TAAM:Inductive Graph-Class Incremental Learning with Task-Aware Adaptive Modulation

Jingtao Liu
University of Science and Technology of China
HeFei, China
jingt-liu@mail.ustc.edu.cn

Xinming Zhang*
University of Science and Technology of China
HeFei, China
xinming@ustc.edu.cn

ABSTRACT

Graph Continual Learning (GCL) aims to solve the challenges of streaming graph data. However, current methods often depend on replay-based strategies, which raise concerns like memory limits and privacy issues, while also struggling to resolve the stability-plasticity dilemma. In this paper, we suggest that lightweight, task-specific modules can effectively guide the reasoning process of a fixed GNN backbone. Based on this idea, we propose **T**ask-**A**ware **A**daptive **M**odulation (TAAM). The key component of TAAM is its lightweight Neural Synapse Modulators (NSMs). For each new task, a dedicated NSM is trained and then frozen, acting as an "expert module." These modules perform detailed, node-attentive adaptive modulation on the computational flow of a shared GNN backbone. This setup ensures that new knowledge is kept within compact, task-specific modules, naturally preventing catastrophic forgetting without using any data replay. Additionally, to address the important challenge of unknown task IDs in real-world scenarios, we propose and theoretically prove a novel method named Anchored Multi-hop Propagation (AMP). Notably, we find that existing GCL benchmarks have flaws that can cause data leakage and biased evaluations. Therefore, we conduct all experiments in a more rigorous inductive learning scenario. Extensive experiments show that TAAM comprehensively outperforms state-of-the-art methods across eight datasets.Code and Datasets are available at: https://github.com/1iuJT/TAAM_AAMAS2026.

CCS CONCEPTS

• **Computing methodologies** → **Knowledge representation and reasoning**; **Neural networks**; *Multi-agent planning*; • **Information systems** → **Data mining**.

KEYWORDS

Continual Graph Learning,Graph representation learning,Inductive Learning,Multiple Experts,Graph neural network

ACM Reference Format:
Jingtao Liu and Xinming Zhang. 2026. TAAM:Inductive Graph-Class Incremental Learning with Task-Aware Adaptive Modulation. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 9 pages. https://doi.org/10.65109/WSBG8309

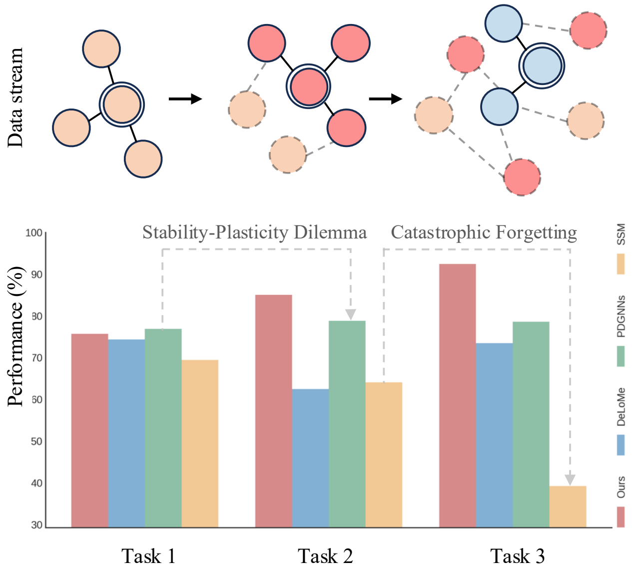


Figure 1: Top: A typical CGL scenario where a model must adapt to a stream of new tasks (indicated by new node colors) in an evolving graph. Bottom: An illustration of the Stability-Plasticity Dilemma on the Citeseer benchmark. While strong replay-based methods effectively mitigate the catastrophic forgetting of early tasks, they exhibit diminished plasticity.

1 INTRODUCTION

While Graph Neural Networks (GNNs) have demonstrated remarkable success on static graphs, real-world graph data is often dynamic and evolving.To address this, Continual Graph Learning (CGL) has become a key research field [23], where models must learn from an evolving stream of tasks while retaining previously acquired knowledge (Figure 1, top). Among CGL paradigms, Graph Class-Incremental Learning (GCIL) presents a particularly challenging and practical frontier. Its core tenet is that task identifiers (IDs) are unavailable during inference, a scenario mirroring real-world applications like drug discovery, where new compound families emerge [30], or social networks, where user communities evolve over time [24]. This task-agnostic requirement introduces a dual challenge: first, overcoming catastrophic forgetting, and second, resolving the "inter-task class separation" problem—the difficulty of distinguishing between similar classes from different tasks without explicit context [21].

---

*Corresponding author.

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/WSBG8309

To address these challenges, researchers have primarily explored three paradigms: regularization-based, replay-based, and parameter-isolation methods [28, 30]. Replay-based approaches, which store and rehearse historical data, can effectively mitigate forgetting but introduce significant computational and storage overhead, along with potential data privacy risks. Critically, they face a fundamental trade-off between preserving old knowledge (stability) and acquiring new information (plasticity)(Figure 1, Bottom).In contrast, parameter-isolation methods offer a more direct path to overcoming forgetting. By dedicating distinct parameters to different tasks, these approaches structurally prevent knowledge interference [28, 32].

Recently, Prompt Learning has emerged as a highly effective parameter-efficient fine-tuning strategy, achieving great success in computer vision [28, 30]. This paradigm was quickly adapted to the graph domain [14, 25], with pioneering works like TPP conditioning a frozen GNN with small, learnable "graph prompts" based on inferred task IDs [18].However, the effectiveness of this paradigm is often predicated on the availability of a powerful, pre-trained GNN backbone to ensure robust knowledge transferability [2, 9]. Moreover, many graph prompting methods employ universal prompts that offer only coarse-grained control over the GNN's internal computational flow, thus limiting their adaptive capacity [18]. This reliance on pre-training presents unique challenges in the graph domain. Unlike in NLP and Computer vision standardized, universal pre-trained GNNs are not yet readily available, and aligning knowledge across heterogeneous graph structures remains a formidable task [6, 22]. This confluence of challenges motivates a fundamental research question: Is it possible to design a CGL approach that eliminates the dependency on a heavily pre-trained model, yet still achieves high performance in a efficient, replay-free manner?

In this paper, we tackle this challenge by rethinking how the stability-plasticity trade-off can be managed in a resource-efficient, replay-free context. Our approach maintains stability by leveraging a frozen GNN backbone and encapsulating task-specific knowledge within lightweight modulators. We argue that the key to unlocking plasticity lies in designing highly expressive, task-specific modules and, critically, a robust mechanism to identify and deploy the correct knowledge for a given task at inference time.

To this end, we introduce **T**ask-**A**ware **A**daptive **M**odulation (TAAM), a novel approach that operationalizes this principle. The architecture of TAAM is composed of two core innovations:

First, for plasticity, we design dynamic and lightweight **N**eural **S**ynapse **M**odulators (NSMs). For each task, a new NSM is created and trained to perform fine-grained, node-attentive modulation on the internal representations of the frozen GNN backbone. These modules act as task-specific experts, effectively steering the GNN's computational flow without altering its stable, foundational parameters.

Second, for stability, we address the critical challenge of task-ID inference. We propose and theoretically ground a novel method named **A**nchored **M**ulti-hop **P**ropagation (AMP) to generate highly robust and discriminative task prototypes. For inference, AMP allows TAAM to accurately identify the task affiliation of incoming data, ensuring the correct expert NSM is selected. This precise retrieval mechanism is fundamental to preventing catastrophic forgetting. The modulated output is then processed by a unified classifier whose output dimension expands for new classes, preserving knowledge at the decision layer.We summarize our main contributions as follows.

- We propose **TAAM**, a novel CGL approach that operates without reliance on data replay or extensive model pre-training, offering a truly resource-efficient solution.
- We introduce and provide theoretical grounding for Anchored Multi-hop Propagation (AMP), a robust method for task-ID inference that ensures model stability and prevents forgetting, validated across eight datasets.
- We propose a lightweight, and node-attentive modulator (NSM) that effectively incorporates task-specific knowledge and provides more expressive, fine-grained control over GNN representations than traditional static prompting methods.
- We identify limitations in existing GCL benchmarks and are the first to validate our approach in the more challenging and realistic inductive graph continual learning scenario, demonstrating its practical effectiveness.

2 RELATED WORKS

Replay-based Methods. A dominant paradigm in Continual Graph Learning (CGL) is the use of replay-based methods, which maintain a memory buffer of representative exemplars from past tasks. During training on a new task, these stored exemplars are rehearsed alongside the new data, mitigating catastrophic forgetting through joint optimization [12, 13, 29, 31]. The sophistication of what is stored has evolved, from representative nodes [33] to more complex structures. For instance, CaT [12] and PUMA [13] focus on graph condensation techniques to create small, synthetic graphs for replay. Others compress topological information, either by storing sparsified computation subgraphs (SSM [31]) or by decoupling the model from the graph via compressed topology-aware embeddings (PDGNNs [29]). DeLoMe [17] constructs a memory composed of lossless node prototypes, aiming to fully capture and store the graph information of old tasks, and further proposes a de-biased GCL loss function to address category imbalance. While these methods often demonstrate strong performance, their reliance on storing and re-processing historical data introduces significant computational and memory overhead, along with potential data privacy concerns.

Prompt Learning and Parameter-Efficient Methods. An alternative direction is rooted in Parameter-Efficient Transfer Learning (PETL), a paradigm where large pre-trained models are adapted to new tasks by updating only a small subset of parameters. Representative methods like Adapters achieve this by inserting lightweight modules between the layers of a backbone network to reduce fine-tuning costs [3, 5]. Building on this foundation, prompt learning has emerged as a leading PETL technique. It guides a model's behavior for specific tasks by injecting learnable prompts," and is widely adopted for its excellent few-shot and transferability performance under thepre-train, prompt, predict" framework [11]. Inspired by its success, researchers have recently extended this paradigm to Graph Neural Networks (GNNs). For instance, UPT conditions a frozen GNN on structural prompts to enhance few-shot and cross-domain performance [2], while GraphPrompt unifies pre-training and downstream tasks through a shared interface to improve generalization [14]. Other works like TPP infers task IDs to select the appropriate task-specific prompt for conditioning the GNN [18]. While these methods achieve impressive parameter-efficient knowledge transfer, their effectiveness critically depends on the availability of a powerful, often resource-intensive pre-trained backbone. This prerequisite poses a significant challenge in the graph domain, where universally effective and readily available pre-trained GNNs are not as common as in other fields like NLP or vision. This limitation motivates our research into a new paradigm that delivers the parameter efficiency of prompt-based conditioning, but without the dependency on a pre-trained model, thereby offering a more flexible and accessible solution for continual graph learning.

3 PRELIMINARY

3.1 Problem Formulation

The work of this paper is oriented towards Graph Class-Incremental Learning (GCIL), where a model must continually learn new classes of nodes from a sequence of tasks.

Formally, we are given a sequence of N tasks, $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_N\}$. Each task τ_k introduces a new set of node classes C_k, where all class sets are disjoint, i.e., $C_i \cap C_j = \emptyset$ for any $i \neq j$. After learning up to task τ_k, the model f must be able to classify nodes from the union of all observed classes, $C_{1:k} = \bigcup_{i=1}^{k} C_i$. A critical constraint in the GCIL setting is that the task identity of data is **unavailable** during inference.

Unlike earlier GCL benchmarks [30], whichoperate in a transductive scenario (where training and testing nodes reside within the same graph), our study tackles the more demanding and practical **inductive** setting. Here, training and testing data for each task are fully isolated into independent subgraphs. Specifically, for each task τ_k, the model is trained on a training set of graphs $\mathcal{G}\text{train}^k$ and evaluated on a distinct set of unseen test graphs $\mathcal{G}\text{test}^k$, where $\mathcal{G}\text{train}^k \cap \mathcal{G}\text{test}^k = \emptyset$. This setup challenges the model to generalize to new classes as well as to entirely novel graph structures and node instances that it has never encountered during training.

3.2 Graph Neural Networks

A GNN computes the hidden representation for a node $h_i^{(l)}$ at a given layer l by aggregating features from its neighbors $\mathcal{N}(i)$[7]. This process is generally defined by applying a transformation matrix $\mathbf{W}^{(l)}$ and a non-linear activation function $\sigma(\cdot)$:

$$h_i^{(l)} = \sigma\left(\sum_{j \in \mathcal{N}(i)} \mathcal{A}_{ij} h_j^{(l-1)} \mathbf{W}^{(l)}\right) \tag{1}$$

Here, $\mathcal{A}$ is a matrix defining the aggregation strategy, and $h_i^{(0)}$ is the initial node feature. SGC [26] streamlines the GNN architecture by removing intermediate non-linear activation functions. This decouples the model into two distinct stages:

(1) *Neighborhood Aggregation* The initial node features $\mathbf{X}$ are aggregated over a K-hop neighborhood by being multiplied by the K-th power of the normalized adjacency matrix $\mathbf{S}$. This creates a fixed, pre-processed feature matrix $\mathbf{X}'$:

$$\mathbf{X}' = \mathbf{S}^K \mathbf{X} \tag{2}$$

(2) *Feature Transformation:* A trainable Multi-Layer Perceptron (MLP) is then applied to the propagated features $\mathbf{X}'$.:

$$\hat{\mathbf{Y}} = \text{softmax}\left(\sigma\left(\mathbf{X}'\mathbf{W}_1\right)\mathbf{W}_2\right) \tag{3}$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ are the trainable weight matrices of the MLP, and $\sigma(\cdot)$ is a non-linear activation function, such as ReLU.

4 METHODOLOGY

The proposed Task-Aware Adaptive Modulation (TAAM) framework offers a replay-free and resource-efficient solution for continual learning on graphs. TAAM utilizes a frozen Simple Graph Convolution (SGC) backbone as a stable feature extractor. For every new task in the sequence $\mathcal{T} = \tau_1, \tau_2, \ldots, \tau_N$, it introduces a lightweight, task-specific Neural Synapse Modulator (NSM) to dynamically adapt the backbone's output. Each task is assigned its own NSM ($\text{NSM}_1, \text{NSM}_2, \ldots, \text{NSM}_N$), as illustrated in Figure 2. At inference time, the system selects the appropriate NSM using robust task-ID inference, which is based on prototypes created by the Anchored Multi-hop Propagation (AMP) technique.

4.1 Anchored Multi-hop Propagation for Task-ID Inference

A critical component for stability in a continual learning setting without task labels at inference is the ability to accurately identify the task to which incoming data belongs. To achieve this, we propose generating a single, highly representative prototype vector $\mathbf{p}_k$ for each task τ_k.

We introduce **A**nchored **M**ulti-hop **P**ropagation (AMP), a method designed to create robust prototypes by capturing multi-scale neighborhood information. AMP leverages the principles of Approximate Personalized PageRank (APPNP) [8], which diffuses node features across the graph while retaining a connection to the initial features via a teleport probability α. The propagation rule for the node feature matrix $\mathbf{H}$ is:

$$\mathbf{Z}^{(i+1)} = (1-\alpha)\mathbf{S}\mathbf{Z}^{(i)} + \alpha\mathbf{H}^{(0)} \tag{4}$$

where $\mathbf{Z}^{(0)} = \mathbf{H}^{(0)}$ are the initial node features, and $\mathbf{S}$ is the symmetrically normalized adjacency matrix.

Inspired by SIGN [20], we create a multi-scale representation by concatenating the diffused features from different propagation depths (hops). The final enriched feature matrix for prototype generation, $\mathbf{H}_{\text{AMP}}$, is formed as:

$$\mathbf{H}_{\text{AMP}} = \text{CONCAT}(\mathbf{Z}^{(h_1)}, \mathbf{Z}^{(h_2)}, \ldots, \mathbf{Z}^{(h_m)}) \tag{5}$$

where $\{h_1, \ldots, h_m\}$ is the set of selected hops. The prototype $\mathbf{p}_k$ for task τ_k is then the mean of these enriched features over its training nodes $\mathcal{V}_k^{\text{train}}$:

$$\mathbf{p}_k = \frac{1}{|\mathcal{V}_k^{\text{train}}|} \sum_{i \in \mathcal{V}_k^{\text{train}}} (\mathbf{H}_{\text{AMP}})_i \tag{6}$$

During inference, we generate a prototype for the test data and identify the task ID by finding the most similar stored prototype using cosine similarity. The pseudocode for AMP is detailed in Algorithm 1.

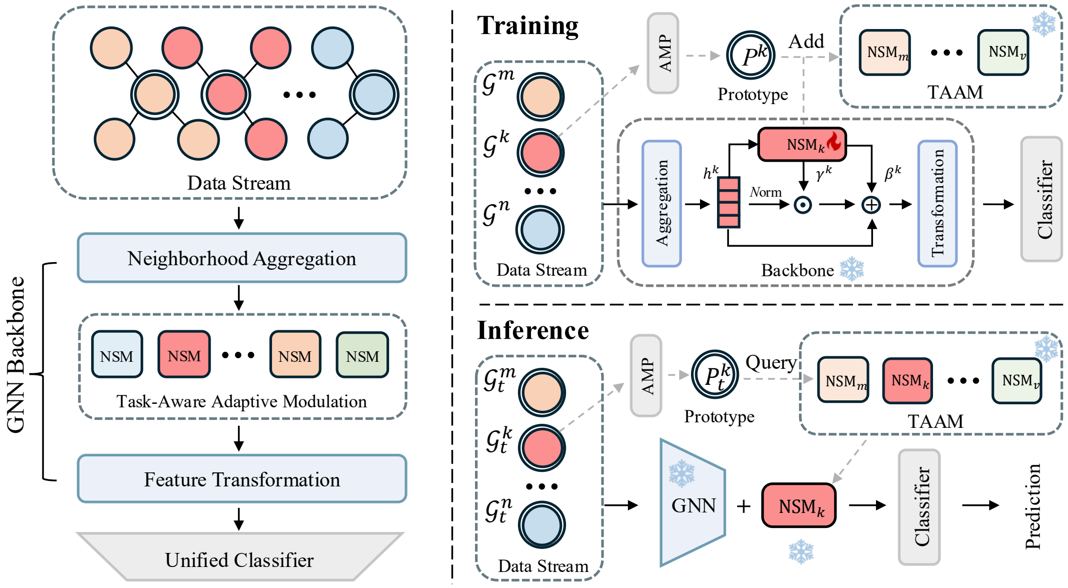


Figure 2: Overview of TAAM framework.TAAM is inserted the frozen GNN backbone,to steer a GNN's reasoning process in stream of graph data.The diagram illustrates the strategy of TAAM.Training Phase: For a new task $\mathcal{G}_k$, a new Neural Synapse Modulator (NSM$_k$) is created from scratch and trained. Concurrently, its corresponding task prototype P_k is generated and stored.After training is complete, NSM$_k$ frozen and added to the TAAM module bank.Inference Phase:For incoming test graph $\mathcal{G}_t^k$ with an unknown task ID , a prototype P_t^k generated using AMP,for selects the most relevant "expert" NSM.

```
Algorithm 1: Anchored Multi-hop Propagation
  Input: Graph $\mathcal{G}$, Initial features $\mathbf{H}^{(0)}$, Training node set
    $\mathcal{V}_k^{\text{train}}$, AMP parameters $(L_{hops}, \alpha, K)$;
  Output: Task prototype vector $\mathbf{p}_k$;
  begin
    $\mathcal{H}_{collect} \leftarrow \emptyset$; $\mathbf{Z} \leftarrow \mathbf{H}^{(0)}$; if $0 \in L_{hops}$ then
      $\mathcal{H}_{collect} \leftarrow \mathcal{H}_{collect} \cup \{\mathbf{Z}\}$;
    end
    for $i \leftarrow 1$ to $K$ do
      $\mathbf{Z} \leftarrow (1-\alpha)\mathbf{S}\mathbf{Z} + \alpha\mathbf{H}^{(0)}$; if $i \in L_{hops}$ then
        $\mathcal{H}_{collect} \leftarrow \mathcal{H}_{collect} \cup \{\mathbf{Z}\}$;
      end
    end
    $\mathbf{H}_{\text{AMP}} \leftarrow \text{CONCAT}(\mathcal{H}_{collect})$;
    $\mathbf{p}_k \leftarrow \frac{1}{|\mathcal{V}_k^{\text{train}}|} \sum_{v \in \mathcal{V}_k^{\text{train}}} (\mathbf{H}_{\text{AMP}})_v$;
    return $\mathbf{p}_k$;
  end
```

4.2 Neural Synapse Modulator

For each task τ_k, we introduce a new, lightweight **N**eural **S**ynapse **M**odulator (NSM). The NSM acts as a task-specific expert that adapts the behavior of the frozen SGC backbone. It is a dynamic, node-attentive module that generates Feature-wise Linear Modulation (FiLM) [1, 19] parameters, $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$, for each node.

Specifically, the NSM for task τ_k contains a unique, learnable task embedding $\mathbf{e}_k \in \mathbb{R}^{D_{task}}$. To generate modulation parameters efficiently, we use a low-rank factorization. First, a set of H base modulation vectors for H attention heads, $\mathbf{M}_k \in \mathbb{R}^{H \times 2F'}$, is generated from the task embedding:

$$\mathbf{M}_k = (\mathbf{W}_A \mathbf{e}_k) \cdot \mathbf{W}_B \tag{7}$$

where $\mathbf{W}_A$ and $\mathbf{W}_B$ are learnable low-rank projection matrices.

Next, for each node i with feature vector $\mathbf{h}_i$, we compute attention weights over the H heads:

$$\mathbf{a}_i = \text{softmax}(\mathbf{h}_i \mathbf{W}_{\text{att}}) \tag{8}$$

where $\mathbf{W}_{\text{att}} \in \mathbb{R}^{F \times H}$. The final per-node FiLM parameters are a weighted combination of the base modulations:

$$[\boldsymbol{\gamma}_i, \boldsymbol{\beta}_i] = \mathbf{a}_i^\top \mathbf{M}_k \tag{9}$$

These parameters are then used to modulate the node's feature vector $\mathbf{h}_i$ after the SGC's neighborhood aggregation step, followed by a residual connection:

$$\mathbf{h}'_i = (\boldsymbol{\gamma}_i \odot \text{LayerNorm}(\mathbf{h}_i) + \boldsymbol{\beta}_i) + \mathbf{h}_i \tag{10}$$

This node-attentive mechanism allows the NSM to provide fine-grained, context-specific adaptation of the backbone's representations.

4.3 Unified Classifier and Training Objective

To accommodate the expanding set of classes, we employ a unified linear classifier that grows dynamically. Upon the arrival of a new task τ_k with class set C_k, the classifier's weight matrix $\mathbf{W}_{k-1} \in \mathbb{R}^{D_{hid} \times C_{k-1}}$ is expanded to a new matrix $\mathbf{W}_k \in \mathbb{R}^{D_{hid} \times C_k}$, where $C_k = C_{k-1} + |C_k|$. To preserve prior knowledge, the weights

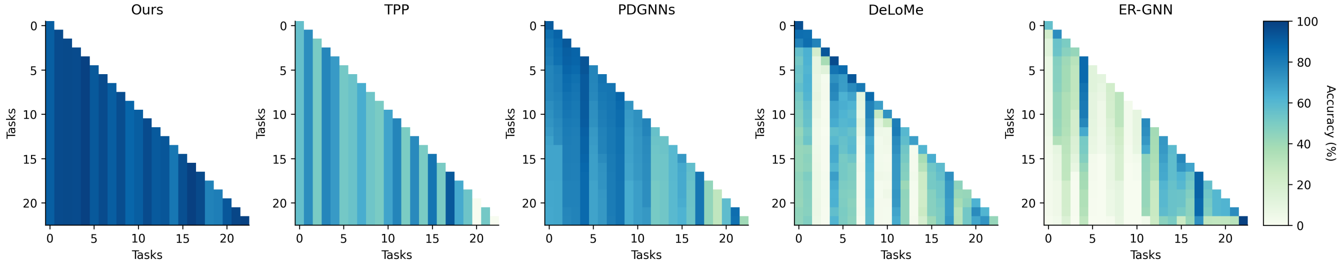


Figure 3: Performance matrices of differernt method on Products dataset.Darker colors indicate higher accuracy. TAAM (Ours) demonstrates consistently high accuracy across all tasks, indicating zero catastrophic forgetting.

for existing classes are copied directly from $\mathbf{W}_{k-1}$, while only the weights corresponding to the new classes are randomly initialized.

During the training phase for task τ_k, we freeze all parameters except for those of the new NSM and the newly added weights in the classifier. This design strictly isolates parameter updates and prevents catastrophic forgetting at the decision layer.

To mitigate class imbalance, our training objective for task τ_k is a weighted cross-entropy loss. The weight for each class $c \in C_k$ is its inverse frequency in the training data, $w_c = 1/N_c$, where N_c is the number of training instances for class c. Crucially, the loss is computed exclusively over the set of new classes C_k:

$$\mathcal{L}_k = -\sum_{v\in\mathcal{V}_k^{\text{train}}} w_{y_v} \log\left(\frac{\exp(\mathbf{z}_{v,y_v})}{\sum_{c'\in C_k}\exp(\mathbf{z}_{v,c'})}\right) \tag{11}$$

where $\mathcal{V}_k^{\text{train}}$ are the training nodes for task τ_k, $\mathbf{z}_v$ is the full logit vector for node v, and $y_v \in C_k$ is its corresponding label.

5 EXPERIMENTS

In this section, we conduct a comprehensive empirical evaluation to validate the effectiveness and efficiency of our proposed Task-Aware Adaptive Modulation (TAAM) framework. Our experiments are designed to answer four key research questions:

Q1: How does TAAM perform against state-of-the-art (SOTA) Graph Continual Learning (GCL) methods in terms of accuracy and catastrophic forgetting?

Q2: How efficient is TAAM concerning computational time, memory usage, and parameter overhead?

Q3: How crucial is TAAM's task-ID inference mechanism, and how does it compare to existing approaches?

Q4: What is the specific contribution of each core component within the TAAM architecture?

5.1 Experimental Setup

Datasets and Baselines. We perform evaluations on eight benchmark datasets, which span several domains. Specifically, these include three citation networks: CoraFull [16], Arxiv [6], and Citeseer [27]; one social network: Reddit [4]; three product co-purchase networks: Products [6], Photo [15], and Computer [15]; and one webpage-based network: WikiCS [10], with statistics detailed in Table 2. Our baselines cover the spectrum of GCL strategies,including regularization, replay-base, and parameter isolation leading methods. We use Joint training and Oracle models as upper bounds.

Implementation Details. To ensure a rigorous and practical evaluation, our experimental setup adapts the CGL benchmark [30] to a strict **inductive learning scenario**. In this setting, the training, validation, and test sets are constructed from entirely disjoint subgraphs. This deliberate separation is a crucial correction to prior evaluation protocols, as it prevents any potential data leakage from the test distribution during the training phase. Consequently, this setup guarantees a more realistic and unbiased assessment of a model's generalization capabilities in a continual learning environment.

For a fair comparison across all methods, we employ a unified backbone feature extractor: a two-layer Simplifying Graph Convolutional Network (SGC) [26] with a 256-dimensional hidden layer. The SGC is configured without batch normalization, dropout, or bias. Its parameters are randomly initialized and subsequently frozen throughout the entire continual learning process, ensuring that all performance gains come from the continual learning modules themselves.

All models were trained for 200 epochs per task using the Adam optimizer, with a learning rate of 0.005 and a weight decay of 5×10^{-4}. We use the same set of random seeds for all experiments to ensure reproducibility. Hyperparameters for all baseline methods were configured according to their original publications. For our proposed model, TAAM, the dimension of the learnable task embedding, $\mathbf{e}_\tau$, is set to 6. The internal node-attentive mechanism in each Node-Specific Modulator (NSM) uses $K = 2$ heads. Critically, for each new task, only the parameters of the newly instantiated NSM and its corresponding classifier are trainable, embodying our lightweight learning paradigm.

Evaluation Metrics. We evaluate performance using Average Accuracy (AA) and Average Forgetting (AF), where a higher AA indicates better overall accuracy and a lower AF indicates less catastrophic forgetting.

5.2 Main results

*State-of-the-Art Performance (**Q1**).* As shown in Table 1, TAAM establishes a new state-of-the-art by achieving the highest Average Accuracy (AA) across all eight datasets while simultaneously eliminating catastrophic forgetting (0.0% AF). This superiority stems from its unique architectural design, which overcomes the limitations of prior paradigms. Unlike replay-based methods, TAAM avoids their inherent computational and privacy overhead while

Table 1: Comparison of different models on different data sets in the GCIL setting. The bold results are the best performance excluding Joint and Oracle. A '↑' indicates that a greater value represents better performance, while a '↓' indicates the opposite. 'Joint' can access the data of all tasks; 'Oracle' can access the data and task IDs of all tasks.

Category	Model	CoraFull		Arxiv		Reddit		Products		Photo		WikiCS		Computer		Citeseer	
		AA%(↑)	AF%(↓)	AA%(↑)	AF%(↓)	AA%(↑)	AF%(↓)	AA%(↑)	AF%(↓)	AA%(↑)	AF%(↓)	AA%(↑)	AF%(↓)	AA%(↑)	AF%(↓)	AA%(↑)	AF%(↓)
Lower bound	Finetuning	1.8±0.2	13.7±2.5	4.9±0.0	11.8±4.1	4.9±0.0	4.9±0.0	4.7±0.8	31.1±1.6	23.2±0.0	38.6±6.5	18.8±0.1	41.5±7.9	19.0±0.1	28.1±12.0	31.4±0.1	52.3±5.3
	LwF(2017)	1.8±0.2	18.2±2.1	4.9±0.0	30.8±4.8	5.6±0.9	58.4±5.3	21.4±0.9	33.4±0.7	12.5±8.0	46.2±3.1	14.7±1.2	78.4±6.0	8.5±7.8	41.8±32.0	31.4±0.1	70.7±0.8
Regularisation	MAS(2018)	2.8±0.3	67.4±1.4	4.8±0.0	80.9±4.2	5.0±0.1	97.5±0.1	3.9±0.3	88.7±0.4	18.6±0.1	92.3±2.0	18.6±0.1	92.3±1.8	23.1±0.2	87.3±14.2	29.9±0.1	80.9±0.4
	TWP(2021)	7.2±0.9	65.7±1.3	4.9±0.0	60.5±5.3	5.0±0.3	92.8±1.5	5.5±0.2	78.3±1.7	23.2±0.1	73.8±7.5	18.6±0.1	89.2±2.5	19.2±0.3	64.7±15.4	36.5±0.4	66.9±1.2
	ER-GNN(2021)	2.1±0.0	14.1±1.8	16.3±2.7	40.4±2.4	85.0±0.8	12.3±0.8	22.5±0.9	33.1±1.2	94.3±0.2	2.4±0.2	84.1±0.3	7.3±0.5	87.6±0.6	10.6±0.9	61.3±0.4	34.5±0.6
Replay	SSM(2022)	60.3±0.6	11.2±0.3	7.7±2.1	8.8±2.5	91.9±0.3	1.5±0.2	32.9±1.5	15.3±1.6	92.6±0.7	1.9±0.5	74.3±1.6	4.4±1.1	88.7±0.9	3.1±0.8	47.1±4.5	6.5±2.2
	PDGNNs(2024)	49.8±27.0	11.5±7.9	44.5±0.2	14.2±0.1	94.9±0.1	1.4±0.0	45.2±0.1	7.1±0.4	94.6±0.3	2.2±0.2	84.2±0.5	3.8±0.4	91.4±0.8	5.5±0.9	71.0±0.1	10.9±0.1
	DeLoMe(2024)	30.5±0.1	47.0±0.5	31.0±0.2	24.1±0.2	94.1±0.6	0.2±0.1	37.8±0.5	35.8±0.2	47.4±0.2	64.9±0.3	35.6±0.3	67.1±0.3	37.1±0.1	73.7±0.2	48.4±0.2	32.3±0.2
Parameter isolation	TPP(2024)	88.9±0.8	0.0±0.0	84.4±1.0	0.0±0.0	98.5±0.2	0.0±0.0	84.9±0.2	0.0±0.0	23.3±0.7	0.0±0.0	36.6±0.5	0.0±0.0	19.9±0.1	46.2±0.5	77.7±1.6	0.0±0.0
	TAAM (Ours)	**92.5±0.0**	**0.0±0.0**	**88.6±0.4**	**0.0±0.0**	**99.0±0.0**	**0.0±0.0**	**92.9±0.5**	**0.0±0.0**	**96.2±0.4**	**0.0±0.0**	**92.8±0.3**	**0.0±0.0**	**97.5±0.6**	**0.0±0.0**	**83.5±1.0**	**0.0±0.0**
Upper bound	Joint	64.1±0.3	-	45.2±0.1	-	95.6±0.1	-	65.2±0.3	-	95.1±0.5	-	83.7±0.3	-	94.9±0.1	-	70.7±0.1	-
	Oracle	94.3±0.3	-	88.5±0.3	-	99.2±0.0	-	94.0±0.6	-	97.0±0.1	-	92.9±0.2	-	97.7±0.1	-	85.2±0.2	-

Table 2: Statistics of the datasets.

Dataset	Nodes	Edges	Feats.	Classes	Tasks
Citeseer	3,327	9,228	3,703	6	3
Photo	48,362	500,928	745	8	4
WikiCS	11,701	431,726	300	10	5
Computer	13,752	491,722	767	10	5
CoraFull	19,793	130,622	8,710	70	35
Arxiv	169,343	1,166,243	128	40	20
Reddit	232,965	114,615,892	602	40	20
Products	2,449,029	61,859,036	100	47	23

still delivering dominant performance on large-scale graphs like Products (92.9% vs. 45.2% for PDGNNs). More critically, its advancement over other parameter-isolation methods like TPP is driven by its novel Task-Aware module and fine-grained Neural Synapse Modulators (NSMs). This mechanism provides superior adaptability, leading to remarkable performance gains such as the 77.6% improvement on the Computer dataset. Ultimately, TAAM presents a more effective solution to the **stability-plasticity trade-off**: its core framework guarantees perfect stability (zero forgetting), while its adaptive modulators deliver SOTA plasticity.

Visualizing Knowledge Retention. To provide a more intuitive understanding of this zero-forgetting property, we visualize the model's task-wise performance after the final training phase. As depicted in Figure 3, the heatmaps vividly illustrate our method's superior knowledge retention. The matrix for TAAM (Ours) exhibits a consistently dark and uniform color across the lower triangle. This signifies both excellent plasticity (the dark diagonal, indicating high accuracy on newly learned tasks) and perfect stability (no color degradation in the off-diagonal elements, confirming zero forgetting). In stark contrast, the heatmaps for replay-based methods like DeLoMe and regularization-based ER-GNN reveal a clear pattern of knowledge decay. While their diagonals indicate an ability to learn, the pronounced color fading down each column provides clear visual evidence of catastrophic forgetting, corroborating the high AF% values reported in Table 1.

5.3 Efficiency Analysis

*Efficiency Analysis (**Q2**).* A core design philosophy of TAAM is resource efficiency, a critical factor for practical GCL deployment. We analyze this through the lens of the performance-cost trade-off, considering parameters, memory, and time.

As illustrated in Figure 5, the optimal efficiency profile—high accuracy with low computational cost—resides in the top-left quadrant. **TAAM is consistently positioned in this optimal region**, achieving state-of-the-art accuracy with minimal running time and memory footprint. However, this plot only reveals part of the story. For instance, while TPP appears competitive in inference time, this view conceals the **heavy computational overhead of its mandatory pre-training phase**. Similarly, some methods are omitted entirely for clarity due to their extreme inefficiency; **DeLoMe**, for example, requires substantial time to generate node prototypes, with a single experiment on the large-scale Reddit dataset consuming approximately one hour.

Parameter Efficiency. This computational efficiency is rooted in our model's design. As shown in Table 4, TAAM demonstrates exceptional parameter efficiency by introducing only 0.02M - 3.08M additional parameters per task (measured by the number of float32 values), which is orders of magnitude lower than methods like ER-GNN and PDGNNs. This makes TAAM a truly lightweight solution. For scenarios where peak performance is paramount, our variant **TAAM-L** offers a flexible trade-off, further boosting accuracy with only a modest increase in parameters (Table 3), demonstrating the adaptability of our framework to different efficiency-performance requirements.

5.4 Analysis of Core Components

*Superior Task-ID Inference (**Q3**).* Reliable task-ID inference is the foundation of our replay-free approach. We compare our Anchored Multi-hop Propagation (AMP) against the Laplacian smoothing (LS) method used in TPP. Figure 4 shows that AMP achieves near-perfect task-ID accuracy across diverse datasets, significantly outperforming LS. This robust inference capability prevents error propagation and is key to TAAM's stability and consistent performance.

Table 3: Ablation results of TAAM and its variants.

Ablated Components	CoraFull		Arxiv		Reddit		Products		Photo		WikiCS		Computer		Citeseer	
	AA%(↑)	AF%(↓)	AA%(↑)	AF%(↓)	AA%(↑)	AF%(↓)	AA%(↑)	AF%(↓)	AA%(↑)	AF%(↓)	AA%(↑)	AF%(↓)	AA%(↑)	AF%(↓)	AA%(↑)	AF%(↓)
w/o Task-Aware	2.0±0.3	5.0±0.9	4.9±0.0	14.8±2.4	5.3±0.9	27.9±4.6	2.7±0.5	11.6±3.0	21.2±0.8	55.6±10.5	18.9±0.1	32.1±14.2	18.9±0.1	32.3±14.6	29.7±0.6	37.4±14.4
w/o NSMs	5.8±0.8	0.0±0.0	7.8±2.8	0.0±0.0	9.0±3.7	0.0±0.0	8.8±2.0	0.0±0.0	30.6±7.2	0.0±0.0	23.5±9.1	0.0±0.0	25.5±9.0	0.0±0.0	31.7±4.5	0.0±0.0
w/o Unified Classifier	92.2±0.5	0.0±0.0	88.3±0.4	0.0±0.0	99.0±0.0	0.0±0.0	93.0±0.5	0.0±0.0	95.7±0.7	0.0±0.0	92.9±0.2	0.0±0.0	97.7±0.5	0.0±0.0	82.0±0.8	0.0±0.0
TAAM	92.5±0.5	0.0±0.0	88.6±0.4	0.0±0.0	99.0±0.0	0.0±0.0	92.9±0.5	0.0±0.0	96.2±0.4	0.0±0.0	92.8±0.3	0.0±0.0	97.5±0.6	0.0±0.0	83.5±1.0	0.0±0.0
TAAM-L (w/o LoRA)	93.6±0.3	0.0±0.0	87.9±0.5	0.0±0.0	99.1±0.1	0.0±0.0	93.0±0.6	0.0±0.0	95.9±0.7	0.0±0.0	93.4±0.2	0.0±0.0	98.0±0.4	0.0±0.0	84.9±0.3	0.0±0.0

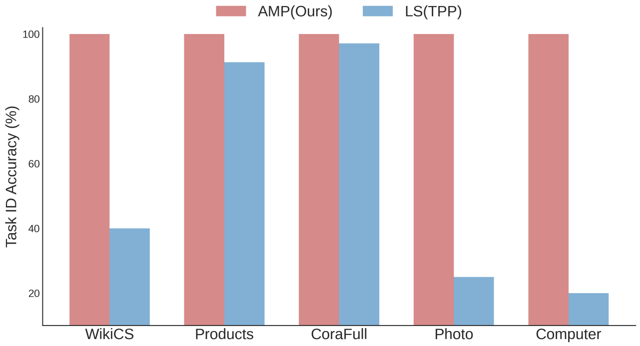
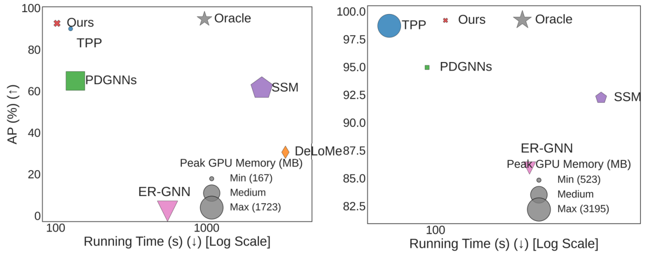


Figure 4: Task ID prediction accuracy of AMP and Laplacian smoothing (LS) of TPP



(a) CoraFull (b) Reddit

Figure 5: Comparison of running time and AA

Table 4: Parameter efficiency comparison.

Method	Total Avg AA(↑)	Additional Cost	
		Min.	Max.
Oracle	93.6	2.12 M	82.5638 M
ER-GNN	56.7	1.80 M	103.2285 M
SSM	61.9	0.30 M	51.6099 M
DeLoMe	45.2	0.18 M	34.4037 M
PDGNNs	71.9	1.06 M	100.2838 M
TPP	64.3	0.01 M	1.7949 M
TAAM-L	93.2	0.05 M	9.1812 M
TAAM	92.9	0.02 M	3.0861 M

Ablation Study (Q4). To dissect the model architecture, we conducted a comprehensive ablation study, The key findings from our ablation study are as follows:

- **w/o Task-Aware Module**: The removal of this component leads to a catastrophic collapse in performance. This confirms that the task-aware module is the cornerstone of our architecture, enabling the frozen GNN backbone to effectively adapt to diverse task distributions.
- **w/o NSMs**: Excluding the NSMs results in a severe degradation in accuracy, even though forgetting is not induced. This validates our hypothesis that fine-grained, instance-level adaptation is crucial for capturing the unique characteristics of nodes within each task.
- **w/o Unified Classifiers**: Compared with using independent classifiers for each task, adopting a unified classifier will lead to a persistent slight decline in performance, which indicates that decoupled classification heads can effectively avoid interference between tasks..
- **TAAM vs. TAAM-L (w/o LoRA)**: A core design choice in our framework is the integration of LoRA to parameterize the NSMs. This is a deliberate trade-off: its primary purpose is to ensure that the number of new parameters per task remains minimal, preventing the model size from growing uncontrollably as more tasks are learned. To validate this strategy, we compare our full TAAM model against TAAM-L, a variant employing a simpler modulator without the LoRA parameterization. The results show that complete model is slightly better than that of the variant without LoRA It has been proven that LoRA has achieved an excellent balance between ensuring parameter efficiency and maintaining model performance.

5.5 Parameter Efficiency

Finally, we analyze the parameter efficiency of TAAM. Table 4 compares the additional parameter cost required by different continual learning methods. It is evident that TAAM and its variant TAAM-L are exceptionally parameter-efficient. Our method requires only 0.02M to 3.08M additional parameters, a cost that is orders of magnitude lower than most replay-based and regularization-based methods like ER-GNN and PDGNNs, and even more efficient than the parameter-isolation baseline TPP. This demonstrates that TAAM not only achieves superior performance but does so with remarkable efficiency, making it a practical and scalable solution for real-world graph continual learning scenarios.

6 CONCLUSION

In this paper, we introduce TAAM, a new replay-free framework for Graph Continual Learning (GCL) designed to address the stability-plasticity dilemma without the memory and privacy issues associated with replay-based methods. The core of TAAM balances performance and parameter efficiency: it combines a frozen GNN backbone, which maintains prior knowledge, with lightweight, task-specific Neural Synapse Modulators (NSMs) that are trained and kept fixed for each new task. This design intentionally adds only a small number of parameters per task, preventing the model size from growing uncontrollably while still allowing enough plasticity to learn new concepts. Additionally, we propose Anchored Multi-hop Propagation (AMP), a theoretically grounded method to infer task IDs in real-world scenarios where they are unknown. Importantly, we identify flaws in existing GCL benchmarks and perform a thorough evaluation in a more rigorous and realistic inductive learning setting. Extensive experiments across eight datasets show that TAAM consistently and significantly outperforms state-of-the-art methods.

REFERENCES

[1] Marc Brockschmidt. 2020. GNN-FiLM: Graph Neural Networks with Feature-wise Linear Modulation. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event (Proceedings of Machine Learning Research, Vol. 119)*. PMLR, 1144–1152. http://proceedings.mlr.press/v119/brockschmidt20a.html

[2] Taoran Fang, Yunchao Zhang, Yang Yang, Chunping Wang, and Lei Chen. 2023. Universal Prompt Tuning for Graph Neural Networks. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (Eds.). http://papers.nips.cc/paper_files/paper/2023/hash/a4a1ee071ce0fe63b83bce507c9dc4d7-Abstract-Conference.html

[3] Xinyuan Gao, Songlin Dong, Yuhang He, Qiang Wang, and Yihong Gong. 2024. Beyond Prompt Learning: Continual Adapter for Efficient Rehearsal-Free Continual Learning. In *Computer Vision - ECCV 2024 - 18th European Conference, Milan, Italy, September 29-October 4, 2024, Proceedings, Part LXXXV (Lecture Notes in Computer Science, Vol. 15143)*, Ales Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (Eds.). Springer, 89–106. https://doi.org/10.1007/978-3-031-73013-9_6

[4] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 1024–1034. https://proceedings.neurips.cc/paper/2017/hash/5dd9db5e033da9c6fb5ba83c7a7ebea9-Abstract.html

[5] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-Efficient Transfer Learning for NLP. arXiv:1902.00751 [cs.LG] https://arxiv.org/abs/1902.00751

[6] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open Graph Benchmark: Datasets for Machine Learning on Graphs. *ArXiv* abs/2005.00687 (2020). https://api.semanticscholar.org/CorpusID:218487328

[7] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=SJU4ayYgl

[8] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. 2019. Predict then Propagate: Graph Neural Networks meet Personalized PageRank. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. https://openreview.net/forum?id=H1gL-2A9Ym

[9] Chuang Liu, Yibing Zhan, Xueqi Ma, Liang Ding, Dapeng Tao, Jia Wu, and Wenbin Hu. 2023. Gapformer: Graph Transformer with Graph Pooling for Node Classification. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*. ijcai.org, 2196–2205. https://doi.org/10.24963/IJCAI.2023/244

[10] Hao Liu, Jiarui Feng, Lecheng Kong, Ningyue Liang, Dacheng Tao, Yixin Chen, and Muhan Zhang. 2024. One For All: Towards Training One Graph Model For All Classification Tasks. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net. https://openreview.net/forum?id=4IT2pgc9v6

[11] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. arXiv:2107.13586 [cs.CL] https://arxiv.org/abs/2107.13586

[12] Yilun Liu, Ruihong Qiu, and Zi Huang. 2023. CaT: Balanced Continual Graph Learning with Graph Condensation. *Arxiv* abs/2309.09455 (2023). https://arxiv.org/abs/2309.09455

[13] Yilun Liu, Ruihong Qiu, Yanran Tang, Hongzhi Yin, and Zi Huang. 2024. PUMA: Efficient Continual Graph Learning for Node Classification with Graph Condensation. *Arxiv* abs/2312.14439 (2024). http://arxiv.org/abs/2312.14439

[14] Zemin Liu, Xingtong Yu, Yuan Fang, and Xinming Zhang. 2023. GraphPrompt: Unifying Pre-Training and Downstream Tasks for Graph Neural Networks. In *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*, Ying Ding, Jie Tang, Juan F. Sequeda, Lora Aroyo, Carlos Castillo, and Geert-Jan Houben (Eds.). ACM, 417–428. https://doi.org/10.1145/3543507.3583386

[15] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-Based Recommendations on Styles and Substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Santiago, Chile) *(SIGIR '15)*. Association for Computing Machinery, New York, NY, USA, 43–52. https://doi.org/10.1145/2766462.2767755

[16] Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. 2000. Automating the Construction of Internet Portals with Machine Learning. *Inf. Retr.* 3, 2, 127–163. https://doi.org/10.1023/A:1009953814988

[17] Chaoxi Niu, Guansong Pang, and Ling Chen. 2024. Graph Continual Learning with Debiased Lossless Memory Replay. In *ECAI 2024 - 27th European Conference on Artificial Intelligence, 19-24 October 2024, Santiago de Compostela, Spain - Including 13th Conference on Prestigious Applications of Intelligent Systems (PAIS 2024) (Frontiers in Artificial Intelligence and Applications, Vol. 392)*, Ulle Endriss, Francisco S. Melo, Kerstin Bach, Alberto José Bugarín Diz, Jose Maria Alonso-Moral, Senén Barro, and Fredrik Heintz (Eds.). IOS Press, 1808–1815. https://doi.org/10.3233/FAIA240692

[18] Chaoxi Niu, Guansong Pang, Ling Chen, and Bing Liu. 2024. Replay-and-Forget-Free Graph Class-Incremental Learning: A Task Profiling and Prompting Approach. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (Eds.). http://papers.nips.cc/paper_files/paper/2024/hash/a07e87ecfa8a651d62257571669b0150-Abstract-Conference.html

[19] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. 2018. FiLM: Visual Reasoning with a General Conditioning Layer. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, Sheila A. McIlraith and Kilian Q. Weinberger (Eds.). AAAI Press, 3942–3951. https://doi.org/10.1609/AAAI.V32I1.11671

[20] Emanuele Rossi, Fabrizio Frasca, Benjamin Paul Chamberlain, Davide Eynard, Michael M. Bronstein, and Federico Monti. 2020. SIGN: Scalable Inception Graph Neural Networks. *ArXiv* abs/2004.11198 (2020). https://api.semanticscholar.org/CorpusID:216080440

[21] Junwei Su, Difan Zou, Zijun Zhang, and Chuan Wu. 2023. Towards Robust Graph Incremental Learning on Evolving Graphs. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA (Proceedings of Machine Learning Research, Vol. 202)*, Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (Eds.). PMLR, 32728–32748. https://proceedings.mlr.press/v202/su23a.html

[22] Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang. 2024. GraphGPT: Graph Instruction Tuning for Large Language Models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024, Washington DC, USA, July 14-18, 2024*, Grace Hui Yang, Hongning Wang, Sam Han, Claudia Hauff, Guido Zuccon, and Yi Zhang (Eds.). ACM, 491–500. https://doi.org/10.1145/3626772.3657775

[23] Zonggui Tian, Du Zhang, and Hong-Ning Dai. 2024. Continual Learning on Graphs: A Survey. *CoRR* abs/2402.06330. https://doi.org/10.48550/ARXIV.2402.06330 arXiv:2402.06330

[24] Chen Wang, Yuheng Qiu, Dasong Gao, and Sebastian A. Scherer. 2022. Lifelong Graph Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*. IEEE, 13709–13718. https://doi.org/10.1109/CVPR52688.2022.01335

[25] Qi Wang, Tianfei Zhou, Ye Yuan, and Rui Mao. 2025. Prompt-Driven Continual Graph Learning. *CoRR* abs/2502.06327 (2025). https://doi.org/10.48550/ARXIV.

2502.06327 arXiv:2502.06327

[26] Felix Wu, Amauri H. Souza Jr., Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Q. Weinberger. 2019. Simplifying Graph Convolutional Networks. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 6861–6871. http://proceedings.mlr.press/v97/wu19e.html

[27] Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. 2016. Revisiting Semi-Supervised Learning with Graph Embeddings. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016 (JMLR Workshop and Conference Proceedings, Vol. 48)*, Maria-Florina Balcan and Kilian Q. Weinberger (Eds.). JMLR.org, 40–48. http://proceedings.mlr.press/v48/yanga16.html

[28] Peiyan Zhang, Yuchen Yan, Chaozhuo Li, Senzhang Wang, Xing Xie, Guojie Song, and Sunghun Kim. 2023. Continual Learning on Dynamic Graphs via Parameter Isolation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, Hsin-Hsi Chen, Wei-Jou (Edward) Duh, Hen-Hsen Huang, Makoto P. Kato, Josiane Mothe, and Barbara Poblete (Eds.). ACM, 601–611. https://doi.org/10.1145/3539618.3591652

[29] Xikun Zhang, Dongjin Song, Yixin Chen, and Dacheng Tao. 2024. Topology-aware Embedding Memory for Continual Learning on Expanding Networks. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2024, Barcelona, Spain, August 25-29, 2024*, Ricardo Baeza-Yates and Francesco Bonchi (Eds.). ACM, 4326–4337. https://doi.org/10.1145/3637528.3671732

[30] Xikun Zhang, Dongjin Song, and Dacheng Tao. 2022. CGLB: Benchmark Tasks for Continual Graph Learning. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (Eds.). http://papers.nips.cc/paper_files/paper/2022/hash/548a41b9cac6f50dccf7e63e9e1b1b9b-Abstract-Datasets_and_Benchmarks.html

[31] Xikun Zhang, Dongjin Song, and Dacheng Tao. 2022. Sparsified Subgraph Memory for Continual Graph Representation Learning. In *2022 IEEE International Conference on Data Mining (ICDM)*. 1335–1340. https://doi.org/10.1109/ICDM54844.2022.00177

[32] Xikun Zhang, Dongjin Song, and Dacheng Tao. 2023. Hierarchical Prototype Networks for Continual Graph Representation Learning. *IEEE Trans. Pattern Anal. Mach. Intell.* 45, 4 (2023), 4622–4636. https://doi.org/10.1109/TPAMI.2022.3186909

[33] Fan Zhou and Chengtai Cao. 2021. Overcoming Catastrophic Forgetting in Graph Neural Networks with Experience Replay. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. AAAI Press, 4714–4722. https://doi.org/10.1609/AAAI.V35I5.16602