TEDDY

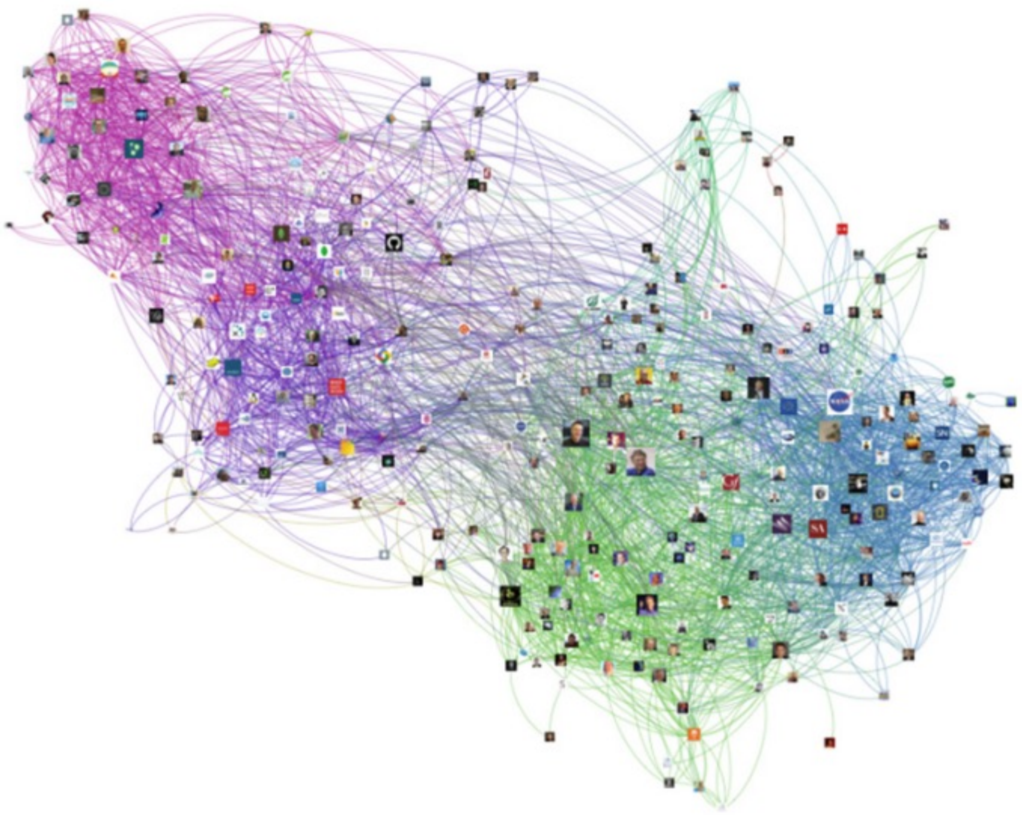
: Trimming Edges with Degree-based Discrimination strategY

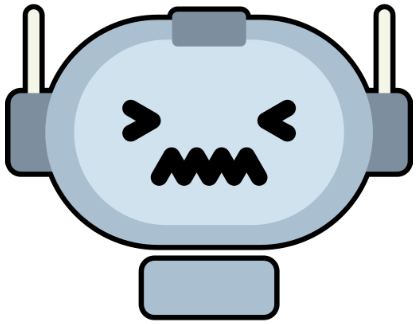
Hyunjin Seo*, Jihun Yun*, Eunho Yang

KAIST MLILAB Machine Learning & Intelligence

ICLR

Graph datasets and GNNs have progressively become more INTRICATE

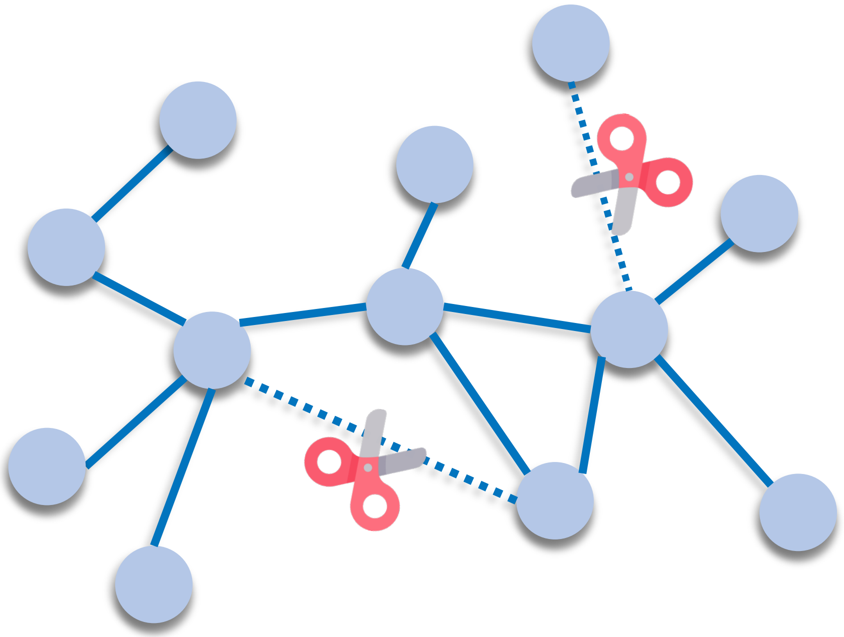




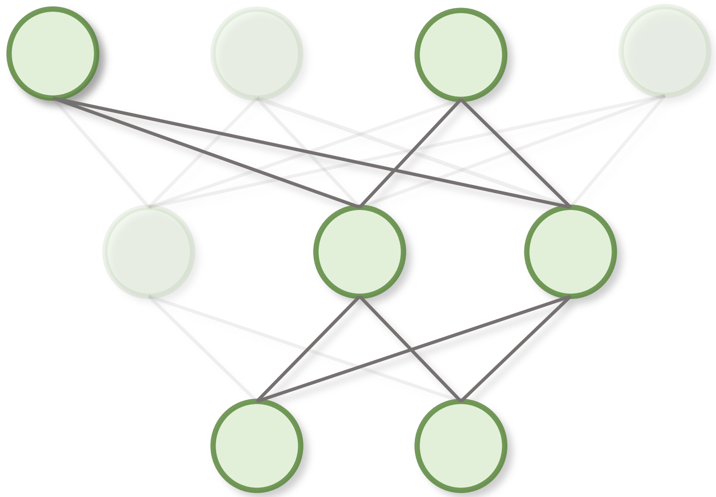
Notorious COMPUTATIONAL OVERHEAD

during training / inference

SOLUTION: Finding Graph Lottery Tickets (GLT) has become one of the pivotal focus

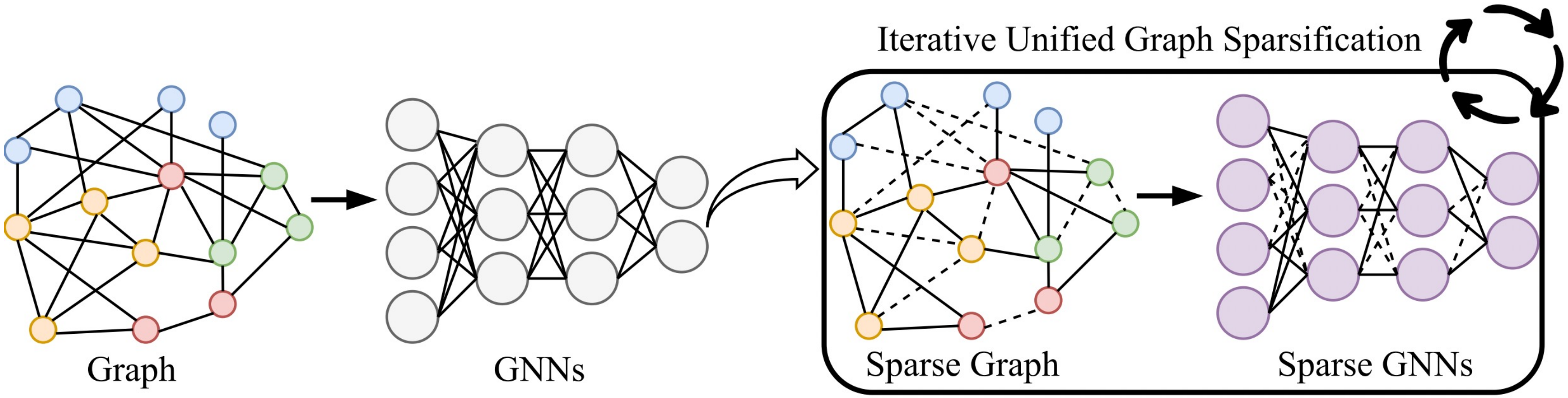


Graph Sparsification



Parameter Sparsification

HOWEVER, previous studies have largely overlooked the paramount importance of structural information

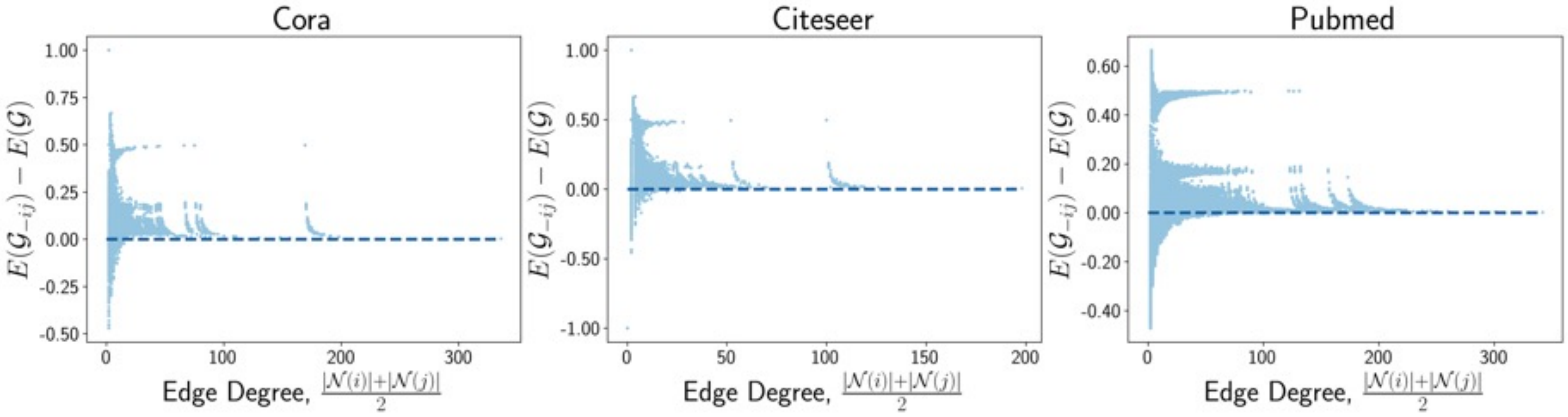


- Magnitude-based pruning ➡ NOT fully exploit inherent pathways in the graph
- Identified lottery tickets in an *iterative* manner (progressive pruning)

1) Chen et al., A Unified Lottery Ticket Hypothesis for Graph Neural Networks, ICML 2021

Motivation: Low-degree Edge Matters Spectrally 1

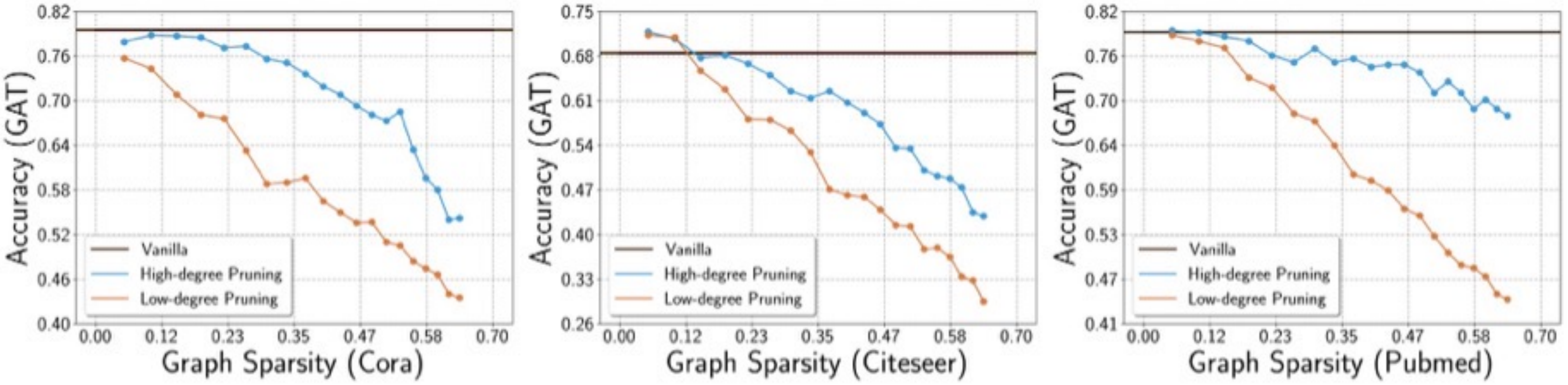
Eliminating low-degree edges makes the graph **spectrally unstable**



- $E(\mathcal{G}_{-ij}) - E(\mathcal{G})$: Changes of normalized graph Laplacian energy E when removing a single edge (i, j) from original edges
- Higher energy indicates low graph stability

Motivation: Low-degree Edge Matters Empirically 2

Eliminating low-degree edges **significantly degrades** node classification performance



- Edge sparsification by (1) the highest edge degree and (2) the lowest edge degree

We introduce **TEDDY**,

a **One-shot** edge sparsification framework

incorporating **Edge Degree** information

with projected gradient descent on ℓ_0 Ball

TEDDY significantly surpasses conventional iterative approaches **UP TO 20.4%** in node classification task, within a **Single Training**

Published as a conference paper at ICLR 2024

TEDDY: TRIMMING EDGES WITH DEGREE-BASED DISCRIMINATION STRATEGY

Hyunjin Seo[1*], Jihun Yun[1*], Eunho Yang[1,2]
Korea Advanced Institute of Science and Technology (KAIST)[1], AITRICS[2]
{bella72,arcprime,eunhoy}@kaist.ac.kr

ABSTRACT

Since the pioneering work on the lottery ticket hypothesis for graph neural networks (GNNs) was proposed in Chen et al. (2021), the study on finding graph lottery tickets (GLT) has become one of the pivotal focus in the GNN community, inspiring researchers to discover sparser GLT while achieving comparable performance to original dense networks. In parallel, the graph structure has gained substantial attention as a crucial factor in GNN training dynamics, also elucidated by several recent studies. Despite this, contemporary studies on GLT, in general, have not fully exploited inherent pathways in the graph structure and identified tickets in an iterative manner, which is time-consuming and inefficient. To address these limitations, we introduce TEDDY, a one-shot edge sparsification framework that leverages structural information by incorporating *edge-degree* information. Following edge sparsification, we encourage the parameter sparsity during training via simple projected gradient descent on the ℓ_0 ball. Given the target sparsity levels for both the graph structure and the model parameters, our TEDDY facilitates efficient and rapid realization of GLT within a *single* training. Remarkably, our experimental results demonstrate that TEDDY significantly surpasses conventional iterative approaches in generalization, even when conducting one-shot sparsification that solely utilizes graph structures, without taking node features into account.

1 INTRODUCTION

Graph neural networks (GNNs) have emerged as a powerful tool for modeling graph-structured data and addressing diverse graph-based tasks, such as node classification (Kipf & Welling, 2016; Hamilton et al., 2017; Xu et al., 2018b; Wang et al., 2020; Park et al., 2021), link prediction (Zhang & Chen, 2018; Li et al., 2018; Yun et al., 2021b; Ahn & Kim, 2021; Zhu et al., 2021), and graph classification (Hamilton et al., 2017; Xu et al., 2018b; Lee et al., 2018; Sui et al., 2022; Hou et al., 2022). In conjunction with the notable performance achieved in GNNs, a substantial number of recent attempts have been made to handle large-scale real-world datasets. Owing to this, datasets and network architectures in graph-related tasks have progressively become more intricate, which incurs the notorious computational overhead both in training and inference.

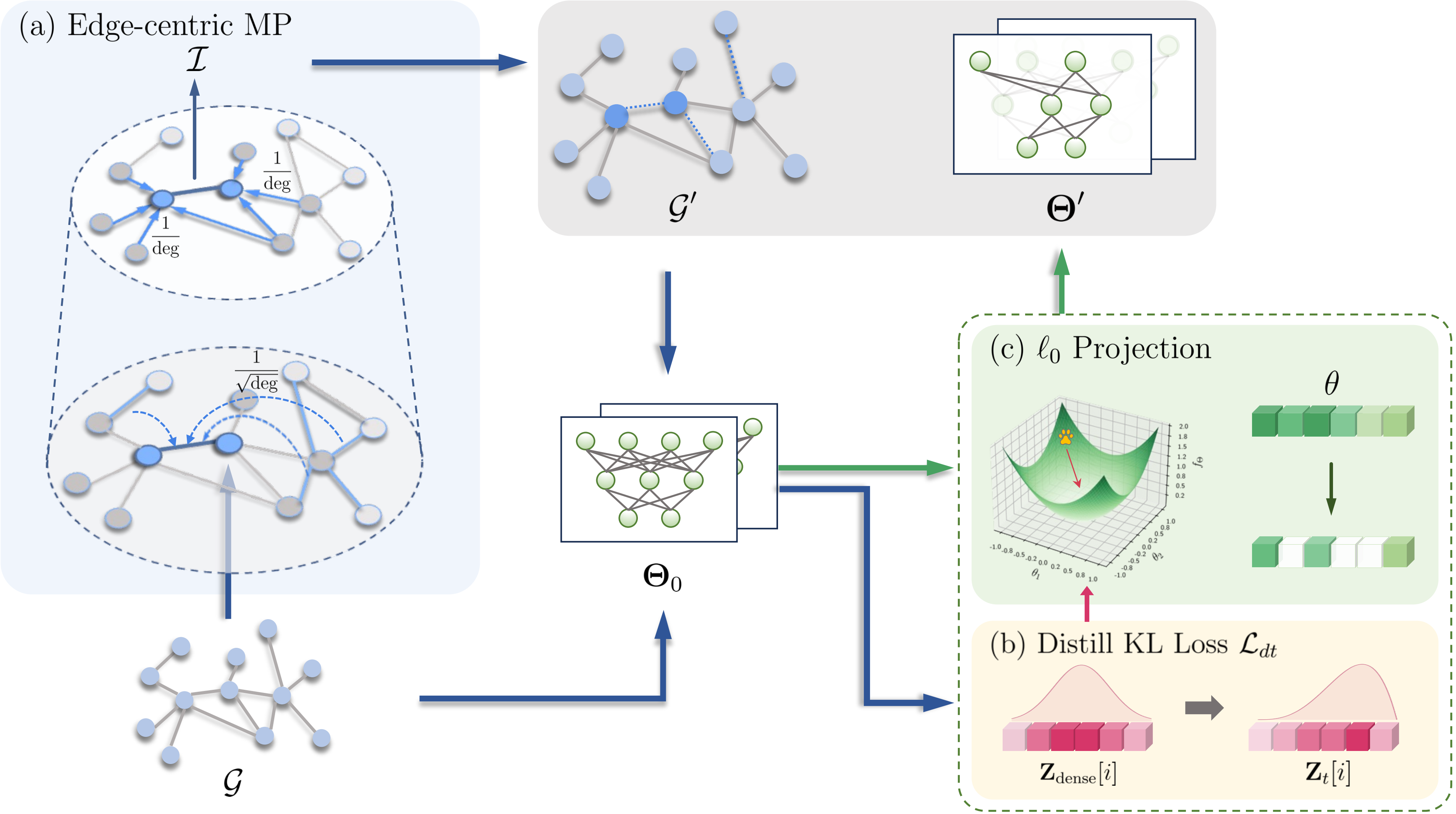
In response to this challenge, GNN compression has emerged as one of the main research areas in GNN communities, and the Graph Lottery Ticket (GLT) hypothesis was articulated (Chen et al., 2021), serving as an extension of the conventional lottery ticket hypothesis (LTH, Frankle & Carbin (2019)) for GNN. Analogous to the conventional LTH, Chen et al. (2021) claimed that the GNNs possess a pair of core sub-dataset and sparse sub-network with admirable performance, referred to as GLT, which can be jointly identified from the original graph and the original dense model. In order to identify GLT, Chen et al. (2021) employ an iterative pruning as LTH where the edges/parameters are pruned progressively through multiple rounds until they arrive at the target sparsity level.

In parallel with advancements in GNN compression, a surge of recent studies has begun to underscore the increasing significance of graph structure over node features in GNN training dynamics. Notably, Tang & Liu (2023) have derived generalization error bounds for various GNN families. Specifically, they have discovered that the model generalization of GNNs predominantly depends on the graph

[*]Equal contribution.

1

Overview of TEDDY



Component 1 - Graph Sparsification

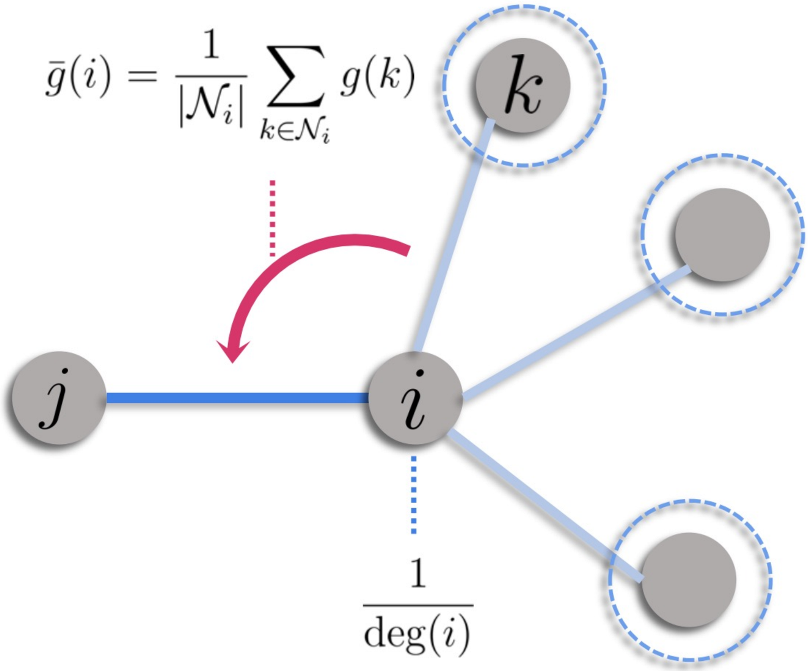
#1 **Aggregate** degree information in a multi-level perspective

$$\widetilde{g}(i) = \frac{\bar{g}(i)}{\deg(i)}, \quad \bar{g}(i) = \frac{1}{|\mathcal{N}_i|} \sum_{k \in \mathcal{N}_i} g(k)$$

#2 Compute **edge-wise score** T_{edge} via outer-product

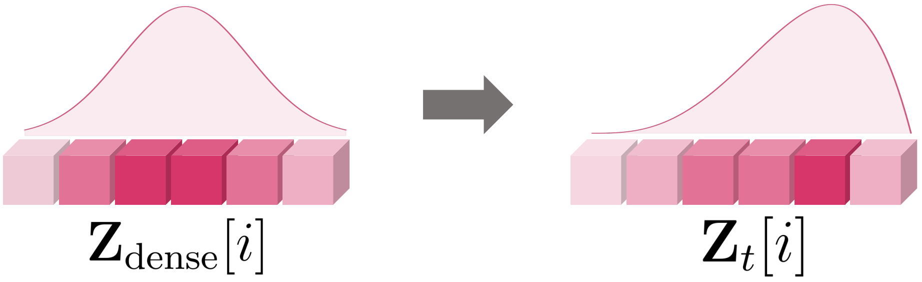
$$T_{\text{edge}} = \widetilde{g}\widetilde{g}^{\mathsf{T}}$$

In practice, TEDDY requires O(N+M) space complexity, as T_{edge} is only computed for the *existing edges*



Component 2 - Parameter Sparsification

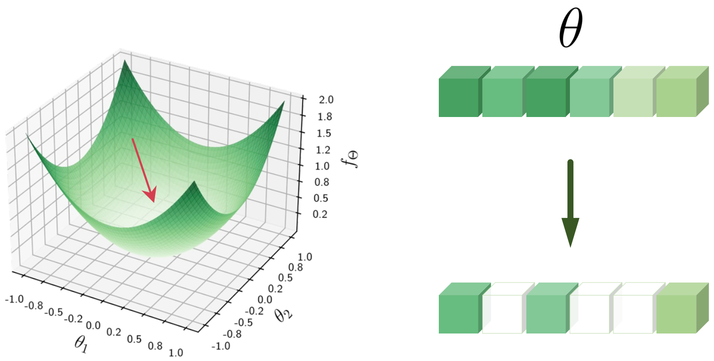
#1 Distillation from Dense GNNs



$$\mathcal{L}_{dt}(\mathcal{G}, \Theta) := \mathcal{L}(\mathcal{G}, \Theta) + \lambda_{dt}\mathbb{KL}(\text{softmax}(Z), \text{softmax}(Z_{\text{dense}}))$$

To improve the model generalization,
match the logits learned from the entire graph

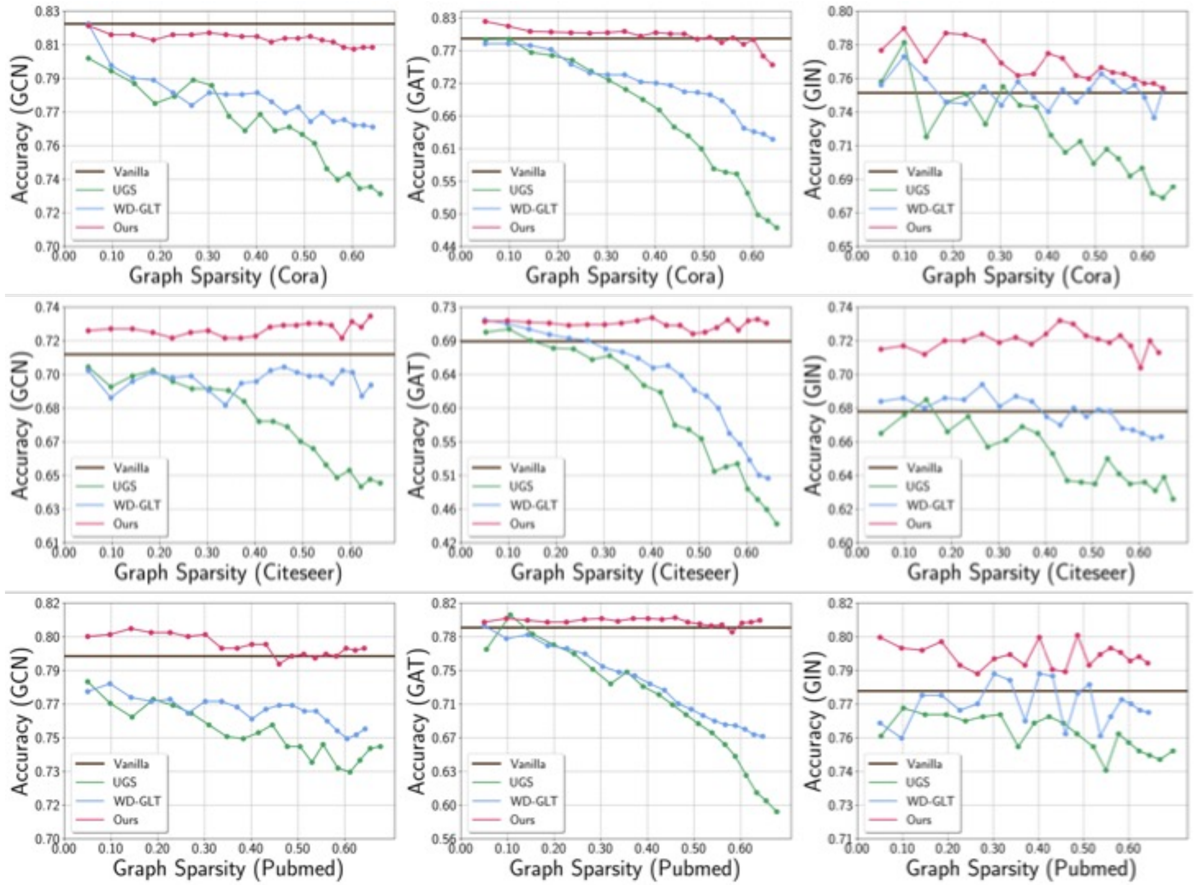
#2 Projected Gradient Descent on ℓ_0 ball



$$\Theta_{t+1} = \text{proj}_{\mathcal{C}_h}(\Theta_t - \eta\nabla_\Theta\mathcal{L}_{dt}(\mathcal{G}, \Theta_t))$$

Encourage the desired level of sparsity h
without iterative process
$\mathcal{C}_h := \{\Theta \in \mathbb{R}^d : \|\Theta\|_0 = h\}$: h-sparse ℓ_0 ball

Results on Small- and Medium-scale Graphs



#1 TEDDY outperforms vanilla GNNs in ***more than half*** of the settings, even though the main goal is the **preservation** of the original accuracy

#2 Consistently achieves SOTA by a **huge margin (12.8-20.4% ⬆ in GAT)**, compared to the optimal baseline performance

Results on Extreme Sparsity Regimes

TEDDY still surpasses the original performance in 14 out of 15 settings

Simulations	Vanilla	20-th	25-th	30-th	35-th	40-th
GS(%)	0	64.14	72.25	78.53	83.38	87.14
WS(%)	0	64.15	72.26	78.54	83.39	87.15
Cora	76.34 ± 0.79	76.20 ± 0.69	$\mathbf{76.64 \pm 0.76}$	$\mathbf{77.38 \pm 0.97}$	$\mathbf{77.20 \pm 1.12}$	$\mathbf{76.82 \pm 1.00}$
Citeseer	68.10 ± 0.77	$\mathbf{71.16 \pm 0.66}$	$\mathbf{70.58 \pm 1.43}$	$\mathbf{71.54 \pm 0.52}$	$\mathbf{71.42 \pm 0.56}$	$\mathbf{71.12 \pm 0.51}$
Pubmed	77.90 ± 0.14	$\mathbf{79.70 \pm 0.26}$	$\mathbf{79.36 \pm 0.71}$	$\mathbf{79.68 \pm 0.37}$	$\mathbf{80.48 \pm 0.50}$	$\mathbf{80.98 \pm 0.42}$

Progressive improvement is observed with the increment in sparsity ratio (**3.08% ⬆ on Pubmed**)

Summary

Contact Info

Paper Link

- We presented a novel edge sparsification method that considers the **Graph Structural** Information

- TEDDY successfully identifies GLT within a **Single Training**

- As future work, we plan to investigate how to incorporate the node feature information into our framework

For more details,
please join our poster session!
Thank you ☺