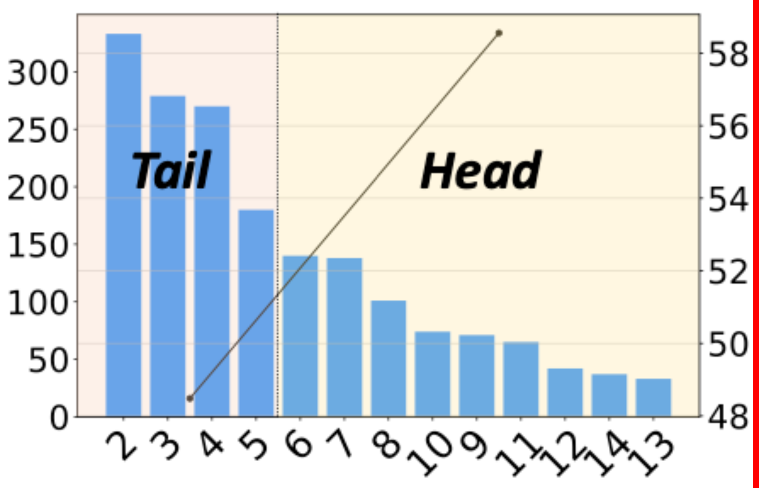
Actual	Positive	Negative
Positive	TP	FN
Negative	FP	TN

MOTIVATION: CAN WE JOINTLY ADDRESS BOTH LONG-TAILEDNESS?

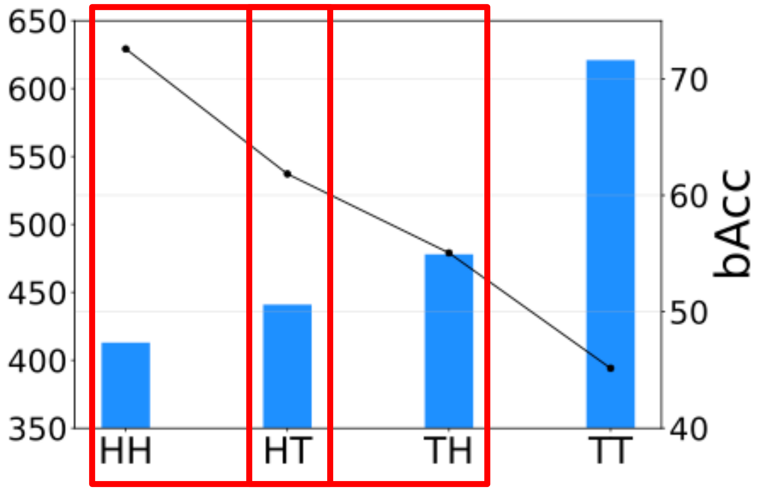
- **Existing works only consider either class- or degree-longtailedness**
  - ***Can we view the both class- and degree- long-tailedness jointly?***



(a) Class



(b) Degree



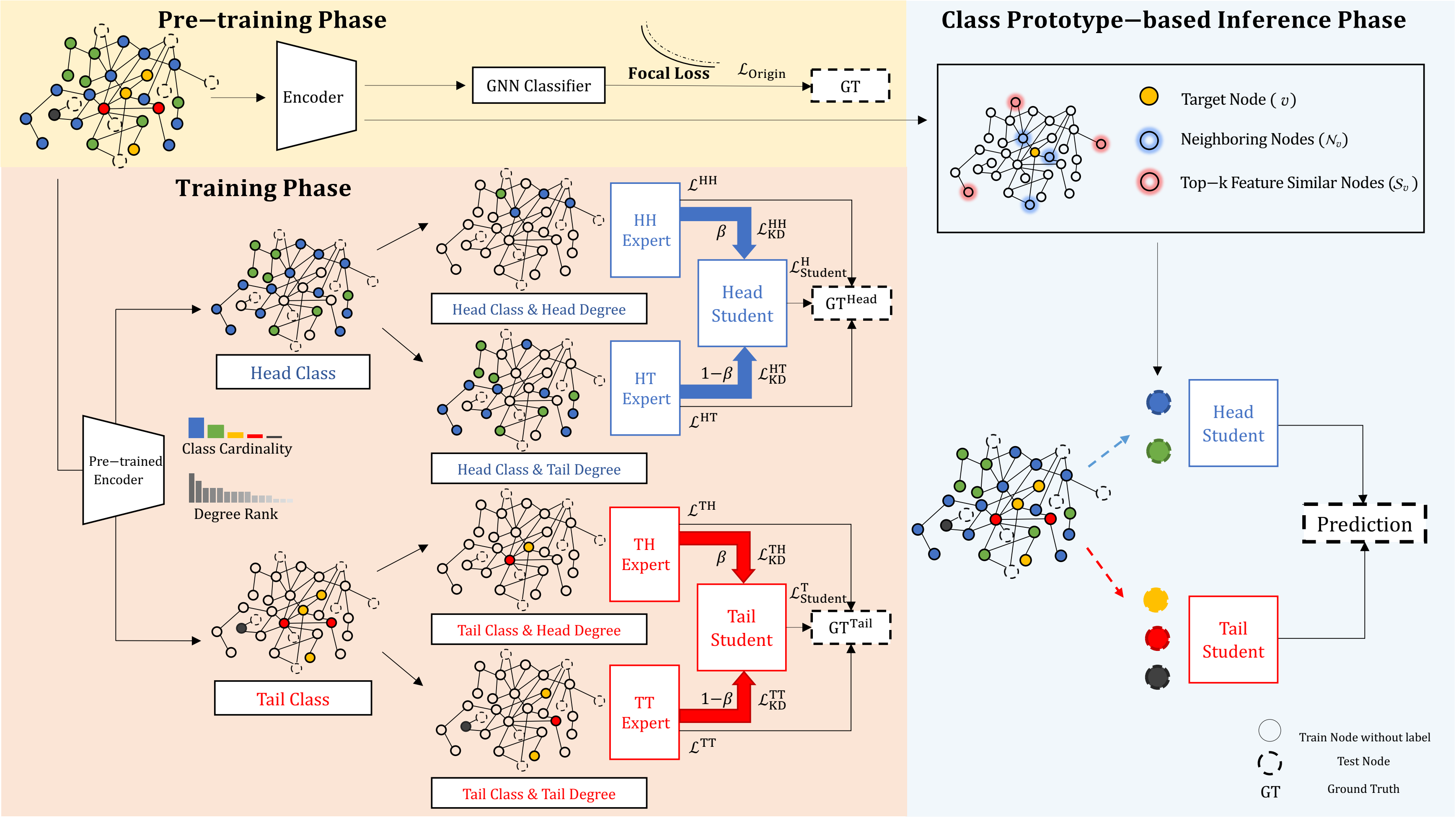
(c) Joint Consideration

Head Class nodes do *not* always generalize well (HH > HT)

Head Degree nodes do *not* always generalize well (HT > TH)

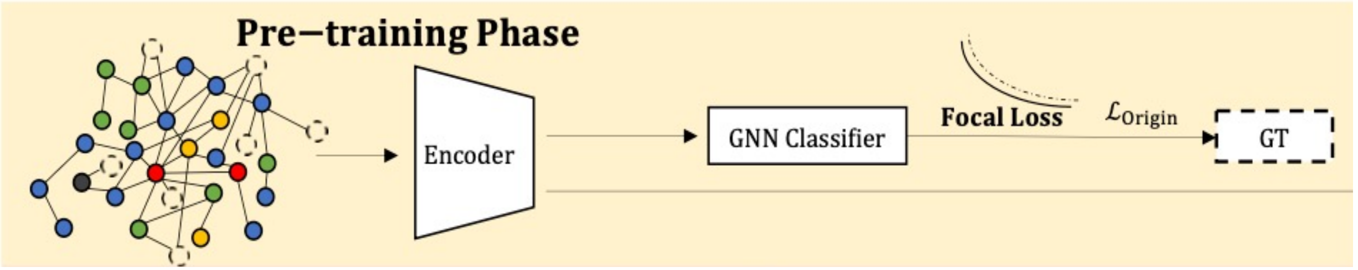
LTE4G: LONG-TAIL EXPERTS FOR GRAPH NEURAL NETWORKS

- **Overall Framework of LTE4G**



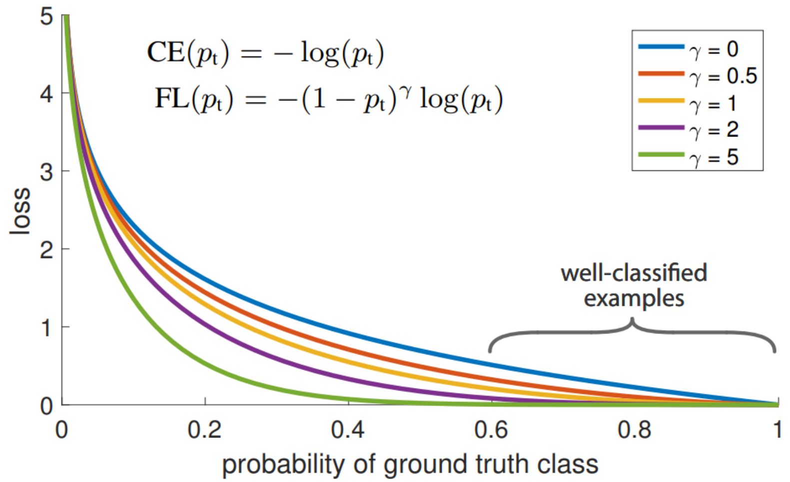
LTE4G: LONG-TAIL EXPERTS FOR GRAPH NEURAL NETWORKS

- **Pre-training Phase**
  - **Obtain Pre-trained Encoder with focal loss**



$$\mathbf{H}^{\text{pre}} = \sigma(\hat{\mathbf{D}}^{-1/2}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-1/2}\mathbf{X}\mathbf{W}^{\text{pre}})$$

$$\mathbf{P}^{\text{og}} = \text{softmax}(\mathbf{Z}^{\text{og}}), \quad \mathbf{Z}^{\text{og}} = \sigma(\hat{\mathbf{D}}^{-1/2}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-1/2}\mathbf{H}^{\text{pre}}\mathbf{W}^{og}_{\text{GNN}})\mathbf{W}^{og}_{\text{MLP}}$$

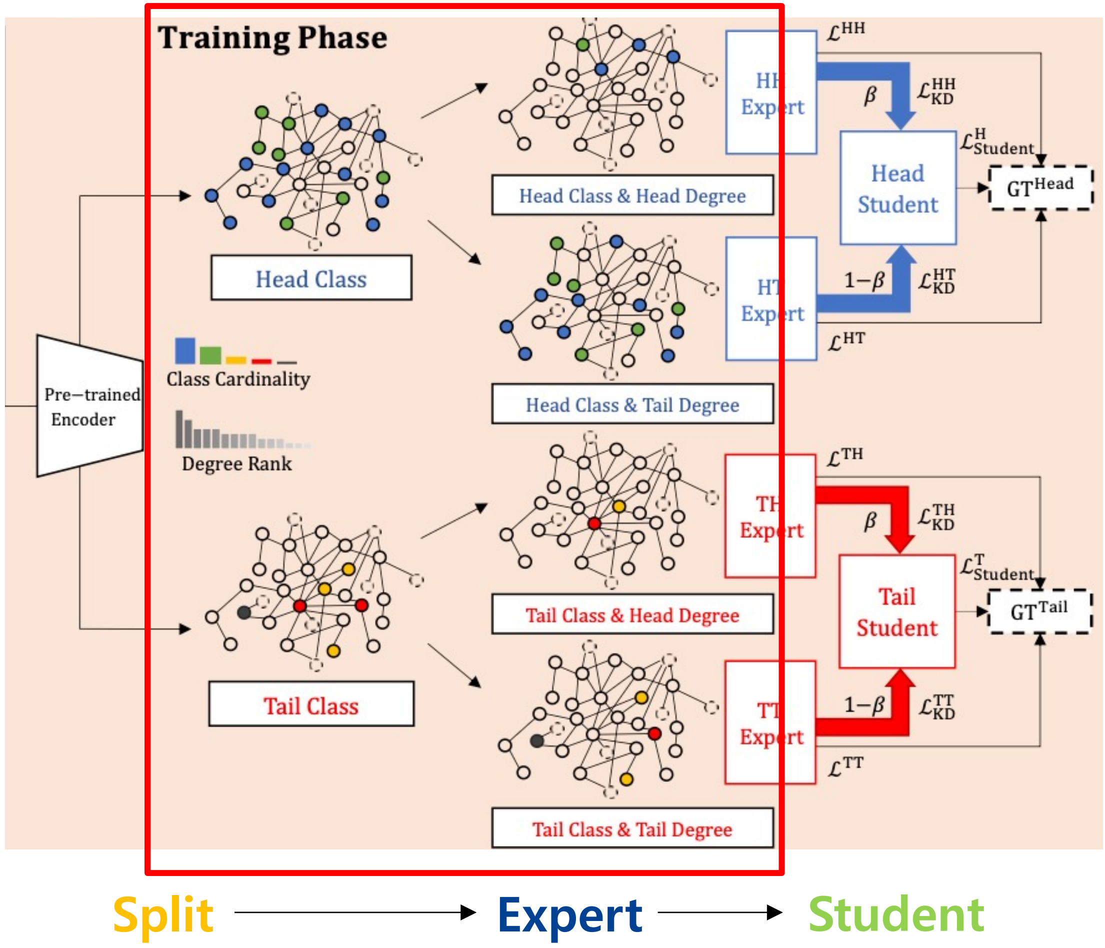


Focal Loss

[1] Lin, Tsung-Yi, et al. "Focal loss for dense object detection." *Proceedings of the IEEE international conference on computer vision*. 2017.

LTE4G: LONG-TAIL EXPERTS FOR GRAPH NEURAL NETWORKS

- **Training Phase**
  - **Split** → Expert → Student



Split

	class	degree
Head	top−p%	Over 5
Tail	100−p%	Less than 5

1) Head Class & Head Degree (HH)

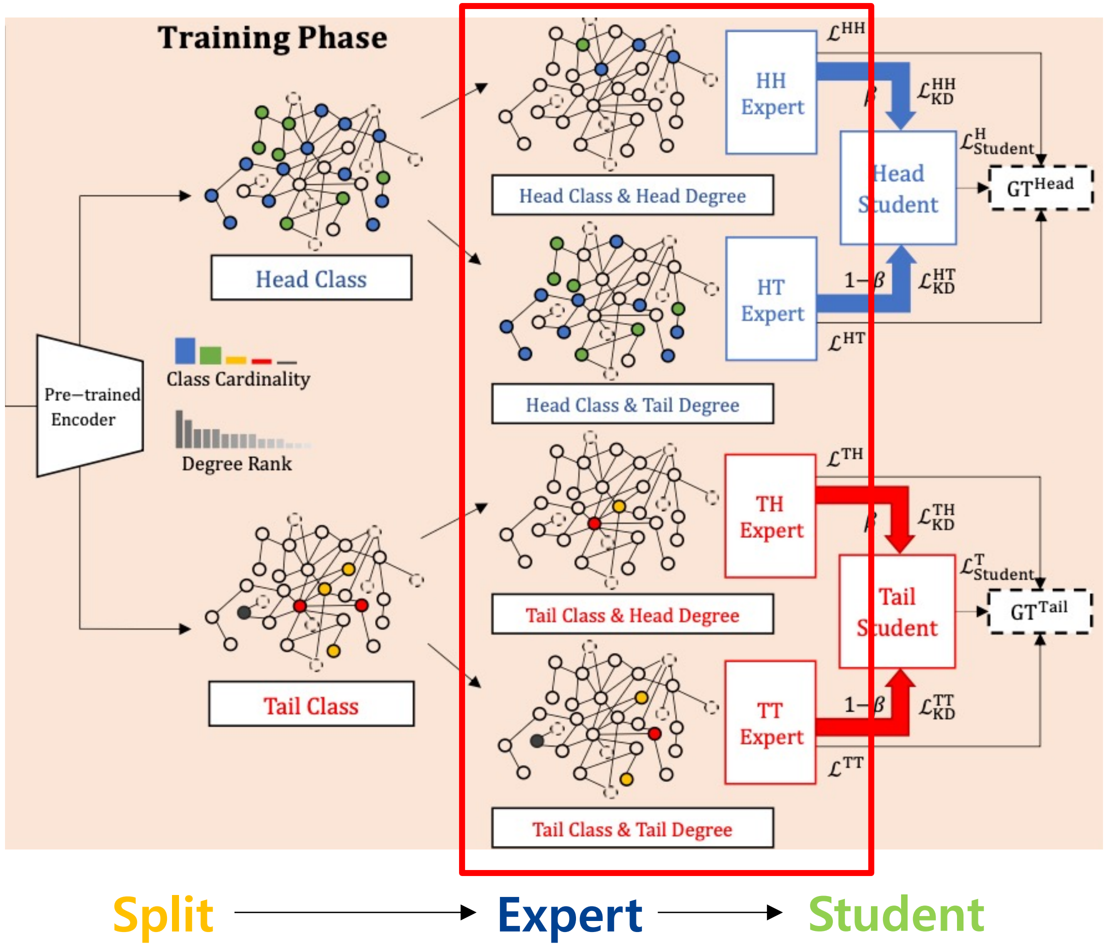
2) Head Class & Tail Degree (HT)

3) Tail Class & Head Degree(TH)

4) Tail Class & Tail Degree (TT)

LTE4G: LONG-TAIL EXPERTS FOR GRAPH NEURAL NETWORKS

- **Training Phase**
  - Split → **Expert** → Student



Expert

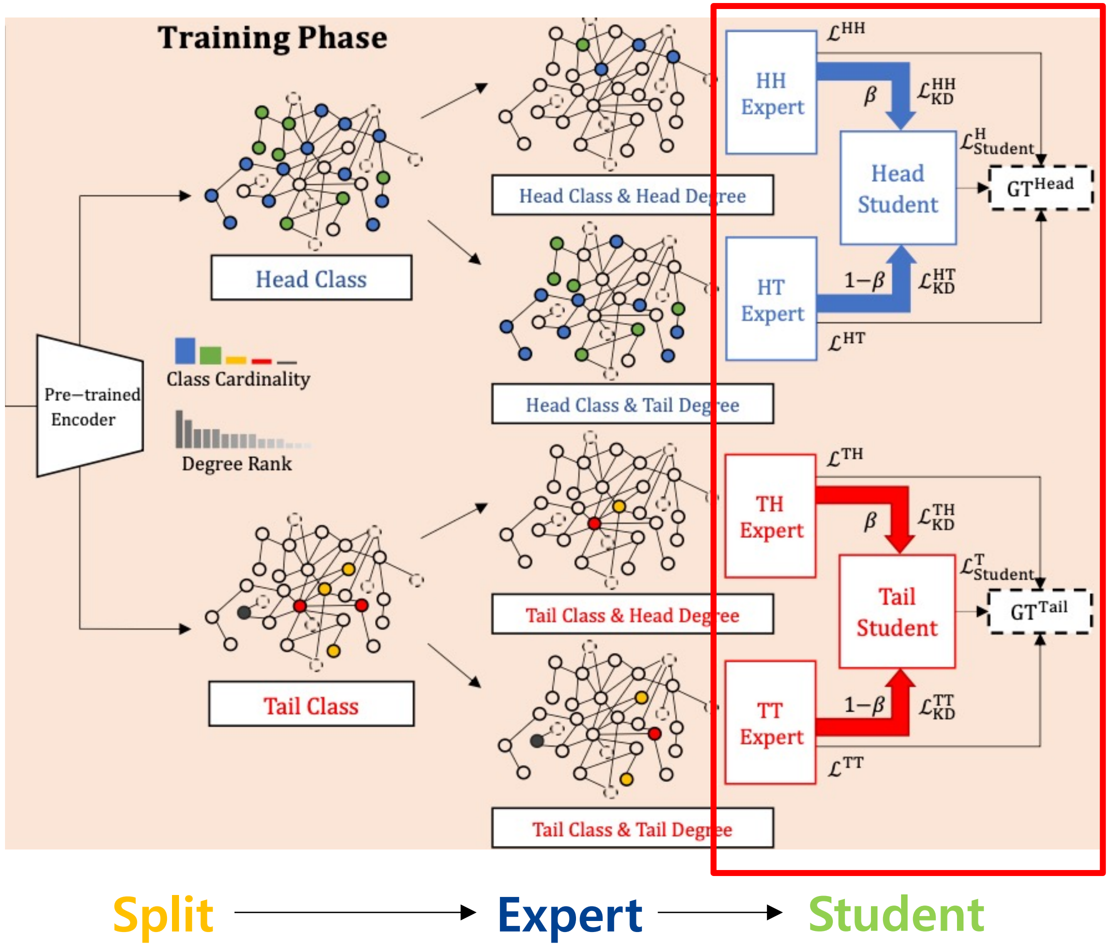
$$\mathbf{P}^* = \mathrm{softmax}(\mathbf{Z}^*), \quad \mathbf{Z}^* = \sigma(\hat{\mathbf{D}}^{-1/2}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-1/2}\mathbf{H}^{\mathrm{pre}}\mathbf{W}^*_{\mathrm{GNN}})\mathbf{W}^*_{\mathrm{MLP}}$$

$$\mathcal{L}^*_{\mathrm{Expert}} = \sum_{v \in V^*} \sum_{c \in C^*} CE(\mathbf{P}^*_v[c])$$

$* \in \{HH, HT, TH, TT\}$

LTE4G: LONG-TAIL EXPERTS FOR GRAPH NEURAL NETWORKS

- **Training Phase**
  - Split → Expert → **Student**



Student

$$\mathbf{P}^{\star} = \mathrm{softmax}(\mathbf{Z}^{\star}), \quad \mathbf{Z}^{\star} = \sigma(\hat{\mathbf{D}}^{-1/2}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-1/2}\mathbf{H}^{\mathrm{pre}}\mathbf{W}^{\star}_{\mathrm{GNN}})\mathbf{W}^{\star}_{\mathrm{MLP}}$$

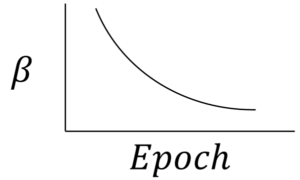
$\star \in \{H, T\}$

- Knowledge Distillation from Expert to Student

$$\mathcal{L}^{\mathrm{HH}}_{\mathrm{KD}} = D_{\mathrm{KL}}[\mathbf{P}^{\mathrm{HH}} \| \mathbf{P}^{\mathrm{H}}], \quad \mathcal{L}^{\mathrm{HT}}_{\mathrm{KD}} = D_{\mathrm{KL}}[\mathbf{P}^{\mathrm{HT}} \| \mathbf{P}^{\mathrm{H}}]$$

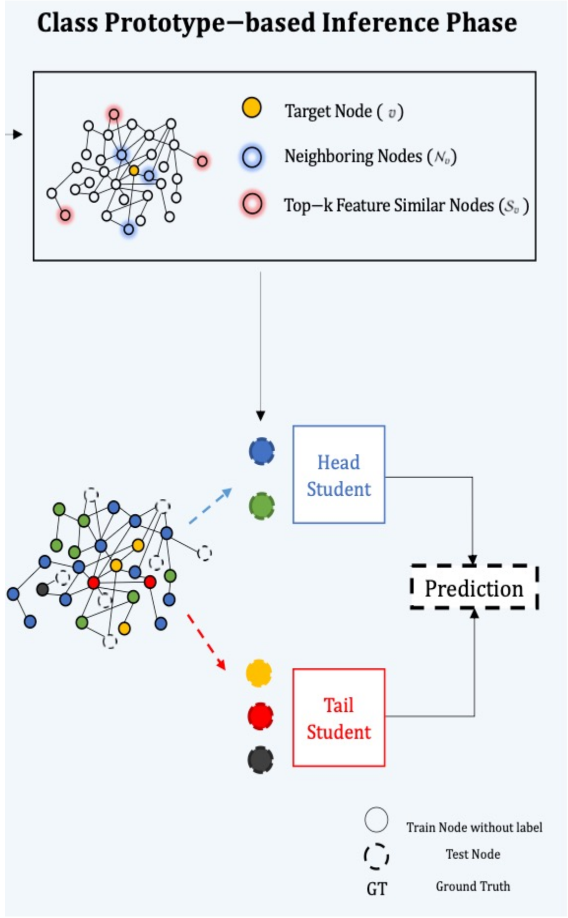
- Head-to Tail Curriculum Learning

$$\mathcal{L}^{\mathrm{H}}_{\mathrm{Student}} = \beta \mathcal{L}^{\mathrm{HH}}_{\mathrm{KD}} + (1-\beta)\,\mathcal{L}^{\mathrm{HT}}_{\mathrm{KD}}$$



LTE4G: LONG-TAIL EXPERTS FOR GRAPH NEURAL NETWORKS

- **Class Prototype-based Inference Phase**
  - **Generate Class Prototype by expanded candidates**
  - **Assign test node on certain Student based on its similarity between class prototypes**



- **Candidates for class prototype**

$$v^c_{\text{train}} \in \mathcal{V}^c_{\text{train}}$$

Neighboring Nodes

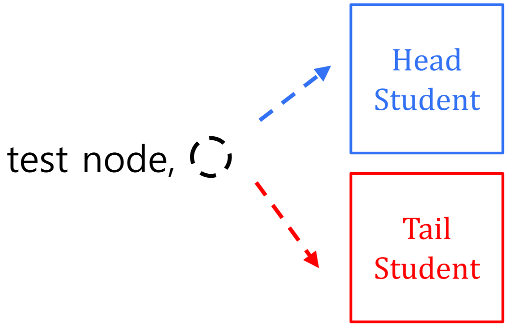
$$\mathcal{N}^c_{\text{train}} = \bigcup_{v^c_{\text{train}} \in \mathcal{V}^c_{\text{train}}} \mathcal{N}_{v^c_{\text{train}}}$$

Nodes with similar features

$$\mathcal{S}^c_{\text{train}} = \bigcup_{v^c_{\text{train}} \in \mathcal{V}^c_{\text{train}}} \mathcal{S}_{v^c_{\text{train}}} \qquad S = \frac{X \cdot X^T}{\|X\| \cdot \|X^T\|}$$

$$\mathcal{V}^c_{\text{train}} \leftarrow \{\mathcal{V}^c_{\text{train}} \cup \mathcal{N}^c_{\text{train}} \cup \mathcal{S}^c_{\text{train}}\} \qquad \mathrm{p}^c = \frac{1}{|\mathcal{V}^c_{\text{train}}|} \sum_{v^c_{\text{train}} \in \mathcal{V}^c_{\text{train}}} \mathrm{H}^{\text{pre}}_{v^c_{\text{train}}}$$

$$c^* = \arg\max_c \, \mathrm{sim}(\mathrm{p}_c, \mathrm{H}^{\text{pre}}_{v_{\text{test}}}), \ \forall c \in C$$

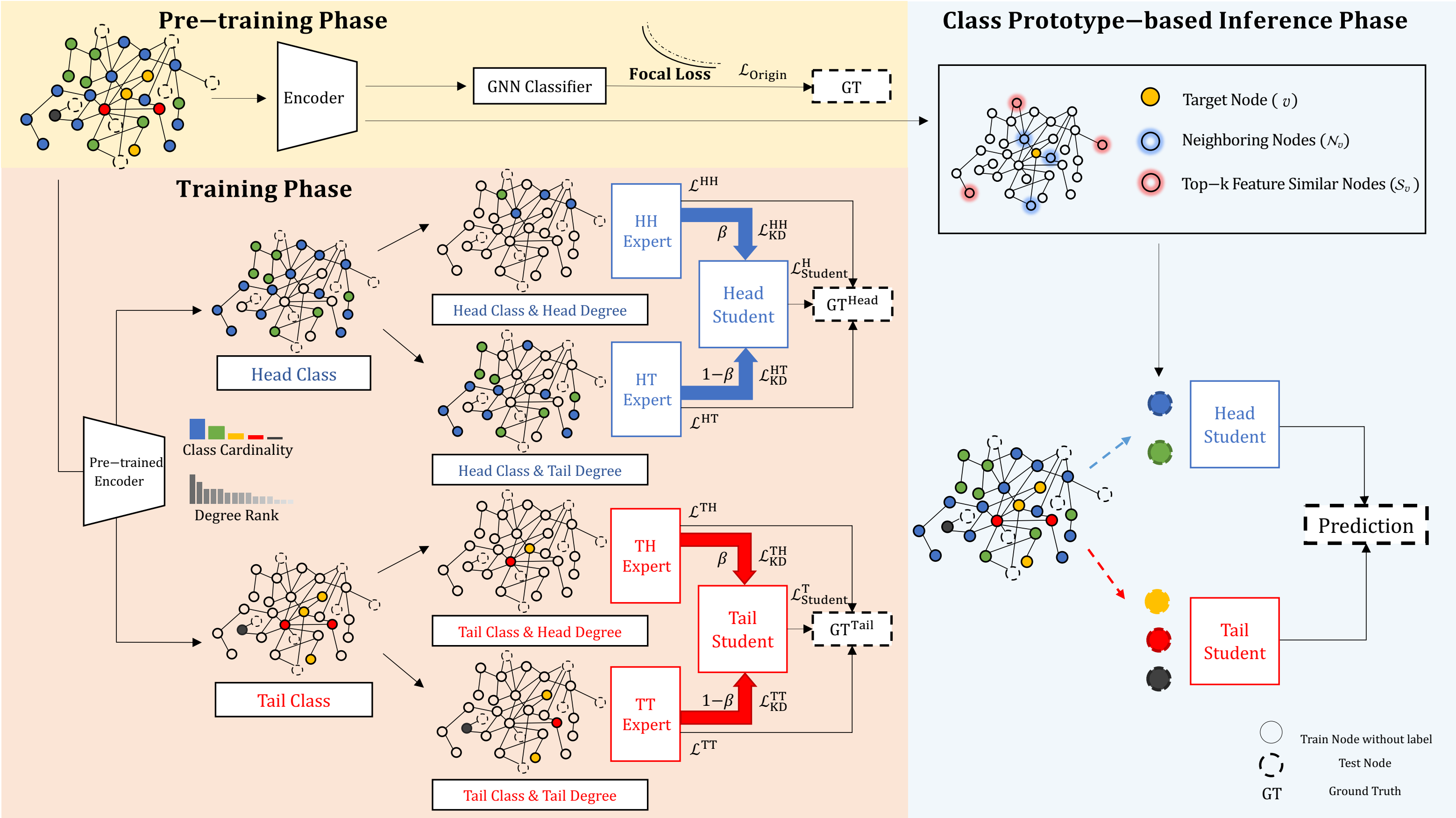


if $c^* \in C^{\mathrm{H}}$

if $c^* \in C^{\mathrm{T}}$

LTE4G: LONG-TAIL EXPERTS FOR GRAPH NEURAL NETWORKS

- **In a nutshell,**



Pre-training Phase

- Obtain a Pre-trained Encoder

Training Phase

- Split nodes in a balanced manner
- Obtain Experts and Students
- Using Knowledge Distillation
- Using Head-to-Tail Curriculum Learning

Class Prototype-based Inference Phase

- Generate Class Prototype
- Assign each test node to a student

EXPERIMENTS

- **Data Statistics & Evaluation Metrics**

Dataset	#Nodes	#Edges	#Features	#Classes
Cora	2,708	5,429	1,433	7
CiteSeer	3,327	4,732	3,703	6
Cora-Full	19,793	146,635	8,710	70

Dataset	Imb. class	Imb. ratio	L_0	L_1	L_2	L_3	L_4	L_5	L_6
Cora	3	10%	23.3	23.3	23.3	23.3	2.4	2.4	2.4
		5%	24.1	24.1	24.1	24.1	1.2	1.2	1.2
	5	10%	40.0	40.0	4.0	4.0	4.0	4.0	4.0
		5%	44.4	44.4	2.2	2.2	2.2	2.2	2.2
	LT	1%	54.0	25.0	11.6	5.4	2.4	1.2	0.5
CiteSeer	3	10%	30.3	30.3	30.3	3.0	3.0	3.0	-
		5%	31.7	31.7	31.7	1.6	1.6	1.6	-
	5	10%	66.7	6.7	6.7	6.7	6.7	6.7	-
		5%	80.0	4.0	4.0	4.0	4.0	4.0	-
	LT	1%	60.7	24.1	9.5	3.8	1.5	0.5	-
Cora-Full	-	1.1%	34.0	18.9	14.1	10.9	6.9	4.8	2.6

Predict

Actual	Positive	Negative
Positive	TP	FN
Negative	FP	TN

Balanced Accuracy (bAcc.) = $\frac{(\text{True Positive Rate} + \text{True Negative Rate})}{2}$

F1-Score = $2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$

Geometric Means (G-Means) = $\sqrt{\text{True Positive Rate} * \text{True Negative Rate}}$

Accuracy (Acc.) = $\frac{\text{TP+TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}}$

EXPERIMENTS

- **Performance on Node Classification**

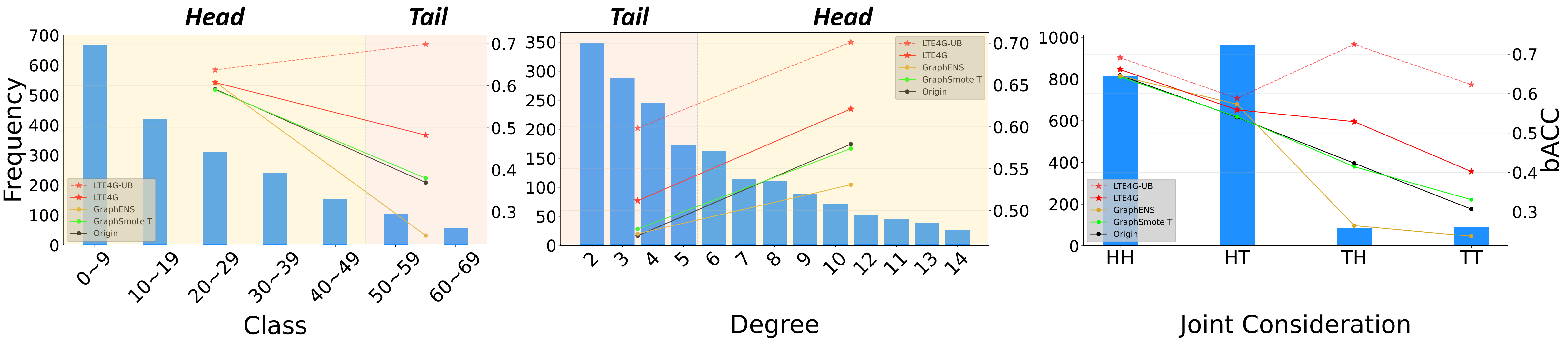
Dataset	Method	Imb. class num: 3						Imb. class num: 5					
		Imbalance_ratio: 10%			*Imbalance_ratio: 5%*			*Imbalance_ratio: 10%*			*Imbalance_ratio: 5%*		
		bAcc.	Macro-F1	G-Means	bAcc.	Macro-F1	G-Means	bAcc.	Macro-F1	G-Means	bAcc.	Macro-F1	G-Means
Cora	Origin	68.8±4.0	67.6±5.0	80.8±2.6	60.0±0.4	56.6±0.7	74.8±0.3	64.9±6.2	64.7±5.7	78.1±4.2	55.1±2.6	51.4±2.4	71.4±1.8
	Over-sampling	65.6±4.3	63.4±5.6	78.6±2.9	59.0±2.2	53.9±2.6	74.2±1.5	58.9±5.6	56.9±7.6	74.0±3.9	49.1±4.1	45.6±5.4	67.0±3.1
	Re-weight	70.6±4.3	69.9±5.1	81.9±2.8	60.8±1.8	56.7±2.4	75.4±1.3	65.2±7.6	65.0±8.2	78.3±5.2	57.9±4.3	54.8±5.5	73.3±3.0
	SMOTE	65.1±4.0	62.3±5.1	78.3±2.7	59.0±2.2	53.9±2.6	74.2±1.5	60.3±7.6	58.7±8.9	74.9±5.3	49.1±4.1	45.6±5.4	67.0±3.1
	Embed-SMOTE	61.0±3.6	58.0±5.5	75.5±2.5	55.5±2.1	50.0±3.1	71.7±1.5	53.2±5.2	50.9±6.4	70.0±3.8	40.7±2.8	36.5±3.0	60.5±2.2
	GraphSMOTE$_T$	70.0±3.4	68.6±4.9	81.6±2.2	62.5±1.8	58.7±2.1	76.5±1.2	66.3±6.6	65.3±7.7	79.0±4.5	55.8±5.6	52.4±4.7	71.9±4.0
	GraphSMOTE$_O$	67.3±3.8	65.5±5.0	79.7±2.5	61.0±0.6	58.4±0.7	75.5±0.4	62.8±6.1	62.0±6.1	76.7±4.2	59.6±4.3	55.9±4.2	74.5±3.0
	GraphSMOTE$_{preT}$	70.6±6.3	68.9±7.9	81.9±4.1	67.5±5.7	64.5±8.0	79.9±3.8	69.4±5.2	68.3±5.2	81.1±3.4	66.0±6.0	63.4±6.5	78.8±4.0
	GraphSMOTE$_{preO}$	69.8±5.7	67.9±7.1	81.4±3.7	67.2±5.2	64.0±7.3	79.6±3.5	69.1±7.7	67.8±7.9	80.9±5.1	66.8±4.9	64.6±4.9	79.4±3.2
	GraphENS	59.3±7.0	55.4±10.6	74.2±4.9	55.1±4.9	48.1±7.9	71.3±3.5	44.3±6.5	41.0±7.0	63.3±5.0	36.1±10.1	31.1±12.3	56.3±8.4
	Tail-GNN	64.6±3.6	62.0±5.4	77.9±2.4	57.2±1.5	51.8±1.8	72.9±1.1	55.9±4.5	54.0±5.0	71.9±3.2	41.7±1.4	36.8±2.8	61.4±1.1
	LTE4G	**73.2**±5.4	**72.1**±6.1	**83.6**±3.5	**70.9**±2.5	**69.6**±2.8	**82.1**±1.6	**75.4**±5.6	**75.4**±5.4	**85.0**±3.6	**70.2**±4.5	**68.8**±4.7	**81.7**±3.0
CiteSeer	Origin	49.5±2.1	43.1±2.3	66.7±1.5	48.2±0.9	39.3±0.4	65.7±0.7	48.9±1.4	45.3±1.3	66.2±1.1	42.4±6.5	39.1±7.3	61.1±5.1
	Over-sampling	51.5±3.0	43.7±2.1	68.2±2.2	47.8±0.8	38.9±1.9	65.4±0.6	43.0±3.4	40.3±1.7	61.7±2.7	40.1±2.0	34.2±1.5	59.4±1.6
	Re-weight	52.1±2.7	46.2±3.2	68.6±2.0	48.0±0.4	39.2±1.1	65.6±0.3	48.4±3.9	44.5±3.9	65.8±2.9	41.3±4.5	35.6±5.3	60.3±3.6
	SMOTE	48.7±2.5	40.1±1.8	66.1±1.9	47.8±0.8	38.9±1.9	65.4±0.6	44.9±4.4	41.9±4.1	63.2±3.4	40.1±2.0	34.2±1.5	59.4±1.6
	Embed-SMOTE	47.5±2.1	37.9±1.7	65.2±1.6	46.7±3.0	35.7±2.8	64.5±2.3	43.2±6.5	38.3±5.8	61.8±5.2	33.2±6.6	28.3±7.9	53.4±5.9
	GraphSMOTE$_T$	51.2±3.7	43.4±4.2	67.9±2.8	49.3±2.0	40.1±1.3	66.5±1.5	50.3±5.0	46.1±4.5	67.2±3.7	46.5±3.7	**41.5**±4.1	64.4±2.9
	GraphSMOTE$_O$	52.7±2.3	45.3±2.8	69.1±1.7	49.5±2.6	40.3±1.8	66.7±2.0	49.5±3.5	44.5±2.9	66.7±2.6	42.3±6.6	36.9±6.6	61.0±5.3
	GraphSMOTE$_{preT}$	44.7±1.7	37.3±2.1	63.1±1.3	48.2±3.9	39.4±4.9	65.7±3.0	41.8±4.1	39.5±4.1	60.7±3.3	38.0±2.6	33.6±2.5	57.7±2.1
	GraphSMOTE$_{preO}$	44.1±2.0	36.6±1.7	62.6±1.6	45.7±2.6	37.1±3.1	63.8±2.0	43.4±6.6	42.9±6.3	62.7±5.1	39.2±1.8	34.7±2.4	58.7±1.5
	GraphENS	44.2±3.5	35.9±1.0	62.7±2.7	43.5±2.6	33.4±1.9	62.1±2.1	33.0±3.2	28.6±4.4	53.4±2.9	28.5±6.7	23.1±6.2	49.1±6.2
	Tail-GNN	48.8±1.9	40.4±2.9	66.2±1.5	48.2±1.7	39.4±1.2	65.7±1.3	42.4±6.1	38.9±6.1	61.1±4.8	34.2±4.8	28.2±4.1	54.4±4.2
	LTE4G	**54.2**±4.5	**51.8**±4.1	**70.2**±3.3	**52.7**±2.1	**48.3**±3.7	**69.1**±1.5	**52.1**±3.7	**47.2**±3.6	**68.6**±2.7	**47.3**±1.1	41.2±2.1	**65.0**±0.9

Method	Cora-Full			
	bAcc.	Macro-F1	G-Means	Acc.
Origin	52.0±1.0	52.5±0.8	71.9±0.7	60.5±0.2
Over-sampling	51.4±1.0	52.4±0.9	71.5±0.7	60.9±0.3
Re-weight	52.1±0.9	52.6±0.7	72.0±0.6	60.7±0.1
SMOTE	52.0±0.7	52.6±0.6	71.9±0.5	60.7±0.1
Embed-SMOTE	52.3±0.7	**53.8**±0.7	72.1±0.5	62.6±0.5
GraphSMOTE$_T$	52.1±0.9	52.4±0.7	72.0±0.6	60.6±0.3
GraphSMOTE$_O$	52.0±0.9	52.4±0.8	71.9±0.6	60.7±0.5
GraphSMOTE$_{preT}$	48.0±2.1	48.4±2.2	69.0±1.5	56.8±1.9
GraphSMOTE$_{preO}$	47.7±1.7	47.7±1.6	68.8±1.3	56.3±1.5
GraphENS	49.6±0.6	51.5±0.5	70.2±0.4	**62.5**±0.3
Tail-GNN	OOM	OOM	OOM	OOM
LTE4G	**54.2**±0.7	53.0±0.4	**73.4**±0.5	60.9±0.5

LTE4G performs well on both the manual and natural imbalanced settings

EXPERIMENTS

- **Performance on each class, degree and joint consideration**



LTE4G outperforms other baselines on Class Separation, Degree Separation, and their Joint Consideration

DSAIL@KAIST

EXPERIMENTS

- **Ablation of key components & Importance of Balanced Splits**

Components						CiteSeer-10% (Imb. Class 5)			CiteSeer-5% (Imb. Class 5)		
#	*C*	*D*	*KD*	*T2H*	*H2T*	bAcc.	Macro-F1	G-Means	bAcc.	Macro-F1	G-Means
(a)	✓					51.5±4.2	47.1±4.0	68.2±3.1	43.5±0.5	38.5±1.8	62.1±0.4
(b)		✓				39.6±4.6	34.7±5.8	58.9±3.8	29.8±2.5	24.1±2.4	50.6±2.3
(c)	✓	✓				45.6±2.3	41.1±3.2	63.7±1.8	37.6±5.7	32.9±5.8	57.2±4.8
(d)	✓	✓	✓			50.7±3.3	45.5±2.8	67.6±2.5	44.9±3.6	39.4±1.5	63.2±2.8
(e)	✓	✓	✓	✓		50.5±2.8	45.9±2.0	67.4±2.1	46.5±3.1	41.9±3.7	64.4±2.3
(f)	✓	✓	✓		✓	**52.1±3.7**	**47.2±3.6**	**68.6±2.7**	**47.3±1.1**	**41.2±2.1**	**65.0±0.9**

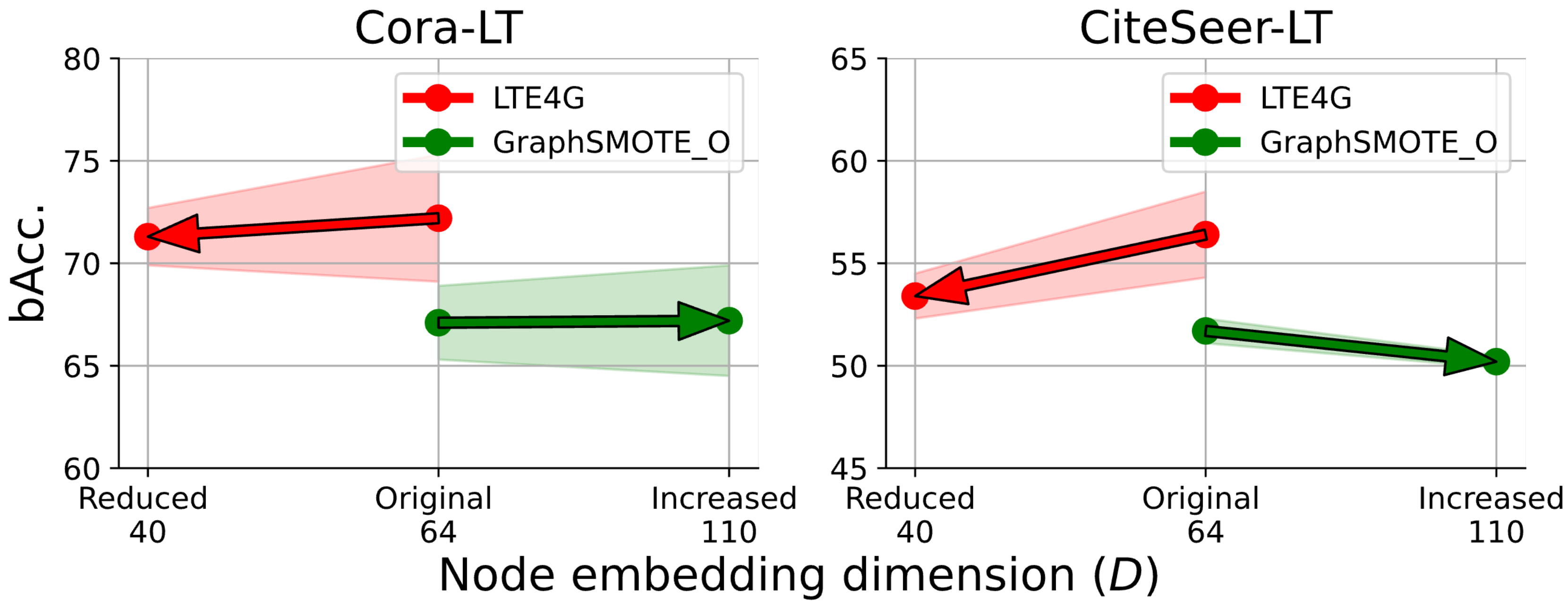
Considering both the class and degree long-tailedness with knowledge distillation scheme is effective

Balanced Split		Cora-5% (Imb. Class 3)			CiteSeer-5% (Imb. Class 3)		
Class	Degree	bAcc.	Macro-F1	G-Means	bAcc.	Macro-F1	G-Means
✗	✗	51.9±3.5	44.4±4.8	69.1±2.5	38.1±1.9	26.6±0.5	57.7±1.5
✗	✓	47.4±2.0	36.7±2.2	65.8±1.5	38.8±1.7	27.0±1.0	58.3±1.4
✓	✗	69.4±2.6	68.2±2.8	81.2±1.7	52.2±1.6	48.2±3.1	68.7±1.2
✓	✓	**70.9**±2.5	**69.6**±2.8	**82.1**±1.6	**52.7**±2.1	**48.3**±3.7	**69.1**±1.5

The beauty of alleviating long-tailedness comes in where the both class and degree long-tailedness is jointly considered

EXPERIMENTS

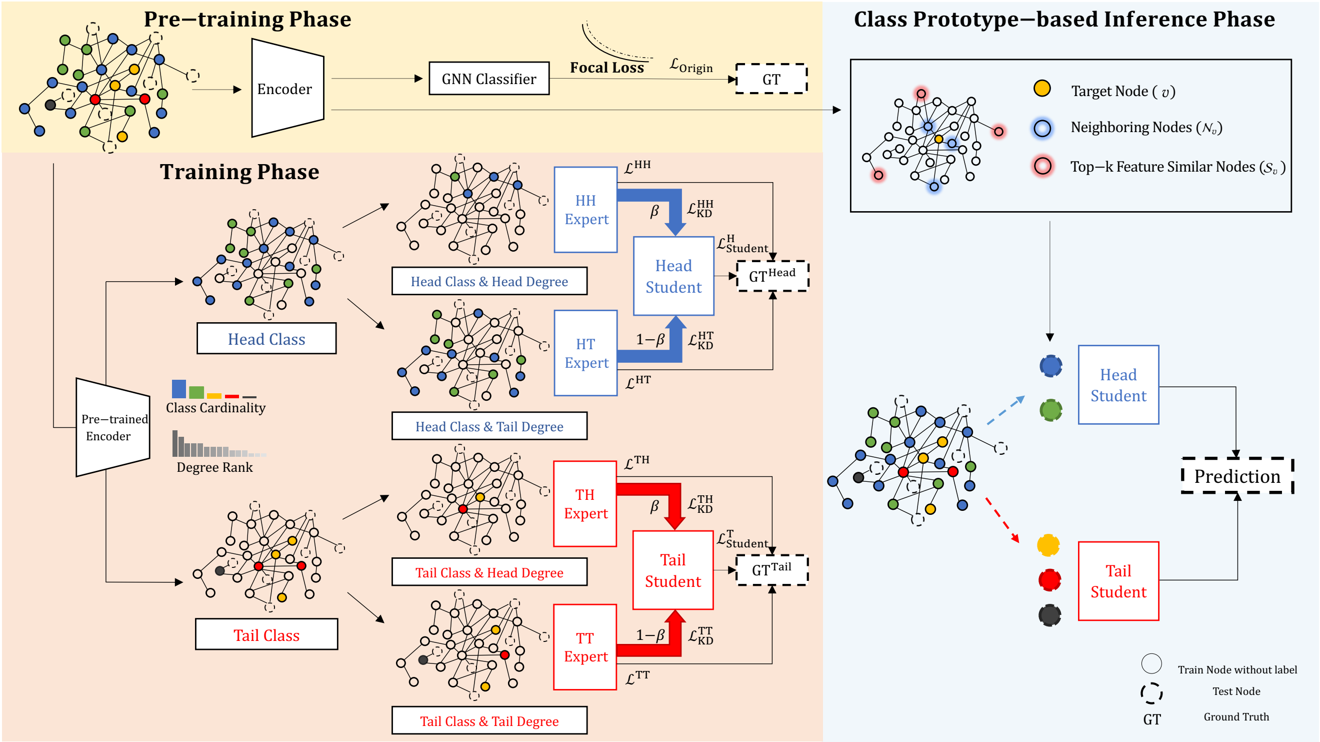
- **Complexity Analysis**



It is important to assign parameters in the right place where they are needed

CONCLUSION

- Existing GNNs assume balanced situation where both the class and degree distributions are balanced.
- However, in real-world scenarios, we often encounter long-tail problem (i.e., Head dominates).
- Recent Studies focused on either the class or degree long-tailedness.
- To this end, we propose LTE4G, which jointly alleviates the class and degree long-tailedness.
- *Keywords for LTE4G*
  - Split (in a balanced manner)
  - Experts (for joint subsets)
  - Students (in a class-wise manner)
  - Class-Prototypes (for inference)



DSAIL@KAIST

SUPPLEMENTARY MATERIALS

[Full paper] https://arxiv.org/abs/2208.10205

[Code] https://github.com/SukwonYun/LTE4G

[Author Email] swyun@kaist.ac.kr