On the Limitation and Experience Replay for GNNs in Continual Learning

Junwei Su
Department of Computer Science
The University of Hong Kong
Hong Kong
jwsu@cs.hku.hk

Difan Zou
Department of Computer Science
The University of Hong Kong
Hong Kong
dzou@cs.hku.hk

Chuan Wu
Department of Computer Science
The University of Hong Kong
Hong Kong
cwu@cs.hku.hk

Abstract

Continual learning seeks to empower models to progressively acquire information from a sequence of tasks. This approach is crucial for many real-world systems, which are dynamic and evolve over time. Recent research has witnessed a surge in the exploration of Graph Neural Networks (GNN) in Node-wise Graph Continual Learning (NGCL), a practical yet challenging paradigm involving the continual training of a GNN on node-related tasks. Despite recent advancements in continual learning strategies for GNNs in NGCL, a thorough theoretical understanding, especially regarding its learnability, is lacking. Learnability concerns the existence of a learning algorithm that can produce a good candidate model from the hypothesis/weight space, which is crucial for model selection in NGCL development. This paper introduces the first theoretical exploration of the learnability of GNN in NGCL, revealing that learnability is heavily influenced by structural shifts due to the interconnected nature of graph data. Specifically, GNNs may not be viable for NGCL under significant structural changes, emphasizing the need to manage structural shifts. To mitigate the impact of structural shifts, we propose a novel experience replay method termed Structure-Evolution-Aware Experience Replay (SEA-ER). SEA-ER features an innovative experience selection strategy that capitalizes on the topological awareness of GNNs, alongside a unique replay strategy that employs structural alignment to effectively counter catastrophic forgetting and diminish the impact of structural shifts on GNNs in NGCL. Our extensive experiments validate our theoretical insights and the effectiveness of SEA-ER.

1 Introduction

Continual learning, also known as incremental learning or life-long learning, investigates machine learning approaches that enable a model to continuously acquire new knowledge while retaining previously obtained knowledge. This capability is essential as it allows the model to adapt to new information without forgetting past information. In the general formulation of continual learning, a stream of tasks arrives sequentially, and the model undergoes rounds of training sessions to accumulate knowledge for specific objectives, such as classification. The primary goal is to find a learning algorithm that can continually update the model's parameters based on the new task without suffering from *catastrophic forgetting* (Delange et al., 2021), which refers to the inability to retain previously learned information when learning new tasks. Continual learning is crucial for the practicality of machine learning systems as it enables the model to adapt to new information without the need for frequent retraining, which can be costly in the deep learning regime today.

On the other hand, graph neural networks (GNNs) have gained widespread popularity as effective tools for modeling graph and relational data structures (Wu et al., 2020; Su et al., 2024a). However, most studies on GNNs have predominantly focused on static settings, assuming that the underlying graph and learning task remain unchanged. This approach, while valuable for certain applications, does not reflect the dynamic and evolving nature of real-life networks. For example, networks such as citation networks (Zliobaite, 2010) and financial networks (Gama et al., 2014) naturally evolve over time, accompanied by the emergence of new tasks. Therefore, to further advance the practicality and effectiveness of GNNs, it is essential to study their effectiveness and applicability in graph continual learning (GCL).

Due to its practical significance, recent research has seen an upsurge in the investigation of the application of GNNs in GCL (Febrinanto et al., 2023; Yuan et al., 2023). Nonetheless, there is a noticeable gap in the theoretical foundations, particularly regarding the node-level prediction task, commonly referred to as *Node-wise Graph Continual Learning* (NGCL) (Su et al., 2023; Zhang et al., 2022). In NGCL, the GNN model undergoes multiple training rounds (tasks) to accumulate knowledge for specific objectives, such as node classification. In this scenario, vertices across different tasks can be interconnected, and the introduction of new tasks may alter the structure of the existing graph, as illustrated in Figure 1. Given that GNNs rely on graph structures as input, these structural changes can lead to shifts in the data distributions of previous tasks, a phenomenon known as *structural shift*. Su et al. (2023) has shown that the risk (upper bound) of the performance of GNNs in NGCL is closely related to structural shift. Nevertheless, it is not yet clear whether and how structural shifts can affect the learnability of GNNs in NGCL. Learnability concerns whether there exists a learning algorithm that can produce a good candidate model from the hypothesis/weight space, and the learnability of GNN in NGCL is crucial to NGCL development. This has motivated the central question of this study:

Is GNN always learnable (a suitable model choice) for NGCL under structural shift?

In this work, we address the aforementioned question by formally formulating and analyzing the learnability of GNNs in NGCL. To the best of our knowledge, this work presents the first theoretical studies on the learnability of GNNs in NGCL, and our first contribution is highlighted as follows:

- We present a mathematical formulation for studying the learnability of GNNs in NGCL. This formulation is important for a rigorous exploration of the learnability problem. In particular, we prove that a large structural shift can render GNNs unlearnable in NGCL, as articulated in Theorem 3.1. This implies that GNNs may not be a suitable choice for NGCL in volatile network systems (experiencing rapid changes in graph structure), answering the central question of this study. Moreover, this insight is particularly valuable for NGCL development in practice, as it can guide the model selection process by providing insights on when to select or rule out GNN-like model architectures for NGCL.

Given the pivotal role of structural shifts in learnability, we conclude the importance of controlling structural shifts even in scenarios where GNNs are learnable in NGCL. To address this challenge, we focus on experience replay, a principal method for continual learning, which has been empirically proven to be effective in addressing catastrophic forgetting in NGCL (Ahrabian et al., 2021; Zhou & Cao, 2021a). We introduce a novel experience replay method designed to mitigate the impact of structural shifts for GNNs in NGCL. Our contribution in this regard is as follows:

- We introduce Structure-Evolution-Aware Experience Replay (SEA-ER), a novel experience replay method that targets both catastrophic forgetting and the unique challenge of structural shifts in NGCL. SEA-ER leverages two key principles - structural alignment and topological awareness of GNNs - to select experience samples and assign different replaying weights based on their structural similarity to the rest of the graphs. This approach effectively mitigates structural shifts and catastrophic forgetting for GNNs in NGCL, offering theoretical assurances on long-term performance (Proposition 4.1).

To further validate the above theoretical results and the effectiveness of our proposed experience replay method, we conduct an empirical evaluation consisting of real-world and synthetic datasets. The results align with our theoretical findings and demonstrate the effectiveness of our proposed approach. Our novel experience replay method adeptly addresses both the structural shift and catastrophic forgetting, outperforming contemporary experience replay methods in NGCL (Zhou & Cao, 2021a; Ahrabian et al., 2021; Kim et al., 2022).

2 Related Work

2.1 Continual Learning

Continual learning, also known as incremental or lifelong learning, has gained increasing attention in recent years and has been extensively explored on Euclidean data. We refer readers to surveys (Delange et al., 2021; Parisi et al., 2019; Biesialska et al., 2020) for a more comprehensive review of these works. The primary challenge of continual learning is to address the catastrophic forgetting problem, which refers to the degradation in the performance of a model on previous tasks after being trained on new tasks.

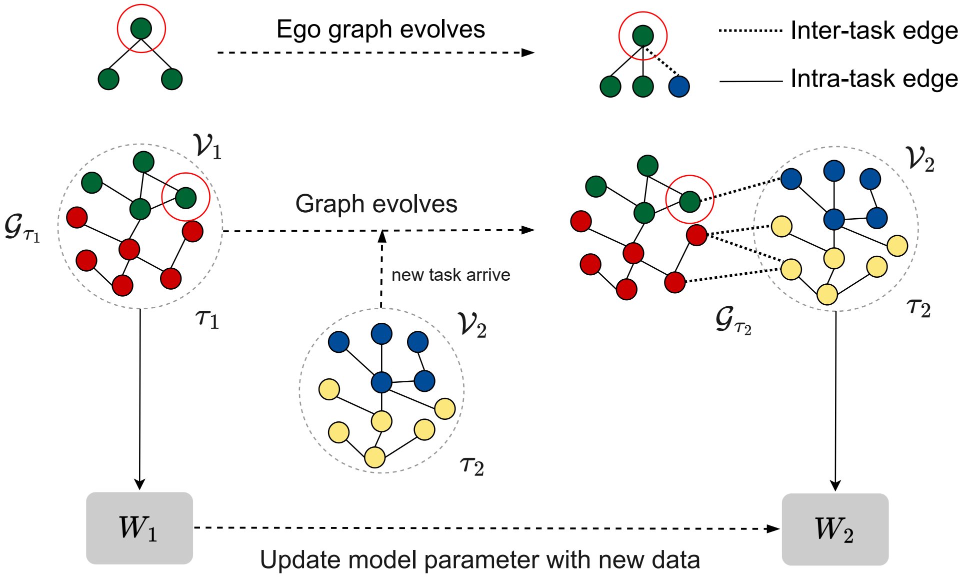


Figure 1: Illustration of the progression of NGCL. Task 2 introduces a new batch of vertices and results in an update to the parameters of the model from W_2 to W_1 using data from the new task. As new vertex batches associated with the new task τ_2 are introduced, the graph structure changes, potentially altering the graph structure (inputs to GNNs) of the existing vertices, as captured by the changes in their ego graphs. This is referred to as the structural shift.

Existing approaches for addressing this problem can be broadly categorized into three types: regularization-based methods, experience-replay-based methods, and parameter-isolation-based methods. Regularization-based methods aim to maintain the model's performance on previous tasks by penalizing large changes in the model parameters (Jung et al., 2016; Li & Hoiem, 2017; Kirkpatrick et al., 2017; Farajtabar et al., 2020; Saha et al., 2021). Parameter-isolation-based methods prevent drastic changes to the parameters that are important for previous tasks by continually introducing new parameters for new tasks (Rusu et al., 2016; Yoon et al., 2017; 2019; Wortsman et al., 2020; Wu et al., 2019). Experience-replay-based methods, inspired by the spaced repetition in human learning theory, can be viewed as a form of memory consolidation for neural networks (Zhou & Cao, 2021b). Experience replay methods consist of two key components: 1) experience selection and 2) replay strategy. The goal of experience selection is to select a small set of representative/strong samples from previous data and store them in an experience buffer. Then, the replay strategy is to strategically retrain the model on the data from the experience buffer to consolidate past knowledge and counteract catastrophic forgetting.

In this study, we delve into both the theoretical foundations and practical implementations of GNN in NGCL. On the theoretical front, we leverage established statistical learning frameworks previously utilized in domains such as domain adaptation (David et al., 2010), multi-task learning (Ben-David & Borbely, 2008), and continual learning (Benavides-Prado & Riddle, 2022) at large. Specifically, we employ these frameworks to delve into the nature of structural shifts and their implications for the learnability of GNN within NGCL contexts. Methodologically, we focus on the experience replay method for its simplicity and effectiveness (Lopez-Paz & Ranzato, 2017; Shin et al., 2017; Aljundi et al., 2019; Caccia et al., 2020; Chrysakis & Moens, 2020; Knoblauch et al., 2020; Su et al., 2024b). Because of the non-I.I.D. nature of graph-structured data, experience replay for NGCL requires different treatment (Zhou & Cao, 2021b). In this work, we propose a novel experience replay method tailored for counteracting the catastrophic forgetting and structural shift for GNNs in NGCL.

2.2 Graph Continual Learning (GCL)

Recently, there has been a surge of interest in GCL due to its practical significance in various applications (Wang et al., 2022; Xu et al., 2020; Daruna et al., 2021; Kou et al., 2020; Ahrabian et al., 2021; Cai et al., 2022; Wang et al., 2020a; Liu et al., 2021; Zhang et al., 2021; Zhou & Cao, 2021c; Carta et al., 2021; Zhang et al., 2022; Kim et al., 2022; Tan et al., 2022). For a detailed review of GCL methodologies and their efficacy, we direct readers to recent reviews and benchmarks (Zhang et al., 2022; Yuan et al., 2023; Febrinanto et al., 2023). However, most existing NGCL studies primarily focus on scenarios where the entire graph structure is predefined or where interconnections between tasks are disregarded. The NGCL setting, where the graph evolves with the introduction of new tasks, remains relatively unexplored and warrants a deeper theoretical investigation. To our knowledge, (Su et al., 2023) is the only existing theoretical study on NGCL with an evolving graph. Su et al. (2023) prove that the catastrophic forgetting risk (upper bound) of GNNs is determined by the structural shift of the underlying graph structure. In contrast, our work addresses the learnability of GNNs in NGCL under structural shift, which can be interpreted as a sort of "lower bound". We present the first impossibility result regarding the learnability of GNNs in NGCL, underscoring the critical role of structural shifts. Additionally, we introduce a novel experience replay method that effectively tackles both catastrophic forgetting and structural shifts in NGCL.

2.3 Other Related Works

While our primary focus is on GCL, it is worth noting there are parallel lines of inquiry within dynamic graph learning and graph unlearning that intersect with our work in terms of context and terminology. Dynamic graph learning focuses on enabling GNNs to capture the changing graph structures (Galke et al., 2021; Wang et al., 2020b; Han et al., 2020; Yu et al., 2018; Nguyen et al., 2018; Ma et al., 2020; Feng et al., 2020; Bielak et al., 2022; Su et al., 2024b). The goal of dynamic graph learning is to capture the temporal dynamics of the graph into the representation vectors while having access to all previous information. In contrast, GCL addresses the problem of catastrophic forgetting, in which the model's performance on previous tasks degrades after learning new tasks. For evaluation, a dynamic graph learning algorithm is only tested on the latest data, while GCL models are also evaluated on past data. On the other hand, the goal of graph unlearning is to "unlearn" or "forget" specific patterns, associations, or information that the GNN has acquired during training on a given dataset or set of tasks. Therefore, dynamic graph learning, graph unlearning, and GCL are independent and distinct research directions with different focuses and should be considered separately.

3 Learnability of GNN in NGCL

3.1 Preliminary of NGCL

We use lowercase letters to denote scalars and graph-related objectives. Furthermore, we use lower and uppercase boldface letters to denote vectors and matrices, respectively. NGCL assumes the existence of a stream of training tasks $\tau_1, \tau_2, ..., \tau_m$, characterized by observed vertex batches $\mathcal{V}_1, \mathcal{V}_2, ..., \mathcal{V}_m$. Each vertex v is associated with a node feature vector $\mathbf{x}_v \in \mathcal{X}$ and a target label $y_v \in \mathcal{Y}$. The observed graph structure at training task τ_i is induced by the accumulative vertices and given by $\mathcal{G}_{\tau_i} = \mathcal{G}[\bigcup_{j=1}^{i} \mathcal{V}_j]$. In this setting, the graph structure is evolving as the learning progresses through different training tasks, as illustrated in Figure. 1.

Node-level learning tasks with GNNs rely on information aggregation within the k-hop neighbourhood of a node as input. To accommodate this nature, we adopt a local view in the learning problem formulation. We denote $\mathcal{N}_k(v)$ as the k-hop neighborhood of vertex v, and the nodes in $\mathcal{N}_k(v)$ form an ego-graph $\mathbf{G}_v$, which consists of a (local) node feature matrix $\mathbf{X}_v = \{x_u | u \in \mathcal{N}_k(v)\}$ and a (local) adjacency matrix $\mathbf{A}_v = \{a_{uw} | u, w \in \mathcal{N}_k(v)\}$. We denote $\mathbb{G}$ as the space of possible ego-graph and $\mathbf{G}_v = (\mathbf{A}_v, \mathbf{X}_v)$ as the ego-graph for the target vertex v. Let $\mathbf{G}_\mathcal{V} = \{\mathbf{G}_v | v \in \mathcal{V}\}$ denote the set of ego graphs associated with vertex set $\mathcal{V}$. From the perspective of GNNs, the ego-graph $\mathbf{G}_v$ is the Markov blanket containing all necessary information for the prediction problem for the root vertex v. Therefore, we can see the prediction problem associated with data $\{(\mathbf{G}_v, y_v)\}_{v \in \mathcal{V}_i}$ from training session τ_i as drawn from a joint distribution $\mathbb{P}(\mathbf{y}_\mathbf{V}, \mathbf{G}_v | \mathcal{V}_i)$. The dependency of the vertex is captured and manifested through the underlying graph structure in the above formulation.

Let $\mathcal{F}$ denote the hypothesis space and $F \in \mathcal{F}$ be a classifier with $\widehat{y}_v = F(\mathbf{G}_v)$ and $L(.,.) \mapsto \mathbb{R}$ be a given loss function. We use $R^L_{\mathbb{P}(\mathbf{y}_v, \mathbf{G}_v | \mathcal{V})}(F)$ to denote the generalization performance of the classifier f with respect to L and $\mathbb{P}(\mathbf{y}_v, \mathbf{G}_v | \mathcal{V})$, and it is defined as follows:

$$R^L_{\mathbb{P}(\mathbf{y}_v, \mathbf{G}_v | \mathcal{V})}(F) = \mathbb{E}_{\mathbb{P}(\mathbf{y}_v, \mathbf{G}_v | \mathcal{V})}[L(F(\mathbf{G}_v), y_v)]. \tag{3.1}$$

Catastrophic Forgetting. With the formulation above, the catastrophic forgetting (**CF**) of a classifier F after being trained on τ_m can be characterized by the retention of performance on previous vertices/tasks, given by:

$$\mathbf{CF}(F) := R^L_{\mathbb{P}(\mathbf{y}_v, \mathbf{G}_v | \mathcal{V}_1, ..., \mathcal{V}_{m-1})}(F), \tag{3.2}$$

which translates to the retention of performance of the classifier F from τ_m on the previous tasks $\tau_1, ..., \tau_{m-1}$.

3.2 Structural Shift and Learnability

This paper delves into the relationship between structural shifts and the learnability of GNNs in NGCL. We begin with defining the expressiveness of the hypothesis space and characterizing the structural shift.

Definition 1 (Expressiveness of hypothesis space)**.** *For a set of distributions $\{\mathbb{P}_1, \mathbb{P}_2, ..., \mathbb{P}_k\}$ and a hypothesis space $\mathcal{F}$, we define the joint prediction error of the hypothesis space on the given distributions as,*

$$\lambda_{\mathcal{F}}(\{\mathbb{P}_1, \mathbb{P}_2, ..., \mathbb{P}_k\}) = \inf_{F \in \mathcal{F}} \left[\sum_{\mathbb{P} \in \{\mathbb{P}_1, \mathbb{P}_2, ..., \mathbb{P}_k\}} R^{\mathcal{L}}_{\mathbb{P}}(F) \right].$$

The joint prediction error characterizes the capability (expressiveness) of the hypothesis space to simultaneously fit a given set of distributions. Clearly, the expressiveness of the hypothesis space is a trivial necessary condition for the learnability of a given setting. If the expressiveness of the hypothesis space is insufficient, even the best classifier within the hypothesis space would yield poor performance. Throughout the paper, we focus our analysis on structural shifts, assuming that the training samples for each task and the expressive power of the hypothesis space are sufficient. This means that when all the tasks are presented together, there exists a learning algorithm that can produce a classifier within the hypothesis space that incurs minimal error. Further discussion on expressiveness and sample complexity is provided in Appendix A.

Next, we formalize the notion of structural shift among different vertex batches (tasks).

Definition 2. *Let $\mathbb{P}_1$ and $\mathbb{P}_2$ be two distributions over some domain $\mathcal{X}$, $\mathcal{F}\Delta\mathcal{F}$-distance between $\mathbb{P}_1$ and $\mathbb{P}_2$ is defined as,*

$$d_{\mathcal{F}\Delta\mathcal{F}}(\mathbb{P}_1, \mathbb{P}_2) = 2 \sup_{A \in \mathcal{F}\Delta\mathcal{F}} \|\mathbb{P}_1(A) - \mathbb{P}_2(A)\|,$$

where $\mathcal{F}\Delta\mathcal{F} = \{\{\mathbf{x} \in \mathcal{X} : F(\mathbf{x}) \neq F'(\mathbf{x})\} : F, F' \in \mathcal{F}\}$.

$\mathcal{F}\Delta\mathcal{F}$-distance is commonly used to capture the relatedness (distance) of distributions with respect to the hypothesis space $\mathcal{F}$ (Redko et al., 2020). Here, we use it to characterize the effect of the emerging task on the distribution of the previous vertex tasks. If the structural shift among different tasks is large, then the appearance of the new vertex batch would shift the distribution of the previous vertex batch to a larger extent. Consequently, the $\mathcal{F}\Delta\mathcal{F}$-distance between the updated distribution and the previous distribution would be large

Without loss of generality, we may gauge our learnability analysis toward the case of two training sessions, referred to as the NGCL-2 problem. If the NGCL-2 problem is not learnable, then trivially, the general NGCL-k is not learnable. Furthermore, the corresponding insight can also be easily extended to multiple training tasks by recursively applying the analysis and treating consecutive tasks as a joint task. The NGCL-2 problem is characterized by three distributions: $\mathbb{P}(y, \mathbf{G}_v|\mathcal{V}_1, \mathcal{G}_{\tau_1})$, $\mathbb{P}(y, \mathbf{G}_v|\mathcal{V}_1, \mathcal{G}_{\tau_2})$, and $\mathbb{P}(y, \mathbf{G}_v|\mathcal{V}_2, \mathcal{G}_{\tau_2})$, which are the distributions of $\mathcal{V}_1$ in graphs $\mathcal{G}_{\tau_1}$ and $\mathcal{G}_{\tau_2}$, and the distribution of $\mathcal{V}_2$ in graph $\mathcal{G}_{\tau_2}$. Then, the following theorem captures the learnability of GNN in NGCL-2.

Theorem 3.1 (necessity of controllable structural shift, informal)**.** *Let $\mathcal{F}$ be a hypothesis space over $\mathbb{G} \times \mathcal{Y}$ captured by GNNs. If there is no control on*

$$d_{\mathcal{F}\Delta\mathcal{F}}\big(\mathbb{P}(y, \mathbf{G}_v|\mathcal{V}_1, \mathcal{G}_{\tau_1}), \mathbb{P}(y, \mathbf{G}_v|\mathcal{V}_2, \mathcal{G}_{\tau_2})\big) \quad \text{and} \quad d_{\mathcal{F}\Delta\mathcal{F}}\big(\mathbb{P}(y, \mathbf{G}_v|\mathcal{V}_1, \mathcal{G}_{\tau_1}), \mathbb{P}(y, \mathbf{G}_v|\mathcal{V}_1, \mathcal{G}_{\tau_2})\big)$$

then, for every $c > 0$, there exists an instance of NGCL-2 problem satisfying the following:

1. *$\lambda_{\mathcal{F}}(\{\mathbb{P}(y, \mathbf{G}_v|\mathcal{V}_1, \mathcal{G}_{\tau_1}), \mathbb{P}(y, \mathbf{G}_v|\mathcal{V}_1, \mathcal{G}_{\tau_2})\}) \leq c$,*
2. *for any given continual learning algorithms, there is at least $1/2$ probability that it would produce a classifier F from $\mathcal{F}$ with **CF**(F) at least $1/2$.*

The formal version and proof of Theorem 3.1 can be found in Appendix A. Theorem 3.1 indicates that even if the hypothesis space is expressive enough (i.e., each individual task and their combination are learnable by the GNNs), the NGCL problem might still not be learnable without control on the structural shift among different vertex batches. We make the following remarks.

Remark 3.2. *Theorem 3.1 suggests that when considering continual learning for a volatile network (a large structural shift), a GNN-like model is not a suitable candidate, as its learnability is heavily damaged by the structural shift. Different treatment is required. This renders an interesting future direction but beyond the scope of this paper.*

Remark 3.3. *The structural shift can harm the learnability of GNNs in NGCL. Recognizing this pivotal role of structural shift, we conclude that it is important to control structural shifts even in scenarios where GNNs are learnable in NGCL. This motivates our design of the experience replay method described in the next section and can be extended to modify other continual learning methods for GNNs in NGCL.*

4 Structure-Evolution-Aware Experience Replay (SEA-ER)

In the preceding sections, we explored the learnability of GNN within the NGCL context, highlighting the significance of structural shifts in this learnability. Shifting focus from theoretical aspects to practical implementations in this section, we examine the application of the experience replay method in NGCL. We introduce SEA-ER, which integrates the topology awareness of GNNs with structural alignment to mitigate catastrophic forgetting and address structural shifts in the NGCL problem. Experience replay involves two key components: 1) experience buffer selection, which entails choosing representative data samples from the past, and 2) replay strategy, determining how the experience buffer is utilized in training with new tasks. The pseudo-code for our methods can be found in Appendix E.

4.1 Topology-Aware Experience Buffer Selection

We denote the experience buffer for task τ_i as $\mathcal{P}_i = \{P_1, P_2, ..., P_{i-1}\}$, where P_j represents samples selected from task τ_j. GNNs have demonstrated superior generalization performance on vertices with closely related structural characteristics (Ma et al., 2021). Accordingly, we propose an experience buffer selection strategy that chooses samples with the closest structural similarity to the remaining vertices. This strategy is formulated as the optimization problem:

$$\begin{aligned} &\operatorname*{argmin}_{P_j \subset \mathcal{V}_j} \max_{u \in \mathcal{V}_j \setminus P_j} \min_{v \in P_j} d_{\text{spd}}(u, v), \\ &s.t. \quad |P_j| = b. \end{aligned} \tag{4.1}$$

Here, $d_{\text{spd}}(.)$ represents the shortest path distance in the graph, and b is the number of samples (budget) to select from each task. The optimization above returns a set P_j with close structural similarity to the rest of the vertices within the same task. For our experiments, we adopt the classic farthest-first traversal algorithm to obtain a solution.

4.2 Replay Strategy with Structural Alignment

The standard replay strategy incorporates the experience buffer into the training set of the newest task, treating the samples in the buffer as normal training data. However, structural shifts induced by evolving graph structures from $\mathcal{G}_{\tau_{i-1}}$ to $\mathcal{G}_{\tau_i}$ can alter the distribution of vertices from the previous task, potentially leading to suboptimal learning. To mitigate the impact of structural shift, we propose a replay strategy with structural alignment. This strategy assigns higher weights to replay samples that are more similar to the distribution before the structural shift, ensuring that the model prioritizes relevant information. The learning objective with structural alignment is given by:

$$L = \frac{1}{|\mathcal{V}_i^{\text{train}}|} \sum_{v \in \mathcal{V}_i^{\text{train}}} L(v) + \sum_{P_j \in \mathcal{P}_i} \frac{1}{|P_j|} \sum_{v \in P_j} \beta_v L(v), \tag{4.2}$$

where β_v (highlighted in blue) is the main difference from the standard replay strategy and represents the weight for vertex v in the experience buffer. To determine β (collection of β_v), we use kernel mean matching (KMM) Gretton et al. (2009) with respect to the embeddings of vertices in the experience buffer. This involves solving the following convex quadratic problem:

$$\begin{aligned} &\min_{\beta} \Big\| \sum_{P_j \in \mathcal{P}_i} \sum_{v \in P_j} \beta_v \phi(h_v) - \phi(h'_v) \Big\|^2, \\ &s.t. \quad B_l \leq \beta_v < B_u, \end{aligned} \tag{4.3}$$

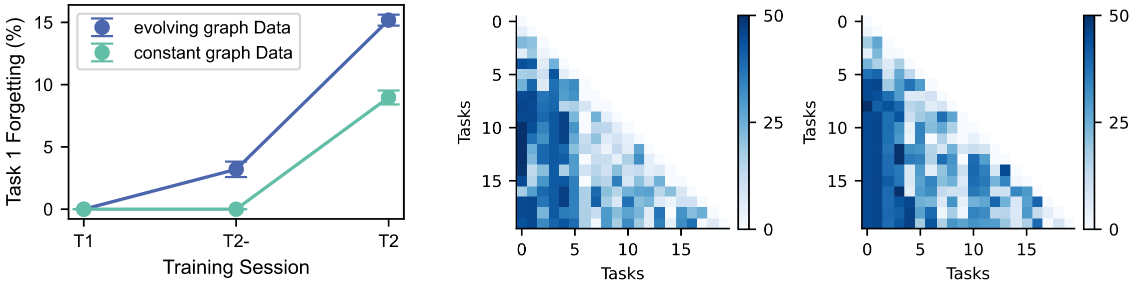
where h_v is the node embedding of v (output of the GNN) with respect to $\mathcal{G}_{\tau_i}$ and h'_v is the node embedding of v with respect to $\mathcal{G}_{\tau_{i-1}}$. Eq. 4.3 is equivalent to match the mean elements in some kernel space $\mathcal{K}(.,.)$ and $\phi(.)$ is the feature map to the reproducing kernel Hilbert space induced by the kernel $\mathcal{K}(.,.)$. In our experiment, we use a mixture of Gaussian kernel $\mathcal{K}(x, y) = \sum_{\alpha_i} e^{-\alpha_i ||x-y||_2}$, $\alpha_i = 1, 0.1, 0.01$. The lower bound B_l and upper bound B_u constraints are to ensure reasonable weight for most of the instances.

4.3 Theoretical Analysis of SEA-ER

In this section, we provide a theoretical foundation for our proposed method by introducing the concept of distortion rate, which helps characterize the topological awareness of GNNs.

Definition 3 (distortion rate)**.** *Given two metric spaces $(\mathcal{Q}, d)$ and $(\mathcal{Q}', d')$ and a mapping $\gamma : \mathcal{Q} \mapsto \mathcal{Q}'$, γ is said to have a distortion $\alpha \geq 1$, if there exists a constant $r > 0$ such that $\forall u, v \in \mathcal{E}$,*

$$rd(u, v) \leq d'(\gamma(u), \gamma(v)) \leq \alpha r d(u, v).$$



(a) Evolution of Task 1 Performance

(b) Arxiv, Constant

(c) Arxiv, Evolving

Figure 2: Forgetting Dynamics of Bare Model on Arxiv under Settings with Constant and Evolving Graphs. (a) captures the change of catastrophic forgetting of task 1 when transitioning into task 2 under the settings with constant and evolving graphs. (b) and (c) are the complete catastrophic forgetting matrix (x, y-axis are the i, j in $r_{i,j} - r_{i,i}$ correspondingly) of Bare model in the inductive and transductive settings. The colour density indicates the amount of forgetting(denser color means larger forgetting).

The distortion rate quantifies the ratio of distances in the embedding space to those in the original graph space. A low distortion rate, close to 1, signifies that the embedding effectively preserves the structural information from the original graph. This measure allows us to assess the topological awareness of GNNs and forms the basis for our theoretical analysis of the experience selection strategy in relation to the underlying graph topology.

Proposition 4.1. *Given a collection task $\tau_1, ..., \tau_m$ and their associated vertex set $\mathcal{V}_1, ..., \mathcal{V}_m$. Let $\mathcal{P}_m = \{P_1, ..., P_{m-1}\}$ be the experience buffer where each $P_i \in \mathcal{P}_m$ has the same cardinality and cover all classes in τ_i. Let F be the learned GNN model with distortion rate α. Assuming the loss function L is bounded, continuous and smooth, we have that,*

$$R^L_{\mathbb{P}(\mathbf{y}_v, \mathbf{G}_v | \mathcal{V}_i \backslash P_i)}(F) \leq R^L_{\mathbb{P}(\mathbf{y}_v, \mathbf{G}_v | P_i)}(F) + \mathcal{O}(\alpha D_s(\mathcal{V}_i \backslash P_i, P_i)),$$

where

$$D_s(\mathcal{V}_i \backslash P_i, P_i) = \max_{u \in \mathcal{V}_i \backslash P_i} \min_{v \in P_i} d_{\text{spd}}(v, u).$$

The proof for Propostion 4.1 can be found in Appendix C. This proposition demonstrates that catastrophic forgetting in the model, as measured by its ability to retain performance on past data, is bounded by the training loss on the experience buffer and is further influenced by a combination of factors involving the distortion rate of the model and the structural distance within the experience buffer. This underscores the critical role of graph structure when selecting a suitable experience buffer, as the structural distance substantially impacts the performance bound. Consequently, Proposition 4.1 validates the optimization objective, Eq. 4.1, for selecting an experience buffer and provides a performance guarantee for our proposed experience selection strategy.

5 Experiments

In this section, we present an empirical validation of our theoretical results and evaluation of our proposed experience replay method for NGCL. In this empirical study, we aim to answer the following questions: (1) Does structural shift indeed have an impact on the learnability and catastrophic forgetting of GNNs in NGCL? (2) Does the distortion rate of GNNs, as discussed in Proposition 4.1, hold true in practice? (3) How does our proposed experience replay method compare to existing experience replay methods for GNNs (Zhou & Cao, 2021a; Ahrabian et al., 2021; Kim et al., 2022) in NGCL? Due to space limitations, we only summarize the key details of the experiments in the main paper and defer a more comprehensive description of the datasets, experiment set-up and additional results in Appendix D.

Datasets and General Set-up. In general, we follow closely the experiment settings in (Su et al., 2023; Zhang et al., 2022). We conduct experiments on both real-world and synthetic datasets. Real-world datasets include OGB-Arxiv (Hu et al., 2020), Reddit (Hamilton et al., 2017), and CoraFull (Bojchevski & Günnemann, 2017). To generate the synthetic dataset, we use the popular contextual stochastic block model (cSBM) (Deshpande et al., 2018) (further details of the cSBM can be found in Appendix D.4). The experimental set-up follows the widely adopted task-continual-learning (task-CL)(Su et al., 2023; Zhang et al., 2022), where a k-class classification task is extracted from

Table 1: Performance comparison of SEA-ER to existing experience replay NGCL frameworks (↑ higher means better). Results are averaged among five trials. **Bold letter** indicates that the entry admits the best performance for that given dataset. 5% of the training set size is used for the size of the experience replay buffer for all the experience replay methods.

Dataset	CoraFull		OGB-Arxiv		Reddit	
Performance Metric	FAP (%) ↑	FAF (%) ↑	FAP (%) ↑	FAF (%) ↑	FAP(%) ↑	FAF (%) ↑
Bare model	61.41 ± 1.2	-30.36 ± 1.3	59.24 ± 3.6	-33.72 ± 3.3	68.60 ± 1.8	-23.87 ± 1.7
Joint Training	92.34 ± 0.6	N.A.	94.36 ± 0.4	N.A.	95.27 ± 1.2	N.A.
ER:IID-Random	77.24 ± 5.8	-16.73 ± 5.6	79.72 ± 4.6	-14.89 ± 4.5	80.08 ± 3.3	-10.89 ± 2.5
ER-deg	86.24 ± 0.7	-8.73 ± 0.7	85.28 ± 0.8	-10.89 ± 1.9	85.08 ± 0.9	-8.89 ± 0.9
ER-infl	87.99 ± 0.6	-8.31 ± 0.5	86.42 ± 1.2	-7.65 ± 1.3	86.98 ± 1.4	-6.82 ± 1.6
ER-rep	88.95 ± 0.9	-7.55 ± 0.8	88.12 ± 0.9	-8.23 ± 0.9	86.02 ± 1.2	-7.21 ± 1.3
SEA-ER w.o. sa (ours)	89.13 ± 0.8	-7.21 ±0.9	89.19 ± 0.7	-7.13 ± 0.5	88.48 ± 1.7	-6.10 ± 1.6
ER:IID-Random w. sa	80.24 ± 6.9	-14.73 ± 4.8	81.72 ± 5.2	-13.23 ± 5.2	83.43 ± 2.8	-10.32 ± 3.8
ER-deg w. sa	87.32 ± 0.5	-7.89 ± 0.5	86.88 ± 0.6	-9.32 ± 2.0	87.11 ± 1.0	-6.89 ± 1.0
ER-infl w. sa	88.88 ± 0.5	-8.01 ± 0.5	89.42 ± 1.3	-5.65 ± 1.7	90.64 ± 1.6	-3.63 ± 1.3
ER-rep w. sa	89.06 ± 0.7	-7.00 ± 0.9	88.92 ± 0.8	-8.00 ± 1.1	88.99 ± 1.4	-4.21 ± 1.1
SEA-ER (ours)	**91.67** ± 1.3	**-4.01** ± 1.4	**92.88** ± 1.1	**-3.08** ± 0.7	**92.89** ± 1.5	**-2.72** ± 1.4

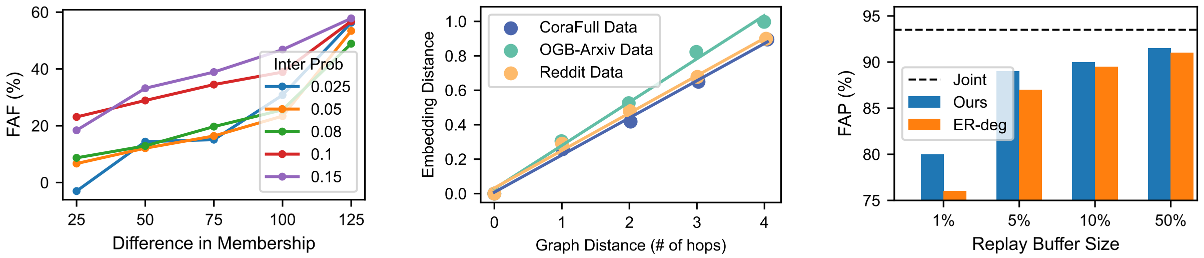
Datasets	OGB-Arxiv	Reddit	CoraFull
# vertices	169,343	227,853	19,793
# edges	1,166,243	114,615,892	130,622
# class	40	40	70
# tasks	20	20	35
# vertices / # task	8,467	11,393	660
# edges / # task	58,312	5,730,794	4,354

Table 2: Continual learning settings for each dataset.

the dataset for each training session. For example, in the OGB-Arxiv dataset, which has 40 classes, we divide them into 20 tasks: Task 1 is a 2-class classification task between classes 0 and 1, task 2 is between classes 2 and 3, and so on. In each task, the system only has access to the graph induced by the vertices at the current and earlier learning stages, following the formulation in Sec.3.1. A brief description of the datasets and how they are divided into different node classification tasks is given in Table 2. We adopt the implementation from the recent NGCL benchmark (Zhang et al., 2022) for creating the task-CL setting and closely following their set-up (such as the train/valid/test set split).

Evaluation Metrics. Let $r_{i,j}$ denote the performance (accuracy) on task j after the model has been trained over a sequence of tasks from 1 to i. Then, the forgetting of task j after being trained over a sequence of tasks from 1 to i is measured by $r_{i,j} - r_{j,j}$. We use the final average performance (FAP) $:= \sum_j^m r_{m,j}/m$ and the final average forgetting (FAF) $:= (\sum_j^m r_{m,j} - r_{j,j})/m$ to measure the overall effectiveness of an NGCL framework. These metrics measure the performance of the model after being presented with all the tasks. The order of the task permutation should not affect the performance of the experience reply with the two chosen metrics (further result and discussion on this regard can be found in Appendix D

Baselines. We compare the performance of SEA-ER with the following experience replay baselines. All the baselines are defaulted for the standard replay strategy, i.e. Eq. 4.2 without β. **ER-IID-Random** updates each task with an experience buffer with vertices randomly selected from each task. **ER-IID-Random** serves as a baseline for showing the effectiveness of different experience buffer selection strategies. We adopt state-of-the-art experience replay frameworks for GNNs for comparison. **ER-rep** (Zhou & Cao, 2021a) selects an experience buffer based on the representations of the vertex. **ER-deg** (Ahrabian et al., 2021) selects an experience buffer selected based on the degree of vertices. **ER-infl** (Kim et al., 2022) selects an experience buffer selected based on the influence score of the representations of the vertex. We adopt the implementation of the two baselines (Ahrabian et al., 2021; Zhou & Cao, 2021a) from the benchmark paper (Zhang et al., 2022) and implement a version of (Kazemi et al., 2020) based on our understanding, as the code is not open-sourced. **SEA-ER w.o. sa** is our proposed experience replay selection strategy with the standard replay strategy, and **SEA-ER** is our proposed experience replay selection strategy with learning objective Eq. 4.2. The suffix **w. sa** indicate the usage of Eq. 4.2. In addition to the above frameworks, we also include



(a) FAF of cSBM

(b) Distortion

(c) Performance vs. buffer size

Figure 3: Fig. 3(a) is the experiments with different configurations of cSBM models. Fig. 3(b) is the distortion between graph distance and embedding distance. Fig. 3(c) is the experiment on the effect of replay buffer size (without the structural alignment).

two natural baselines for NGCL: the **Bare model** and **Joint Training**. The **Bare model** denotes the backbone GNN without any continual learning techniques, serving as the lower bound on continual learning performance. On the other hand, **Joint Training** trains the backbone GNN on all tasks simultaneously, resulting in no forgetting problems and serving as the upper bound for continual learning performance.

5.1 Summary of the Empirical Results

Impact of Structural Shift. We first show the effect of evolving graph structure (structural shift) on the continual learning system. We do so by comparing the dynamic of forgetting (change of $r_{i,j} - r_{i,i}$ for different i, j) of the Bare model. The result of the OGB-Arxiv dataset is illustrated in Fig. 2(a). In Fig. 2(a), T_1 and T_2 on the x-axis denote the first and the second tasks (training sessions), respectively, and T_2- represents the state when Task 2 has arrived but the model has not been trained on Task 2 data yet. As shown in the figure, in the case of an evolving graph, the catastrophic forgetting on Task 1 occurs twice when entering stage 2: when vertices of Task 2 are added to the graph (structural shift) and when the model is trained on Task 2's data. Fig. 2(b) and Fig. 2(c) are the complete catastrophic forgetting matrix (with $r_{i,j} - r_{i,i}$ as entries) for the two settings. The colour density difference of the two performance matrices illustrates that the existing tasks suffer more catastrophic forgetting in the setting with evolving graphs. The results of this experiment align with our expectations, indicating that the existence of structural shifts can lead to a larger catastrophic forgetting.

We then further quantify the relation between the structural shift and the performance of a continual learning system. We use cSBM to create a two-session NGCL with a two-community graph of the size 300 for each training session (task), and the learning objective is to differentiate and classify vertices between the two communities. The community membership configuration is described by a four-tuple, $(c_0^{(1)}, c_1^{(1)}, c_0^{(2)}, c_1^{(2)}) = \big(150+\delta, 150-\delta, 150-\delta, 150+\delta\big)$, where $c_i^{(j)}$ is the number of vertices from community i at session j. Then, we vary the structural shift by changing (1) the inter-connectivity p_{inter} (probability of an edge between tasks/sessions) and (2) the difference (δ) of each community member in different stages. Intuitively, as we increase δ and the inter-connectivity p_{inter}, the structural shift between the two training sessions increases and therefore we should observe a larger FAF converging to random guess as predicted by the theoretical results. As shown in Fig. 3(a), this is indeed the case and therefore validating Theorem 3.1.

Distortion Rate. We next validate that the distortion rate of GNNs is indeed small. We train a vanilla GraphSAGE model using the default setting for each dataset. We extract vertices within the 5-hop neighbourhood of the training set and group the vertices based on their distances to the training set. We then compute the average embedding distance of each group to the training set in the embedding space. In Fig. 3(b), the distortion rate and scaling factor of the GNN are reflected by the slope of the near-linear relation between the embedding distance and the graph distance. We can see that the distortion factors in the performance bound in Proposition 4.1 are indeed small (closed to 1), validating our proposed experience selection strategy.

Effectiveness of SEA-ER. In Table 1, we show the FAP and FAF of each method after learning the entire task sequence. On average, the Bare model without any continual learning technique performs the worst, and Joint training performs the best. FAF is inapplicable to joint-trained models because they do not follow the continual learning setting and are simultaneously trained on all tasks. Our experience buffer selection strategy, SEA-ER w.o. sa, outperforms ER-rep, ER-infl and ER-deg in the proposed setting. In addition, the modified learning objective Eq. 4.2 with importance re-weighting can boost the performance of all experience replay frameworks by adjusting the structural shifts induced by the evolving graph structure.

Ablation Study on the Size of Experience Replay Buffer. As illustrated in Fig. 3(c), the performance of SEA-ER-re, predictably, converges to the Joint Training setting as the size of the experience replay buffer increases. Nonetheless, the size of the experience replay buffer is often subject to system constraints, such as the available storage capacity. The experiment demonstrates that SEA-ER is highly effective in preserving the performance of the system (as indicated by the final average task performance, FAP), even when the size of the experience replay buffer is limited.

6 Conclusion

This paper offers a comprehensive theoretical exploration of the learnability of GNNs in NGCL within the context of evolving graphs. Our formal analysis reveals a crucial insight: GNNs may not be always learnable in NGCL when structural shifts are uncontrolled. Furthermore, we introduce a novel experience replay framework, SEA-ER, adept at addressing the challenges of catastrophic forgetting and structural shifts inherent in this problem. Empirical results validate our theoretical findings and underscore the effectiveness of our proposed approach.

6.1 Limitation and Future Work

The lower bound of catastrophic forgetting for GNNs in NGCL presented in this paper is framed with the existence of a learning algorithm. While it offers insight into the extent of structural shifts that can induce catastrophic forgetting of GNNs, establishing a more quantitative relationship between structural shifts and catastrophic forgetting of GNNs in NGCL would be a valuable avenue for further research. This could enhance our understanding of GNNs in NGCL and facilitate the development of improved methods.

Our study here primarily investigates widely-used GNN architectures, including GCN, GAT, and SAGE. Nevertheless, in NGCL scenarios characterized by pronounced heterophily, it becomes imperative to explore heterophily-specific GNN models, which may employ distinct aggregation and update mechanisms. Such models could fundamentally alter the approach to addressing the problem. This highlights a potential avenue for significant future research within the domain of graph continual learning. However, it falls outside the purview of our current study.

Acknowledgement

We would like to thank the anonymous reviewers and area chairs for their helpful comments. JS and CW are supported in part by Hong Kong RGC grants 17207621, 17203522 and C7004-22G (CRF). DZ is supported by NSFC 62306252 and central fund from HKU IDS.

References

Kian Ahrabian, Yishi Xu, Yingxue Zhang, Jiapeng Wu, Yuening Wang, and Mark Coates. Structure aware experience replay for incremental learning in graph-based recommender systems. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pp. 2832–2836, 2021.

Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. *Advances in neural information processing systems*, 32, 2019.

Shai Ben-David and Reba Schuller Borbely. A notion of task relatedness yielding provable multiple-task learning guarantees. *Machine learning*, 73:273–287, 2008.

Shai Ben-David and Ruth Urner. On the hardness of domain adaptation and the utility of unlabeled target samples. In *International Conference on Algorithmic Learning Theory*, pp. 139–153. Springer, 2012.

Diana Benavides-Prado and Patricia Riddle. A theory for knowledge transfer in continual learning. In *Conference on Lifelong Learning Agents*, pp. 647–660. PMLR, 2022.

Piotr Bielak, Kamil Tagowski, Maciej Falkiewicz, Tomasz Kajdanowicz, and Nitesh V Chawla. Fildne: a framework for incremental learning of dynamic networks embeddings. *Knowledge-Based Systems*, 236:107453, 2022.

Magdalena Biesialska, Katarzyna Biesialska, and Marta R Costa-jussà. Continual lifelong learning in natural language processing: A survey. *arXiv preprint arXiv:2012.09823*, 2020.

Aleksandar Bojchevski and Stephan Günnemann. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. *arXiv preprint arXiv:1707.03815*, 2017.

Lucas Caccia, Eugene Belilovsky, Massimo Caccia, and Joelle Pineau. Online learned continual compression with adaptive quantization modules. In *International conference on machine learning*, pp. 1240–1250. PMLR, 2020.

Jie Cai, Xin Wang, Chaoyu Guan, Yateng Tang, Jin Xu, Bin Zhong, and Wenwu Zhu. Multimodal continual graph learning with neural architecture search. In *Proceedings of the ACM Web Conference 2022*, pp. 1292–1300, 2022.

Antonio Carta, Andrea Cossu, Federico Errica, and Davide Bacciu. Catastrophic forgetting in deep graph networks: an introductory benchmark for graph classification. *arXiv preprint arXiv:2103.11750*, 2021.

Aristotelis Chrysakis and Marie-Francine Moens. Online continual learning from imbalanced data. In *International Conference on Machine Learning*, pp. 1952–1961. PMLR, 2020.

Angel Daruna, Mehul Gupta, Mohan Sridharan, and Sonia Chernova. Continual learning of knowledge graph embeddings. *IEEE Robotics and Automation Letters*, 6(2):1128–1135, 2021.

Shai Ben David, Tyler Lu, Teresa Luu, and Dávid Pál. Impossibility theorems for domain adaptation. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 129–136. JMLR Workshop and Conference Proceedings, 2010.

Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

Yash Deshpande, Subhabrata Sen, Andrea Montanari, and Elchanan Mossel. Contextual stochastic block models. *Advances in Neural Information Processing Systems*, 31, 2018.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 3762–3773. PMLR, 2020.

Falih Gozi Febrinanto, Feng Xia, Kristen Moore, Chandra Thapa, and Charu Aggarwal. Graph lifelong learning: A survey. *IEEE Computational Intelligence Magazine*, 18(1):32–51, 2023.

Yutong Feng, Jianwen Jiang, and Yue Gao. Incremental learning on growing graphs. *openReview preprint https://openreview.net/forum?id=nySHNUlKTVw*, 2020.

Lukas Galke, Benedikt Franke, Tobias Zielke, and Ansgar Scherp. Lifelong learning of graph neural networks for open-world node classification. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2021.

João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM computing surveys (CSUR)*, 46(4):1–37, 2014.

Arthur Gretton, Alex Smola, Jiayuan Huang, Marcel Schmittfull, Karsten Borgwardt, and Bernhard Schölkopf. Covariate shift by kernel mean matching. *Dataset shift in machine learning*, 3(4):5, 2009.

William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 1025–1035, 2017.

William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs, 2018.

Yi Han, Shanika Karunasekera, and Christopher Leckie. Graph neural networks with continual learning for fake news detection from social media. *arXiv preprint arXiv:2007.03316*, 2020.

Dorit S Hochbaum. Approximating covering and packing problems: set cover, vertex cover, independent set, and related problems. In *Approximation algorithms for NP-hard problems*, pp. 94–143. 1996.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *arXiv preprint arXiv:2005.00687*, 2020.

Heechul Jung, Jeongwoo Ju, Minju Jung, and Junmo Kim. Less-forgetting learning in deep neural networks. *arXiv preprint arXiv:1607.00122*, 2016.

Seyed Mehran Kazemi, Rishab Goel, Kshitij Jain, Ivan Kobyzev, Akshay Sethi, Peter Forsyth, and Pascal Poupart. Representation learning for dynamic graphs: A survey. *J. Mach. Learn. Res.*, 21(70):1–73, 2020.

Seoyoon Kim, Seongjun Yun, and Jaewoo Kang. Dygrain: An incremental learning framework for dynamic graphs. In *31st International Joint Conference on Artificial Intelligence, IJCAI 2022*, pp. 3157–3163. International Joint Conferences on Artificial Intelligence, 2022.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2017.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Jeremias Knoblauch, Hisham Husain, and Tom Diethe. Optimal continual learning has perfect memory and is np-hard. In *International Conference on Machine Learning*, pp. 5327–5337. PMLR, 2020.

Xiaoyu Kou, Yankai Lin, Shaobo Liu, Peng Li, Jie Zhou, and Yan Zhang. Disentangle-based continual graph representation learning. *arXiv preprint arXiv:2010.02565*, 2020.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

Huihui Liu, Yiding Yang, and Xinchao Wang. Overcoming catastrophic forgetting in graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 8653–8661, 2021.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017.

Jiaqi Ma, Junwei Deng, and Qiaozhu Mei. Subgroup generalization and fairness of graph neural networks, 2021.

Yao Ma, Ziyi Guo, Zhaocun Ren, Jiliang Tang, and Dawei Yin. Streaming graph neural networks. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 719–728, 2020.

Giang Hoang Nguyen, John Boaz Lee, Ryan A Rossi, Nesreen K Ahmed, Eunyee Koh, and Sungchul Kim. Continuous-time dynamic network embeddings. In *Companion proceedings of the the web conference 2018*, pp. 969–976, 2018.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

Ievgen Redko, Emilie Morvant, Amaury Habrard, Marc Sebban, and Younès Bennani. A survey on domain adaptation theory: learning bounds and theoretical guarantees. *arXiv preprint arXiv:2004.11829*, 2020.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. *arXiv preprint arXiv:2103.09762*, 2021.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30, 2017.

Junwei Su, Difan Zou, Zijun Zhang, and Chuan Wu. Towards robust graph incremental learning on evolving graphs. 2023.

Junwei Su, Lingjun Mao, and Chuan Wu. Bg-hgnn: Toward scalable and efficient heterogeneous graph neural network. *arXiv preprint arXiv:2403.08207*, 2024a.

Junwei Su, Difan Zou, and Chuan Wu. Pres: Toward scalable memory-based dynamic graph neural networks. *arXiv preprint arXiv:2402.04284*, 2024b.

Zhen Tan, Kaize Ding, Ruocheng Guo, and Huan Liu. Graph few-shot class-incremental learning. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pp. 987–996, 2022.

Chen Wang, Yuheng Qiu, and Sebastian Scherer. Bridging graph network to lifelong learning with feature interaction. 2020a.

Chen Wang, Yuheng Qiu, Dasong Gao, and Sebastian Scherer. Lifelong graph learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13719–13728, 2022.

Junshan Wang, Guojie Song, Yi Wu, and Liang Wang. Streaming graph neural networks via continual learning. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pp. 1515–1524, 2020b.

Mitchell Wortsman, Vivek Ramanujan, Rosanne Liu, Aniruddha Kembhavi, Mohammad Rastegari, Jason Yosinski, and Ali Farhadi. Supermasks in superposition. *Advances in Neural Information Processing Systems*, 33:15173–15184, 2020.

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 374–382, 2019.

Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems*, 32(1):4–24, 2020.

Yishi Xu, Yingxue Zhang, Wei Guo, Huifeng Guo, Ruiming Tang, and Mark Coates. Graphsail: Graph structure aware incremental learning for recommender systems. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pp. 2861–2868, 2020.

Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. *arXiv preprint arXiv:1708.01547*, 2017.

Jaehong Yoon, Saehoon Kim, Eunho Yang, and Sung Ju Hwang. Scalable and order-robust continual learning with additive parameter decomposition. *arXiv preprint arXiv:1902.09432*, 2019.

Wenchao Yu, Wei Cheng, Charu C Aggarwal, Kai Zhang, Haifeng Chen, and Wei Wang. Netwalk: A flexible deep embedding approach for anomaly detection in dynamic networks. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2672–2681, 2018.

Qiao Yuan, Sheng-Uei Guan, Pin Ni, Tianlun Luo, Ka Lok Man, Prudence Wong, and Victor Chang. Continual graph learning: A survey. *arXiv preprint arXiv:2301.12230*, 2023.

Xikun Zhang, Dongjin Song, and Dacheng Tao. Hierarchical prototype networks for continual graph representation learning. *arXiv preprint arXiv:2111.15422*, 2021.

Xikun Zhang, Dongjin Song, and Dacheng Tao. Cglb: Benchmark tasks for continual graph learning. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

Fan Zhou and Chengtai Cao. Overcoming catastrophic forgetting in graph neural networks with experience replay. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 4714–4722, 2021a.

Fan Zhou and Chengtai Cao. Overcoming catastrophic forgetting in graph neural networks with experience replay. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 4714–4722, 2021b.

Fan Zhou and Chengtai Cao. Overcoming catastrophic forgetting in graph neural networks with experience replay. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 4714–4722, 2021c.

Indre Zliobaite. Learning under concept drift: an overview. *arXiv preprint arXiv:1010.4784*, 41, 2010.

A Further Discussion on Learnability and Proof of the Theorem 3.1

In this appendix, we provide a proof for Theorem 3.1. The key idea of the proof is to establish a formal connection between the NGCL-2 problem with the domain adaptation problem (henceforth we refer to it as the DA problem). We begin by giving a brief introduction to the domain adaptation (DA) problem. Without loss of generality, we focus the analysis on binary classification tasks. Obviously, if binary classification is impossible, then an arbitrary k-way classification is also impossible.

A.1 Domain Adaptation

Let $\mathcal{X}$ and $\mathcal{Y} = \{0, 1\}$ be the input and output space, and $l : \mathcal{X} \mapsto \mathcal{Y}$ be the labelling function. In the DA problem, there are two different distributions over $\mathcal{X}$, the source distribution $\mathcal{S}$ and the target distribution $\mathcal{T}$. The goal of domain adaptation is to learn a classifier $h : \mathcal{X} \mapsto \mathcal{Y}$ that performs well on the target distribution, i.e., $R^l_{\mathcal{T}}(h)$ is small, with the knowledge from the source domain $\mathcal{S}$. Similar to the NGCL-2 problem, the learnability of the DA problem can be defined as follow.

Definition 4 (DA-learnability). *Let $\mathcal{S}$ and $\mathcal{T}$ be distribution over $\mathcal{X}$, $\mathcal{H}$ a hypothesis class, $\epsilon > 0, \delta > 0$. We say that the DA problem is learnable if there exists a learning algorithm that produces a classifier h from $\mathcal{H}$ with probability at least $1 - \delta$, and h incurs an error smaller than ϵ, i.e., $Pr[R^l_{\mathcal{T}}(h) \leq \epsilon] \geq 1 - \delta$, when given access to labelled samples L of size m from $\mathcal{S}$ and unlabelled samples U of size n from $\mathcal{T}$.*

A.2 Connection between DA and NGCL-2

Recall that The NGCL-2 problem is characterized by three distributions: $\mathbb{P}(\mathbf{y}, \mathbf{g}_v|\mathcal{V}_1, \mathcal{G}_{\tau_1})$, $\mathbb{P}(\mathbf{y}, \mathbf{g}_v|\mathcal{V}_1, \mathcal{G}_{\tau_2})$, and $\mathbb{P}(\mathbf{y}, \mathbf{g}_v|\mathcal{V}_2, \mathcal{G}_{\tau_2})$. To simplify the notation, we re-write $\mathbb{P}(\mathbf{y}, \mathbf{g}_v|\mathcal{V}_1, \mathcal{G}_{\tau_1})$, $\mathbb{P}(\mathbf{y}, \mathbf{g}_v|\mathcal{V}_1, \mathcal{G}_{\tau_2})$, $\mathbb{P}(\mathbf{y}, \mathbf{g}_v|\mathcal{V}_2, \mathcal{G}_{\tau_2})$ as $\mathbb{P}^{(1)}_1, \mathbb{P}^{(2)}_1, \mathbb{P}^{(2)}_2$. Then, the learnability of NGCL-2 problem can be formally defined as,

Definition 5 (NGCL-2-learnability). *Given a NGCL-2 with distributions $\mathbb{P}^{(1)}_1, \mathbb{P}^{(2)}_1, \mathbb{P}^{(2)}_2$ and a labelling function l, let $\mathcal{H}$ a hypothesis class, $\epsilon > 0, \delta > 0$. We say that the NGCL-2 problem is learnable relative to $\mathcal{H}$, if there exists a learning algorithm that outputs a classifier h from $\mathcal{H}$ with probability of at least $1 - \delta$ and h has error less than ϵ, i.e., $Pr[R^l_{\mathbb{P}}(h) \leq \epsilon] \geq 1 - \delta$, when given access to labeled samples L from $\mathbb{P}$ ($\mathbb{P}$ is the mixed distribution of $\mathbb{P}^{(1)}_1$ and $\mathbb{P}^{(2)}_2$) of size $m + n$ and unlabelled samples U of size k from $\mathbb{P}^{(2)}_1$*

To establish a formal connection between NGCL-2 and DA, we consider a relaxed version of the NGCL-2 problem where the learning algorithm allow to access the same amount of data in an arbitrary order from distribution $\mathbb{P}^{(1)}_1$ and $\mathbb{P}^{(2)}_2$ and only needs to perform well on the updated distribution $\mathbb{P}^{(2)}_1$. We refer to this relaxed version of NGCL-2 problem as NGCL-2-weak and its learnability is defined as follows.

Definition 6 (NGCL-2-weak-learnability). *Given a NGCL-2 with distributions $\mathbb{P}^{(1)}_1, \mathbb{P}^{(2)}_1, \mathbb{P}^{(2)}_2$ and a labelling function l, let $\mathcal{H}$ a hypothesis class, $\epsilon > 0, \delta > 0$. We say that the NGCL-2 problem is learnable relative to $\mathcal{H}$, if there exists a learning algorithm that outputs a classifier h from $\mathcal{H}$ with probability of at least $1 - \delta$ and h has error less than ϵ, i.e., $Pr[R^l_{\mathbb{P}^{(2)}_1}(h) \leq \epsilon] \geq 1 - \delta$, when given access to labeled samples L from P (P is the mixed distribution of $\mathbb{P}^{(1)}_1$ and $\mathbb{P}^{(2)}_2$) of size $m + n$ and unlabelled samples U of size k from $\mathbb{P}^{(2)}_1$*

It is obvious that the NGCL-2-weak-learnability definition subsume the NGCL-2-learnability. In other words, we can obtain the following corollary immediately from the definition.

Corollary A.1. *If the NGCL-2-weak problem is not learnable, then the NGCL-2 problem is not learnable.*

Next, we establish a formal connection between the DA problem and the NGCL-2-weak problem.

Theorem A.2 (Connection of DA and NGCL-2-weak). *Let $\mathcal{X}$ be a domain and $\mathcal{H}$ the hypothesis class on $\mathcal{X} \times \{0, 1\}$. Suppose an instance of the DA problem with distributions $\mathcal{S}$ and $\mathcal{T}$ on $\mathcal{X}$, and there is access to m' labelled samples from $\mathcal{S}$ and k' unlabelled samples from $\mathcal{T}$. Similarly, suppose an instance of an NGCL-2-weak problem with distributions $\mathbb{P}^{(1)}_1, \mathbb{P}^{(2)}_1, \mathbb{P}^{(2)}_2$ over $\mathcal{X}$,and there is access to $m + n$ labelled samples from $\mathbb{P}^{(1)}_1, \mathbb{P}^{(2)}_1$ and k unlabelled samples from $\mathbb{P}^{(2)}_1$. For the same hypothesis sapce $\mathcal{H}$, assuming that $m' = m + n, k' = k$, $\lambda_{\mathcal{H}}(\{\mathcal{S}, \mathcal{T}\}) = \lambda_{\mathcal{H}}(\{P, \mathbb{P}^{(2)}_1\})$ and $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S}, \mathcal{T}) = d_{\mathcal{H}\Delta\mathcal{H}}(P, \mathbb{P}^{(2)}_1)$, then we have that the DA is learnable if and only if the NGCL-2-weak problem is learnable.*

Proof. We first show that under the premise of the theorem if the NGCL-2-weak learnable, then the DA problem is learnable.

Suppose we are given an instance of DA problem of distributions $\mathcal{S}$, $\mathcal{T}$ and a $\mathcal{H}$. Let $\mathcal{S}$ be a mixed distribution of $\mathcal{S}_1$ and $\mathcal{S}_2$. Then, we can map the DA to a NGCL-2-weak problem as follows.

$$\mathcal{S}_1 = \mathbb{P}_1^{(1)}, \mathcal{S}_2 = \mathbb{P}_1^{(2)}, \mathcal{T} = \mathbb{P}_1^{(2)}$$

Under the mapping above, we have $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S}, \mathcal{T}) = d_{\mathcal{H}\Delta\mathcal{H}}(P, \mathbb{P}_1^{(2)})$ and $\lambda_{\mathcal{H}}(\{\mathcal{S}, \mathcal{T}\}) = \lambda_{\mathcal{H}}(\{P, \mathbb{P}_1^{(2)}\})$, where P is the mixed distribution of distributions $\mathbb{P}_1^{(1)}$ and $\mathbb{P}_1^{(2)}$.

By the assumption, the NGCL-2-weak problem is learnable. By Def. 6, this means that there exist a learning algorithm that takes in $m + n$ labelled data from P and k unlabelled data from $\mathbb{P}_1^{(2)}$, and with probability at least $1 - \delta$, outputs a classifier h wiht at most ϵ error with respect to $\mathbb{P}_1^{(2)}$. By the mapping above and the premise of the theorem that $m + n = m', k' = k$, the same learning algorithm would satisfy Def. 4. This shows that the learnability of NGCL-2-weak implies the learnability of DA problem.

Now we show the opposite that under the premise of the theorem if the DA problem is learnable, then the NGCL-2-weak learnable.

Let's reverse the construction direction. Suppose we are given an instance of NGCL-2-weak problem of distributions $\mathbb{P}_1^{(1)}, \mathbb{P}_1^{(2)}, \mathbb{P}_1^{(2)}$ and a $\mathcal{H}$. Let $\mathcal{P}$ be a mixed distribution of $\mathbb{P}_1^{(1)}$ and $\mathbb{P}_1^{(2)}$. Then, we can map the NGCL-2-weak problem to DA problem as follows.

$$P = \mathcal{S}, \mathbb{P}_1^{(2)} = \mathcal{T}$$

Similarly, under the mapping above, we have $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S}, \mathcal{T}) = d_{\mathcal{H}\Delta\mathcal{H}}(P, \mathbb{P}_1^{(2)})$ and $\lambda_{\mathcal{H}}(\{\mathcal{S}, \mathcal{T}\}) = \lambda_{\mathcal{H}}(\{P, \mathbb{P}_1^{(2)}\})$.

By the assumption, the DA problem is learnable. By Def. 4, this means that there exist a learning algorithm that takes in m' labelled data from $\mathcal{S}$ and k' unlabelled data from $\mathcal{T}$, and with probability at least $1 - \delta$, outputs a classifier h wiht at most ϵ error with respect to $\mathcal{T}$. By the mapping above and the premise of the theorem that $m + n = m', k' = k$, the same learning algorithm would satisfy Def. 6. This shows that the learnability of DA problem implies the learnability of the NGCL-2-weak problem.

This completes the proof of Theorem A.2. □

Now that we have shown that the learnability of DA problem is equivalent to the learnability of the NGCL-2-weak problem. Combining Theorem A.2 and Corollary A.1, we immediately obtain the following corollary on the connection between the NGCL-2 problem with Def. 5 and the domain adaptation problem.

Corollary A.3 (Connection of DA and NGCL-2)**.** *Let $\mathcal{X}$ be a domain and $\mathcal{H}$ the hypothesis class on $\mathcal{X} \times \{0, 1\}$. Suppose an instance of the DA problem with distributions $\mathcal{S}$ and $\mathcal{T}$ on $\mathcal{X}$, and there is access to m' labelled samples from $\mathcal{S}$ and k' unlabelled samples from $\mathcal{T}$. Similarly, suppose an instance of a NGCL-2 problem with distributions $\mathbb{P}_1^{(1)}, \mathbb{P}_1^{(2)}, \mathbb{P}_1^{(2)}$ over $\mathcal{X}$,and there is sequential access to m labelled samples from $\mathbb{P}_1^{(1)}$, n labelled samples from $\mathbb{P}_1^{(2)}$ and k unlabelled samples from $\mathbb{P}_1^{(2)}$. For the same hypothesis sapce $\mathcal{H}$, assuming that $m' = m + n, k' = k$, $\lambda_{\mathcal{H}}(\{\mathcal{S}, \mathcal{T}\}) = \lambda_{\mathcal{H}}(\{P, \mathbb{P}_1^{(2)}\})$ and $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S}, \mathcal{T}) = d_{\mathcal{H}\Delta\mathcal{H}}(P, \mathbb{P}_1^{(2)})$, then we have that the if the DA is not learnable, then the NGCL-2 problem is not learnable.*

A.3 Proof for Theorem 3.1

Corollary A.3 above establishes a formal connection between the DA problem and the NGCL-2 problem. Under the same proposed mapping, the infeasibility of DA implies the infeasibility of NGCL-2. DA is a well-studied problem with mature understanding on its learnability.

David et al. (2010) has shown that small divergent distance between source domain and target domain is a "necessary" condition for the success of domain adaptation. We restate the theorem below.

Theorem A.4 ((David et al., 2010))**.** *Let $\mathcal{X}$ be some domain set, and $\mathcal{H}$ a class of functions over $\mathcal{X} \times \{0, 1\}$. For every $c > 0$ there exists probability distributions $\mathcal{S}, \mathcal{T}$ over $\mathcal{X}$ such that for every domain adaptation learner, every integers $m, n > 0$, there exists a labeling function $l : \mathcal{X} \mapsto 0, 1$ such that $\lambda_{\mathcal{H}}(P, Q) \leq c$ and the DA problem is not learnable.*

The proof of Theorem A.4 can be found in (David et al., 2010). Then the proof of Theorem 3.1 follows immediately from Theorem A.4 and Corollary A.3.

B Expressiveness and Sample Complexity

In this appendix, building upon the connection established in the previous appendix, we provide a further discussion of the sample complexity and expressiveness.

Theorem B.1 (Necessity of the Expressiveness of Hypothesis Space)**.** *Let $\mathcal{X}$ be some domain set, and $\mathcal{H}$ be a class of functions over $\mathcal{X} \times \{0, 1\}$ whose VC dimension is much smaller than $|\mathcal{X}|$. Then for every $c > 0$, there exists a NGCL-2 problem such that $d_{\mathcal{H}\Delta\mathcal{H}}(\mathbb{P}, \mathbb{P}_1^{(2)}) \leq c$, where $\mathbb{P}$ is the mixed distribution of $\mathbb{P}_1^{(1)}$ and $\mathbb{P}_2^{(2)}$, and the NGCL-2 problem is not learnable with any amount of labelled data from $\mathbb{P}_1^{(1)}, \mathbb{P}_2^{(2)}$ and unlablled data from $\mathbb{P}_1^{(2)}$.*

Theorem B.1 formalizes the intuition that the hypothesis class (GNN model) needs to be expressive enough to capture different distributions in NGCL, for it to be learnable.

Theorem B.2 (Sample Complexity)**.** *Let $\mathcal{X}$ be some compact domain set in $\mathbf{R}^d$, and $\mathcal{H}$ be a class of functions over $\mathcal{X} \times \{0, 1\}$. Suppose the labeling function $l : \mathcal{X} \mapsto \{0, 1\}$ is α-Lipschitz continuous. Then, if $m + n + k < \sqrt{(1 - 2(\epsilon + \delta))(\alpha + 1)^d}$, for every $c > 0$, there exists a NGCL-2 problem such that $\lambda_{\mathcal{H}}(\{\mathbb{P}_1^{(1)}, \mathbb{P}_2^{(2)}, \mathbb{P}_1^{(2)}\}) \leq c$, $d_{\mathcal{H}\Delta\mathcal{H}}(P, \mathbb{P}_1^{(2)}) \leq c$, where $\mathbb{P}$ is the mixed distribution of $\mathbb{P}_1^{(1)}$ and $\mathbb{P}_2^{(2)}$, and the NGCL-2 problem is not learnable.*

Theorem B.2 suggests that even if the two "necessary" conditions in Theorem 3.1 and Theorem B.1 are satisfied, there is still a sampling requirement depending on the dimension of the input domain and the smoothness of the labelling function. This shows the importance of having access to previous data for NGCL.

B.1 Proof for Theorem B.1 and Theorem B.2

The idea of the proof is similar to the one presented earlier. (David et al., 2010) has shown that expressive power of the hypothesis space is a "necessary" condition for the success of domain adaptation. We restate the theorem below.

Theorem B.3. *Let $\mathcal{X}$ be some domain set, and $\mathcal{H}$ a class of functions over $\mathcal{X} \times \{0, 1\}$ whose VC dimension is much smaller than $|\mathcal{X}|$. Then for every $c > 0$ there exists probability distributions Q, P over $\mathcal{X}$ such that for every domain adaptation learner, every integers $m, n > 0$, there exists a labeling function $l : \mathcal{X} \mapsto 0, 1$ such that $d_{\mathcal{H}\Delta\mathcal{H}}(P, Q) \leq c$ and the DA problem is not learnable.*

The proof of Theorem B.3 can be found in (David et al., 2010). Then the proof of Theorem B.1 follows immediately from Theorem B.3 and Corollary A.3.

(Ben-David & Urner, 2012) has shown that even if DA problem has small divergent distance and the hypothesis space is expressive enough, there is still a sampling requirement for the DA problem to be learnable. We restate the theorem below.

Theorem B.4. *Let $\mathcal{X} \subset \mathbf{R}^d$ and $\mathcal{H}$ a class of functions over $\mathcal{X} \times \{0, 1\}$. Suppose the labelling function l is α-Lipschitz continuous. For every $c > 0$ there exist $\mathcal{S}, \mathcal{T}$ over $\mathcal{X}$ suc htat $\lambda_{\mathcal{H}} \leq c$ and $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S}, \mathcal{T}) \leq c$ and the DA problem is not learnable if $m + n \geq \sqrt{(\alpha + 1)^d(1 - 2(\epsilon + \delta))}$, where m, n, ϵ, δ are given in Def. 4.*

The proof of Theorem B.4 can be found in (Ben-David & Urner, 2012). Then the proof of Theorem B.2 follows immediately from Theorem B.4 and Corollary A.3.

Remark B.5. *In addition to the learnability studies, there exist a large body of studies investigating different techniques for domain adaptation. The formal connection A.3 allow to rigorously connection the NGCL-2 problem and DA problem and to carefully transfer or explore the understanding and techniques from DA to NGCL-2.*

C Proof of Proposition 4.1

In this appendix, we provide proof for Proposition 4.1. We begin with restating the definition of distortion.

Definition 7 (distortion rate)**.** *Given two metric spaces $(\mathcal{Q}, d)$ and $(\mathcal{Q}', d')$ and a mapping $\gamma : \mathcal{Q} \mapsto \mathcal{Q}'$, γ is said to have a distortion $\alpha \geq 1$, if there exists a constant $r > 0$ such that $\forall u, v \in \mathcal{E}$, $rd(u, v) \leq d'(\gamma(u), \gamma(v)) \leq \alpha r d(u, v)$.*

Proof. Suppose $P_i \subset \mathcal{V}_i$ is the experience replay set selected by solving Eq. 4.1. Let s be the structure of interest with distance measure d_s. Let g be a given GNN model with distortion α and scaling factor r and its prediction function g.

Let $\sigma : \mathcal{V}_i \mapsto P_i$ denote a mapping that map a vertex v from $\mathcal{V}_i$ to the closest vertex in P_i, i.e.,

$$\sigma(v) = \operatorname*{argmin}_{u \in P_i} d_s(u, v). \tag{C.1}$$

Let's consider the loss of vertex v, $L(g \circ g(v), y_v)$. By assumption, L is smooth and let's denote B_L^{up1} to be the upper bound for the first derivative with respect to $g \circ g(v)$ and B_L^{up2} to be the upper bound for the first derivative with respect to y_v. Then, consider the Taylor expansion of $L(g \circ g(v), y_v)$ with respect to $\sigma(v)$, which is given as follows

$$\begin{aligned} L(g \circ g(v), y_v) \leq L(g \circ g(\sigma(v)), y_{\sigma(v)}) + \\ B_L^{up1} ||g \circ g(v) - g \circ g(\sigma(v))|| \\ + B_L^{up2} ||y_v - y_{\sigma(v)}|| \end{aligned} \tag{C.2}$$

Next, let's examine the inequality above term by term and start with $||g \circ g(v) - g \circ g(\sigma(v))||$. Let's denote h_v and $h_{\sigma(v)}$ the embedding for vertex v and $\sigma(v)$. Then, we have that,

$$||g \circ g(v) - g \circ g(\sigma(v))|| = ||g(h_v) - g(h_{\sigma(v)})|| \tag{C.3}$$

By definition of distortion as given in Def. 7, we have that,

$$||h_v - h_{\sigma(v)}|| \leq r\alpha d_s(v, \sigma(v)) \tag{C.4}$$

By assumption, the prediction function g is smooth. Let B_g denote the upper bound of first derivative of the prediction function g. Then, we have

$$||g(h_v) - g(h_{\sigma(v)})|| \leq B_g^{up} ||h_v - h_{\sigma(v)}|| \tag{C.5}$$

Substitute all these back, we have

$$||g \circ g(v) - g \circ g(\sigma(v))|| \leq B_g^{up} r\alpha d_s(v, \sigma(v)) \tag{C.6}$$

Next, let's consider $||y_v - y_{\sigma(v)}||$. Similarly, by the data smoothness assumption and distortion, we have that

$$||y_v - y_{\sigma(v)}|| \leq B_l ||h_v - h_{\sigma(v)}|| \leq B_l^{up} r\alpha d_s(v, \sigma(v)) \tag{C.7}$$

Substitute these back to the inequality we start with, we have that

$$\begin{aligned} & L(g \circ g(v), y_v) \leq L(g \circ g(\sigma(v)), y_{\sigma(v)}) + \\ & \quad B_L^{up1} B_g^{up} r\alpha d_s(v, \sigma(v)) + B_L^{up2} B_l^{up} r\alpha d_s(v, \sigma(v)) \\ & \leq L(g \circ g(\sigma(v)), y_{\sigma(v)}) + \\ & (B_l^{up} B_L^{up1} + B_g^{up} B_L^{up2}) r\alpha \max_{v' \in \mathcal{V}_i} d_s(v', \sigma(v')) \\ & = L(g \circ g(\sigma(v)), y_{\sigma(v)}) + \\ & (B_l^{up} B_L^{up1} + B_g^{l} B_L^{up2}) r\alpha D_s(\mathcal{V}_i, P_i). \end{aligned} \tag{C.8}$$

Since the inequality above holds for every vertex v, then we have

$$\begin{aligned} R^L(g \circ g, \mathcal{V}_i) \leq R^L(g \circ g, P_i) + \\ (B_l^{up} B_L^{up1} + B_g^{up} B_L^{up2}) r\alpha D_s(\mathcal{V}_i, P_i) \\ = R^L(g \circ g, P_i) + \mathcal{O}(\alpha D_s(\mathcal{V}_i, P_i)) \end{aligned} \tag{C.9}$$

□

D Additional Experimental Details

In this appendix, we provide additional experimental results and include a detailed set-up of the experiments for reproducibility.

D.1 Hardware and System

All the experiments of this paper are conducted on the following machine

CPU: two Intel Xeon Gold 6230 2.1G, 20C/40T, 10.4GT/s, 27.5M Cache, Turbo, HT (125W) DDR4-2933

GPU: four NVIDIA Tesla V100 SXM2 32G GPU Accelerator for NV Link

Memory: 256GB (8 x 32GB) RDIMM, 3200MT/s, Dual Rank

OS: Ubuntu 18.04LTS

D.2 Dataset and Processing

D.2.1 Dataset Description

OGB-Arxiv. The OGB-Arxiv dataset Hu et al. (2020) is a benchmark dataset for node classification. It is constructed from the arXiv e-print repository, a popular platform for researchers to share their preprints. The graph structure is constructed by connecting papers that cite each other. The node features include the text of the paper's abstract, title, and its authors' names. Each node is assigned one of 40 classes, which corresponds to the paper's main subject area.

Cora-Full. The Cora-Full Bojchevski & Günnemann (2017) is a benchmark dataset for node classification. Similarly to OGB-Arxiv, it is a citation network consisting of 70 classes.

Reddit. The Reddit dataset Hamilton et al. (2017) is a benchmark dataset for node classification that consists of posts and comments from the website Reddit.com. Each node represents a post or comment and each edge represents a reply relationship between posts or comments.

Table 3: Continual learning settings for each dataset.

Datasets	OGB-Arxiv	Reddit	CoraFull
# vertices	169,343	227,853	19,793
# edges	1,166,243	114,615,892	130,622
# class	40	40	70
# tasks	20	20	35
# vertices / # task	8,467	11,393	660
# edges / # task	58,312	5,730,794	4,354

D.2.2 License

The datasets used in this paper are curated from existing public data sources and follow their licenses. OGB-Arxiv is licensed under Open Data Commons Attribution License (ODC-BY). Cora-Full dataset and the Reddit dataset are two datasets built from publicly available sources (public papers and Reddit posts) without a license attached by the authors.

D.2.3 Data Processing

For the datasets, we remove the 41-th class of Reddit-CL, following closely in Zhang et al. (2022). This aims to ensure an even number of classes for each dataset to be divided into a sequence of 2-class tasks. For all the datasets, the train-validation-test splitting ratios are 60%, 20%, and 20%. The train-validation-test splitting is obtained by random sampling, therefore the performance may be slightly different with splittings from different rounds of random sampling.

D.3 Continual Training Procedure

Suppose $\tau_1, \tau_2, ...\tau_m$ is the set of classification tasks created from the dataset. Then, the standard continual training procedure is given in Alg. 1 and a graphical illustration is given in Fig. 7.

D.4 Contextual Stochastic Block Model

We use the contextual stochastic block model (Deshpande et al., 2018) to create a graph for each learning stage. For each graph, we create 300 vertices and assign them a community label $\{0, 1\}$. We refer to a specific ratio of vertices from different communities as a configuration and the formula we use to vary the community ratio is as follows, configuration $i = (150 + (i-1) * 25, 150 - (i-1) * 25, 150 - (i-1) * 25, 150 + (i-1) * 25)$. As i increases, the difference in the ratio of the two learning stages increases. Intuitively, this should allow us to control the distribution difference between different learning stages.

Then we follow the standard stochastic block model. We use $p_{intra} = 0.15$ to randomly create edges among vertices of the same community and $p_{inter} = 0.1$ to randomly create edges among vertices of different communities. Then, we use $p_{stage} = [0.025, 0.05, 0.08, 0.1, 0.15]$ to create edges among vertices of different learning stages.

Then, the node features x_v of each vertex is created based on the following formula,

$$x_v = \sqrt{\frac{\mu}{n}} y_v u + \frac{Z_v}{p},$$

where $y_v \in \{0, 1\}$ is the community label of vertex v, $u, Z_v \sim \mathcal{N}(0, I)$ are random Gaussian of dimension p, n is the number of vertices in the graph and μ is a hyper-parameter of the model to control how separable of the features of the vertices from different communities. For our experiments, we use $\mu = 5, n = 300, p = 500$ for the graph in each stage.

D.5 Graph Neural Network Details

For our experiments, we use two popular GNN models, GCN (Kipf & Welling, 2017) and GraphSage (Hamilton et al., 2018). For implementation, we use the default implementation provided by DGL for realizing these models. For all our experiments, we use two GNN layers followed by an additional readout function of an MLP layer. We use Sigmoid as the activation function for the last hidden layer and Relu for the rest of the layers. We only vary the hidden dimension of the GNN layer for each experiment.

D.6 Difference between Inductive and Transductive

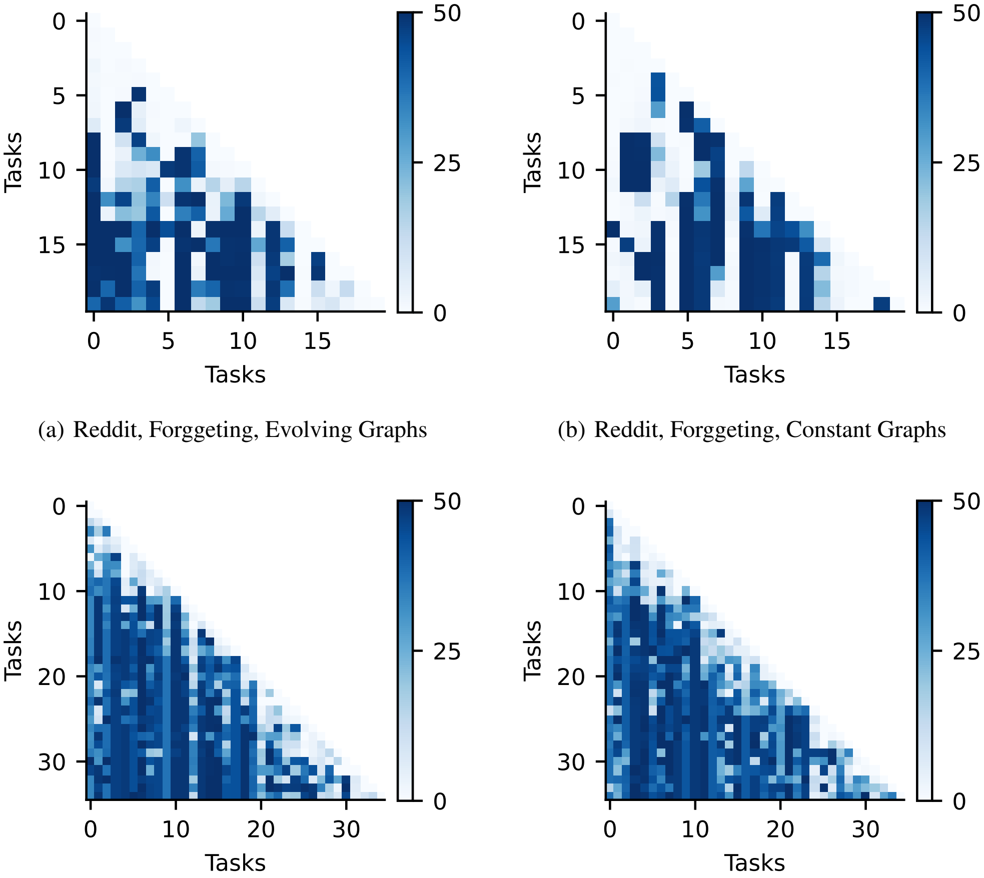
In this subsection, we provide the additional results on the remaining dataset for the setting with evolving graphs and constant graphs and additional illustrations on the difference between transductive and inductive NGCL (with structural shift). The additional experimental results are reported in Fig. 4 and 5. The graphical illustration of the difference between transductive and inductive NGCL is given in Fig. 8

D.7 Permutation of Task Order

In the empirical study, the task order is constructed once for each dataset. The reasons behind this are as follows. The focused continual learning strategy in this paper is experience replay, where the model is presented with (a small subset of) previous tasks regardless of their appearance order. Furthermore, the performance measures we used in the paper are FAP (Final Average Performance) and FAF (Final Average Forgetting). These metrics measure the performance of the model after being presented with all the tasks. As such, the order of the task permutation might affect the intermediate learning dynamic but should not affect the performance of the experience reply with the two chosen metrics. This is further validated by Fig. 6 shows the permutation of task order has minimal effect on the performance of experience replay on the chosen metrics.

Algorithm 1 Continual Training

```
Input: $\tau_1, \tau_2, ...\tau_m$ //classification task created from the dataset.
For $\tau$ in $\tau_1, \tau_2, ...\tau_m$:
    1. update the graph structure with data from $\tau_i$
    2. get $V_i^{\text{train}}$ for $\tau_i$
    3. update the existing GNN model and train a new task prediction head on $V_i^{\text{train}}$
    4. if evaluation, evaluate the model on the current and previous task
```



(a) Reddit, Forggeting, Evolving Graphs

(b) Reddit, Forggeting, Constant Graphs

(c) CoraFull, Forggeting, Evolving Graphs

(d) CoraFull, Forggeting, Constant Graphs

Figure 4: Performance Matrix of Bare Model on Transductive and Inductive Setting on CoraFull and Reddit Datasets.

E Complete Framework and Pesudo-code

E.1 Overall framework

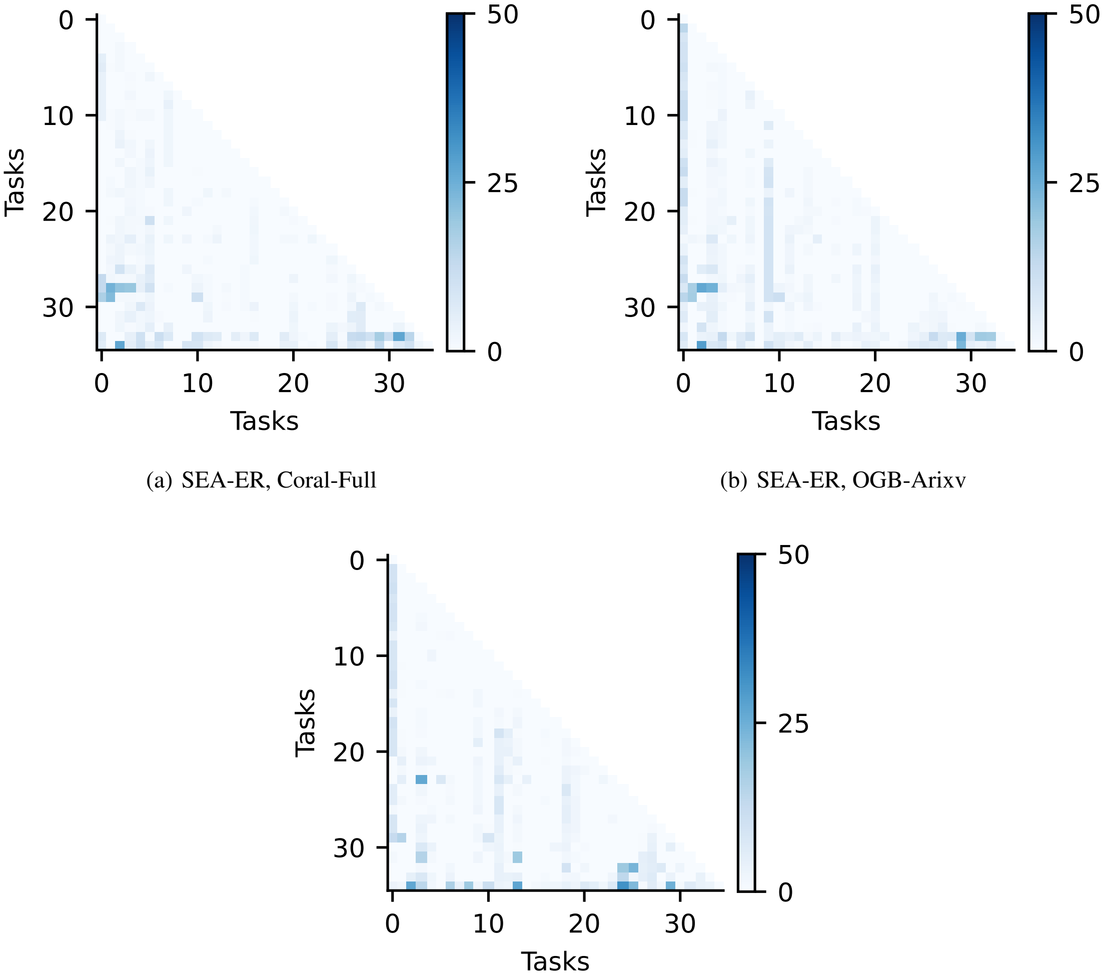
The continual learning procedure of our framework is summarized in Alg. 2 and Fig. 9 provide a graphical illustration of the training of the framework on a two-stage continual learning.

E.2 Selecting Replay Set

The replay set selection strategy is given in Eq. equation 4.1. It is obvious that when $D_i = V_i$, the problem can be reduced to the k-center problem (Hochbaum, 1996) which is to select a set of k vertices so that the distance from the rest of the vertices to these selected k vertices is minimized. While the k-center problem is NP-hard, there exist efficient greedy approximation algorithms, e.g., by selecting a vertex that is furthest away from the established set (Hochbaum, 1996). However, such a heuristic might run into the problem of selecting vertices that are at the end of the long path and barely connect to other vertices. To mitigate this, we propose to weight the distance with the degree of the vertices. In addition, like other vertex sampling algorithms, we modify the above selection process by assigning a probability to each vertex based on the greedy criteria and its degree to give the algorithm some room for exploration.

E.3 Complexity Analysis

The time complexity of our algorithm is readily apparent from its procedural steps, and it exhibits a computational complexity of O(N), where N denotes the number of vertices in the graph. Notably, our approach shares a common



(a) SEA-ER, Coral-Full

(b) SEA-ER, OGB-Arixv

(c) SEA-ER, Reddit

Figure 5: Forgetting Performance Matrix of SEA-ER on CoraFull and Reddit Datasets.

characteristic with baseline methods, where the problem formulation might inherently be NP-hard to solve. However, both our method and the baselines leverage efficient heuristics to obtain practical solutions. As a result, the time complexity of our proposed methods and the baseline approaches remains similar, with all falling within the order of O(N). This underscores the computational efficiency and feasibility of our methodology in addressing the challenges posed by the problem at hand.

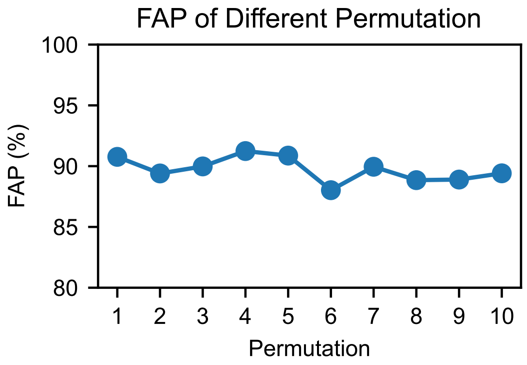


Figure 6: FAP performance of our method with respect to different permutations. X-axis indicate different permutation 1-10. The figure shows that different permutations have a small effect (1%) on the performance.

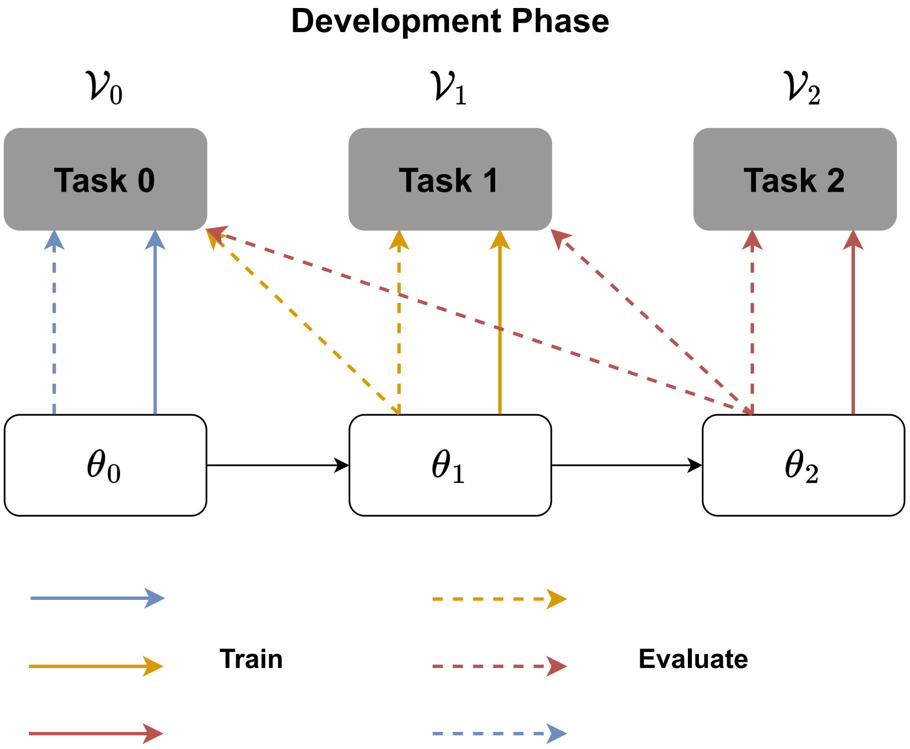


Figure 7: Continual Training in Development Phase. The figure above illustrates what task data the model use for training and evaluation.

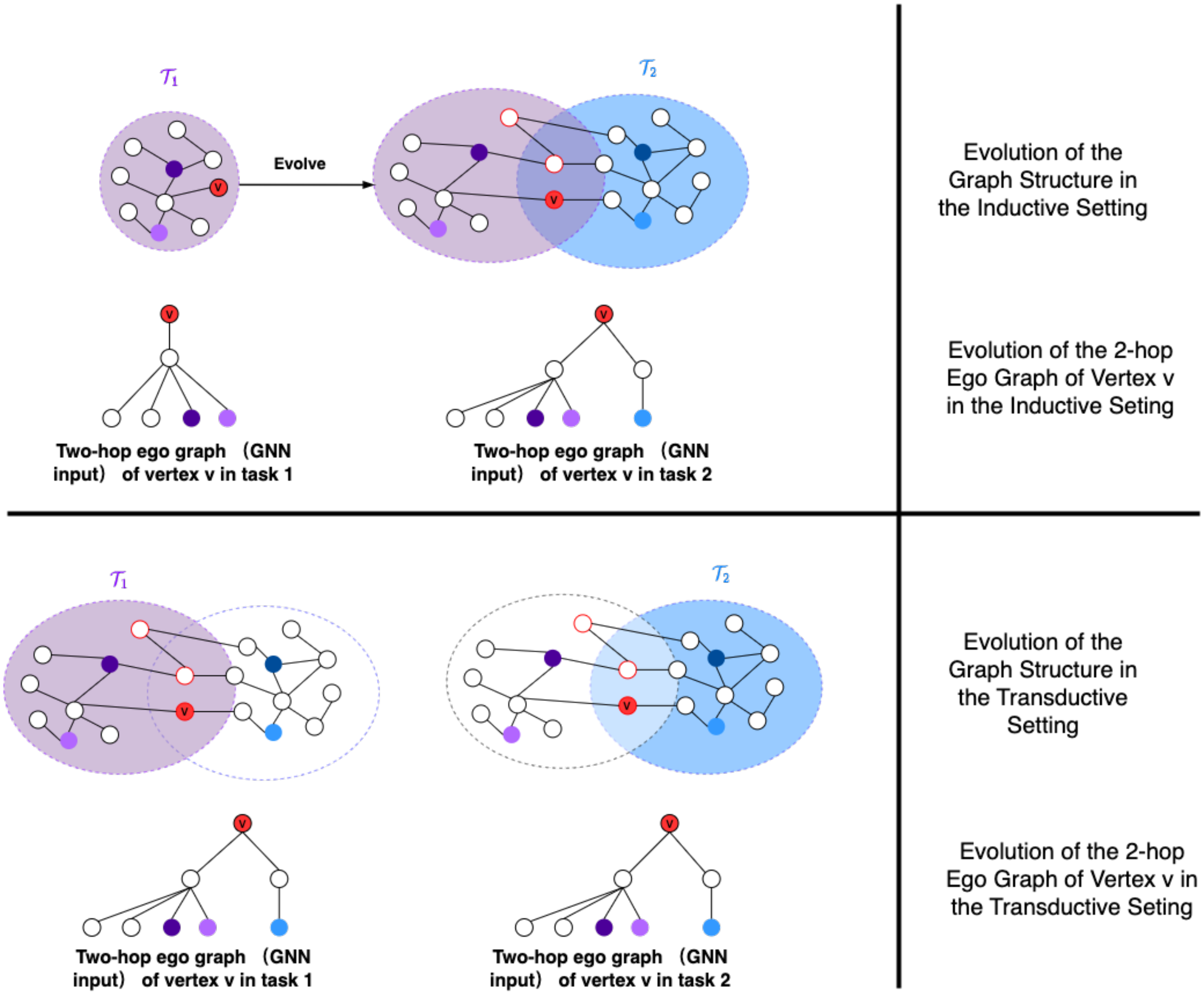

$\mathcal{T}_1$
Evolve
$\mathcal{T}_2$
v
Evolution of the Graph Structure in the Inductive Setting
Two-hop ego graph （GNN input） of vertex v in task 1
Two-hop ego graph （GNN input） of vertex v in task 2
Evolution of the 2-hop Ego Graph of Vertex v in the Inductive Seting
$\mathcal{T}_1$
$\mathcal{T}_2$
Evolution of the Graph Structure in the Transductive Setting
Two-hop ego graph （GNN input） of vertex v in task 1
Two-hop ego graph （GNN input） of vertex v in task 2
Evolution of the 2-hop Ego Graph of Vertex v in the Transductive Seting

Figure 8: Difference between Inductive (evolving graphs) GCL and Transductive (constant graphs) GCL. As illustrated in the top half of the figure, in the evolving graph setting, the appearing tasks expand/change the existing graph structure, resulting in changes in the neighborhood (input to the GNN) of some previous vertices. As discussed in the main body of the paper, this structural changes/dependency would cause distribution shift in the previously learnt information of the model. On the other hand, in the constant graph setting, the complete graph structure is available from the very first beginning. The neighborhoods of all the vertices are static/stay unchanged throughout different learning stages. Therefore, structural dependency is a unique challenge in the continual learning in evolving graphs.

Algorithm 2 SEA-ER-GNN

```
  Input: $\tau_i$ //new node classification task
Require: $\mathcal{P}_{i-1} = \{P_1, P_2, ..., P_{i-1}\}$ //replay buffer
Require: $g$ //current GNN model
Require: $\mathcal{G}_{\tau_{i-1}}$ //graph induced by vertices of $\tau_{i-1}$
Require: $\beta_l, \beta_u$
  Create the training set for this learning stage
      $\widehat{\mathcal{V}}_i = \mathcal{V}^{\text{train}} \cup \mathcal{P}_{i-1};$
  Compute $\beta$ with Eq. 4.3
  Run the training procedure (forward computation and backward propagation) with loss function
      $L = \frac{1}{|\mathcal{V}_i^{\text{train}}|} \sum_{v \in \mathcal{V}_i^{\text{train}}} L(v) + \sum_{P_j \in \mathcal{P}_i} \frac{1}{|P_j|} \sum_{v \in P_j} \beta_v L(v),$
  Update the replay buffer with replay set $P_i$ from $\tau_i$
      $\mathcal{P}_i = \mathcal{P}_{i-1} \cup P_i;$
```

Algorithm 3 Replay Set

```
  Input: $D_i \subset V_i$ //labelled set from task i ($\tau_i$)
Require: $\mathcal{G}_{i-1} = \mathcal{G}[\mathcal{V}_{i-1}]$ //existing graph
  Sample/select the replay set $P_i$ with the following procedures:
      1. initialize each vertice $v \in D_i$ with $p_v$ with its degree and sample an initial vertex $u$ based on a probability proportion to the degree (e.g., normalized each the degree of each vertex with the sum of total degree)
      2. assignment $u$ to the replay set $P_i = \{u\}$
      3. compute the shortest-path distance between each vertex $v \in D_i \setminus P_i$ to $P_i$ with the formula $d(v, P_i) = \min\{d(v, u) | u \in P_i\}$
      4. compute the weight for each vertex $v \in D_i \setminus P_i$ based on product of the distance to $P_i$ and its degree, i.e., $p_v = \text{degree}(v) * d(v, P_i)$
      5. sample $v$ from $D_i$ based on probability proportion to $p_v$, e.g., normalized it with the sum from the remaining vertices of $D_i$
      6. repeat step 2-5 until $|P_i| = k$
```

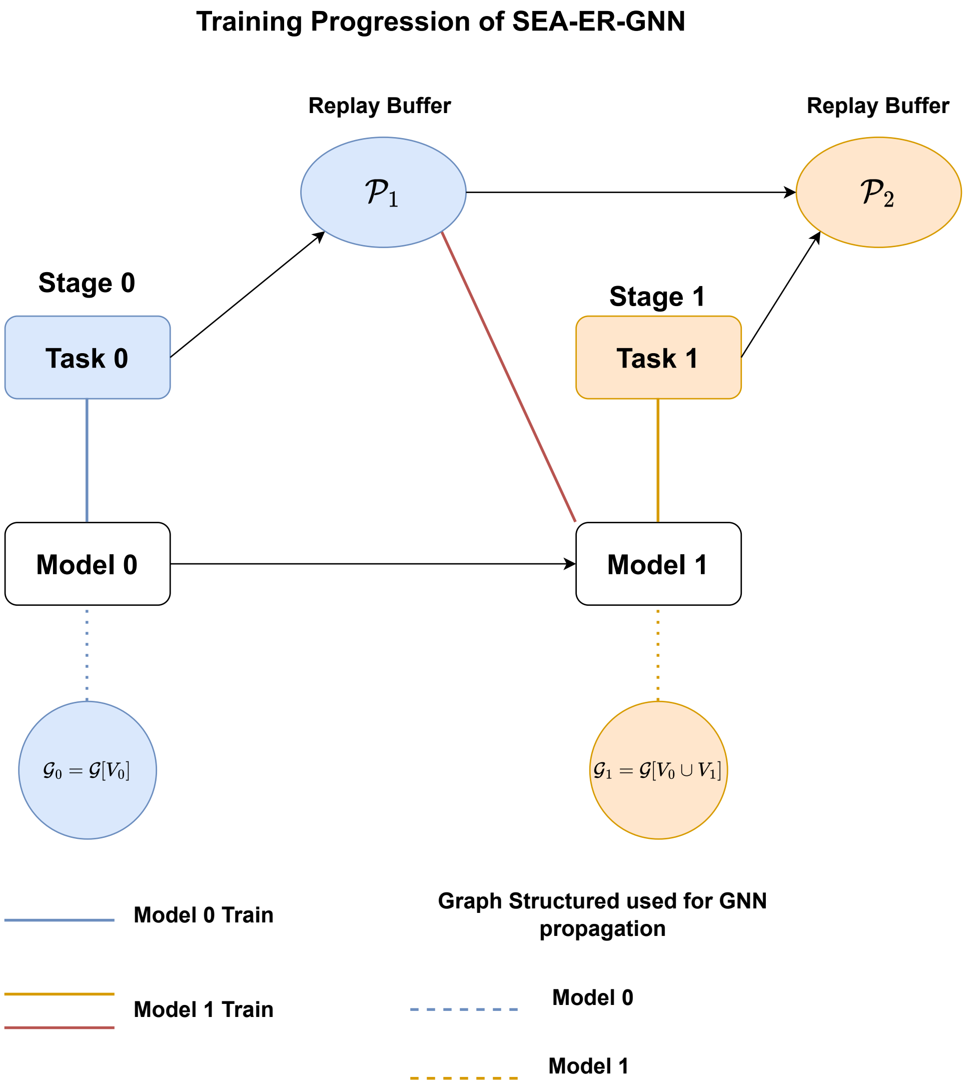

Training Progression of SEA-ER-GNN
Replay Buffer
$\mathcal{P}_1$
Replay Buffer
$\mathcal{P}_2$
Stage 0
Task 0
Stage 1
Task 1
Model 0
Model 1
$\mathcal{G}_0 = \mathcal{G}[V_0]$
$\mathcal{G}_1 = \mathcal{G}[V_0 \cup V_1]$
Model 0 Train
Model 1 Train
Graph Structured used for GNN propagation
Model 0
Model 1

Figure 9: Two-stage SEA-ER-GNN Training