DyGRAIN: An Incremental Learning Framework for Dynamic Graphs

Seoyoon Kim[1] , **Seongjun Yun**[2] and **Jaewoo Kang**[2*]
[1]LG AI Research
[2]Department of Computer Science and Engineering, Korea University
seoyoon.kim@lgresearch.ai, {ysj5419, kangj}@korea.ac.kr

Abstract

Graph-structured data provide a powerful representation of complex relations or interactions. Many variants of graph neural networks (GNNs) have emerged to learn graph-structured data where underlying graphs are static, although graphs in various real-world applications are dynamic (e.g., evolving structure). To consider the dynamic nature that a graph changes over time, the need for applying incremental learning (i.e., continual learning or lifelong learning) to the graph domain has been emphasized. However, unlike incremental learning on Euclidean data, graph-structured data contains dependency between the existing nodes and newly appeared nodes, resulting in the phenomenon that receptive fields of existing nodes vary by new inputs (e.g., nodes and edges). In this paper, we raise a crucial challenge of incremental learning for dynamic graphs as *time-varying receptive fields*, and propose a novel incremental learning framework, DyGRAIN, to mitigate *time-varying receptive fields* and *catastrophic forgetting*. Specifically, our proposed method incrementally learns dynamic graph representations by reflecting the influential change in receptive fields of existing nodes and maintaining previous knowledge of informative nodes prone to be forgotten. Our experiments on large-scale graph datasets demonstrate that our proposed method improves the performance by effectively capturing pivotal nodes and preventing catastrophic forgetting.

1 Introduction

Over the last few years, graph-structured data has been widely used in diverse domains, including social network analysis [Qiu *et al.*, 2018], recommender systems [Berg *et al.*, 2017], computer vision [Qi *et al.*, 2017], and bioinformatics [De Cao and Kipf, 2018]. To deal with graph-structured data, various graph neural networks (GNNs) have been presented. GNNs learn powerful representations from non-Euclidean and high-dimensional graph information, resulting in state-of-the-art performance in many applications. While most of GNNs proposed so far assume that a given graph is static, many graph-structured data in the real world are *dynamic graphs* where nodes and edges vary over time as it evolves. A general approach for learning dynamic graphs is to retrain the full graph at every time step, requiring numerous amounts of cost. However, incremental learning is efficient and effective to learn representations of a dynamic graph. Recently, a few incremental learning methods for graph-structured data have emerged, but they investigate a static setting where aims to improve class learning (e.g., class-incremental setting) and focus to overcome only the catastrophic forgetting problem [Zhou and Cao, 2021; Liu *et al.*, 2021]. Contrary to static graphs, incremental learning for dynamic graphs suffers from not only catastrophic forgetting but *time-varying receptive fields* of existing data. Since incremental inputs (e.g., nodes and edges) in dynamic graphs have dependencies with existing data that changes in receptive fields, learning a dynamic graph with an incremental approach should consider the influences propagated through dependencies with existing nodes. In the message-passing mechanism, the receptive field of a node is determined with L-hop neighborhood nodes. If the node receives new nodes and edges in the neighborhood, its receptive field varies by new incoming influences propagated from the input through dependencies connected to. This is an important concern since time-varying receptive fields can cause a prediction change that may shift from correct labels to incorrect labels even with the same parameters. In addition, incremental learning has catastrophic forgetting problem that the learner model forgets the learned knowledge from the past while learning new information currently. Therefore, incremental learning for dynamic graphs confronts two important challenges which are time-varying receptive fields and catastrophic forgetting.

To address those challenges, we propose **Dy**namic **Gra**ph **In**cremental learning (DyGRAIN), a framework to train a dynamic graph, which captures the influential change in receptive fields over time and distills previously learned knowledge through the following components: **(i)** *Influence Propagation* identifies key existing nodes that are most influenced by new inputs in view of structure and feature. Our insight is that new dependencies can be captured by propagation from new inputs to existing data. Influence Propagation mit-

[*]Corresponding Author

igates time-varying receptive fields. **(ii)** *Knowledge Distillation with Truth*(KDwTruth) selects existing nodes that are prone to forget previously learned knowledge by estimating loss discrepancy. It distills previously learned knowledge of selected nodes by leveraging ground truth labels. KDwTruth mitigates catastrophic forgetting. Our experiments show that our method outperforms state-of-the-art incremental learning techniques and graph incremental learning methods. Our framework is straightforward yet effective in capturing nodes influenced by new inputs and reducing catastrophic forgetting in a dynamic graph. Our contributions are as follows:

- We raise *Time-varying receptive fields*, a new problem occurring in dynamic graph incremental learning, where receptive fields of existing nodes vary due to dependencies with new inputs. This problem has a risk to the prediction shifting of existing nodes.
- We propose a novel and straightforward DyGRAIN that effectively captures crucial dependencies on a graph through both structure- and feature-based approaches and reduces catastrophic forgetting through more precise Knowledge Distillation of vulnerable nodes.
- We demonstrate that our method is effective to learn representations of dynamic graphs incrementally and experimentally verify on dynamic graphs processed from benchmark datasets including Open Graph Benchmark (OGB), where it consistently improves performance on both a given task and catastrophic forgetting.

2 Related Work

Incremental learning (i.e., continual learning or lifelong learning) aims to acquire increasing knowledge and extend it to future learning. However, incremental learning algorithms encounter catastrophic forgetting that the base learner model gradually loses the acquired knowledge when arriving data instances have different patterns or distribution shift from the previous time. Several studies on incremental learning overcoming catastrophic forgetting have been introduced.

Replay-based methods retrain sample data stored in memory [Chaudhry *et al.*, 2019b] or pseudo-samples from the distribution of hidden representation [Shin *et al.*, 2017] to relieve catastrophic forgetting. Alternatively, random sampling from a memory is utilized for estimating the gradient direction to constrain the optimization on new task from previous tasks [Chaudhry *et al.*, 2019a]. These replay-based approaches show promising results on consistent fair scenarios. Another strategy to tackle catastrophic forgetting is an additional regularization term that consolidates previous parameters. There are regularization terms designed to distill the knowledge from a previous model to a current training model such as Learning Without Forgetting (LWF) [Li and Hoiem, 2017]. On the other hand, several regularization terms penalize important parameters that are estimated from a distribution over the model parameters. For example, Elastic Weight Consolidation (EWC) [Kirkpatrick *et al.*, 2017] penalizes the parameter space to include important parameters from previous tasks. [Aljundi *et al.*, 2018] proposes the regularization that can penalize parameters by estimating unsupervised importance, while [Zenke *et al.*, 2017] estimates parameter importance in an online manner.

While conventional incremental learning methods for grid data are abundant, few approaches for incremental learning on graphs have been presented. GraphSAIL [Xu *et al.*, 2020] proposed the online learning framework for dynamic user preferences, but it is limited to recommender systems and catastrophic forgetting. TWP [Liu *et al.*, 2021] introduced the regularization-based incremental learning method that preserves important parameters from previous tasks and topological attention between adjacent nodes. ER-GNN [Zhou and Cao, 2021] proposed incremental graph learning using the replay approach. It selects nodes based on intra-class features, embedding distance, and influence function and retrains the selected nodes while learning a new task.

3 Methodology

The goal of our framework is to effectively learn incremental inputs of graph topology with dynamic nature. Since graph-structured data have dependency between data instances (i.e., nodes), incremental learning on dynamic graphs has major challenges: existing nodes influenced by new inputs (i.e., time-varying receptive fields) and catastrophic forgetting. To address these challenges, we propose an incremental learning framework for dynamic graphs, **Dy**namic **Gra**ph **In**cremental learning (DyGRAIN) which learns useful representations from pivotal nodes among existing nodes. In this section, we begin with the basic concepts of dynamic graphs and demonstrate the challenges of incremental learning on dynamic graphs, and then introduce the components of our framework DyGRAIN: Influence Propagation and KD with Truth.

3.1 Notation

Consider a dynamic graph $\mathcal{G} = \{G^{(1)}, G^{(2)}, \cdots, G^{(T)}\}$ that $G^{(t)} = (\mathcal{V}^{(t)}, \mathcal{E}^{(t)})$ denotes the graph at time step t. $\mathcal{V}^{(t)} = \{v_1, v_2, \cdots, v_N\}$ represents a set of nodes interacted during a time step t and $\mathcal{E}^{(t)} = \{e_{ij} | v_i, v_j \in \mathcal{V}^{(t)}\}$ represents a set of edges where nodes $v_i, v_j \in \mathcal{V}^{(t)}$ are connected at a time step t. Both $\mathcal{V}^{(t)}$ and $\mathcal{E}^{(t)}$ change across time, and the cardinality in each time step may differ from the another. Likewise, $\mathcal{A} = \{\mathbf{A}^{(1)}, \mathbf{A}^{(2)}, \cdots, \mathbf{A}^{(T)}\}$ denotes the adjacency matrices where $\mathbf{A}^{(t)} \in \mathbb{R}^{N^{(t)} \times N^{(t)}}$ is defined by $\mathbf{A}^{(t)}_{ij} = 1$ if $e_{ij} \in \mathcal{E}^{(t)}$. $\mathcal{X} = \{\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \cdots, \mathbf{X}^{(T)}\}$ represents a sequence of node features where $\mathbf{X}^{(t)} \in \mathbb{R}^{N^{(t)} \times F}$ is defined as a set of feature vectors of each node $x^{(t)}_i \in \mathbb{R}^{1 \times F}$. As an input of graph neural networks, input data consist of an adjacency matrix and features, i.e., $\mathcal{D}^{(t)} = (\mathbf{A}^{(t)}, \mathbf{X}^{(t)})$.

Time-varying Receptive Fields

As new nodes and edges arrive incrementally on dynamic graphs, incremental learning (i.e., continual learning) that mainly trains new arriving inputs is essential to learn representations of dynamic graphs with efficiency. Unlike image domain or natural language domain, graph-structured data has a dependency with incremental inputs that leads to the unique challenge. Receptive fields of existing nodes can be

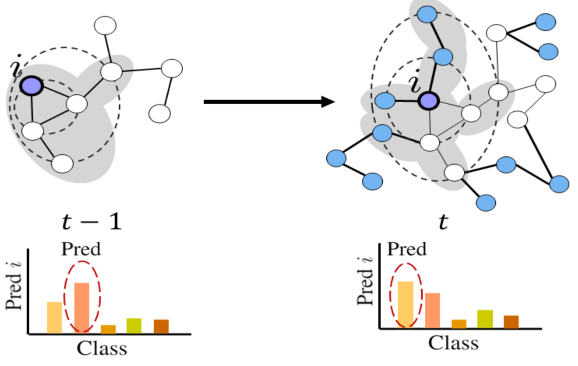


Figure 1: Blue nodes and corresponding edges indicate the new input, the purple node is the target node i, and gray-scale indicates its receptive field. New nodes and edges may change a topology of existing nodes, ultimately influencing on receptive fields of existing nodes which can lead to different predictions. We termed this phenomenon as *time-varying receptive fields*.

changed when new nodes $\mathcal{V}^{(t)}$ and edges $\mathcal{E}^{(t)}$ appear because of the dependency. Specifically, $\mathcal{V}^{(t)}$ influences the existing nodes that are connected in L-hop neighborhood where L is the number of layers in a GNN learner, changing the receptive fields of those existing nodes from the previous time step. We define this specific phenomenon as *time-varying receptive fields* and corresponding nodes as *influence-receiving nodes*.

3.2 Our Method

We propose DyGRAIN, a novel incremental learning framework for dynamic graphs that is designed to be aware of dependencies between new nodes and existing nodes while alleviating catastrophic forgetting. Specifically, our framework identifies the nodes most influenced new inputs, which we termed influence-receiving nodes, and vulnerable nodes that carry the previous knowledge but are prone to be forgotten. We employed the memory buffer $\mathcal{M}$ which applies *reservoir sampling* [Vitter, 1985] for previous nodes and then trains the retrieved nodes from the memory buffer $\mathcal{M}$ for parameter update. Our framework retrieves nodes in two approaches to mitigate time-varying receptive fields and catastrophic forgetting.

Mitigating Time-varying Receptive Fields

To mitigate time-varying receptive fields, it should identify the existing nodes that are influenced by dependencies between new input $\mathcal{D}^{(t)}$. Thus, we propose *Influence Propagation* that is designed to measure the influence of the new input graph $G^{(t)}$ on the existing graph $G^{(1:t-1)}$ based on two perspectives: structure-based influence and feature-based influence. In other words, Influence Propagation is capable of estimating the extent of receptive field change upon a graph structure and a feature space. First, in the graph structure-based perspective, we assess the influence on the existing topology based on the distance from new inputs, i.e., how much new information propagates upon the graph structure. We first define Propagation matrix $\mathbf{P}$ and formulate $\mathbf{V}$, a collection of the binarized values indicating whether node v_i is in $\mathcal{V}^{(t)}$. $\mathbf{V}$ consists of nodes from the memory $\mathcal{M}$ and $\mathcal{V}^{(t)}$. Propagation matrix $\mathbf{P}$ is defined as

$$\mathbf{P}^{(t)} = \mathbf{A}^{(t)}\mathbf{D}^{-1} \tag{1}$$

$$\mathbf{V}_{ii}^{(0)} = \begin{cases} 1, & \text{if } v_i \in \mathcal{V}^{(t)} \\ 0, & \text{if } v_i \in \mathcal{M} \end{cases} \tag{2}$$

$$\mathbf{V}^{(l)} = \mathbf{P}^{(t)}\mathbf{V}^{(l-1)} \tag{3}$$

where $\mathbf{A}^{(t)} \in \mathbb{R}^{N^{(t)}\times N^{(t)}}$ is adjacency matrix at time step t, $\mathbf{D} \in \mathbb{R}^{N^{(t)}\times N^{(t)}}$ is the degree matrix of $\mathbf{A}^{(t)}$ and each node v_i is from either $\mathcal{V}^{(t)}$ or memory $\mathcal{M}$. We obtain the matrix $\mathbf{V}^{(l)} \in \mathbb{R}^{N^{(t)}+N^{(m)}\times N^{(t)}+N^{(m)}}$ at the level of each l-th hop neighborhood when $N^{(m)}$ is the number of nodes from $\mathcal{M}$. $\mathbf{V}^{(l)}$ indicate the amount of influence propagated at l-th hop neighborhood.

Then we define Structural Influence score π_{SI} to measure influence propagated to each node. Given influence matrices $\mathbf{V}^{(1)}, \cdots, \mathbf{V}^{(L)}$ according to the depth of a model L, Structural Influence score π_{SI} is defined as

$$\pi_{SI}(v_i) = \sum_{j=1}^{N^{(t)}+N^{(m)}} \sum_{l=1}^{L} \mathbf{V}_{ij}^{(l)}. \tag{4}$$

If $\pi_{SI}(v_i)$ equals to 0, node v_i is out of L-hop neighborhood of $\mathcal{V}^{(t)}$ or an input node itself. We select the nodes based on K highest scores, i.e., $\mathcal{S}_{struct} = \arg\operatorname{topk}_i \pi_{SI}(v_i)$ where k is determined by sample size B and score ratio r, i.e., $k = B \times r$.

Secondly, in terms of node features $\mathcal{X}$, our approach captures the influence of new input $G^{(t)} = (\mathcal{V}^{(t)}, \mathcal{E}^{(t)})$. Our method calculates how much the receptive field of existing nodes has changed compared to the previous time step. We use embedding vectors from a GNN learner since the receptive fields of nodes are projected into the embedding space. Thus, we estimate influence in feature space with the embedding vector of a node v_j

$$\mathbf{h}_j^{(t)} = h_\theta(\mathbf{x}_j, \mathbf{A}^{(t)}; \theta^{(t-1)}), \tag{5}$$

where h_θ indicates an output layer of arbitrary GNN learner model with L layers (i.e., $f_\theta(x) = \operatorname{softmax}(h_\theta(x))$). $\mathbf{h}_j$ represents the smoothed feature vector of $v_j \in \mathcal{N}(i)$ which indicates neighborhood nodes up to L-hop of new input nodes $u_i \in \mathcal{V}^{(t)}$. $\mathcal{N}(i)$ can be obtained from Structural Influence score π_{SI}, i.e., $\pi_{SI}(v_j) > 0$. $\mathbf{A}^{(t)}$ is the adjacency matrix of $G^{(t)}$. To observe the amount of change in the receptive field of nodes over time, we define Feature Influence score π_{FI} as

$$\pi_{FI}(v_j) = 1 - \frac{\mathbf{h}_j^{(t-1)} \cdot \mathbf{h}_j^{(t)}}{\|\mathbf{h}_j^{(t-1)}\| \cdot \|\mathbf{h}_j^{(t)}\|} \tag{6}$$

indicating a discrepancy of the receptive field by time, where $\mathbf{h}_j^{(t)}$ represents the embedding vector of a node v_j at time step t. We choose nodes with K-highest deviation of the receptive field, i.e., $\mathcal{S}_{feat} = \arg\operatorname{topk}_i \pi_{FI}(v_i)$. To learn the information of nodes influenced by a current input, we replay the nodes from the obtained subsets, i.e., $\mathcal{S}_{IP} = \mathcal{S}_{struct} \cup \mathcal{S}_{feat}$.

Then, a learner model trains the current input data $\mathcal{D}^{(t)}$ with the retrieved node set $\mathcal{S}_{IP}$ to consider dependency. That is, a subset of stored nodes is explicitly retrained, which can be called replay. Let $\mathbf{Y}^{\mathcal{S}}$ be labels of nodes in subset $\mathcal{S}$ and C be the number of classes, then the objective function to mitigate time-varying receptive fields is defined as

$$\mathcal{L}_{IP}^{(t)} = -\sum^{C} \mathbf{Y}^{\mathcal{S}_{IP}} \log f_\theta(\mathcal{D}^{\mathcal{S}_{IP}}). \tag{7}$$

Mitigating Catastrophic Forgetting

As widely known, incremental learning suffers from catastrophic forgetting problem that a learner model gradually forgets previously learned information while new knowledge is learned. The conventional incremental learning methods relieve catastrophic forgetting by adjusting stability-plasticity dilemma [Grossberg, 2012] through replay instances or consolidating parameter space. Graph-structured data can calibrate stability in a similar manner, however, it is neither feasible nor necessary to learn representations of the entire existing nodes. We define the method, *Knowledge Distillation with Truth (KDwTruth)*, to identify informative but vulnerable nodes and distill its knowledge with a guarantee of correctness that further retains previous knowledge.

Identifying informative previous nodes that are vulnerable to new inputs, our approach starts with the matrix $\mathbf{V}^{(l)}$ obtained from Eq.(3). Note that we discriminate the previous nodes that are out of l-hop neighborhood of new input nodes $\mathcal{V}^{(t)}$ based on the structural relevance:

$$\mathcal{V}_C = \{v_j | \sum_{j=1}^{N}\sum_{l=1}^{L} \mathbf{V}_{ij}^{(l)} = 0, \mathbf{V}_{ii} = 0\}. \tag{8}$$

Given node samples $\mathcal{V}_C$ and parameters from the previous time $\theta^{(t-1)}$, we can obtain loss values as

$$\mathcal{L}^{(t-1)} = \ell(\mathcal{V}_C, \tilde{\mathbf{A}}^{(t-1)}; f_{\theta^{(t-1)}}) \ , \tag{9}$$

where $\tilde{\mathbf{A}}^{(t-1)} = \mathbf{A}^{(t-1)} + \mathbf{I}$ is a normalized adjacency matrix. As a loss value implies which information should be stabilized, we select nodes from node samples $\mathcal{V}_C$ by lower q percentile based on Eq.(9), i.e., $\mathcal{V}' = \{v_j \sim \mathcal{V}_C | \mathcal{L}_{v_j}^{(t-1)} \le P_q(\mathcal{L}^{(t-1)})\}$. To retrieve which candidate nodes in $\mathcal{V}'$ are prone to be interfered by $\mathbf{A}^{(t)}$ on parameter space at current step t, the expected loss of $\mathcal{V}'$ is estimated as

$$\widehat{\mathcal{L}}^{(t)} \approx \ell(\mathcal{V}', \tilde{\mathbf{A}}^{t}; f_{\theta^{(t-1)}}). \tag{10}$$

As we aim to train nodes that provide learned knowledge but are prone to be interrupted, we estimate loss-based score π_{CF} and randomly sample nodes as follows:

$$\pi_{CF}(v_j) = \Delta\widehat{\mathcal{L}}_j = \widehat{\mathcal{L}}_j^{(t)} - \mathcal{L}_j^{(t-1)} \tag{11}$$

$$\mathcal{S}_{KD} = \{v_j \sim \arg\operatorname{topk}_j \pi_{CF}(v_j)\} \tag{12}$$

Score π_{CF} equals to estimated deviation of loss per node v_j from $\mathcal{V}'$. We obtain the set of candidate nodes by random sampling from the selected k-highest deviation nodes. The set $\mathcal{S}_{KD}$ contains nodes containing previously learned information but are vulnerable to new inputs at time step t. DyGRAIN retains the previously learned information by Knowledge Distillation [Hinton *et al.*, 2015]. In DyGRAIN framework, previous knowledge is distilled from the previous learner $f_{\theta^{(t-1)}}$ with hidden knowledge:

$$\mathbf{z}_j^{(t)} = \Big\|_{c=1}^{C} \frac{exp(h_{j,c}/\tau)}{\sum_c exp(h_j/\tau)} \ \text{ s.t. } \ h_j = h_\theta(\mathbf{x}_j, \tilde{\mathbf{A}}^{(t)}) \tag{13}$$

$$\arg\min_\theta \mathcal{L} + \gamma \mathbb{E}_{i\in\mathcal{S}_{KD}}[D_{KL}(\mathbf{z}_j^{(t)} \| \mathbf{z}_j^{(t-1)})], \tag{14}$$

where C is the number of classes, $||$ is the concatenation operator, c denotes the index for each class, τ is a temperature to soften probability, and γ is a hyperparameter.

Note that Knowledge Distillation cannot always guarantee hidden knowledge $\mathbf{z}_i$ is correct under some certain conditions such as sudden distribution shift. However, DyGRAIN alleviates such incorrectness by leveraging ground truth labels. Besides, Eq.(14) can be approximated to minimizing Euclidean distance of a given logit from time $t-1$ and t under mild assumption. Thus, the objective function mitigating catastrophic forgetting is defined as

$$\mathcal{L}_{KD}^{(t)} = -\sum^{C} \mathbf{Y}^{\mathcal{S}_{KD}} \log softmax(Z_{\mathcal{S}_{KD}}^{(t)}) + \gamma \cdot \sum_{i}^{\mathcal{S}_{KD}} \|z_j^{(t)} - z_j^{(t-1)}\|_2. \tag{15}$$

Total Loss

Above all, the GNN learner trains input data $\mathcal{D}^{(t)}$. Then, the objective function is obtained as follows:

$$\mathcal{L}^{(t)} = -\sum^{C} \mathbf{Y}^{(t)} \log f_\theta(\mathcal{D}^{(t)}). \tag{16}$$

At time step t, our total objective function that mitigates time-varying receptive fields and catastrophic forgetting is defined as

$$\mathcal{L}_{Total}^{(t)} = \mathcal{L}^{(t)} + \mathcal{L}_{IP}^{(t)} + \mathcal{L}_{KD}^{(t)}. \tag{17}$$

Training procedure of our DyGRAIN is described in Algorithm 1.

Algorithm 1 Training Procedure of DyGRAIN

Required: At time step $t > 1$: Learner parameter $\theta^{(t-1)}$, New input $\mathcal{D}^{(t)}$, Memory buffer $\mathcal{M}$, max epochs K
Output: Parameter $\theta^{(t)}$

```
 1: $\mathbf{V} \leftarrow \mathcal{V}^{(t)} \cup \text{SAMPLE}(\mathcal{M})$  by Eq.(1-3)
 2: for $k = 1$ to $K$ do
 3:     $\mathcal{S}_{IP} \leftarrow Top\text{-}\kappa(\pi_{SI}(v_i \in \mathbf{V})) \bigcup Top\text{-}\kappa(\pi_{FI}(v_i \in \mathbf{V}))$
 4:     $\mathcal{S}_{KD} \leftarrow \text{SAMPLE}(Top\text{-}\kappa(\pi_{CF}(\mathcal{V}')))$  by Eq.(8-11)
 5:     $\mathcal{L} = \ell(\mathcal{D}^{(t)}; \theta^{(t-1)})$
 6:     $\mathcal{L}_{IP} = \ell(\mathcal{S}_{IP}; \theta^{(t-1)})$
 7:     $\mathcal{L}_{KD} = \ell(\mathcal{S}_{KD}; \theta^{(t-1)}) + \gamma \cdot \Sigma\|Z^{(t)} - Z^{(t-1)}\|_2^2$
 8:     $\theta^{(t)} \leftarrow \arg\min \mathcal{L}_{Total} = \mathcal{L} + \mathcal{L}_{IP} + \mathcal{L}_{KD}$
 9: end for
10: $\mathcal{M} \leftarrow Reservoir(\mathcal{M}, \mathcal{D}^{(t)})$
```

Base	Method	OGB-ARXIV		OGB-PRODUCTS		REDDIT		PUBMED	
		Accuracy	Forgetting	Accuracy	Forgetting	Accuracy	Forgetting	Accuracy	Forgetting
GCN	Finetuninng	63.65 ± 0.43	8.50 ± 0.21	63.96 ± 0.21	5.72 ± 0.01	91.68 ± 0.34	4.65 ± 0.26	79.14 ± 0.68	4.47 ± 0.15
	EWC	66.23 ± 0.37	6.48 ± 0.24	69.05 ± 0.16	3.66 ± 0.04	92.95 ± 0.37	2.51 ± 0.09	80.87 ± 0.50	4.92 ± 0.14
	LWF	65.85 ± 0.68	6.55 ± 0.10	69.16 ± 0.06	3.65 ± 0.13	92.37 ± 0.27	2.57 ± 0.19	80.87 ± 1.44	4.46 ± 0.15
	ER	68.66 ± 0.26	7.12 ± 0.07	71.10 ± 0.13	4.89 ± 0.16	93.60 ± 0.05	3.97 ± 0.05	85.83 ± 0.21	4.17 ± 0.04
	ER-GNN	65.49 ± 0.21	7.33 ± 0.19	67.82 ± 0.24	4.71 ± 0.06	93.17 ± 0.47	3.68 ± 0.21	81.36 ± 0.32	4.18 ± 0.37
	TWP	65.66 ± 0.90	4.57 ± 0.07	68.33 ± 0.70	3.64 ± 0.09	93.13 ± 0.34	2.35 ± 0.08	80.56 ± 0.10	3.86 ± 0.09
	DyGRAIN(Ours)	**70.90** ± 0.26	**4.48** ± 0.12	**75.83** ± 0.09	**2.76** ± 0.06	**94.62** ± 0.12	**2.31** ± 0.03	**87.45** ± 0.45	**2.82** ± 0.28
	Full Graph	71.84 ± 0.20	N/A	76.33 ± 0.23	N/A	94.97 ± 0.05	N/A	87.50 ± 0.51	N/A
GAT	Finetuninng	64.25 ± 0.55	7.03 ± 0.14	64.27 ± 0.30	6.94 ± 0.23	88.59 ± 0.30	5.47 ± 0.10	74.54 ± 1.17	7.78 ± 1.19
	EWC	65.49 ± 0.38	6.15 ± 0.29	67.09 ± 0.27	5.19 ± 0.21	89.16 ± 0.37	4.89 ± 0.14	78.46 ± 0.34	6.44 ± 0.18
	LWF	65.54 ± 0.23	5.99 ± 0.29	67.31 ± 0.22	4.97 ± 0.14	89.29 ± 0.30	4.98 ± 0.19	78.31 ± 0.16	6.42 ± 0.74
	ER	68.73 ± 0.23	6.58 ± 0.18	70.62 ± 0.26	5.28 ± 0.06	91.44 ± 0.50	3.58 ± 0.15	82.09 ± 0.75	6.67 ± 0.45
	ER-GNN	65.22 ± 2.32	6.44 ± 0.38	68.66 ± 0.58	5.13 ± 0.03	89.30 ± 0.49	4.38 ± 0.12	74.20 ± 0.84	5.23 ± 0.06
	TWP	65.46 ± 0.72	4.98 ± 0.08	68.79 ± 0.41	4.03 ± 1.12	89.45 ± 0.38	**2.88** ± 0.41	73.97 ± 1.49	6.65 ± 0.63
	DyGRAIN(Ours)	**71.17** ± 0.15	**4.62** ± 0.09	**72.81** ± 0.42	**3.37** ± 0.30	**92.16** ± 0.40	3.02 ± 0.07	**85.86** ± 0.72	**3.30** ± 0.33
	Full Graph	71.45 ± 0.07	N/A	OOM	N/A	OOM	N/A	86.40 ± 0.65	N/A
APPNP	Finetuninng	64.88 ± 0.42	5.01 ± 0.11	66.11 ± 0.46	8.44 ± 0.04	87.09 ± 1.41	5.33 ± 0.12	79.72 ± 0.50	5.49 ± 0.07
	EWC	67.94 ± 0.23	4.72 ± 0.13	69.58 ± 0.13	4.70 ± 0.02	90.95 ± 0.21	4.15 ± 0.93	81.61 ± 0.24	3.25 ± 0.16
	LWF	67.82 ± 0.12	4.73 ± 0.12	68.85 ± 0.08	5.85 ± 0.20	91.98 ± 0.14	3.94 ± 0.06	81.58 ± 0.08	3.35 ± 0.20
	ER	69.93 ± 0.26	5.72 ± 0.14	70.62 ± 0.26	6.22 ± 0.06	93.31 ± 0.22	4.74 ± 0.14	86.79 ± 0.10	4.37 ± 0.09
	ER-GNN	67.34 ± 0.13	6.19 ± 0.03	66.85 ± 0.17	5.34 ± 0.09	92.11 ± 0.19	4.24 ± 0.10	82.45 ± 0.11	4.30 ± 0.14
	TWP	66.42 ± 0.83	**3.89** ± 0.04	66.67 ± 1.25	4.82 ± 0.15	92.39 ± 0.23	4.03 ± 0.24	80.47 ± 0.25	4.22 ± 0.15
	DyGRAIN(Ours)	**71.36** ± 0.18	4.02 ± 0.05	**75.53** ± 0.20	**4.50** ± 0.04	**94.38** ± 0.34	**3.64** ± 0.03	**87.99** ± 0.15	**2.97** ± 0.08
	Full Graph	71.47 ± 0.19	N/A	77.36 ± 0.10	N/A	94.74 ± 0.52	N/A	88.06 ± 0.37	N/A
GCN2	Finetuninng	62.65 ± 0.94	8.05 ± 0.20	66.85 ± 0.20	5.90 ± 0.27	87.75 ± 1.12	4.99 ± 0.13	80.29 ± 1.10	4.91 ± 0.36
	EWC	65.37 ± 0.30	6.86 ± 0.29	67.03 ± 0.54	4.11 ± 0.25	88.40 ± 0.46	3.95 ± 0.12	82.63 ± 0.55	3.61 ± 0.25
	LWF	65.53 ± 0.11	6.78 ± 0.41	67.32 ± 0.28	4.15 ± 0.33	88.28 ± 0.32	3.70 ± 0.40	82.92 ± 0.28	3.55 ± 0.33
	ER	67.73 ± 0.45	7.15 ± 0.34	69.48 ± 0.30	4.92 ± 1.62	89.51 ± 0.48	4.47 ± 0.05	88.04 ± 0.12	4.72 ± 0.24
	ER-GNN	62.70 ± 0.31	7.46 ± 0.07	67.72 ± 0.08	5.13 ± 0.23	87.86 ± 1.55	4.88 ± 0.21	83.32 ± 0.08	4.65 ± 0.23
	TWP	65.93 ± 1.18	5.66 ± 0.06	66.90 ± 0.74	**3.33** ± 0.05	87.84 ± 0.47	3.61 ± 0.14	80.99 ± 0.74	3.13 ± 0.23
	DyGRAIN(Ours)	**70.94** ± 0.23	**4.61** ± 0.00	**73.46** ± 0.23	**3.33** ± 0.12	**92.17** ± 0.53	**3.45** ± 0.11	**89.42** ± 0.19	**2.78** ± 0.20
	Full Graph	71.42 ± 0.29	N/A	OOM	N/A	95.43 ± 0.04	N/A	89.60 ± 0.71	N/A

Table 1: Node Classification performance of DyGRAIN and baselines on Open Graph Benchmark (OGB) datasets, Reddit, and PubMed. The higher Accuracy means better performance whereas the lower the better for Forgetting (OOM: Out of memory). For fair comparison, we reported Accuracy and Forgetting from 5 independent runs, where each seed is from 0 to 4.

4 Experiments

We evaluate our proposed method against a variety of state-of-the-art baselines on four node classification benchmark datasets. We verify our method on datasets simulated like the real world where a dynamic graph arrives in an incremental manner. In this section, we provide the experimental setting, quantative results, ablation studies, and analysis demonstrating the effectiveness of DyGRAIN.

4.1 Experimental Settings

Datasets. We evaluate the effectiveness of our DyGRAIN on the node classification task. We use Open Graph Benchmark (OGB) Arixv, Products [Hu *et al.*, 2020], Reddit [Hamilton *et al.*, 2017] and PubMed [Sen *et al.*, 2008]. To simulate the dynamic scenario, we split the dataset into several incremental blocks. The graph is processed by time information if available. We assume the inductive setting on dynamic graphs [Xu *et al.*, 2020]. Detailed statistics of each dataset are summarized in Appendix A.

Evaluation. We evaluate the performance in dynamic graph incremental learning with three metrics. Accuracy measures the performance on test data composed of nodes to be predicted and nodes predicted previously after learning the incremental block. Forgetting [Chaudhry *et al.*, 2019b] measures the degree of catastrophic forgetting. We modify the Forgetting metric for dynamic graphs, observing the averaged rate of incorrect nodes used to be correct in the previous time.

Baselines To demonstrate the effectiveness of our method in dynamic graphs, we compare DyGRAIN with three well-known incremental learning methods, two incremental learning methods for graph-structured data, fine-tuning which provides a lower bound, and retraining the full graph. We used three familiar incremental learning methods, Elastic Weight Consolidation (EWC) [Kirkpatrick *et al.*, 2017] constraining parameter adaptation with the previous parameter space, Learning without Forgetting (LWF) [Li and Hoiem, 2017] that utilizes the previous model output as soft labels for previous tasks, and Experience Replay (ER) [Rolnick *et al.*, 2019], directly retraining randomly sampled instances stored in the memory buffer. For graph-specified incremental learning, we used ER-GNN [Zhou and Cao, 2021] and Topological Weight Preserving (TWP) [Liu *et al.*, 2021]. Furthermore, we compare our method using four GNN-based models, GCN [Kipf and Welling, 2017], GAT [Veličković *et al.*, 2018], APPNP [Klicpera *et al.*, 2019], and GCN2 [Ming Chen *et al.*, 2020] as a base learner model. Implementation details are described in Appendix B.

4.2 Node Classification Task

Table 1 reports the results on incremental dynamic graph datasets created with official graph benchmarks. First, in terms of accuracy, we can observe our DyGRAIN consistently outperforming state-of-the-art incremental learning algorithms and graph incremental learning methods regardless of base learners. Especially, our method provides a significant improvement on OGB-Arxiv and OGB-Products. DyGRAIN shows 13.81% and 5.36% relative improvement in accuracy on OGB-Arxiv, compared to the lowest baseline and the second-best baseline, respectively. Even more, our method achieves a large relative improvement 7% in accu-

Dataset	DyGRAIN		DyGRAIN (w/o IP)		DyGRAIN (w/o LS)		DyGRAIN (w/o DR)	
	Accuracy	Forgetting	Accuracy	Forgetting	Accuracy	Forgetting	Accuracy	Forgetting
OGB-Arxiv	70.90 ± 0.26	4.48 ± 0.12	68.84 ± 0.36	5.04 ± 0.06	69.05 ± 0.63	7.07 ± 0.12	69.55 ± 0.46	6.28 ± 0.10
OGB-Products	75.83 ± 0.09	2.76 ± 0.06	71.07 ± 0.43	3.48 ± 0.17	73.07 ± 0.35	4.49 ± 0.21	74.59 ± 0.13	4.17 ± 0.14
Reddit	94.62 ± 0.12	2.31 ± 0.03	93.66 ± 0.20	2.53 ± 0.14	94.06 ± 0.37	3.29 ± 0.17	94.46 ± 0.20	2.81 ± 0.31
PubMed	87.45 ± 0.45	2.82 ± 0.28	84.87 ± 0.25	3.12 ± 0.31	85.91 ± 0.41	4.07 ± 0.34	86.83 ± 0.16	3.59 ± 0.10

Table 2: Ablation study analyzing the efficacy of components: Influence Propagation(IP), Loss-based Stabilization(LS), and Distillation Regularization(DR). We evaluate DyGRAIN on the OGB-Arxiv, OGB-Products, Reddit, and PubMed datasets with GCN base learner. Accuracy and Forgetting are computed with 5 independent runs with the fixed seed from 0 to 4.

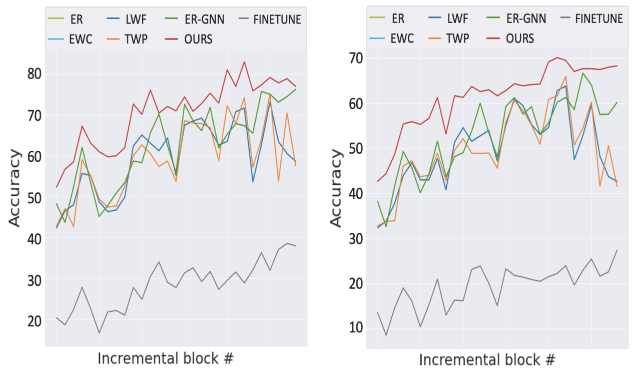
racy on OGB-Products compared to the second-best method, ER. Besides, DyGRAIN shows less than 1% discrepancy on retraining the full graph on most datasets, which indicates our proposed framework DyGRAIN is able to identify useful influence-receiving nodes which affected by incoming influence and properly reflect these nodes on learning incremental inputs. Also, our framework reaches the best or second-best performance in Forgetting evaluation. On the other hand, our DyGRAIN consistently shows state-of-the-art performance in both accuracy and Forgetting evaluation. This implies that addressing other problems (e.g., time-varying receptive fields) beyond catastrophic forgetting plays a key role in incremental learning on dynamic graphs. We further investigate the effectiveness of addressing time-varying receptive fields in the next section.

4.3 Analysis on Learning Influence-receiving Nodes

DyGRAIN is capable of learning existing nodes that have time-varying receptive fields which are critical in dynamic graph incremental learning. To demonstrate the ability to capture the time-varying receptive fields, we evaluate the performance of methods on the entire influence-receiving nodes at each time step. Figure 2 shows accuracy on influence-receiving nodes at each incremental block t. As shown in Figure 2, our method achieves the best performance at every incremental block on both OGB-Arxiv and OGB-Products. Besides, our method shows a smaller variance between incremental blocks compared to other baselines. In OGB-Arxiv dataset, we can notice that our method produces the best results with a less discrepancy between the accuracy of each incremental block. This implies that our method has the ability to capture the most influence-receiving nodes among the existing nodes.

4.4 Ablation Study

In this section, we empirically validate the efficacy of each component in DyGRAIN through ablation experiments. We evaluate average accuracy and average Forgetting with our framework without each component. To validate the efficacy of Knowledge Distillation with Truth, we split the component into Knowledge Distillation (KD) and ground truth label (GL). According to Table 2, our method constantly achieves better performances across all datasets. However, our method without each component (i.e., w/o IP, w/o KD, w/o GL) shows degraded results for accuracy, demonstrating the effectiveness of each component. Specifically, our method (w/o IP) shows the most decreased accuracy score. This implies



(a) OGB-Arxiv (b) OGB-Products

Figure 2: Analysis on nodes with time-varying receptive fields. We evaluate the ability to learn influence-receiving nodes by previous nodes that are highly influenced at every time step.

the time-varying receptive fields, namely, influence-receiving nodes are pivotal in incremental learning for dynamic graphs. Moreover, we can see that our method (w/o KD) and our method (w/o GL) reduce the accuracy but increase Forgetting. As seen in Table 2, Forgetting has clearly increased without Knowledge Distillation and Ground truth labels. This verifies both components contribute to preventing catastrophic forgetting.

5 Conclusion

In this paper, we specified the genuine problem of incremental learning for dynamic graphs that existing nodes influenced by new incremental inputs. Its receptive fields are changing over time, which we named time-varying receptive fields. To address time-varying receptive fields and catastrophic forgetting, we propose DyGRAIN, a novel incremental learning framework for dynamic graphs. Our framework shows promising results on task performance, in terms of capturing new incoming influences on existing nodes and mitigating catastrophic forgetting. In the future, we would further improve DyGRAIN to handle unlabelled data and other graph-related tasks (e.g., link prediction and graph classification).

Acknowledgements

This work was funded by National Research Foundation of Korea (NRF-2020R1A2C3010638), Ministry of Health Welfare, Republic of Korea (HR20C0021), and ICT Creative Consilience program(IITP-2021-0-01819).

References

[Aljundi *et al.*, 2018] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 139–154, 2018.

[Berg *et al.*, 2017] Rianne van den Berg, Thomas N Kipf, and Max Welling. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263*, 2017.

[Chaudhry *et al.*, 2019a] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. In *ICLR*, 2019.

[Chaudhry *et al.*, 2019b] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and M Ranzato. Continual learning with tiny episodic memories. *arXiv preprint arXiv:1902.10486*, 2019.

[De Cao and Kipf, 2018] Nicola De Cao and Thomas Kipf. Molgan: An implicit generative model for small molecular graphs. *arXiv preprint arXiv:1805.11973*, 2018.

[Grossberg, 2012] Stephen T Grossberg. *Studies of mind and brain: Neural principles of learning, perception, development, cognition, and motor control*, volume 70. Springer Science & Business Media, 2012.

[Hamilton *et al.*, 2017] 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*, 2017.

[Hinton *et al.*, 2015] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[Hu *et al.*, 2020] 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.

[Kipf and Welling, 2017] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

[Kirkpatrick *et al.*, 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*, 2017.

[Klicpera *et al.*, 2019] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *International Conference on Learning Representations (ICLR)*, 2019.

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

[Liu *et al.*, 2021] 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, 2021.

[Ming Chen *et al.*, 2020] Zhewei Wei Ming Chen, Bolin Ding Zengfeng Huang, and Yaliang Li. Simple and deep graph convolutional networks. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

[Qi *et al.*, 2017] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[Qiu *et al.*, 2018] Jiezhong Qiu, Jian Tang, Hao Ma, Yuxiao Dong, Kuansan Wang, and Jie Tang. Deepinf: Social influence prediction with deep learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2110–2119, 2018.

[Rolnick *et al.*, 2019] D. Rolnick, Arun Ahuja, Jonathan Schwarz, T. Lillicrap, and Greg Wayne. Experience replay for continual learning. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019.

[Sen *et al.*, 2008] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

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

[Veličković *et al.*, 2018] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *International Conference on Learning Representations*, 2018.

[Vitter, 1985] Jeffrey S Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, 11(1):37–57, 1985.

[Xu *et al.*, 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*, pages 2861–2868, 2020.

[Zenke *et al.*, 2017] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pages 3987–3995. PMLR, 2017.

[Zhou and Cao, 2021] 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, pages 4714–4722, 2021.