Unsupervised Episode Generation for Graph Meta-learning

Jihyeong Jung[1], Sangwoo Seo[1], Sungwon Kim[2], Chanyoung Park[1 2]
Department of Industrial & Systems Engineering[1], Graduate School of Data Science[2], KAIST

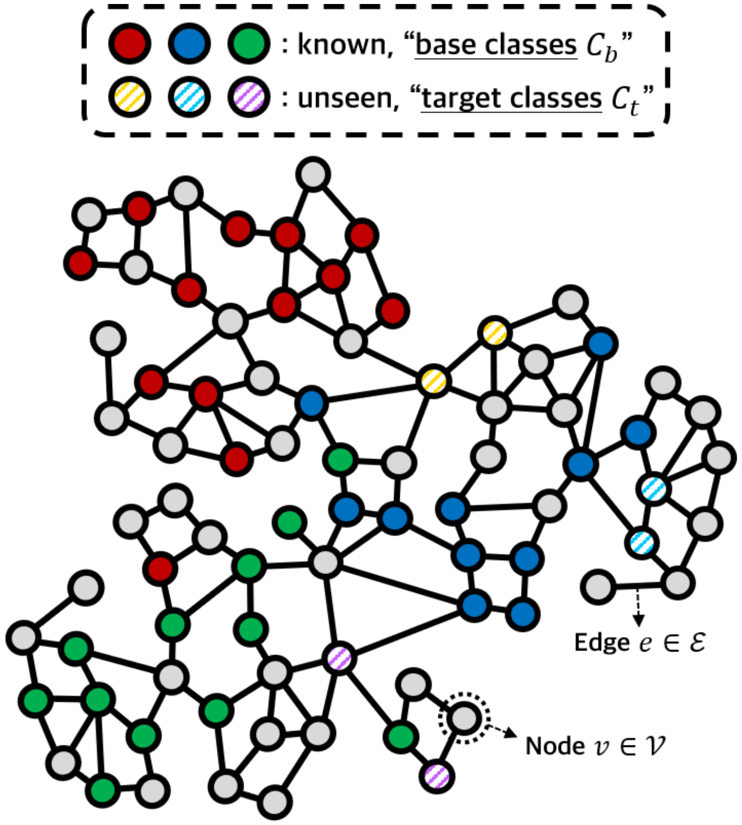
Accepted in the 41st International Conference on Machine Learning (ICML 2024), Vienna, Austria.

KAIST DSAIL Data Science & Artificial Intelligence

Contents

Introduction

- Preliminaries: Frequently used Notations



Graph-structured Data $\mathcal{G}$

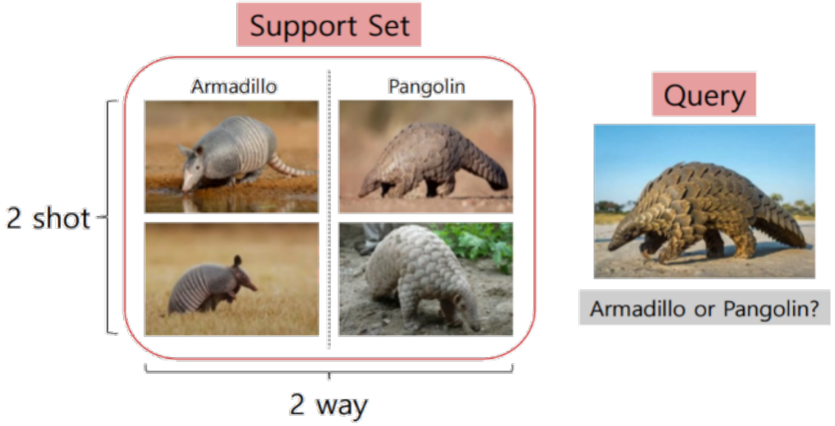
$\mathcal{G}$	$= (\mathcal{V}, \mathcal{E}, X)$; given graph-structured data
$\mathcal{V}$	a set of nodes
$\mathcal{E}$	$\subset \mathcal{V} \times \mathcal{V}$; a set of edges
X	a d-dimensional node feature matrix, or a set of node features $\{x_v : v \in \mathcal{V}\}$
C	a set of total node classes; $C = C_b \cup C_t$
C_b	*base classes*, a set of node classes that can be utilized during training
C_t	*target classes*, a set of node classes that have to be recognized in downstream FSNC tasks
$\mathcal{T}$	$= (S_\mathcal{T}, Q_\mathcal{T})$; a N-way K-shot Q-query (training or testing) episode (task)
$S_\mathcal{T}$	a support set, a set of given a few-labeled samples in $\mathcal{T}$
$Q_\mathcal{T}$	a query set, a set of unlabeled samples have to be predicted in $\mathcal{T}$
N	a number of *way*; i.e., number of distinct classes have to be classify within $\mathcal{T}$
K	a number of labeled samples (support set) given for each class (i.e., way) in $\mathcal{T}$
Q	a number of queries given for each class in $\mathcal{T}$
f_θ	a model have to be trained (i.e., GNN encoder)
θ	a model parameter

Frequently used, important Notations

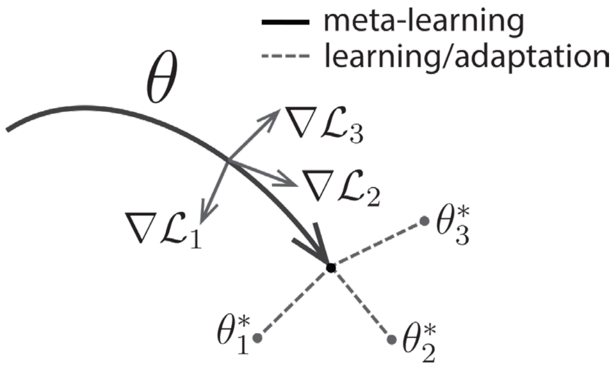
Introduction

- Preliminaries: Few-shot Learning

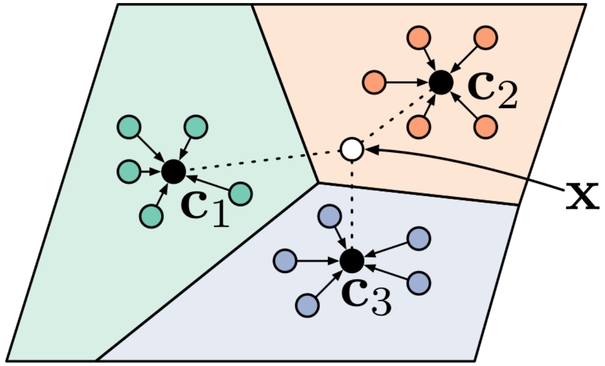
- Few-shot Learning (FSL)
  - Challenge: Deep Neural Networks (DNNs) show poor generalizability for unseen classes with only a few-labeled samples
  - Objective: Like humans, **machines should be able to learn from a few-labeled samples to recognize unseen classes**
  - Dominant paradigm: applying meta-learning methods like MAML [1] and ProtoNet [2] utilizing an **episodic learning framework**



Sample Image: 2-way 2-shot FSL Problem



Description of MAML [1]



Description of ProtoNet [2]

Image: Provided by Sungwon Kim (https://sung-won-kim.github.io)

[1] Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*. PMLR, 2017.

[2] Snell, J., Swersky, K., and Zemel, R. Prototypical networks for few-shot learning. *Advances in Neural Information Processing Systems*, 30, 2017.

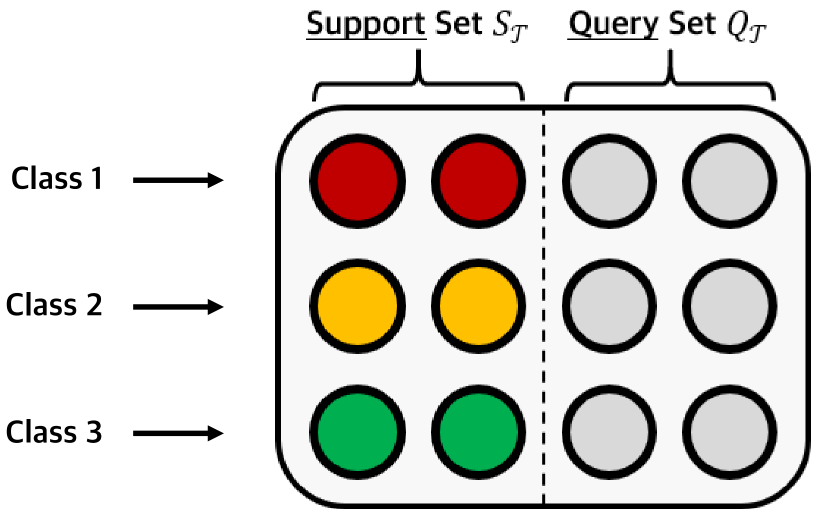
Introduction

- Preliminaries: Few-shot Learning Downstream Task settings

$\mathcal{G}$	$= (\mathcal{V}, \mathcal{E}, X)$; given graph-structured data
$\mathcal{V}$	a set of nodes
$\mathcal{E}$	$\subset \mathcal{V} \times \mathcal{V}$; a set of edges
X	a d-dimensional node feature matrix, or a set of node features $\{x_v : v \in \mathcal{V}\}$
C	a set of total node classes; $C = C_b \cup C_t$
C_b	*base classes*, a set of node classes that can be utilized during training
C_t	*target classes*, a set of node classes that have to be recognized in downstream FSNC tasks
$\mathcal{T}$	$= (S_\mathcal{T}, Q_\mathcal{T})$; a N-way K-shot Q-query (training or testing) episode (task)
$S_\mathcal{T}$	a support set, a set of given a few-labeled samples in $\mathcal{T}$
$Q_\mathcal{T}$	a query set, a set of unlabeled samples have to be predicted in $\mathcal{T}$
N	a number of *way*; i.e., number of distinct classes have to be classify within $\mathcal{T}$
K	a number of labeled samples (support set) given for each class (i.e., way) in $\mathcal{T}$
Q	a number of queries given for each class in $\mathcal{T}$
f_θ	a model have to be trained (i.e., GNN encoder)
θ	a model parameter

Frequently used, important Notations

- Formal Downstream task setting in previous studies
  - Following Vinyals et al. [1], **N-way K-shot Few-shot Learning task is common**
  - N: number of distinct target classes within the downstream task
  - K: number of given a few-labeled samples in each 'support set'
  - Q: number of queries have to be classified



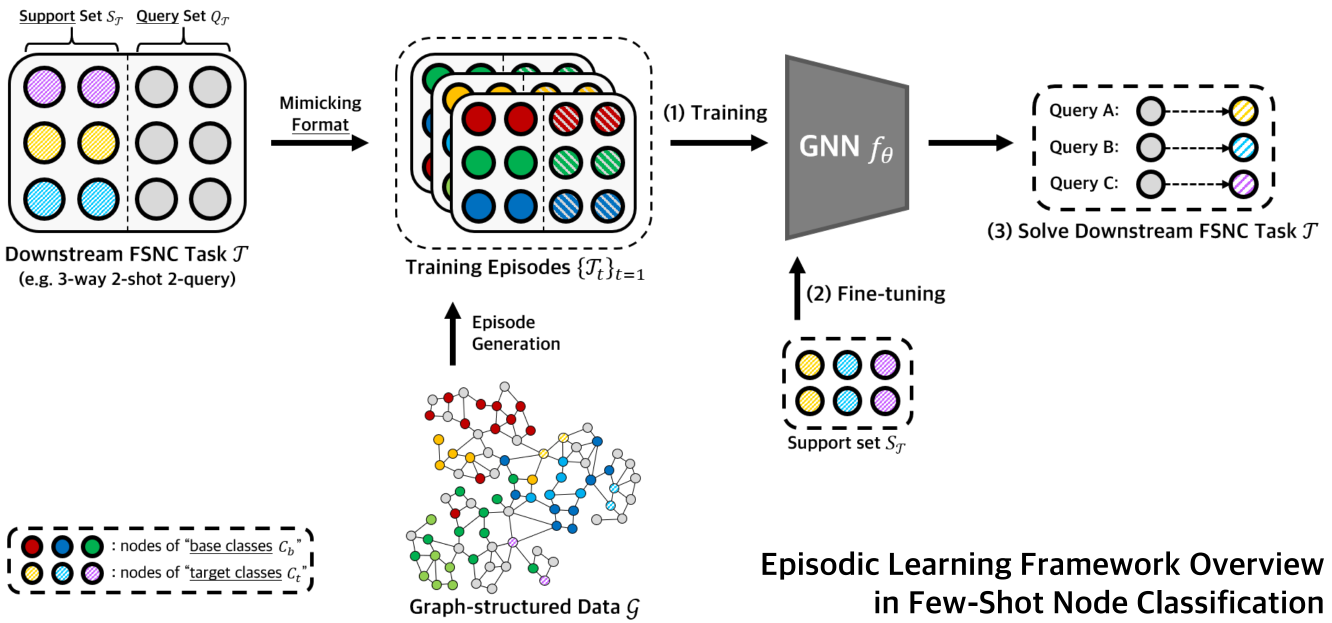
Example: 3-way, 2-shot, 2-query FSL task

[1] Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K. and Wierstra, D. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016.

Introduction

– Preliminaries: Episodic Learning Framework

- Description
  - Instead of using mini-batches, episodic learning trains model by using bundle of tasks $\{\mathcal{T}_t\}_{t=1}^{T}$, where $S_{\mathcal{T}_t} = \{(x_{t,i}^{spt}, y_{t,i}^{spt})\}_{i=1}^{N\times K}$ are support set and $Q_{\mathcal{T}_t} = \{(x_{t,i}^{qry}, y_{t,i}^{qry})\}_{i=1}^{N\times Q}$ for the stochastic optimization
  - By **mimicking the "format" of the downstream task**, **model f_θ is trained to be aware of the task to solve** in the testing phase
  - Most of meta-learning methods follow Episodic Learning Framework [1] for the model training



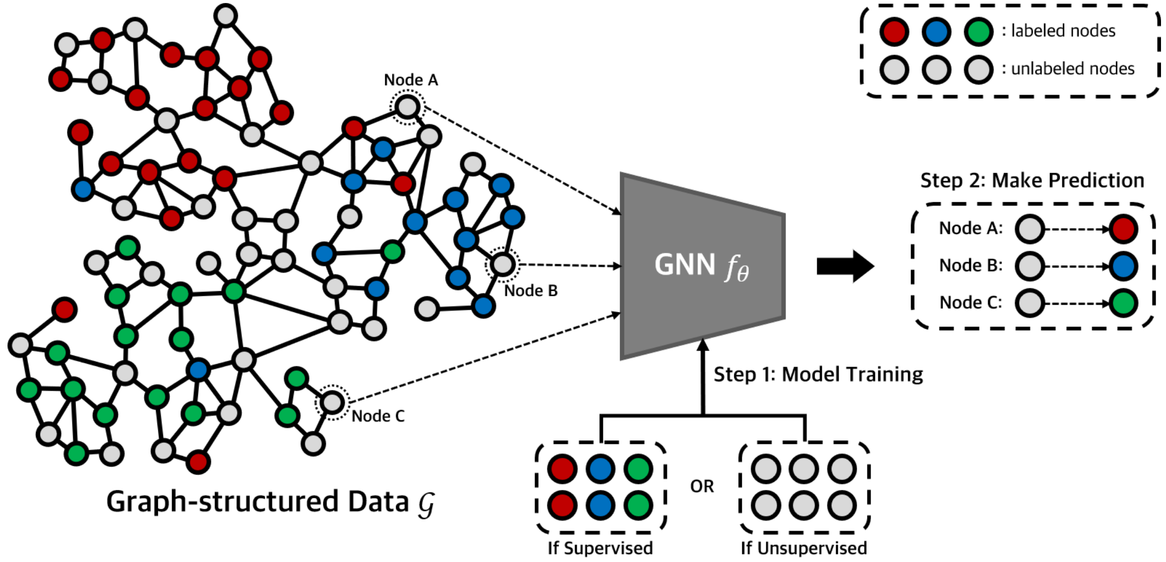
Episodic Learning Framework Overview in Few-Shot Node Classification

[1] Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K. and Wierstra, D. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016.

Introduction

- Preliminaries: Ordinary Node Classification on Graph-structured Data

- Ordinary Node Classification
  - Objective: classifying unlabeled nodes to the one of **known classes**
  - In this setting, **entire classes in the graph are already known**

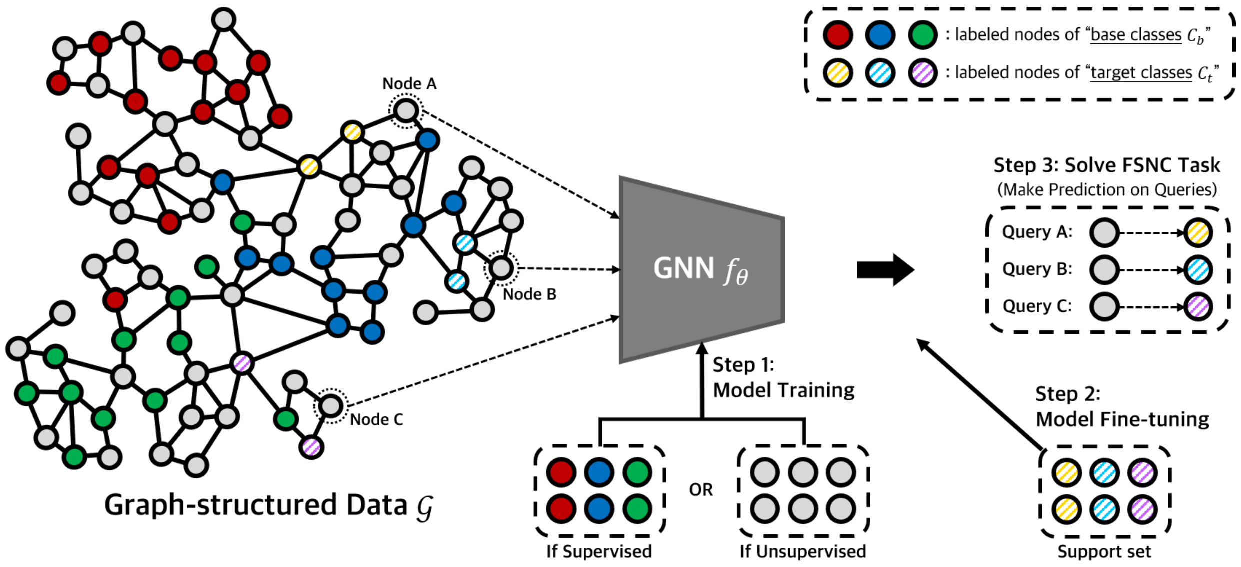


Three-class Example of the Process of the Ordinary Node Classification

Introduction

- Preliminaries: Few-shot Learning in Graph-structured Data

- Few-Shot Node Classification (FSNC)
  - Objective: classifying queries to the one of **unseen classes** (target classes C_t) **with a few-labeled nodes** (support set) in the downstream FSNC task
  - Only some of classes (base classes C_b) are known during training phase in the supervised setting
  - Current Solution: **1) Meta-learning based methods** or **2) utilizing Graph Contrastive Learning (GCL) + Linear probing**

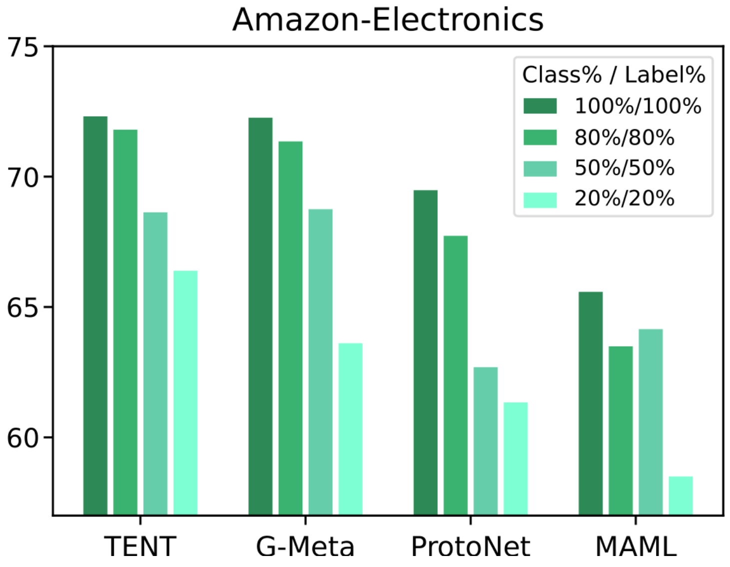


3-way 2-shot Example of the Overall Process of the Few-Shot Node Classification

Introduction

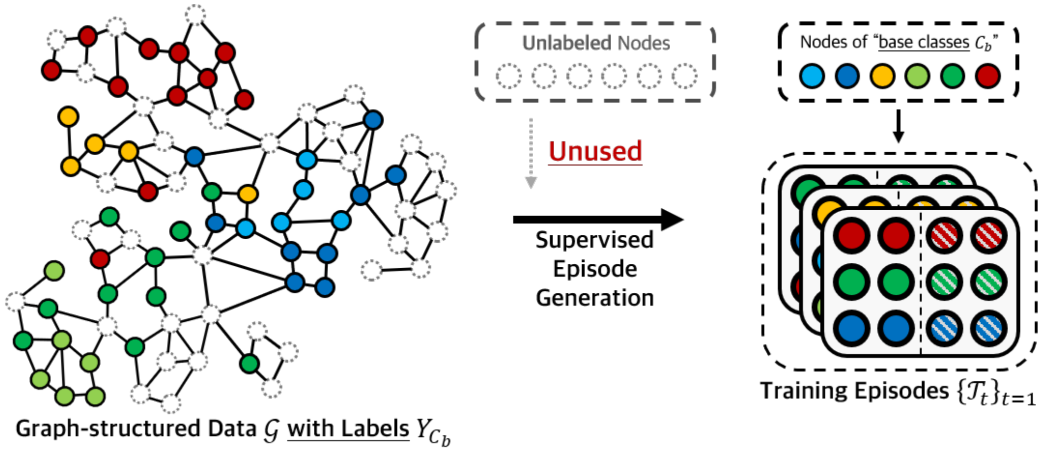
- Challenges in FSNC: Why Supervised Graph Meta-learning methods are Insufficient?

- Label-scarcity Problem
  - Supervised Graph Meta-learning **require <u>enough labeled samples</u> from <u>diverse base classes</u>** for training → **Expensive**
  - <u>Otherwise, their FSNC performances are significantly deteriorated</u> (Kim et al. [1], Wang et al. [2])
  - Moreover, the **Label-scarcity problem hinders the full utilization of the information of all nodes in a graph**



Impact of the Label-scarcity Problem on Supervised Graph Meta-learning Methods

Related with



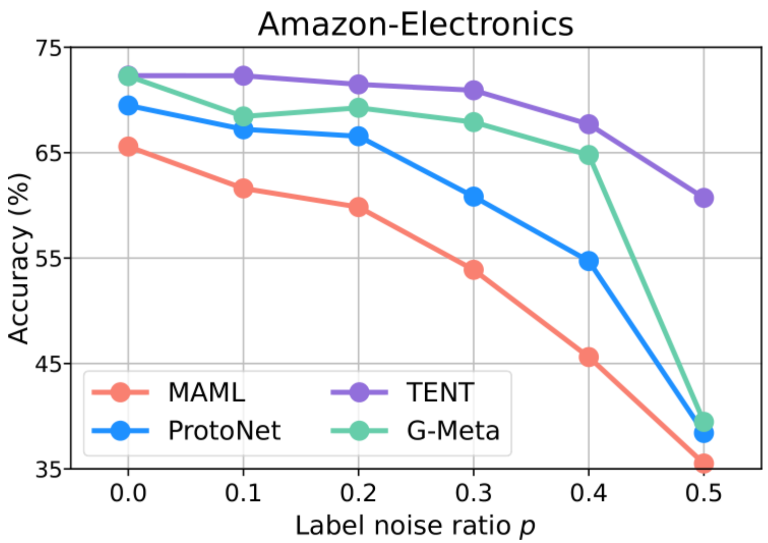
Supervised → Cannot fully utilize all nodes in a graph

[1] Kim, S., Lee, J., Lee, N., Kim, W., Choi, S., and Park, C. Task-equivariant graph few-shot learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023.
[2] Wang, S., Dong, Y., Ding, K., Chen, C. and Li, J. Few-shot node classification with extremely weak supervision. In *Proceedings of the 16th International Conference on Web Search and Data Mining*, 2023.

Introduction

- Challenges in FSNC: Why Supervised Graph Meta-learning methods are Insufficient?

- Vulnerability to the Label Noise
  - **Noisy labels** in base classes also **hurts FSNC performance of existing graph meta-learning methods**
  - It is not always guaranteed that given labels are all clean

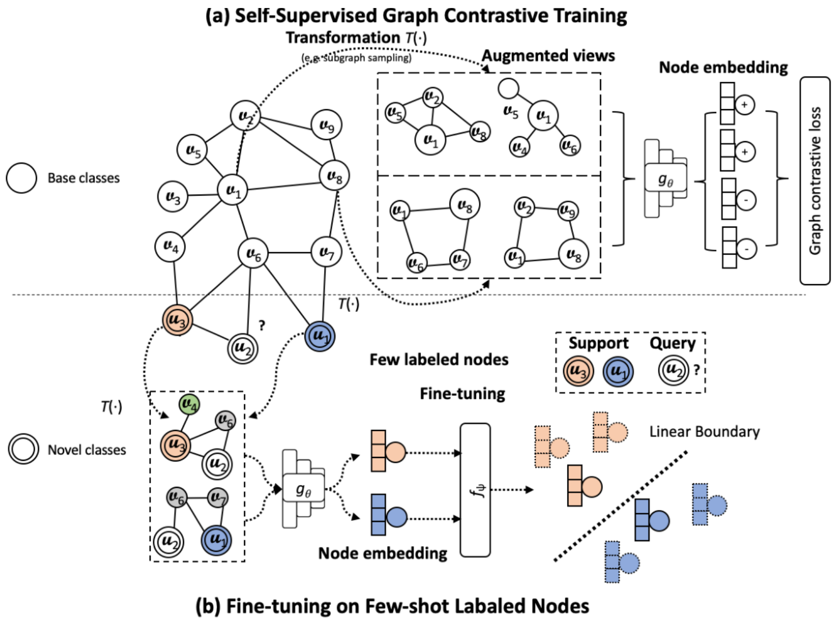


Impact of the Label Noise on Supervised Graph Meta-learning Methods

Introduction

- Challenges in FSNC: Why GCL methods are Insufficient?

- Solving FSNC problem with GCL methods
  - Recently, TLP [1] showed that a **simple linear probing on node embeddings produced by GCL methods is better** than existing supervised graph meta-learning methods
  - This is because **GCL methods involve all nodes in a graph for training**, thus TLP can utilize their **effective and generic node embeddings** for solving FSNC



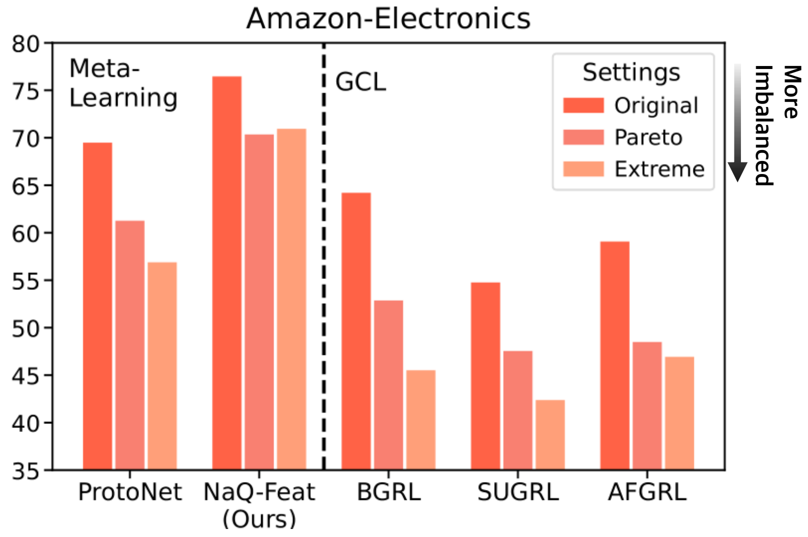
Methodology Overview of Transductive Linear Probing (TLP) [1] with unsupervised GCL methods

[1] Tan, Z., Wang, S., Ding, K., Li, J., and Liu, H. Transductive linear probing: A novel framework for few-shot node classification. In *Learning on Graphs Conference*, 2022.

Introduction

- Challenges in FSNC: Why GCL methods are Insufficient?

- Class Imbalance Problem
  - However, **GCL methods are vulnerable to the Class Imbalance** in the graph;
  - GCL methods have difficulty in learning about nodes from minority classes
  - **Also, without knowledge of the type of downstream task during training**, **GCL methods lacks generalizability** [1] for FSNC,
  - As a result, GCL methods shows much more degraded FSNC performance in more imbalanced setting,



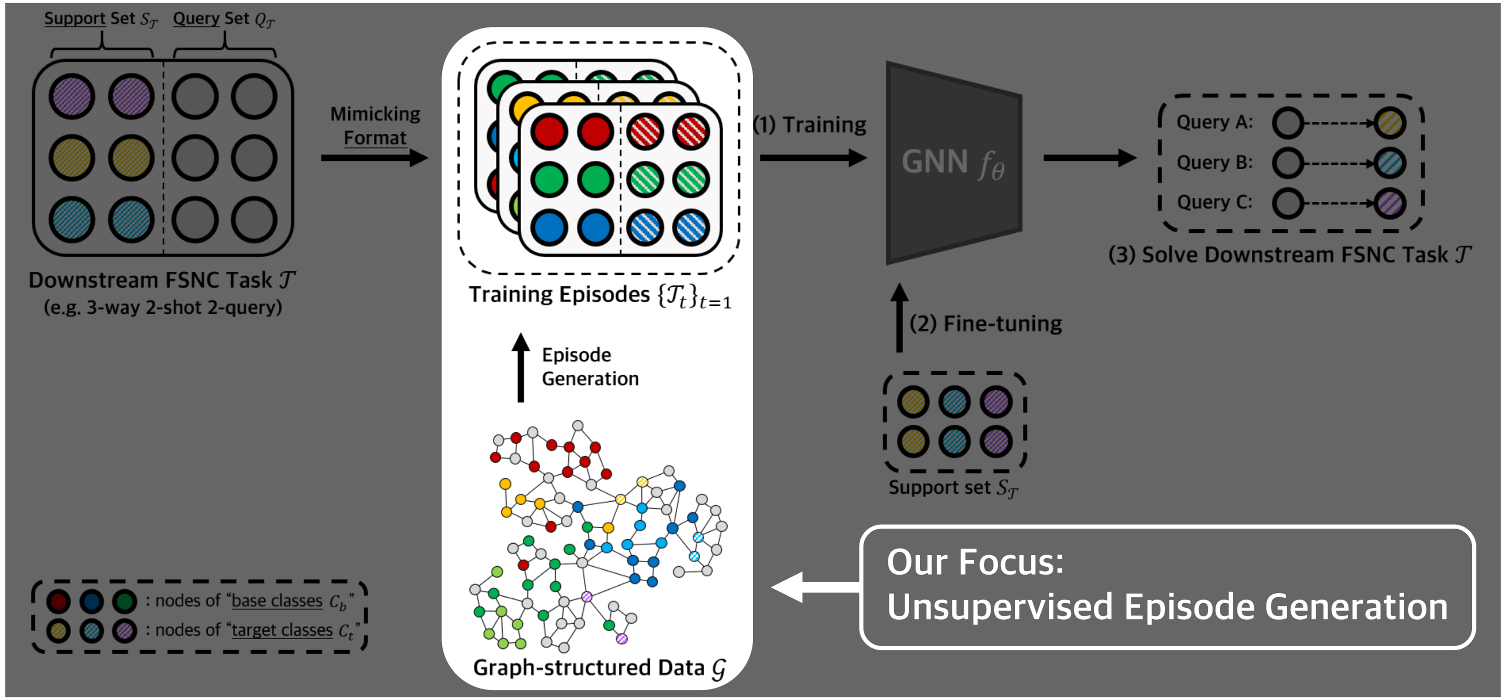
Impact of the Class Imbalance on Meta-learning vs. GCL methods

[1] Lu, Y., Jiang, X., Fang, Y., and Shi, C. Learning to pre-train graph neural networks. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, pp. 4276-4284, 2021.

Introduction

- Solution: Unsupervised Graph Meta-learning

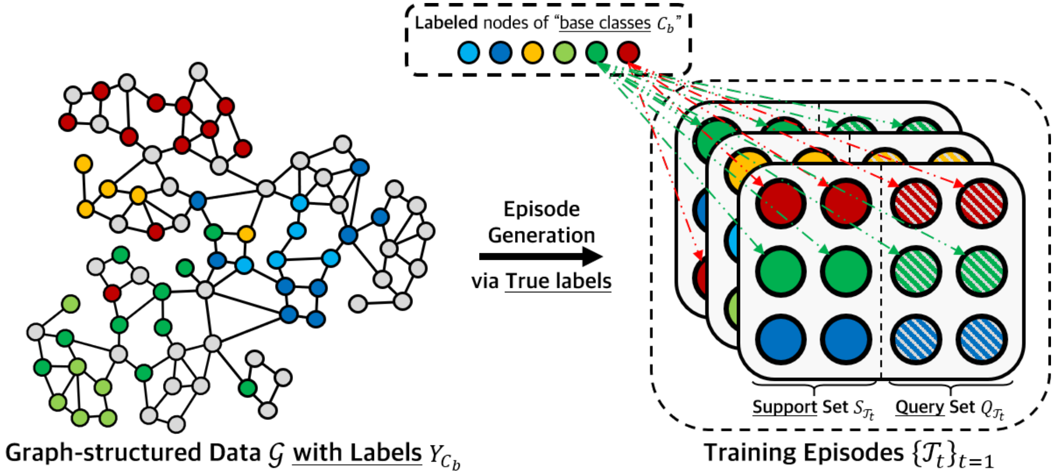
- Solution: "Unsupervised Graph Meta-learning"
  - "**Unsupervised**": we can **utilize all nodes in a graph during training** of graph meta-learning methods
  - "**Meta-learning**": **model can learn downstream task format information** by episodic learning framework
  - Thus, we propose **Unsupervised Episode Generation** **methods to achieve above both properties**



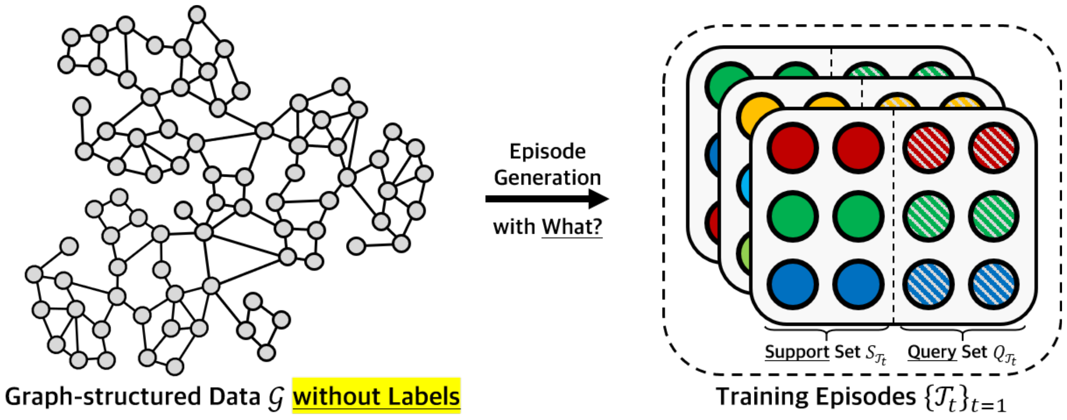
Introduction

- Solution?: Unsupervised Graph Meta-learning

- Challenge
  - **Supervised** Episode Generation: can be done easily with labeled data (X_{C_b}, Y_{C_b}) in base classes C_b
    - After sampling N classes, sample $K + Q$ nodes to make K-shot support set and Q-query query set
  - **Unsupervised** Episode Generation: **only with "unlabeled" data** X, **how can we generate training episodes?**



Ordinary Supervised Episode Generation

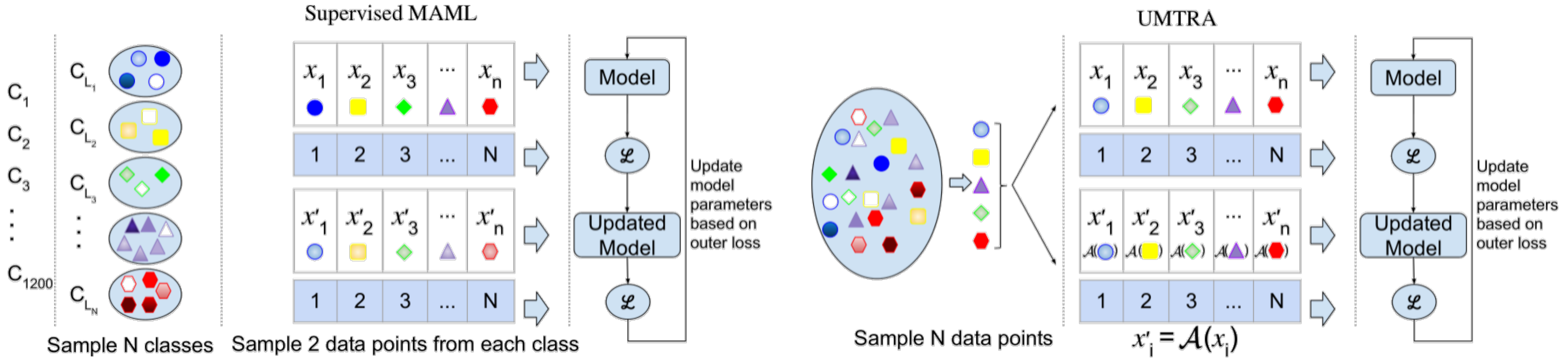


Unsupervised Episode Generation?

Introduction

- Related Works: Unsupervised Meta-learning in Computer Vision

- Unsupervised Meta-learning via Augmentation
  - UMTRA [1] / AAL [2] **utilizes image augmentation to generate queries of randomly sampled N support set**
  - **UMTRA**: randomly sample N samples to make support set, and apply image augmentation on them to make query set
    - **Only generates 1-shot support set to assure that randomly sampled images to have different labels**



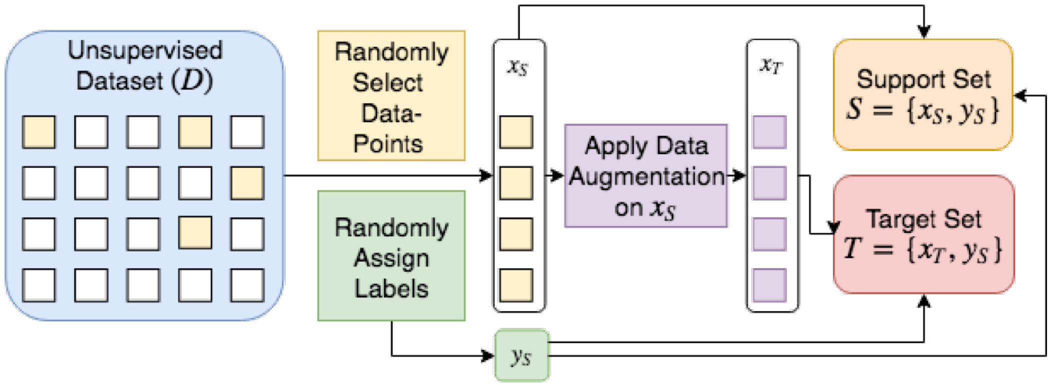
Supervised MAML vs. UMTRA [1]

[1] Khodadadeh, S., Bölöni, L., and Shah, M. Unsupervised meta-learning for few-shot image classification. *Advances in neural information processing systems*, 32, 2019.
[2] Antoniou, A. and Storkey, A. Assume, augment and learn: Unsupervised few-shot meta-learning via random labels and data augmentation. *arXiv preprint arXiv:1902.09884*, 2019.

Introduction

- Related Works: Unsupervised Meta-learning in Computer Vision

- Unsupervised Meta-learning via Augmentation
  - UMTRA [1] / AAL [2] **utilizes image augmentation to generate queries of randomly sampled N support set**
  - **AAL**: **Randomly sample $N \times K$ images**, then make N-way K-shot support set by randomly assigning pseudo-labels



Overview of AAL [2]

Algorithm 2 Unsupervised MAML Sampling Strategy

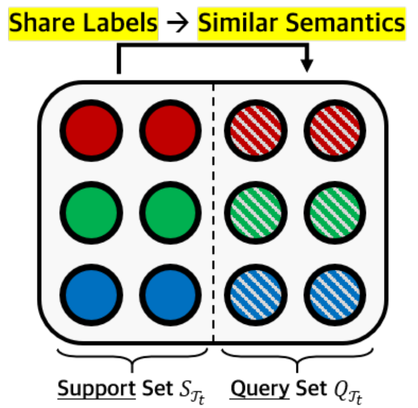
1: **Require:** Dataset $\mathcal{D}$ with I number of data-points
2: Sample $N \times K$ data-points from $\mathcal{D}$, where N is the number of classes per set[1] and K is the number of samples per class $(N \times K) \leqslant I$
3: Build the support set S by assigning random labels to the previously $N \times K$ sampled data-points
4: Build the target (evaluation) set E by augmenting the support set S samples and keeping the labels identical
5: **Return** S, E

Unsupervised Episode Generation of AAL [2]

[1] Khodadadeh, S., Bölöni, L., and Shah, M. Unsupervised meta-learning for few-shot image classification. *Advances in neural information processing systems*, 32, 2019.
[2] Antoniou, A. and Storkey, A. Assume, augment and learn: Unsupervised few-shot meta-learning via random labels and data augmentation. *arXiv preprint arXiv:1902.09884*, 2019.

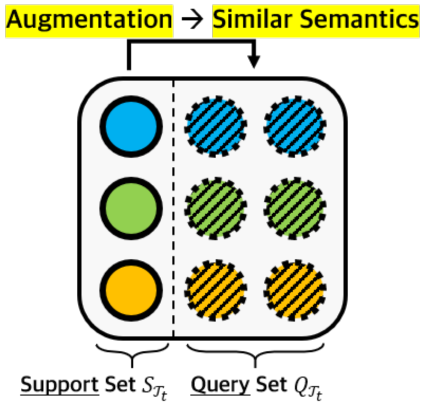
Proposed Methodology: Neighbors as Queries (NaQ)

– Motivation



- Closer Look at Episodic Learning Framework
  - Support set → provides basic information about the task to be solved
  - Query set → enables the model to understand how to solve the given task by making prediction on queries

- Existing Episode Generation methods
  - Supervised: Queries of support set have same labels → Queries and Support set share similar semantics
  - UMTRA/AAL: By augmentation, make queries having similar semantic with support set



Therefore, **queries should share similar semantics with the support set**
→ **"Similarity" Condition on Queries**

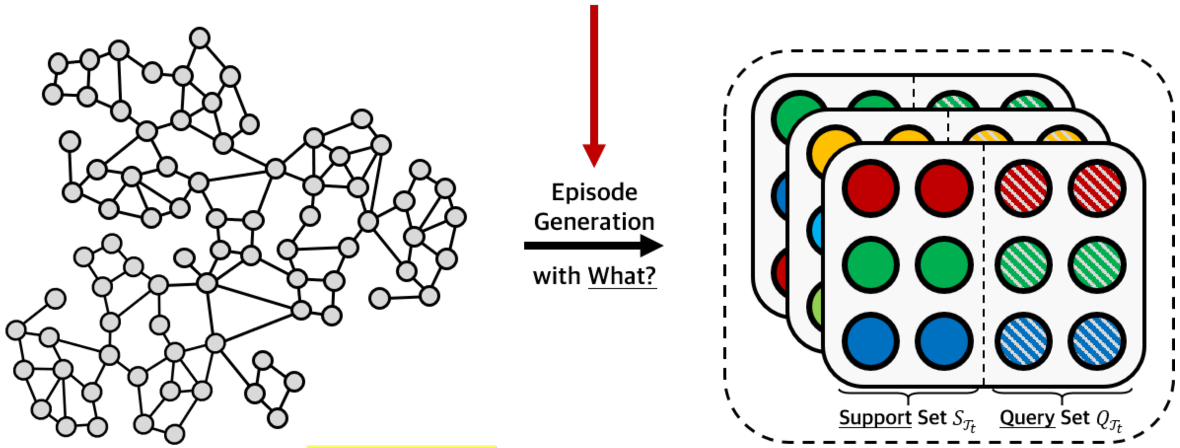
Proposed Methodology: Neighbors as Queries (NaQ)

- Motivation

- Claim: Similarity Condition on Query set
  - Unsupervised Episode Generation: **How to sample queries that share similar semantics with support set samples?**

- Proposed Solution: Neighbors as Queries (NaQ)
  - **Find similar nodes of each support set node as queries!**
  - **NaQ-Feat**: use **raw feature-level similarity / NaQ-Diff**: use **structural-level similarity** measured by graph Diffusion [1]

NaQ: Use feature-level or structure-level similar nodes!



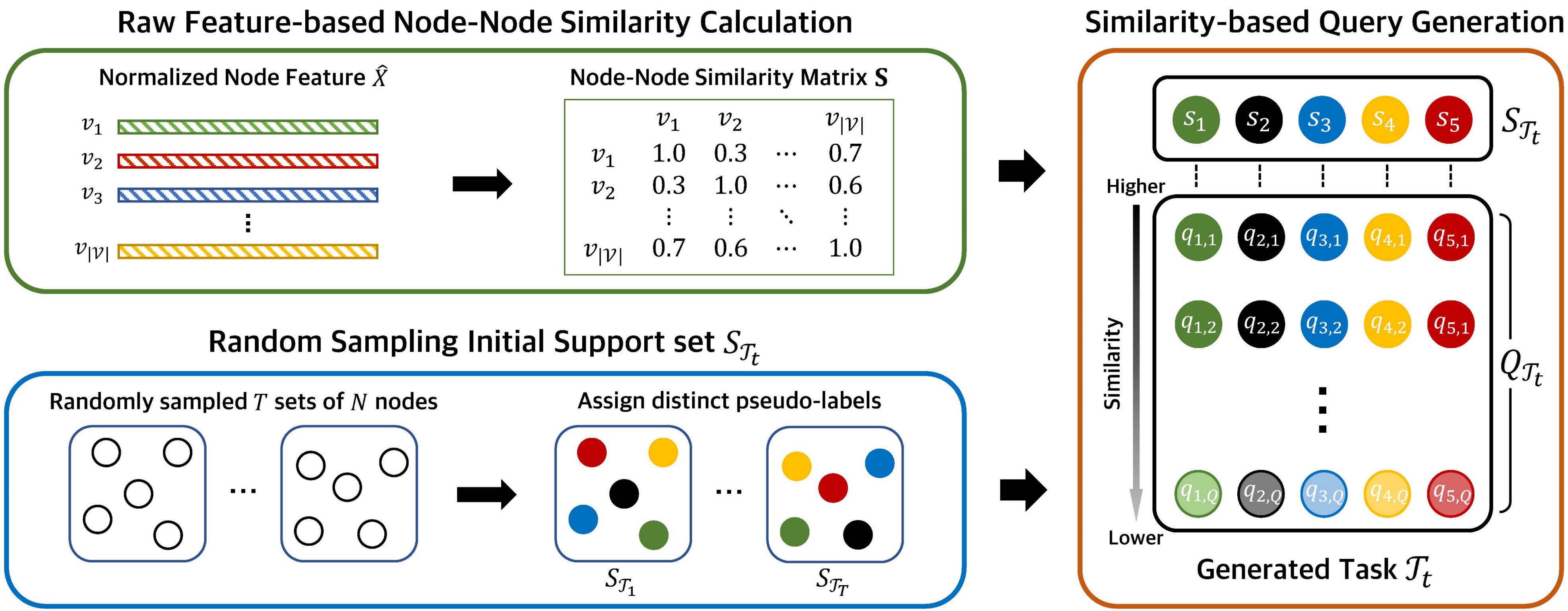
Graph-structured Data $\mathcal{G}$ without Labels

Training Episodes $\{\mathcal{T}_t\}_{t=1}$

[1] Gasteiger, J., Weißenberger, S., and Günnemann, S. Diffusion improves graph learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Proposed Methodology: Neighbors as Queries (NaQ)

- Methodology Overview: NaQ-Feat



Methodology Overview of the NaQ-Feat

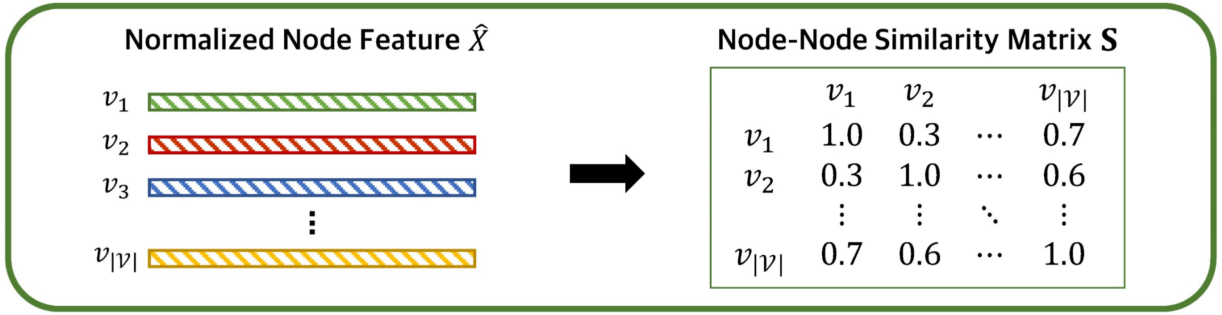
Proposed Methodology: Neighbors as Queries (NaQ)

- Methodology Details

- Node-Node Similarity Calculation
  - **Per dataset**, we calculate node-node similarity matrix with raw node feature for sampling similar node as queries
  - As **it can be done in pre-processing phase, it does not cause large computational cost**

- Similarity Metric Choice
  - For bag-of-words raw node feature, we used cosine similarity
  - For continuous-type raw node feature (e.g. word embeddings), we used Euclidean distance

Raw Feature-based Node-Node Similarity Calculation



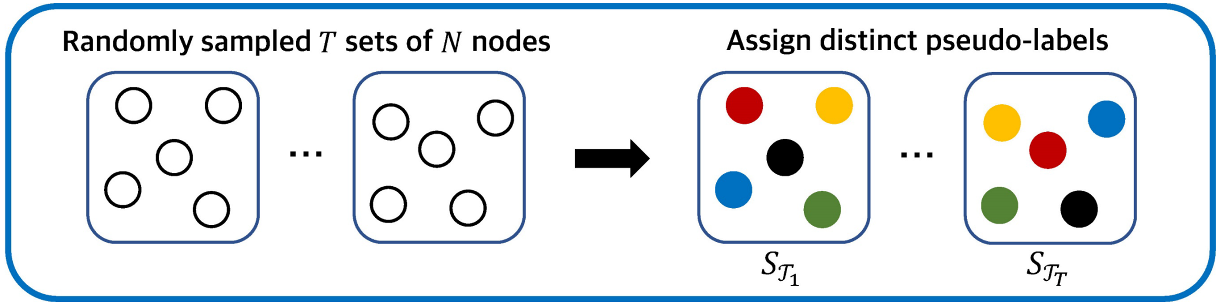
Similarity Calculation Process of the NaQ-Feat

Proposed Methodology: Neighbors as Queries (NaQ)

- Methodology Details

- Support set Generation
  - Similar to UMTRA, we randomly sample N nodes from the entire graph, then regard each of them as distinct support set
  - **To assure sampled N nodes** (corresponding to 'N-way') **are distinguishable as much as possible**, **only 1-shot support set is generated** regardless of the downstream task setting

Random Sampling Initial Support set $S_{\mathcal{T}_t}$



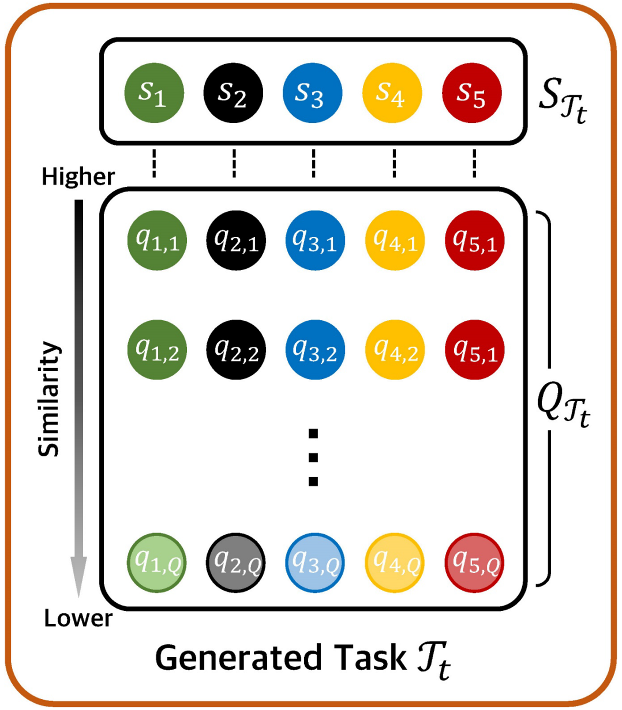
Support set Generation Process of the NaQ-Feat

Proposed Methodology: Neighbors as Queries (NaQ)

- Methodology Details

- Query set Generation
  - For each support set node, **we sample Top-Q similar node as queries**
  - Sampled Q queries are given the same pseudo-label with corresponding support set node
  - <u>Support set node itself is excluded during the query sampling</u> process

Similarity-based Query Generation

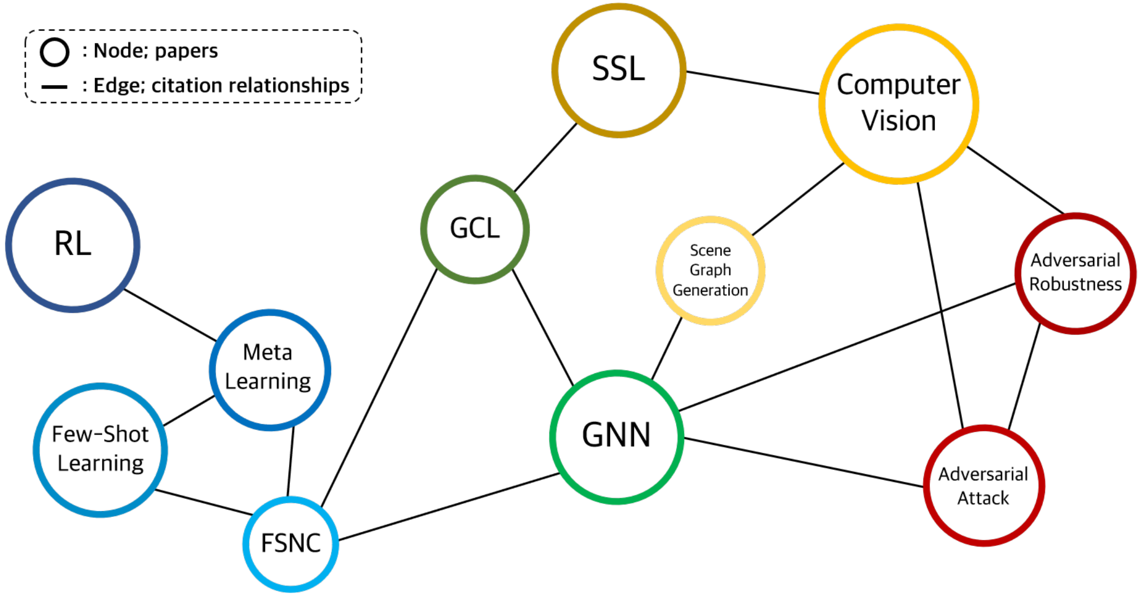


Query set Generation Process of the NaQ-Feat

Proposed Methodology: Neighbors as Queries (NaQ)

- An Extension to NaQ: NaQ-Diff

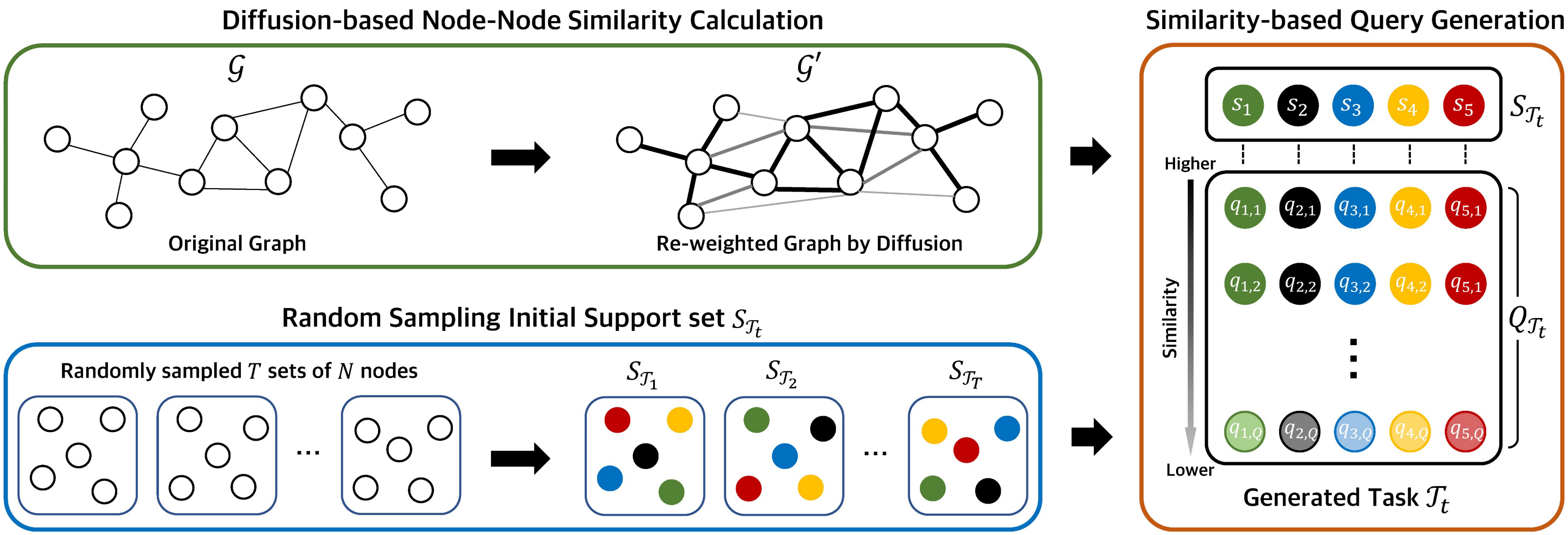
- Motivation
  - NaQ-Feat solely relies on raw node feature X, without considering structural information of the graph
  - However, structural information can be crucial depending on the target domain
  - In citation networks, **citation relationship between papers implies that they share similar semantics** (related topics)
  - Therefore, **considering structurally similar nodes as queries can be more beneficial** in such cases



Toy Example of the Citation Networks

Proposed Methodology: Neighbors as Queries (NaQ)

- Methodology Overview: NaQ-Diff



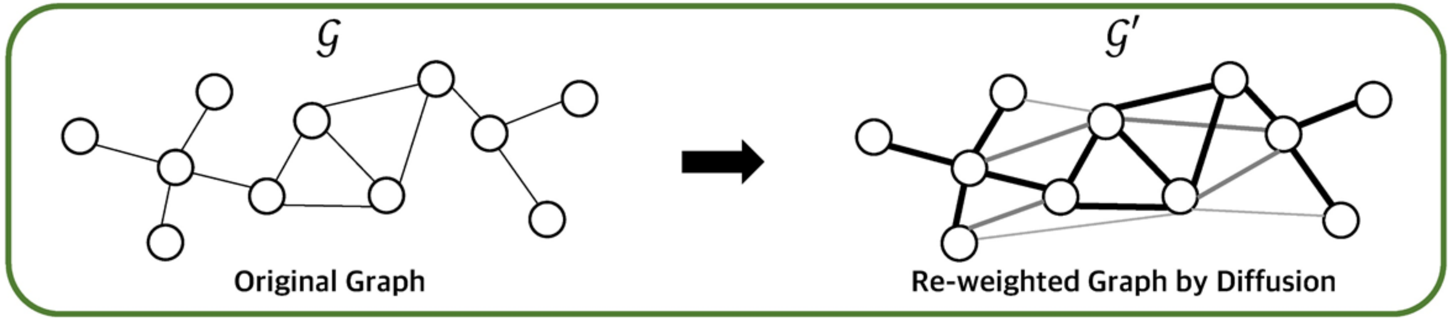
Methodology Overview of the NaQ-Diff

Proposed Methodology: Neighbors as Queries (NaQ)

- Methodology Details

- Node-Node Similarity Calculation
  - NaQ-Diff differs from NaQ-Feat in only the similarity calculation process
  - **Graph Diffusion** [1] **matrix** defined as $\mathbf{S} = \sum_{k=0}^{\infty} \theta_k \mathbf{T}^k$ **is leveraged for measuring structural similarity between nodes**
    - θ_k: weighting coefficients, $\mathbf{T}$: generalized transition matrix calculated with graph adjacency matrix and degree matrix
  - We interpret **edge weights** of diffusion matrix $\mathbf{S}$ **as structural closeness between nodes**

Diffusion-based Node-Node Similarity Calculation



Similarity Calculation Process of the NaQ-Diff

[1] Gasteiger, J., Weißenberger, S., and Günnemann, S. Diffusion improves graph learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Proposed Methodology: Neighbors as Queries (NaQ)

- Model Training with Episodes generated by NaQ

- How to Train existing Graph Meta-learning Methods?
  - **Training Episodes generated by NaQ follow the same, common format of the ordinary supervised episode generation**
  - Hence, **any of existing graph meta-learning methods can be trained in unsupervised manner by NaQ**

- Notes
  - As NaQ generates training episodes with all nodes in a graph, existing graph meta-learning methods can fully utilize all nodes in a graph

---

Algorithm 1 Training Graph Meta-learning methods with NaQ

Require: Bundle of training episodes $\{\mathcal{T}_t\}_{t=1}^{T}$, Meta-learning model $\text{Meta}(\,\cdot\,;\theta)$, learning rate η.

Randomly initialize model parameter θ

for $t = 1, \cdots, T$ **do**

  Step 1: Calculate loss $\mathcal{L}$ by $\text{Meta}(\mathcal{T}_t;\theta)$

  Step 2: Update $\theta \leftarrow \theta - \eta\nabla_\theta\mathcal{L}$

end for

return $\text{Meta}(\mathcal{T}_t;\theta)$

Training Process of existing Graph Meta-learning methods with NaQ

Model Analysis: Why NaQ can work?

- Theoretical Insights: Which similarity condition should NaQ satisfy?

- "Generalization Error" Perspective
  - Assumption: $y = f(x) + \epsilon$ ($\mathbb{E}[\epsilon] = 0, \mathrm{Var}(\epsilon) = \sigma^2 < \infty$), error metric $\mathcal{L}$: Mean-Squared Error;

$$\mathbb{E}\big[\mathcal{L}\big(y', f_S(x')\big)\big] = \big(\mathbb{E}[f_S(x')] - f(x')\big)^2 + \big(\mathbb{E}[f_S(x')^2] - \mathbb{E}[f_S(x')]^2\big) + \sigma^2$$

    - S: **training set**, f_S: **model trained on** S, (x', y'): **test set point,** f: **true, unknown estimation**

- Closer Look at a Single Update Process of MAML [1]
  - Consider a **single** episode $\mathcal{T} = (S_\mathcal{T}, Q_\mathcal{T})$ with encoder f_θ
  - If we regard $S_\mathcal{T}$ as training set, $Q_\mathcal{T}$ as test set, We can interpret that MAML's training process as **"Reducing Generalization Error"** below [2]

$$\mathbb{E}[\mathcal{L}(y^{qry}, f_{\theta'}(x^{qry}))] = \big(\mathbb{E}[f_{\theta'}(x^{qry})] - f_\mathcal{T}(x^{qry})\big)^2 + \big(\mathbb{E}\big[f_{\theta'}(x^{qry})^2\big] - \mathbb{E}[f_{\theta'}(x^{qry})]^2\big) + \sigma^2 \quad \cdots \textbf{(2)}$$

    - (x^{qry}, y^{qry}): **single query,** $f_\mathcal{T}$: **unknown, true estimation on** $\mathcal{T}$

  - **Hence, accurate calculation of Eq. (2) is crucial for better training, since it is used as Loss function** [2]

Inner-loop optimization → "Training" f_θ on $S_\mathcal{T}$

Support set $S_\mathcal{T}$

Sample K datapoints $\mathcal{D} = \{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\}$ from $\mathcal{T}$
Evaluate $\nabla_\theta \mathcal{L}_\mathcal{T}(f_\theta)$ using $\mathcal{D}$ and $\mathcal{L}_\mathcal{T}$
Compute adapted parameters with gradient descent: $\theta' = \theta - \alpha\nabla_\theta\mathcal{L}_\mathcal{T}(f_\theta)$

Sample datapoints $\mathcal{D}' = \{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\}$ from $\mathcal{T}$ for the meta-update

Query set $Q_\mathcal{T}$

Update $\theta \leftarrow \theta - \beta\nabla_\theta \sum_{\mathcal{T} \sim p(\mathcal{T})} \mathcal{L}_\mathcal{T}(f_{\theta'})$ using each $\mathcal{D}'$ and $\mathcal{L}_\mathcal{T}$

Outer-loop optimization → "Reducing Generalization Error" on $Q_\mathcal{T}$

[1] Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*. PMLR, 2017.
[2] Khodadadeh, S., Bölöni, L., and Shah, M. Unsupervised meta-learning for few-shot image classification. *Advances in neural information processing systems*, 32, 2019.

Model Analysis: Why NaQ can work?

- Theoretical Insights: Which similarity condition should NaQ satisfy?

- Analysis
  - For accurate estimation of Eq. (2), **true** label of query and corresponding support set should be the same
  - Otherwise, unexpected error δ s.t. $y^{qry} = f_{\mathcal{T}}(x^{qry}) + \epsilon + \delta$ can occurs, which lead to "suboptimal solution"
  - Supervised episode generation naturally have $\delta = 0$

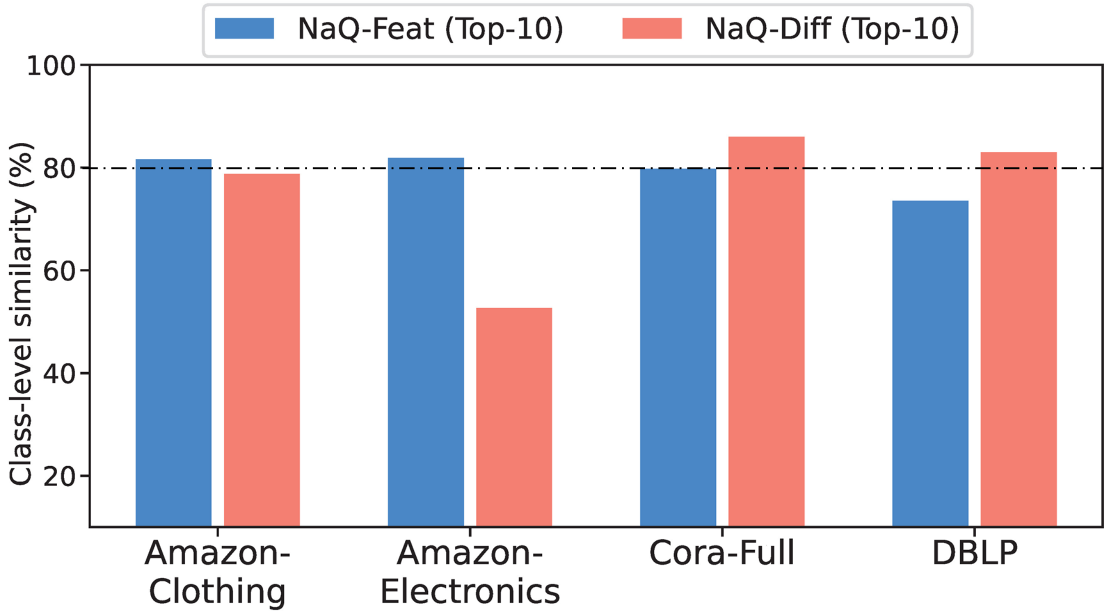
- **Our Claim: "Class-level Similarity" Condition on Queries for Unsupervised Episode Generation**
  - If we can sample "class-level similar" enough queries for each support set node, **undesirable error δ will be small enough**
  - Then, **model f_θ can be trained successfully** with loss function Eq. (2)
  - Therefore, **"Class-level similarity" condition on queries have to be satisfied by NaQ**

$$\mathbb{E}[\mathcal{L}(y^{qry}, f_{\theta'}(x^{qry}))] = \left(\mathbb{E}[f_{\theta'}(x^{qry})] - f_{\mathcal{T}}(x^{qry})\right)^2 + \left(\mathbb{E}\left[f_{\theta'}(x^{qry})^2\right] - \mathbb{E}[f_{\theta'}(x^{qry})]^2\right) + \sigma^2 \quad \cdots (2)$$

Model Analysis: Why NaQ can work?

- Empirical Analysis: NaQ satisfies Class-level Similarity Condition

- Empirical Analysis
  - We measured averaged class-level similarity between each node and top-10 similar nodes found by NaQ
    - Class-level similarity between two nodes: similarity between their **class centroids**
  - In most of cases, **NaQ-Feat and NaQ-Diff** **can discover high enough** (~80%) **class-level similar queries** **in real-world datasets**



Averaged class-level similarity between each node and top-10 similar nodes found via NaQ-Feat and NaQ-Diff

Model Analysis: Why NaQ can work?

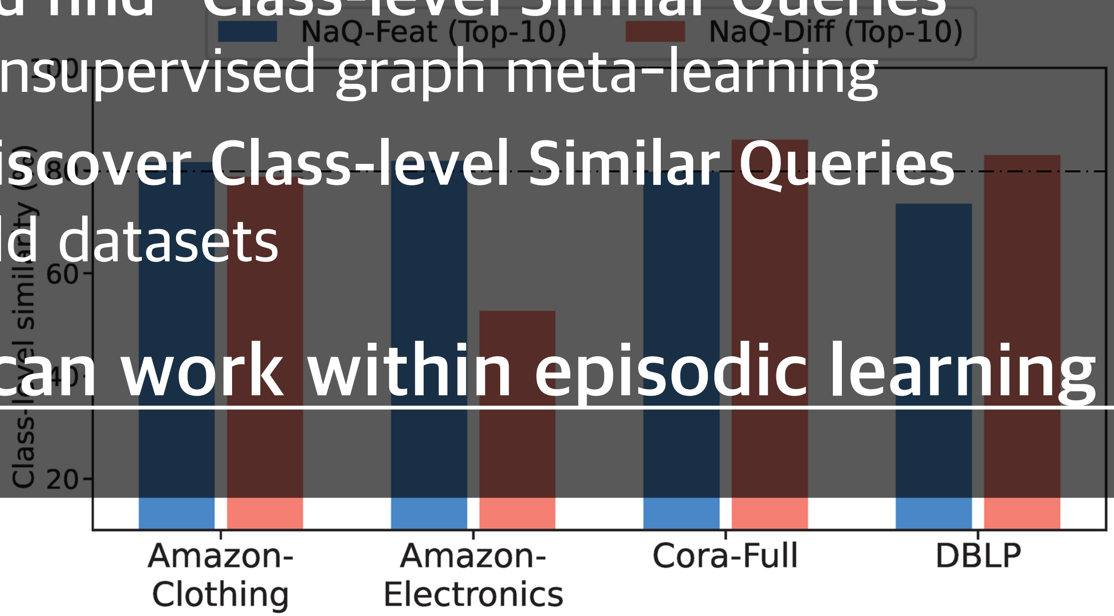
- Empirical Analysis: NaQ satisfies Class-level Similarity Condition

- Empirical Analysis
  - We measured averaged class-level similarity between each node and top-10 similar nodes found by NaQ
    - Class-level similarity between two nodes: <u>similarity between their **class centroids**</u>
  - In most of cases, **NaQ-Feat and NaQ-Diff can discover high enough** (~80%) **class-level similar queries in real-world datasets**

Summary

1) **NaQ should find "Class-level Similar Queries"**
   to enable unsupervised graph meta-learning
2) **NaQ can discover Class-level Similar Queries**
   in real-world datasets

Thus, <u>NaQ can work within episodic learning framework!</u>



Averaged class-level similarity between each node and top-10 similar nodes found via NaQ-Feat and NaQ-Diff

Experiments

- Experimental Settings: Evaluation Datasets

- Evaluation Datasets
  - Total five benchmark datasets were used in evaluation
  - Two product networks (Amazon-Clothing/Electronics) and Three citation networks (Cora-Full, DBLP, ogbn-arxiv) were used
  - 'Class split' means the number of distinct classes used to make episodes in training (supervised only), validation, and testing phase

- Details
  - Amazon-Clothing: edges are 'also-viewed' relationships between products; node class is product category
  - Amazon-Electronics: edges are 'bought-together' relationships between products; node class is product category
  - Node class of Cora-Full: paper topic / DBLP: venue where the paper is published / ogbn-arxiv: subject area in CS papers

Dataset	# Nodes	# Edges	# Features	# Labels	Class split	Hom. ratio
Amazon-Clothing	24,919	91,680	9,034	77	40/17/20	0.62
Amazon-Electronics	42,318	43,556	8,669	167	90/37/40	0.38
Cora-Full	19,793	65,311	8,710	70	25/20/25	0.59
DBLP	40,672	288,270	7,202	137	80/27/30	0.29
ogbn-arxiv	169,343	1,166,243	128	40	15/10/15	0.43

Dataset Statistics

Experiments

- Experimental Settings: Baselines and their Settings

- Compared Baselines
  - Total ten baseline methods were used in evaluation
  - Used six graph meta-learning baselines: MAML, ProtoNet, G-Meta, TENT, GLITTER, and COSMIC
    - MAML, ProtoNet: Representative meta-learning methods
    - G-Meta: Representative Graph meta-learning method
    - TENT, GLITTER, COSMIC: Recent (2022~) Baselines
  - Used three recent (2022~) GCL baselines: BGRL, SUGRL, and AFGRL for the comparison with TLP
  - Lastly, graph transformer-based, unsupervised baseline VNT was used

Experiments

- Results: Overall Performance Analysis

Results in Product Networks

Dataset	Amazon-Clothing					Amazon- Electronics						
Setting	5 way		10 way		Avg.	5 way		10 way		20 way		Avg.
Baselines	1 shot	5 shot	1 shot	5 shot	Rank	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot	Rank
MAML (Sup.)	76.13±1.17	84.28±0.87	63.77±0.83	76.95±0.65	10.25	65.58±1.26	78.55±0.96	57.31±0.87	67.56±0.73	46.37±0.61	60.04±0.52	9.33
ProtoNet (Sup.)	75.52±1.12	89.76±0.70	65.50±0.82	82.23±0.62	7.25	69.48±1.22	84.81±0.82	57.67±0.85	75.79±0.67	48.41±0.57	67.31±0.47	5.83
TENT (Sup.)	79.46±1.10	89.61±0.70	69.72±0.80	84.74±0.59	5.25	72.31±1.14	85.25±0.81	62.13±0.83	77.32±0.67	52.45±0.60	69.39±0.50	4.00
G-Meta (Sup.)	78.67±1.05	88.79±0.76	65.30±0.79	80.97±0.59	7.75	72.26±1.16	84.44±0.83	61.32±0.86	74.92±0.71	50.39±0.59	65.73±0.48	5.67
GLITTER (Sup.)	75.73±1.10	89.18±0.74	64.30±0.79	77.73±0.68	9.00	66.91±1.22	82.59±0.83	57.12±0.88	76.26±0.67	49.23±0.57	61.77±0.52	7.00
COSMIC (Sup.)	82.24±0.99	91.22±0.73	74.44±0.75	81.58±0.63	3.75	72.61±1.05	86.92±0.76	65.24±0.82	78.00±0.64	58.71±0.57	70.29±0.44	3.00
TLP-BGRL	81.42±1.05	90.53±0.71	72.05±0.86	83.64±0.63	4.25	64.20±1.10	81.72±0.85	53.16±0.82	73.70±0.66	44.57±0.54	65.13±0.47	8.67
TLP-SUGRL	63.32±1.19	86.35±0.78	54.81±0.77	73.10±0.63	11.50	54.76±1.06	78.12±0.92	46.51±0.80	68.41±0.71	36.08±0.52	57.78±0.49	11.67
TLP-AFGRL	78.12±1.13	89.82±0.73	71.12±0.81	83.88±0.63	5.25	59.07±1.07	81.15±0.85	50.71±0.85	73.87±0.66	43.10±0.56	65.44±0.48	9.00
VNT	65.09±1.23	85.86±0.76	62.43±0.81	80.87±0.63	10.50	56.69±1.22	78.02±0.97	49.98±0.83	70.51±0.73	42.10±0.53	60.99±0.50	10.83
NaQ-Feat-Best (Ours)	**86.58±0.96**	**92.27±0.67**	**79.55±0.78**	**86.10±0.60**	**1.00**	**76.46±1.11**	**88.72±0.73**	**69.59±0.86**	**81.44±0.61**	**61.05±0.59**	**74.60±0.47**	**1.00**
NaQ-Diff-Best (Ours)	84.40±1.01	91.72±0.69	73.39±0.79	84.82±0.58	2.25	74.16±1.08	87.09±0.75	65.95±0.81	79.13±0.60	60.40±0.59	73.75±0.42	2.00

Results in Large-scale dataset ogbn-arxiv

Dataset	ogbn-arxiv			
Setting	5 way		10 way	
Baselines	1 shot	5 shot	1 shot	5 shot
MAML (Sup.)	40.61±0.89	58.75±0.89	27.32±0.55	43.87±0.56
ProtoNet (Sup.)	43.34±1.01	58.30±0.95	28.17±0.60	46.11±0.60
TENT (Sup.)	48.06±0.97	63.45±0.88	33.85±0.65	48.14±0.59
G-Meta (Sup.)	41.06±0.87	59.43±0.87	27.20±0.53	45.04±0.53
GLITTER (Sup.)	35.64±0.97	34.51±0.85	20.95±0.50	21.84±0.47
COSMIC (Sup.)	50.32±0.95	63.54±0.80	38.41±0.62	49.31±0.51
TLP-BGRL	49.88±1.01	69.10±0.82	36.40±0.62	56.15±0.54
TLP-SUGRL	49.25±0.97	62.15±0.92	32.87±0.61	45.76±0.60
TLP-AFGRL	OOM	OOM	OOM	OOM
VNT	OOM	OOM	OOM	OOM
NaQ-Feat (Ours)	**54.09±1.03**	**69.94±0.84**	**41.61±0.68**	**58.18±0.59**
NaQ-Diff (Ours)	51.45±1.04	66.73±0.89	39.27±0.67	55.93±0.56

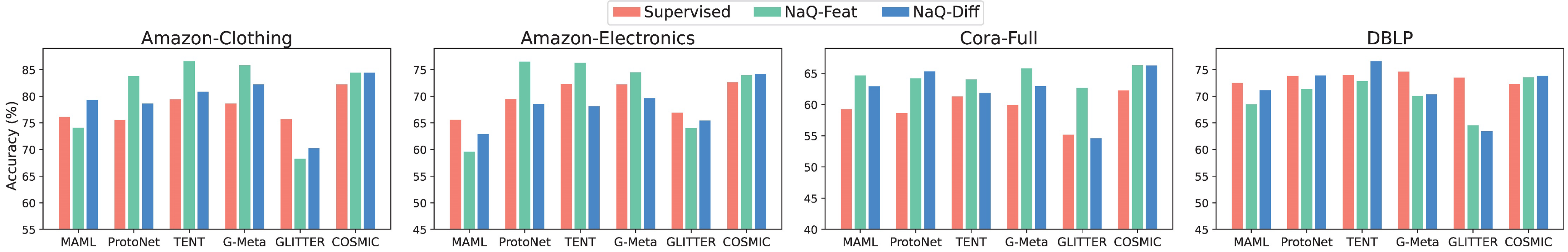
Results in Citation Networks

Dataset	Cora-full							DBLP						
Setting	5 way		10 way		20 way		Avg.	5 way		10 way		20 way		Avg.
Baselines	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot	Rank	1 shot	5 shot	1 shot	5 shot	1 shot	5 shot	Rank
MAML (Sup.)	59.28±1.21	70.30±0.99	44.15±0.81	57.59±0.66	30.99±0.43	46.80±0.38	9.67	72.48±1.22	80.30±1.03	60.08±0.90	69.85±0.76	46.12±0.53	57.30±0.48	8.50
ProtoNet (Sup.)	58.61±1.21	73.91±0.93	44.54±0.79	62.15±0.64	32.10±0.42	50.87±0.40	7.67	73.80±1.20	81.33±1.00	61.88±0.86	73.02±0.74	48.70±0.52	62.42±0.45	4.33
TENT (Sup.)	61.30±1.18	77.32±0.81	47.30±0.80	66.40±0.62	36.40±0.45	55.77±0.39	4.50	74.01±1.20	82.54±1.00	62.95±0.85	73.26±0.77	49.67±0.53	61.87±0.47	2.67
G-Meta (Sup.)	59.88±1.26	75.36±0.86	44.34±0.80	59.59±0.66	33.25±0.42	49.00±0.39	7.50	74.64±1.20	79.96±1.08	61.50±0.88	70.33±0.77	46.07±0.52	58.38±0.47	7.00
GLITTER (Sup.)	55.17±1.18	69.33±0.96	42.81±0.81	52.76±0.68	30.70±0.41	40.82±0.41	11.50	73.50±1.25	75.90±1.19	OOT	OOT	OOM	OOM	9.50
COSMIC (Sup.)	62.24±1.15	73.85±0.83	47.85±0.77	59.11±0.60	42.25±0.43	47.28±0.38	6.33	72.34±1.18	80.83±1.03	59.21±0.80	70.67±0.71	49.52±0.51	59.01±0.42	7.50
TLP-BGRL	62.59±1.13	78.80±0.80	49.43±0.79	67.18±0.61	37.63±0.44	56.26±0.39	3.17	73.92±1.19	82.42±0.95	60.16±0.87	72.13±0.74	47.00±0.53	60.57±0.45	4.83
TLP-SUGRL	55.42±1.08	76.01±0.84	44.66±0.74	63.69±0.62	34.23±0.41	52.76±0.40	6.33	71.27±1.15	81.36±1.02	58.85±0.81	71.02±0.78	45.71±0.49	59.77±0.45	8.17
TLP-AFGRL	55.24±1.02	75.92±0.83	44.08±0.70	64.42±0.62	33.88±0.41	53.83±0.39	7.17	71.18±1.17	82.03±0.94	58.70±0.86	71.14±0.75	45.99±0.53	60.31±0.45	7.83
VNT	47.53±1.14	69.94±0.89	37.79±0.69	57.71±0.65	28.78±0.40	46.86±0.40	11.17	58.21±1.16	76.25±1.05	48.75±0.81	66.37±0.77	40.10±0.49	55.15±0.46	11.17
NaQ-Feat-Best (Ours)	**66.30±1.15**	**80.09±0.79**	**52.23±0.73**	68.87±0.60	**44.13±0.47**	60.94±0.36	**1.33**	73.55±1.16	82.36±0.94	60.70±0.87	72.36±0.73	50.42±0.52	**64.90±0.43**	3.67
NaQ-Diff-Best (Ours)	66.26±1.15	80.07±0.79	52.17±0.74	**69.34±0.63**	44.12±0.47	**60.97±0.37**	1.67	**76.58±1.18**	**82.86±0.95**	**64.31±0.87**	**74.06±0.75**	**51.62±0.54**	64.78±0.44	**1.17**

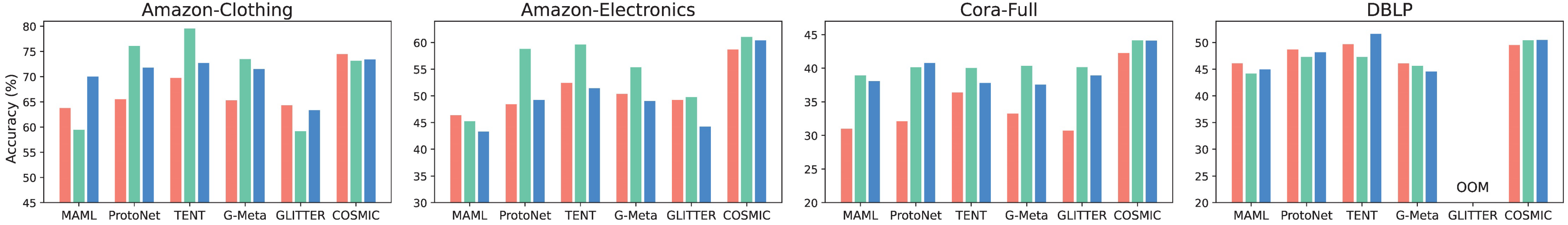
Across all of the settings, proposed NaQ can outperform all the baselines

Experiments

- Results: Model-agnostic Property of NaQ



Results of applying NaQ-Feat and NaQ-Diff to existing graph meta-learning models vs. Supervised (5-way 1-shot)



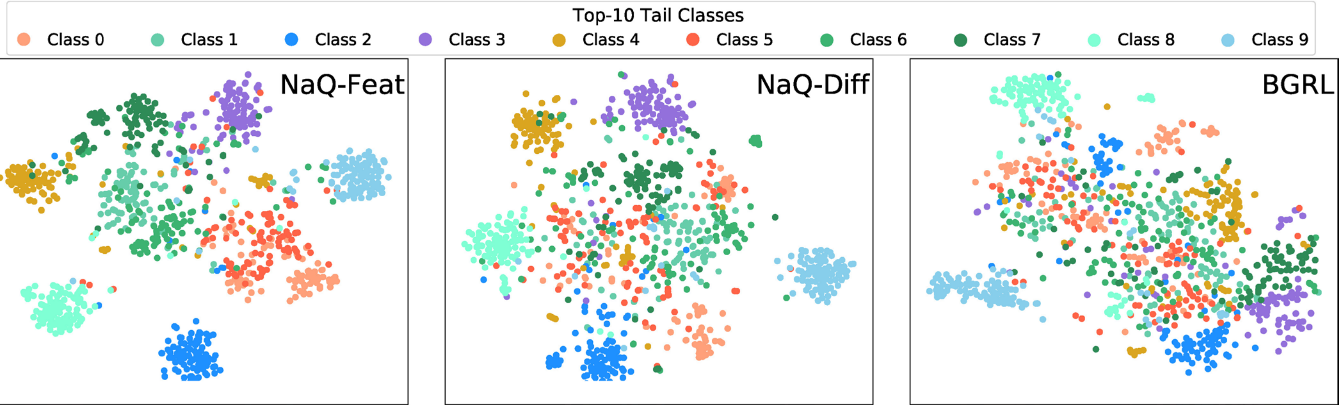
Results of applying NaQ-Feat and NaQ-Diff to existing graph meta-learning models vs. Supervised in Higher way settings
(Amazon Clothing: 10-way 1-shot, Others: 20-way 1-shot)

Generally, proposed NaQ can retain or even improve the performance of graph meta-learning methods

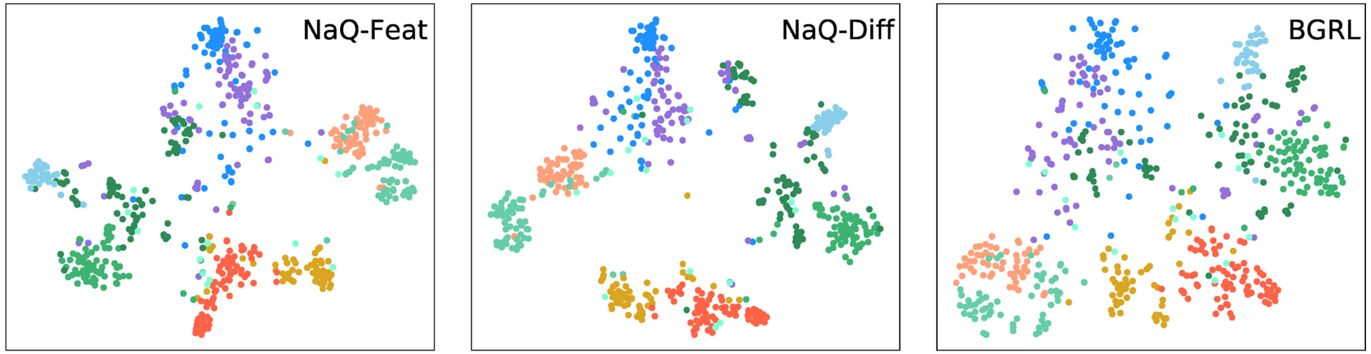
(Note: Supervised methods had access to all, clean labeled samples of entire base classes)

Experiments

- Results: t-SNE Plot of tail-class node embeddings



t-SNE plot of top-10 tail-class node embeddings in Amazon-Electronics Dataset (Product Network)



t-SNE plot of top-10 tail-class node embeddings in Cora-Full Dataset (Citation Network)

NaQ can be more robust to the Class Imbalance in the graph than GCL methods

Experiments

- Additional Empirical Results & Analysis

- Robustness against the Class Imbalance of Graph Meta-learning methods (pp. 40 and 41 in Appendix)
  - NaQ can be robust to the class imbalance since class-level similar queries of tail-class nodes can provide helpful information for learning tail-class node embeddings
  - Downstream task format information obtained by episodic learning is beneficial for attaining robustness

- Impact of Similarity Metric Choice on NaQ-Feat (pp. 42 in Appendix)
  - In summary, proper metric choice is essential for NaQ-Feat

- Impact of the number of queries Q (pp. 43 in Appendix)
  - In summary, when NaQ can find highly class-level similar queries, increasing Q can lead to the better performance

- Regarding Query-overlap Problem of NaQ (pp. 44 in Appendix)
  - Generally, query overlap among distinct query set is negligible for NaQ
  - For some exceptional cases, dropping such overlaps can be a promising solution

Conclusion

- Summary of the dissertation

- Problems of Current Approaches
  - Existing **graph meta-learning methods cannot fully utilize all nodes in the graph**, as they solely rely on the given label information
  - **Naïve application of unsupervised GCL methods on FSNC is vulnerable to Class Imbalance** since there is no information on downstream task format, which also leads to the low generalizability [1] of the trained model when solving downstream tasks

- Solution
  - Proposed NaQ enables the unsupervised graph meta-learning, thus **downstream task format-aware training with all nodes in the graph is allowed**
  - By sampling queries based on pre-calculated node-node similarity, **NaQ can successfully generate training episode that can be applied to existing graph meta-learning methods for their unsupervised training**
  - Extensive experiments and analyses demonstrate effectiveness of our NaQ

[1] Lu, Y., Jiang, X., Fang, Y., and Shi, C. Learning to pre-train graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 4276-4284, 2021.

Conclusion

- Limitation & Future Work

- Computational Issue of NaQ-Diff
  - Current technical issue on sparse matrix multiplication, even truncated approximation of graph Diffusion cannot be computed for datasets having a large number of edges
  - This problem hinders the applicability of NaQ-Diff to large real-world datasets
  - Therefore, **devising an unsupervised episode generation method that can fully leverage the structural information while reducing computational costs will be promising future work**

- Naïve Support set Generation - False-negative Problem
  - NaQ depends on naïve random sampling for support set generation
  - For this reason, there is a possibility that nodes having the same label can be assigned to a distinct support set (False-negative Problem), although NaQ tries to avoid such problem by generating only 1-shot support set
  - Hence, **developing a more sophisticated algorithm that can alleviate the false-negative problem while generate a K-shot ($K \gg 1$) support set will be valuable future work**

Thank you!

Reference

- Full Paper: https://arxiv.org/pdf/2306.15217 / Official Source Code: https://github.com/JhngJng/NaQ-PyTorch
- Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*. PMLR, 2017.
- Snell, J., Swersky, K., and Zemel, R. Prototypical networks for few-shot learning. *Advances in Neural Information Processing Systems*, 30, 2017.
- Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K. and Wierstra, D. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016.
- Kim, S., Lee, J., Lee, N., Kim, W., Choi, S., and Park, C. Task-equivariant graph few-shot learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023.
- Wang, S., Dong, Y., Ding, K., Chen, C. and Li, J. Few-shot node classification with extremely weak supervision. In *Proceedings of the 16th International Conference on Web Search and Data Mining*, 2023.
- Tan, Z., Wang, S., Ding, K., Li, J., and Liu, H. Transductive linear probing: A novel framework for few-shot node classification. In *Learning on Graphs Conference*, 2022.
- Lu, Y., Jiang, X., Fang, Y., and Shi, C. Learning to pre-train graph neural networks. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, pp. 4276-4284, 2021.
- Khodadadeh, S., Bölöni, L., and Shah, M. Unsupervised meta-learning for few-shot image classification. *Advances in neural information processing systems*, 32, 2019.
- Antoniou, A. and Storkey, A. Assume, augment and learn: Unsupervised few-shot meta-learning via random labels and data augmentation. *arXiv preprint arXiv:1902.09884*, 2019.
- Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.
- Huang, K. and Zitnik, M. Graph meta learning via local subgraphs. *Advances in Neural Information Processing Systems*, 33, 2020.
- Wang, S., Ding, K., Zhang, C., Chen, C., and Li, J. Task-adaptive few-shot node classification. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022.
- Gasteiger, J., Weißenberger, S., and Günnemann, S. Diffusion improves graph learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Appendix

- Analysis: Why NaQ can attain robustness against the Class Imbalance?

- Supervised Graph Meta-learning
  - **In a single episode, all classes in base classes are treated equally** regardless of Imbalance
  - With an aid of task format information provided by episodic learning, **supervised graph meta-learning can attain robustness**

- Unsupervised Graph Meta-learning with NaQ
  - **NaQ still can sample "class-level similar" queries to the support set nodes from tail classes**
    - NaQ-Feat can still find high enough similar queries in product networks, while NaQ-Diff find high enough similar queries in citation networks
  - Such **class-level similar queries can provide useful information for learning tail-class embeddings**
  - Also, with task format information provided by episodic learning, **NaQ can attain robustness against Class Imbalance**

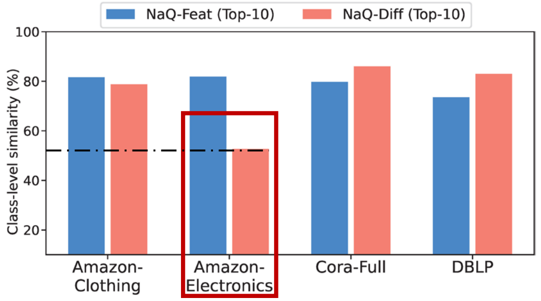
Datasets	**Amazon-Clothing**		**Amazon-Electronics**		**Cora-Full**		**DBLP**	
top-p% tail classes	NaQ-Feat	NaQ-Diff	NaQ-Feat	NaQ-Diff	NaQ-Feat	NaQ-Diff	NaQ-Feat	NaQ-Diff
10%	~78.7%	~75.2%	~72.3%	~48.2%	~69.7%	~77.9%	~66.6%	~75.1%
20%	~81.3%	~78.2%	~74.1%	~51.6%	~70.7%	~77.6%	~68.3%	~78.0%
50%	~81.7%	~80.7%	~77.8%	~53.0%	~74.6%	~81.8%	~70.4%	~80.9%
80%	~80.8%	~79.0%	~78.9%	~52.5%	~77.8%	~84.6%	~71.9%	~82.1%
100%	~81.6%	~78.8%	~81.9%	~52.7%	~79.8%	~86.0%	~73.5%	~83.0%

Averaged class-level similarity between each node from top-p% tail classes and top-10 similar nodes found by NaQ-Feat and NaQ-Diff

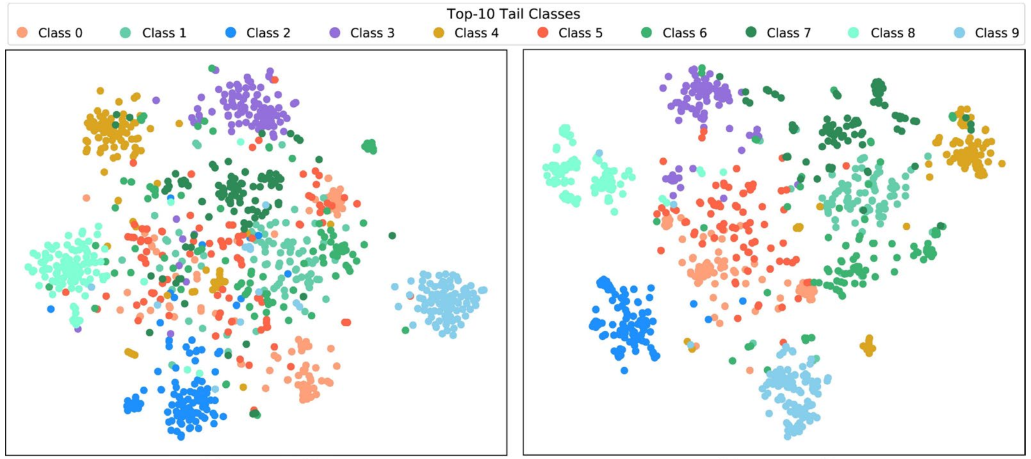
Appendix

- Analysis: Role of the Episodic Learning Framework for attaining robustness against the Class Imbalance

- Is Episodic Learning really beneficial for the Class Imbalance?
  - To demonstrate the effectiveness of downstream task 'format' information provided by episodic learning, we observed the change in tail-class node embedding quality when N-way becomes larger
  - **In Amazon-Electronics, NaQ-Diff have difficulty in finding class-level similar queries**
  - Surprisingly, **training with more challenging episodes** (20-way training episodes) **lead much better tail-class node embedding quality for NaQ-Diff**
  - Therefore, we can conclude that **Episodic Learning does attribute to attain robustness against the Class Imbalance**



Averaged class-level similarity between each node and top-10 similar nodes found via NaQ



(a) NaQ-Diff
(base-model: ProtoNet trained by "5-way" 1-shot episode)

(b) NaQ-Diff
(base-model: ProtoNet trained by "20-way" 1-shot episode)

Impact of higher-way training on tail-class node embedding quality of NaQ-Diff in Amazon-Electronics

Appendix

- Ablation Study: Impact of Similarity Metric Choice on NaQ-Feat

- Similarity Metric Choice of NaQ-Feat
  - Similarity metric is an important factor for NaQ-Feat, as inappropriate choice can lead to wrong selection of queries
  - For datasets having bag-of-words features, Euclidean distance is **inappropriate so thatboth class-level similarity of queries and FSNC performance are degraded**
  - In case of **Jaccard similarity, as it is similar to cosine similarity** when measuring similarities in bag-of-words data, **NaQ-Feat with both similarity metric shows similar FSNC performance**
    - However, Jaccard similarity is cannot be computed with continuous features → cosine similarity is more general
  - In summary, **choosing appropriate similarity metric is important for NaQ-Feat**

Datasets (Feature type: bag-of-words)	Avg. Class-level sim. (Cosine sim.)	Avg. Class-level sim. (Neg. Euclidean dist.)
Amazon-Clothing	$\sim$ 81.6%	$\sim$ 61.0%
Amazon-Electronics	$\sim$ 81.9%	$\sim$ 64.6%
Cora-Full	$\sim$ 79.8%	$\sim$ 40.4%
DBLP	$\sim$ 73.5%	$\sim$ 19.1%

Impact of Similarity Metric Choice on class-level similarity
of top-10 similar nodes found by NaQ-Feat

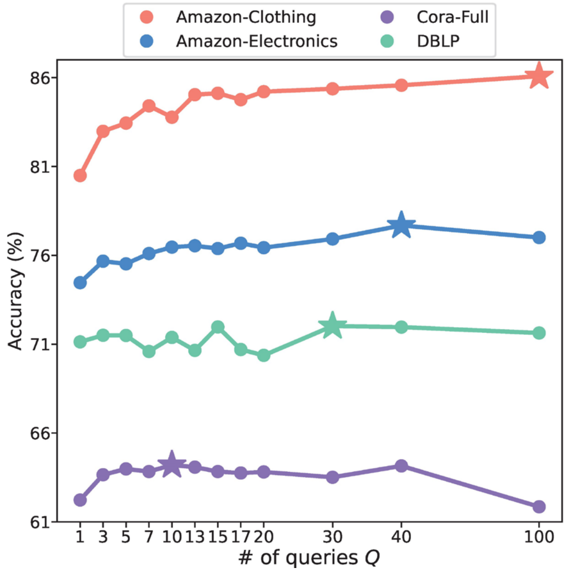
Datasets (Feature type: bag-of-words)	FSNC Accuracy (Cosine sim.)	FSNC Accuracy (Jaccard sim.)	FSNC Accuracy (Neg. Euclidean dist.)
Amazon-Clothing	83.77%	83.35%	80.83%
Amazon-Electronics	76.46%	76.63%	70.68%
Cora-Full	64.20%	63.53%	45.60%
DBLP	71.38%	72.68%	67.53%

Impact of Similarity Metric Choice on FSNC performance
of NaQ-Feat (5-way 1-shot, base-model: ProtoNet)

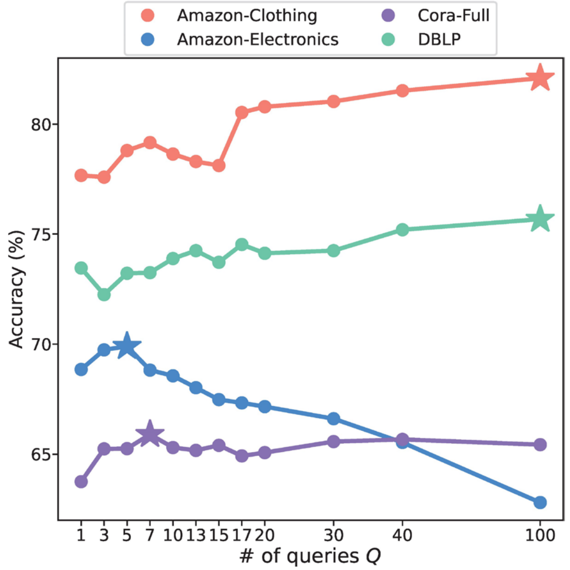
Appendix

- Hyperparameter Sensitivity Analysis: Impact of number of queries Q

- Amazon-Clothing
  - Both NaQ-Feat and NaQ-Diff can discover highly class-level similar queries
    → both show increasing tendency as Q increases
- Amazon-Electronics
  - NaQ-Feat shows increasing tendency as in Amazon-Clothing, due to the same reason
  - NaQ-Feat shows decreasing performance after $Q = 5$, due to relatively low class-level similarity of discovered queries
- DBLP
  - NaQ-Diff shows increasing tendency as Q increases, while NaQ-Feat shows consistent performance by number of queries
- Summary
  - Like the case of NaQ-Diff in Amazon-Electronics, proper choice of Q is essential
    - Otherwise, label noise that can hinder model training can be introduced
  - As NaQ-Diff can find more class-level similar queries than NaQ-Feat in DBLP,
    motivation of utilizing structural neighbors as queries in such datasets is validated



NaQ-Feat



NaQ-Diff

Appendix

- Analysis: Regarding the Query-overlap Problem of NaQ

Datasets	Amazon-Clothing		Amazon-Electronics		Cora-Full		DBLP	
N-way	NaQ-Feat	NaQ-Diff	NaQ-Feat	NaQ-Diff	NaQ-Feat	NaQ-Diff	NaQ-Feat	NaQ-Diff
5	0.1573%	0.9978%	0.0871%	11.1715%	0.2206%	0.4743%	0.1826%	0.0605%
10	0.3855%	2.0769%	0.2118%	16.9618%	0.5101%	1.0138%	0.4108%	0.1389%
20	0.7834%	4.0358%	0.4457%	21.4706%	1.0221%	2.0151%	0.8559%	0.3054%

Averaged query overlap ratio within 16,000 training episodes generated by NaQ

- Query-overlap Problem
  - Situation where sampled query sets corresponding to each distinct support set have intersection can happen for NaQ, which might be problematic during the model training
  - **In real-world datasets, query overlap is generally rare**, as shown in the table above

- Impact of Dropping Query Overlaps
  - When query overlap is significant (NaQ-Diff in Amazon-Electronics), dropping query overlaps have shown remarkable effect
  - However, when query overlap is negligible, dropping queries shows no dramatic improvements on the performance
  - In summary, **query overlap is generally negligible in real-world datasets**, and **dropping query overlaps can be a promising solution for some exceptional cases**

Amazon-Electronics		
Setting	NaQ-Diff (Original ver.)	NaQ-Diff (Overlap drop ver.)
5-way 1-shot	68.56±1.18%	**69.77±1.17%**
10-way 1-shot	59.46±0.86%	**61.98±0.86%**
20-way 1-shot	49.24±0.59%	**52.15±0.60%**

Impact of dropping overlapping queries on NaQ-Diff (When query overlap is significant)

Cora-Full		
Setting	NaQ-Feat (Original ver.)	NaQ-Feat (Overlap drop ver.)
5-way 1-shot	64.20±1.11%	63.37±1.08%
10-way 1-shot	51.78±0.75%	52.32±0.75%
20-way 1-shot	40.11±0.45%	40.27±0.48%

Impact of dropping overlapping queries on NaQ-Feat (When query overlap is negligible)