Pattern Recognition 159 (2025) 111093

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Continual learning with high-order experience replay for dynamic network embedding

Zhizheng Wang, Yuanyuan Sun *, Xiaokun Zhang, Bo Xu, Zhihao Yang, Hongfei Lin

Dalian University of Technology, Dalian, Liaoning, China

ARTICLE INFO

Keywords:
Continual learning
Dynamic pattern recognition
High-order experience replay
Graph auto-encoder

ABSTRACT

Dynamic network embedding (DNE) poses a tough challenge in graph representation learning, especially when confronted with the frequent updates of streaming data. Conventional DNEs primarily resort to parameter updating but perform inadequately on historical networks, resulting in the problem of catastrophic forgetting. To tackle such issues, recent advancements in graph neural networks (GNNs) have explored matrix factorization techniques. However, these approaches encounter difficulties in preserving the global patterns of incremental data. In this paper, we propose CLDNE, a Continual Learning framework specifically designed for Dynamic Network Embedding. At the core of CLDNE lies a streaming graph auto-encoder that effectively captures both global and local patterns of the input graph. To further overcome catastrophic forgetting, CLDNE is equipped with an experience replay buffer and a knowledge distillation module, which preserve high-order historical topology and static historical patterns. We conduct experiments on four dynamic networks using link prediction and node classification tasks to evaluate the effectiveness of CLDNE. The outcomes demonstrate that CLDNE successfully mitigates the catastrophic forgetting problem and reduces training time by 80% without a significant loss in learning new patterns.

1. Introduction

Dynamic networks have become increasingly prevalent across various domains, such as fraud detection [1] and recommendation systems [2,3]. In response, dynamic network embedding (DNE) [4,5] has emerged as a crucial technique for capturing the evolutionary patterns inherent in these networks. The primary objective of DNE is to develop a streaming function $\mathcal{F}$: $v \rightarrow \mathbb{R}^d$ that encapsulates the evolutionary characteristics of each vertex v within its d-dimensional vectors. However, dynamic networks naturally evolve by continuously adding new vertices and links in real-world scenarios. This poses a significant challenge for existing DNE models, which often assume that all vertices are known beforehand and struggle to simultaneously learn global new patterns and preserve previously acquired patterns for the dynamic networks.

Catastrophic forgetting [6] represents a fundamental challenge in traditional approaches to DNE. As illustrated in Fig. 1(a), prevalent methods [7,8] utilize a parameter updating strategy designed to assimilate new patterns from the incremental data. Initially, a new DNE model ($\mathcal{F}_{ini}$) is instantiated using the baseline parameters from the foundational model ($\mathcal{F}_{base}$). Subsequently, $\mathcal{F}_{ini}$ is adapted to conform to the topological structure of the evolving graph snapshot. While this strategy is straightforward, it often leads to a marked decline in performance when revisiting historical graph data, which is known as catastrophic forgetting. This issue arises because, as new data continuously streams into the training pipeline, the updated DNE model ($\mathcal{F}$) gradually diverges from the static features originally encoded in the base model. Thus, addressing catastrophic forgetting to retain the previously acquired patterns in the updated DNE model is a critical challenge in the development of robust DNE systems.

Inadequate learning of global new patterns remains another challenge for traditional DNE models. As depicted in Fig. 1(b), a typical approach to mitigate catastrophic forgetting involves retraining a DNE model from scratch by incorporating all available graph snapshots at each timestamp. This solution essentially treats the graph streaming process as a static network, which will become impractical for networks that are frequently updated, as the training complexity and data volume grow exponentially over time. Consequently, many graph neural network (GNN) models [9,10] attempt to employ matrix factorization techniques to learn new patterns from the incremental data. As illustrated in Fig. 1(c), these GNNs aim to integrate newly emerging vertices (e.g., vertices a, b, d, and e) into historical graph snapshots and update the base model $\mathcal{F}_{base}$ through the application of low-rank decomposition

* Corresponding author.
E-mail addresses: wzz0727@gmail.com (Z. Wang), syuan@dlut.edu.cn (Y. Sun), kun@mail.dlut.edu.cn (X. Zhang), xubo@dlut.edu.cn (B. Xu), yangzh@dlut.edu.cn (Z. Yang), hflin@dlut.edu.cn (H. Lin).

https://doi.org/10.1016/j.patcog.2024.111093
Received 24 August 2023; Received in revised form 22 July 2024; Accepted 20 October 2024
Available online 26 October 2024

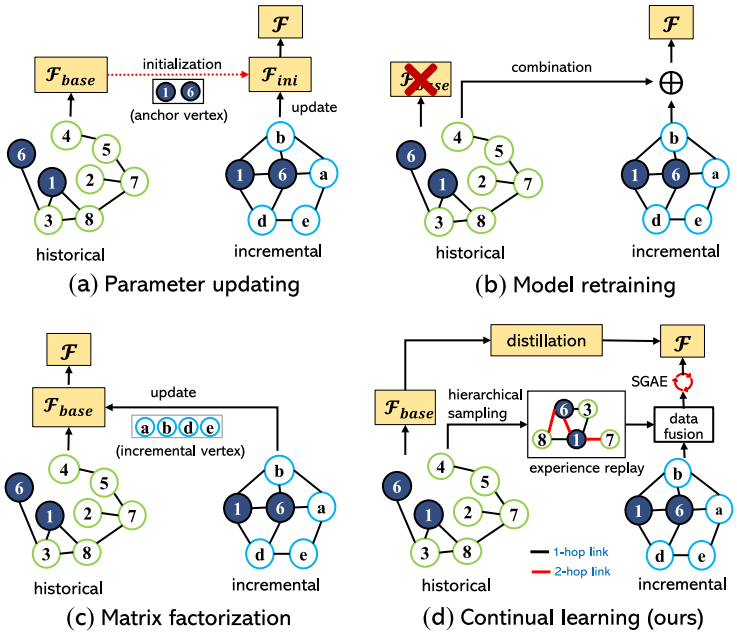

Fig. 1. Examples of different training strategies for dynamic network embedding. All examples are displayed on two adjacency graph snapshots, i.e., "historical" and "incremental".

on the adjacency matrix. Nevertheless, this method inherently segregates new vertices from the incremental graph structure. Consequently, a significant drawback is its tendency to disregard the global features of the incoming data, thereby leading to a partial learning of the emerging global patterns.

Recent advances in continual learning (CL) have positioned it as a viable approach for simultaneously acquiring global new patterns and preserving historical knowledge [11,12]. Within the domain of DNE, notable contributions leveraging CL include streaming GNNs [13,14], of which the objective is to update the key parameters for the specific downstream tasks of DNE under a supervised framework. These GNNs primarily employ strategies such as sampling immediate neighbours of the anchor vertex from historical graph snapshots, focusing on the local topological features of the historical data. In contrast, our study introduces a CL-based off-task DNE model, illustrated in Fig. 1(d), aimed at obtaining generalized representations for dynamic networks. Our model is designed to operate in an unsupervised fashion, using the auto-encoder to obtain the objective function $\mathcal{F}$ engineered to uncover the global evolutionary patterns of new vertices within dynamic streams. Furthermore, we integrate a mechanism to replay high-order topological structures from historical data and preserve the static features encoded in the base model $\mathcal{F}_{base}$, thereby mitigating the effects of catastrophic forgetting and presenting a robust solution for the downstream tasks of DNE such as node classification and link prediction.

Specifically, we present CLDNE, an innovative method composed of three essential components: *Streaming Graph Auto-Encoder* (SGAE), *Experience Replay Buffer*, and *Knowledge Distillation Module*. Central to CLDNE, SGAE plays a pivotal role in obtaining up-to-date vertex embeddings by reconstructing the adjacency matrix of the input graph. It incorporates multiple loss components to capture global patterns of the incremental graph. Notably, the input to SGAE is a synthesized graph, merging the incremental graph snapshot with the replayed historical data. The replayed data is meticulously curated through the experience replay buffer, which employs hierarchical sampling to select distant neighbours of anchor vertices from historical graph snapshots. Therefore, this buffer can not only provide high-order topological features but also enhance the training efficiency by minimizing the data required for training. The knowledge distillation module is a response to preserving static historical features during the training of CLDNE on the synthesized graph. It employs the mean-squared error (MSE) function to transfer the static features from the base model (i.e., teacher model) to the new model (i.e., student model), which ensures that the updated streaming function can continuously keep track of the static historical features. Overall, several noteworthy contributions are summarized as follows:

- We scrutinize the continual learning problem in the task of dynamic network embedding and practically develop an off-task unsupervised framework, CLDNE to derive generalized representations for dynamic networks.
- To adapt to the unsupervised scenarios in the dynamic streaming, we propose a streaming graph auto-encoder in CLDNE to simultaneously learn global and local patterns from the synthesized input graph.
- To address the catastrophic forgetting issue, we configure an experience replay buffer in front of SGAE to provide the high-order topology of historical data. Additionally, we highlight static historical features using a knowledge distillation module.
- We conduct extensive empirical experiments on four dynamic networks using the node classification and link prediction tasks, whose results demonstrate that CLDNE saves 80% training time and provides a 5.9% consolidation for historical knowledge in the best situation.

2. Related work

We summarize some related research works about DNE and CL as follows.

DNE. Generally, current DNE methods can be roughly summarized as dynamic matrix factorization (DMF), temporal point process (TPP), and graph neural network (GNN). The DMF methods such as DynGEM [7], GlodyNE [15], and HoMo-DyHNE [16] first construct the adjacency matrices among different network snapshots based on the random walk, and then impose matrix factorization on the adjacency matrix to maintain the freshness of the end embedding results. The TPP methods such as MMDNE [17], MTNE [18], and DyGCL [19] aim to capture streaming features by modelling the temporal process during a dynamic streaming of graph. They generally assume that given the historical links, a new edge appears in a bounded temporal window

(t) with the conditional intensity probability (λ). The GNN models such as DySAT [8], Dy-SIGN [20], and Ada-DyGNN [21] attempt to design the dynamic information aggregation mechanism to fathom new patterns from the data streaming by optimizing their loss functions in a supervised manner. The DGSCL [22] employs semi-supervised contrastive learning for dynamic network representation, which is to fully utilize the correlative information within the data streaming.

CL. It is presented to tackle the stability-plasticity dilemma in machine learning, i.e., preserving old knowledge while learning new knowledge [11]. Recently, CL has achieved promising performance in recommendation [23,24], computer vision [25,26] and graph data mining [27,28]. For instance, PackNet [29] leverages network pruning to exploit redundancies and free up parameters. GraphSAIL [23] implements a graph-structure preservation strategy to keep each vertex's local structure, global structure, and self-information. TWP [13] overcomes catastrophic forgetting in GNNs by exploring local structures of the input graph and stabilizing the parameters playing pivotal roles in the topological aggregation. ContinualGNN [30] employs GNN with experience replay to incrementally train node representations at each time step, while SGNN-GR [28] designs a generative replay module to make GNN maintain existing knowledge without accessing historical data. ROLAND [14] tries to find a meta-model that serves as a good initialization to derive a new model quickly. After that, the new model continuously updates the node embeddings in different GNN layers. FeCAM [31] aims to overcome the challenges in exemplar-free class-incremental learning by proposing a prototypical network to generate new class prototypes using the frozen feature extractor. IncDE [32] is proposed to train a continual learning model for knowledge graphs (KGs) by introducing the incremental distillation strategy to take full use of the explicit graph structure in KGs.

3. Preliminary knowledge

Definition I (*Dynamic Network*)**.** A dynamic network is formulated as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$, where $\mathcal{V}$ is the vertex set and $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ denote the link set. $\mathcal{A}$ is the adjacency matrix. A streaming splits $\mathcal{G}$ into $\{\mathcal{G}^0, \mathcal{G}^1, \dots, \mathcal{G}^t\}$. $\mathcal{G}^t = (\mathcal{V}_t, \mathcal{E}_t, \mathcal{A}_t)$ denotes the network snapshot at time t. $t = 0$ denotes the initial non-incremental (base) state. $t > 0$ denotes the incremental (streaming) state. For two adjacent network streaming $\mathcal{G}^{t-1}$ and $\mathcal{G}^t$, there will be a certain proportion of co-occurrence vertices, i.e., $\mathcal{V}_{t-1} \cap \mathcal{V}_t \neq \varnothing$.

Definition II (*Dynamic Network Embedding*)**.** Given a network streaming $\{\mathcal{G}^0, \mathcal{G}^1, \dots, \mathcal{G}^t\}$, the purpose of DNE is to learn a mapping function $\mathcal{F}$: $\mathcal{V}_t \to \mathbb{R}^{|\mathcal{V}_t| \times d}$ at time t, which embeds vertex $v \in \mathcal{V}_t$ into d-dimensional vectors. The function $\mathcal{F}$ relies on the network topology implied by $\mathcal{A}_t$. In practice, $\mathcal{F}$ is deduced from the vertex embedding matrix $|\mathcal{V}_t| \times d$ of free parameters.

Definition III (*Continual Learning in DNE*)**.** Given the non-incremental (base) state $\mathcal{G}^0$, the base mapping function $\mathcal{F}^0$: $\mathcal{V}_0 \to \mathbb{R}^{|\mathcal{V}_0| \times d}$ is trained from scratch on $\mathcal{G}^0$. As the incremental network state $\mathcal{G}^t$ streams into the previous network state $\mathcal{G}^{t-1}$, the objective of continual learning is to learn a new mapping function $\mathcal{F}^t$ by using the latest base model $\mathcal{F}^{t-1}$. It encodes vertices in $\{\mathcal{V}_0 \cup \dots \cup \mathcal{V}_{t-1} \cup \mathcal{V}_t\}$ into a new d-dimensional vector space. Notice that the new mapping function $\mathcal{F}^t$ should be learned on the dataset $\mathcal{G} = \mathcal{G}^t \cup \mathcal{Q}$, where $\mathcal{Q}$ denotes the bounded exemplar subset of data from the previous network states $\{\mathcal{G}^0, \dots, \mathcal{G}^{t-1}\}$.

4. Methodology

4.1. Overall framework

As shown in Fig. 2, CLDNE is a parallel framework that aims to continuously learn representations for dynamic networks in an unsupervised manner. It works to consolidate historical knowledge while learning new patterns on the incremental graph snapshot. The overall objective function of CLDNE at time t can be written as

$$\mathcal{O}(\mathcal{F}^t; \mathcal{G}^t) = \mathcal{O}_1(\mathcal{F}^t; (\mathcal{G}^t \cup \mathcal{G}_s^t)) + \mathcal{O}_2(\mathcal{F}^{t-1}; (\mathcal{G}^t \cup \mathcal{G}_s^t)) \tag{1}$$

where the first term (i.e., $\mathcal{O}_1$) is the new-pattern learning process, and the second term (i.e., $\mathcal{O}_2$) is the old-pattern consolidation process. $\mathcal{G}_s^t$ denotes the summary graph obtained through the *experience replay buffer* (Section 4.2) where we apply hierarchical sampling on historical graph snapshots. Note that $\mathcal{G}_s^t$ is a high-order distribution of the historical data, whose vertices are the subset of vertices in $\mathcal{G}^{t-1}$, i.e., $\mathcal{V}_s^t \subseteq \mathcal{V}^{t-1}$.

In the first stage, we propose a *streaming graph auto-encoder* (Section 4.3) to obtain up-to-date vertex representations. At each time step t, the encoder receives both new vertices from $\mathcal{G}^t$ and replayed vertices from $\mathcal{G}_s^t$, which are mixed into the generative graph (i.e., $\mathcal{G}^+$). The encoder also configures multiple loss components to preserve the global and local topological features of $\mathcal{G}^+$ by reconstructing the adjacency matrix.

In the second stage, we utilize a *knowledge distillation* module (Section 4.4) to help the streaming graph auto-encoder keep track of static historical features. It first expands the space of the base model to carry the newly emerging vertices in $\mathcal{G}^{t-1}$. Then, the static features deduced from the recent historical model (i.e., teacher model) are shifted to the new model (i.e., student model) through the mean-squared error (MSE) loss.

4.2. Experience replay

Distinct from streaming GNNs that replay the local topology of historical data, CLDNE aims to introduce high-order topological features from historical graph snapshots. Therefore, at each time step t, CLDNE constructs a *summary graph* ($\mathcal{G}_s^t$) by using the experience replay buffer, where the hierarchical sampling is performed on the graph snapshots before time t.

Formally, we denote the incremental graph snapshot at time t as $\mathcal{G}^t = (\mathcal{V}^t, \mathcal{E}^t, \mathcal{A}^t)$ and the historical graph snapshots from $max(t-\tau, 1)$ to $t-1$ as $\mathcal{G}_{his}^\tau = (\mathcal{V}_{his}^\tau, \mathcal{E}_{his}^\tau, \mathcal{A}_{his}^\tau)$. Take each anchor vertex that co-occurs in both $\mathcal{G}_{his}^\tau$ and $\mathcal{G}^t$ as the root of the random walk process, we sample its k-hop ($k \geq 1$) neighbourhoods from $\mathcal{G}_{his}^\tau$ to establish the high-order abstraction of historical graphs. Therefore, the process of constructing the summary graph can be defined as,

$$\begin{aligned} \mathcal{G}_s^t(\mathbb{V}^t, \mathbb{E}^t, \mathbb{A}^t) &= \bigcup_i^k (\mathbb{V}_i^t, \mathbb{E}_i^t, \mathbb{A}_i^t) \\ &= \bigcup_i^k S_{v \in \mathcal{V}^a}(v, \mathcal{G}_{his}^\tau, i, \Gamma) \end{aligned} \tag{2}$$

where $\mathcal{V}^a = (\mathcal{V}^t \cap \mathcal{V}_{his}^\tau)$ is the set of anchor vertices. The S denotes the random walk process. The $\Gamma \in (0, 1]$ means the ratio of sampled links to be adopted in the summary graph.

Eq. (2) summarizes the available historical data into several topological subgraphs based on the local evolution theory [33]. Each subgraph represents a kind of topological structure of historical graphs. When $k = 1$, the immediate neighbours and the first-order proximity are preserved. When $k \geq 2$, the long-range neighbours and the high-order proximity are maintained. Given that the long-range neighbours may have already been contained in the immediate neighbours, we conduct a link pruning operation before organizing different links into $\mathcal{G}_s^t$: *IF a link appears in multiple subgraphs at the same time, THEN we retain the one in the low-hop subgraph and discard the one in the high-hop subgraph*. In this way, except for providing high-order historical topology for CLDNE, the experience replay buffer also reduces the training data volume when learning new patterns.

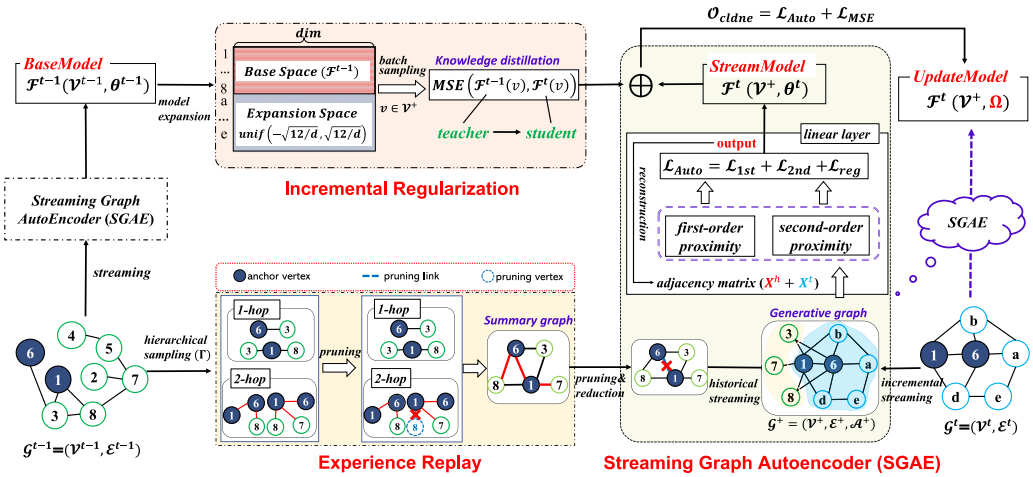

Fig. 2. An overview schematic of proposed CLDNE framework on two timesteps. At each time step t, an SGAE is trained on the generative graph; the experience replay buffer is used to provide the high-order topology of historical graph snapshots; and the knowledge distillation module distils the static historical knowledge from $\mathcal{F}^{t-1}$ to $\mathcal{F}^t$.

4.3. Streaming graph auto-encoder

Instead of following GNNs that employ the matrix factorization strategy, we resort to the matrix reconstruction strategy to preserve the global patterns of new data. Meanwhile, considering an original new graph snapshot in the dynamic streaming without label information, we propose a streaming graph auto-encoder (SGAE) to learn new patterns based on adjacency matrix signals.

Formally, define the combination of the summary graph $\mathcal{G}_s^t$ and the incremental graph snapshot $\mathcal{G}^t$ as the generative graph, i.e., $\mathcal{G}^+ = \mathcal{G}_s^t \bigtriangledown\!\!\!\bigtriangleup \mathcal{G}^t = (\mathcal{V}^+, \mathcal{E}^+, \mathcal{A}^+)$, SGAE aims to learn the new mapping function $\mathcal{F}^t$ for $\mathcal{G}^+$ based on $\mathcal{A}^+$, which projects $\mathcal{G}^+$ from $|\mathcal{V}^+| \times |\mathcal{V}^+|$ to $|\mathcal{V}^+| \times d$, and $d \ll |\mathcal{V}^+|$.

4.3.1. Input features of SGAE

Intuitively, the link sets $\mathcal{E}^+$ can be regarded as a bounded concatenation of $\mathcal{E}^t$ and $\mathbb{E}^t = \{\mathbb{E}_1^t, \mathbb{E}_2^t, \dots, \mathbb{E}_k^t\}$, but the adjacency matrix $\mathcal{A}^+$ cannot be obtained by applying linear addition to $\mathcal{A}^t$ and $\mathbb{A}^t$. This is because the replayed links in $\mathbb{E}^t$ are heterogeneous and may be covered by the links in $\mathcal{E}^t$. Therefore, we decompose the '$\bigtriangledown\!\!\!\bigtriangleup$' operation into two consecutive steps to obtain the input features of SGAE.

Step 1. We perform the pruning and reduction operation here to *homogenize the replayed links in $\mathbb{E}^t$ into the first-order links, and delete those links that have been generated in $\mathcal{E}^t$ from $\mathbb{E}^t$*. After that, the adjacency matrix $\mathcal{A}^+$ of $\mathcal{G}^+$ can be denoted as the union of two independent terms, i.e.,

$$\begin{aligned}\mathcal{A}^+ &= \mathcal{A}^t \bigtriangledown\!\!\!\bigtriangleup \{\mathbb{A}_i^t\}_{i=1}^k \\ &\to \mathcal{A}^t \cup \{\mathbb{A}^t \ominus \mathcal{A}^t\}\end{aligned} \tag{3}$$

where $\ominus$ denotes the pruning and reduction operation. Explained with an instance in Fig. 2 is that: the 2-hop links such as $(1,7)$ are packaged as immediate links and the $(1,6)$ is deleted from $\mathbb{E}^t$, i.e., $\mathbb{A}_{1,7}^t = 1$ and $\mathbb{A}_{1,6}^t = 0$.

Step 2. In Eq. (3), $\mathcal{A}^t$ represents the *pure incremental features* in the incremental graph snapshot, and $\{\mathbb{A}^t \ominus \mathcal{A}^t\}$ represents the *replayed historical features* in the summary graph. It implies that the input features of SGAE consist of network streaming in two directions, and we focus on integrating them based on the Lemma 1.

Lemma 1. *Given the matrix $\mathbf{X}_1 \in \mathbb{R}^{n\times n}$ and $\mathbf{X}_2 \in \mathbb{R}^{n\times n}$, the distribution law will be occurred in $\mathbf{W}(\mathbf{X}_1 + \mathbf{X}_2)$, where $\mathbf{W} \in \mathbb{R}^{n\times n}$ is a weight matrix. Formally, $\mathbf{W}(\mathbf{X}_1 + \mathbf{X}_2) = \mathbf{W}\mathbf{X}_1 + \mathbf{W}\mathbf{X}_2$.*

Proof. The left term can be rewritten as $\sum_k^{|\mathbb{A}^t|} w_{i,k}(x_{k,j}^t + x_{k,j}^h)$. After multiplying the $w_{i,k}$ factor into the bracket, it equals to $\sum_k^{|\mathbb{A}^t|}(w_{i,k}X_{k,j}^t) + (w_{i,k}X_{k,j}^h)$. This formulation is the same as the right term, i.e., $\mathbf{W}\mathbf{X}_1 + \mathbf{W}\mathbf{X}_2$. Thus, the correctness of Lemma 1 is proved.

Therefore, if ensure $\mathcal{A}^t$ with the same size as $\{\mathbb{A}^t \ominus \mathcal{A}^t\}$, we can take $\mathcal{A}^+ = \mathcal{A}^t + (\mathbb{A}^t \ominus \mathcal{A}^t)$ as the input features of SGAE. Here we use the matrix expansion operation to satisfy this condition. Take the $|\mathcal{V}^t| > |\mathbb{V}^t|$ as an example, which means the size of $\mathcal{A}^t$ is larger than $\{\mathbb{A}^t \ominus \mathcal{A}^t\}$, we need to expand $\{\mathbb{A}^t \ominus \mathcal{A}^t\}$ from $|\mathbb{V}^t| \times |\mathbb{V}^t|$ to $|\mathcal{V}^t| \times |\mathcal{V}^t|$ before performing the geometric addition between two matrices. Here, the expanded space of $\{\mathbb{A}^t \ominus \mathcal{A}^t\}$ is filled with 0. The case of $|\mathbb{V}^t| > |\mathcal{V}^t|$ is similar to that of $|\mathcal{V}^t| > |\mathbb{V}^t|$.

4.3.2. Architecture of SGAE

Overall, SGAE is still an encoder–decoder structure. The encoder imposes linear transformation on the input features ($\mathcal{A}^+$) to extract global and local patterns of the input graph. The decoder reconstructs $\mathcal{A}^+$ to optimize the vertex representations in an unsupervised fashion by using the reversed calculation process of the encoder [34]. Specifically, the encoder of SGAE is shown in Eq. (4).

$$\begin{aligned}\mathcal{F}_1^t &= \sigma(\mathbf{W}_1 \cdot \mathcal{A}^+ + \mathbf{b}_1) \\ \mathcal{F}_l^t &= \sigma(\mathbf{W}_l \cdot \mathcal{F}_{l-1}^t + \mathbf{b}_l); l = 2, \dots, L\end{aligned} \tag{4}$$

where L is the number of linear layers. $\sigma(x) = \frac{1}{1+e^{-x}}$ is the non-linear activation function. $\mathbf{W}$ and $\mathbf{b}$ are trainable parameters.

After obtaining $\mathcal{F}_L^t$, the decoder aims to reconstruct the input feature and obtains the output $\hat{\mathcal{A}}^+$, which can be estimated as,

$$\hat{\mathcal{A}}^+_{l-1} = \sigma(\mathbf{H}_l \cdot \mathcal{F}_l^t + \mathbf{b}_l); l = L, \dots, 2 \tag{5}$$

where $\mathbf{H} = (\mathbf{W}^\top\mathbf{W})^{-1}\mathbf{W}^\top$ is the trainable parameter.

The goal of SGAE is also to minimize the reconstruction error of the output $\hat{\mathcal{A}}^+$ and the input $\mathcal{A}^+$, so its objective function can be written as Eq. (6).

$$\mathcal{O} = \| (\hat{\mathcal{A}}^+ - \mathcal{A}^+) \odot \mathbf{M} \|_F^2 = \sum_{i=1}^{|\mathcal{V}^+|} (\| (\hat{\mathcal{A}}^+_i - \mathcal{A}_i^+) \odot \mathbf{m}_i \|_2^2) \tag{6}$$

where $\mathcal{A}_i^+ = \{x_{i,j}\}_{j=1}^{|\mathcal{V}^+|}$ and $x_{i,j} = 1$ (or 0) denotes there is (or not) a link between vertices i and j. $\odot$ denotes the Hadamard product operation. $\mathbf{M}$ is a penalty matrix that imposes more penalty to the reconstruction errors of non-zero elements than that of zero elements [35], w.r.t. $\mathbf{m}_i = \{m_{i,j}\}_{j=1}^{|\mathcal{V}^+|}$. If $x_{i,j} = 0$, $m_{i,j} = 1$, else $m_{i,j} = \beta > 1$.

4.3.3. Training of SGAE

Eq. (6) preserves the global patterns of $\mathcal{A}^+$, which implicitly maintains the *second-order* proximity among vertices. In addition, we also introduce the *first-order* proximity to help SGAE preserve the local patterns of $\mathcal{G}^+$, i.e.,

$$\mathcal{O}_{1st} = \sum_{i,j=1}^{|\mathcal{V}^+|} x_{i,j} \parallel \mathcal{F}_L^t(v_i) - \mathcal{F}_L^t(v_j) \parallel_2^2 \tag{7}$$

Furthermore, we introduce parameter regularization to prevent SGAE from overfitting the graph structure, like,

$$\mathcal{O}_{reg} = \frac{1}{2}\sum_{l=1}^{L}(\parallel \mathbf{W}_l \parallel_F^2 + \parallel \mathbf{H}_l \parallel_F^2) \tag{8}$$

Hence, the new mapping function $\mathcal{F}^t$ trained on $\mathcal{G}^+$ can be learned by minimizing the joint loss function shown in Eq. (9). Here, we rewrite Eq. (6) as $\mathcal{O}_{2nd}$ for description.

$$\mathcal{F}^t(\mathcal{V}^+, \theta^t) = min(\mathcal{O}_{1st} + \alpha\mathcal{O}_{2nd} + \delta\mathcal{O}_{reg}) \tag{9}$$

where α and δ are hyper-parameters. The θ^t denotes the parameter set of $\mathcal{F}^t$.

4.4. Knowledge distillation

As mentioned before, losing track of static historical features in the new DNE model causes the catastrophic forgetting problem. Therefore, we propose a *knowledge distillation* module to keep the freshness of historical knowledge in $\mathcal{F}^t$. This module contains three steps, that is,

Model expansion. First, to carry the vertices that newly appear in $\mathcal{G}^t$, we extend the base model $\mathcal{F}^{t-1}$ from $|\mathcal{V}^{t-1}| \times d$ to $|\mathcal{V}^+| \times d$.

$$\mathcal{F}^{t-1}(v \in \mathcal{V}^+ \backslash \mathcal{V}^{t-1}) = unif(-\sqrt{\frac{12}{d}}, \sqrt{\frac{12}{d}}) \tag{10}$$

where $unif$ represents the uniform sampling. d denotes the dimensional size of vertex embeddings.

Objective function. Second, regarding $\mathcal{F}^{t-1}$ and $\mathcal{F}^t$ as the teacher model and the student model respectively, we optimize the distillation loss between them to ensure the historical knowledge is not lost in $\mathcal{F}^t$. Specifically, for each $v \in \mathcal{V}^+$, we perform the mean-squared error (MSE) loss on $\mathcal{F}^t(v)$ and $\mathcal{F}^{t-1}(v)$, i.e.,

$$\mathcal{O}_{kd} = \sum_{v \in \mathcal{V}^+} \parallel \mathcal{F}^t(v) - \mathcal{F}^{t-1}(v) \parallel_2^2 \tag{11}$$

So far, with the incremental graph snapshot streaming into the training pipeline, we can obtain the new DNE model by minimizing the knowledge distillation loss, denoted as Eq. (12).

$$\mathcal{F}^t(\mathcal{V}^+, \Omega) = min(\mathcal{O}_{kd}) \tag{12}$$

In practice, we jointly train Eqs. (9) and (12) to improve training efficiency and decrease error propagation. Therefore, the overall objective function of CLDNE shown in Eq. (1) can be specifically written as Eq. (13).

$$\begin{aligned}\mathcal{O}_{cldne} &= \mathcal{O}_{1st} + \alpha\mathcal{O}_{2nd} + \gamma\mathcal{O}_{kd} + \delta\mathcal{O}_{reg} \\ &= \sum_{i,j=1}^{|\mathcal{V}^+|}(x_{i,j} \parallel \mathcal{F}_L^t(v_i) - \mathcal{F}_L^t(v_j) \parallel_2^2) \\ &+ \alpha\sum_{i=1}^{|\mathcal{V}^+|}(\parallel (\hat{\mathbf{X}}_i - \mathbf{X}_i) \odot \mathbf{m}_i \parallel_2^2) \\ &+ \gamma\sum_{v \in \mathcal{V}^+}(\parallel \mathcal{F}_L^t(v) - \mathcal{F}^{t-1}(v) \parallel_2^2) \\ &+ \frac{\delta}{2}\sum_{i=1}^{L}(\parallel \mathbf{W}_i \parallel_F^2 + \parallel \mathbf{H}_i \parallel_F^2)\end{aligned} \tag{13}$$

Knowledge transfer. Last, besides the vertices sampled into the summary graph to train the model at time t, other stable vertices in the historical network also have the opportunity to train the new model at time $t+1$. Therefore, the embeddings of stable vertices are directly transferred from $\mathcal{F}^{t-1}$ to $\mathcal{F}^t$ for model stability.

$$\mathcal{F}^t(\mathcal{V}^+, \Omega) = \begin{cases}\mathcal{F}^t(v), & v \in \mathcal{V}^+ \\ \mathcal{F}^{t-1}(v), & v \in \mathcal{V}^{t-1} \backslash \mathcal{V}^+\end{cases} \tag{14}$$

Algorithm 1 CLDNE Algorithm

Input: base model $\mathcal{F}^{t-1}$, network streaming $\{\mathcal{G}^0, \mathcal{G}^1, ..., \mathcal{G}^t\}$
Parameters: dimensional d, initial parameter set $\{\mathbf{W}, \mathbf{b}\}$
historical window τ, sampling hop k, replay ratio Γ
Output: The new mapping function $\mathcal{F}^t$.

```
# Experience Replay
G_his^τ = {G^{t-τ}, G^{t-τ+1}..., G^{t-1}}
G_s^t = ⋃_{i=1}^{k} (V_i^t, E_i^t, A_i^t) = ⋃_{i=1}^{k} S_{v∈V^+}(v, G_his^τ, i, Γ)
# Streaming Graph Auto-Encoder (SGAE)
G^+ = G_s^t ⋈ G^t = (V^+, E^+, A^+), A^+ = A^t ∪ {A^t ⊖ A^t}
Apply matrix expansion on A^t and {A^t ⊖ A^t} to obtain X.
# Base Model Expansion
if v in V^+ and not in V^{t-1} then
    F^{t-1}(v) = unif(-sqrt(12/d), sqrt(12/d))
end if
repeat
    apply Eq.(4) on X to obtain F_L^t = (V^+, θ^t)
    apply Eq.(5) on F_L^t to obtain the output X̂
    # Knowledge Distillation
    apply Eq.(11) to distill historical knowledge
    minimize O_cldne = O_1st + αO_2nd + γO_kd + δO_reg
    use ∂O_cldne/∂θ^t on G^+ to update parameters θ^t → Ω
until converge
Apply Eq.(14) on F^t for model stability
Obtain the new mapping function F^t(V, Ω)
```

Table 1
Topological details of four dynamic networks.

Networks	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	Snapshots	$\rho(\mathcal{V})$	$\rho(\mathcal{E})$	#label
Cora-14t	2708	5429	14	0.00%	100.00%	7
wiki-RfA-4t	11,376	181,073	4	18.29%	100.00%	–
DBLP-8t	28,085	236,894	8	25.64%	76.70%	10
ML-10M-6t	20,537	43,760	6	35.37%	76.33%	–

$\rho_{inc.}(\mathcal{V})$ and $\rho_{inc.}(\mathcal{E})$ denote the average ratio of incremental vertices and links in all new networks.

Algorithm 1 provides the pseudo-code of CLDNE, where we incorporate three modules into the model training.

5. Experiments

5.1. Datasets

In our experiments, we evaluate CLDNE on four dynamic networks, whose details are summarized in Table 1. Vertices of Cora-14t are all contained in the base graph snapshot, and each incremental graph snapshot does not generate new vertices. On the contrary, vertices of wiki-RfA-4t, DBLP-8t, and ML-10M-6t are partially contained in the base graph snapshot, and each incremental graph snapshot will carry out new vertices.

5.2. Experimental setting

Task definition. We mainly evaluate CLDNE on link prediction and node classification tasks in our experiment. (i) For each edge $(i, j, t) \in \mathcal{G}_{test}^t$ in the link prediction task, we evaluate an object $(i, ?, t)$ or a subject $(?, j, t)$ based on the mapping function $\mathcal{F}^t$. Practically, we sample negative links for (i, j, t) and construct the pattern $\mathcal{P}^t$ =

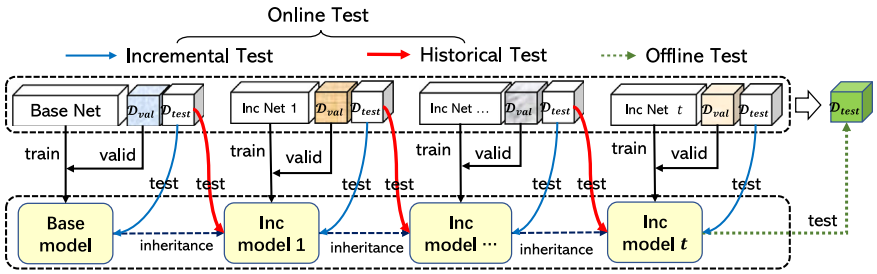

Fig. 3. Training and evaluation protocol of CLDNE. "Online test" refers to evaluate CLDNE on each $\mathcal{G}^t_{test}$ after training it on the $\mathcal{G}^t_{train}$ ("Incremental Test") or $\mathcal{G}^{t+1}_{train}$ ("Historical Test"). "Offline test" refers to evaluating CLDNE on an offline test set after training it on the last graph snapshot.

$\{(i,j,t) \cup (i,j',t) \cup (i',j,t)\}$. Based on $\mathcal{P}^t$, we train a logistic regression (LR) classifier to determine the label y of $(u,v,t) \in \mathcal{P}^t$. (ii) For each vertex $v \in \mathcal{V}^t_{test}$ in the node classification task, we predict its label (y_v) based on the mapping function $\mathcal{F}^t$. Practically, we take $\mathcal{F}^t$ as the pre-trained model to obtain vertex embeddings and design a decoder to predict the labels.

Training and evaluation protocol. Fig. 3 shows the training and evaluation procedure of CLDNE. According to the experimental setup in continuous learning, each network state $\mathcal{G}^t$ is first partitioned into $\mathcal{G}^t_{train}$, $\mathcal{G}^t_{valid}$ and $\mathcal{G}^t_{test}$. Then, we train a base model from scratch on $\mathcal{G}^0_{train}$ and select the best model via $\mathcal{G}^0_{valid}$ and $\mathcal{G}^0_{test}$. Last, we train CLDNE incrementally on other graph snapshots and employ two evaluation manners (i.e., online test and offline test) to measure the performance of CLDNE.

Evaluation metrics. We employ INC-LEARNING and HIS-FORGETTING to evaluate CLDNE in the online test manner and introduce FULL-ACCURACY to evaluate CLDNE in the offline test manner. The evaluation metrics are selected to $\bar{\text{AUC}}$ (averaged Area of Under Cover), $\bar{\text{ACC}}$ (averaged Accuracy) and Ma-$\bar{\text{F}}_1$ (averaged Macro F_1-value).

(i) INC-LEARNING. It is used to evaluate the performance of CLDNE in learning new patterns. When obtaining $\mathcal{F}^t$ at time step t, it imposes $\mathcal{F}^t$ on the test set $\mathcal{G}^t_{test}$ to calculate three measure metrics. After all $\mathcal{F}^t, t \in \{1,2,\ldots,T\}$ are finished to evaluate, the averaged $\{\text{AUC}, \text{ACC}, Ma\text{-F}_1\}$ is the INC-LEARNING evaluation of CLDNE.

(ii) HIS-FORGETTING. It is used to observe the degree of knowledge forgetting in CLDNE. After using the recent historical model $\mathcal{F}^{i-1}$ to incrementally learn $\mathcal{F}^i$, the performance of $\mathcal{F}^i$ on the historical graph snapshot $\mathcal{G}^{i-1}$ is denoted as $\mathcal{F}^{\mathcal{H}}_{i,i-1}$. The INC-LEARNING evaluation of historical model $\mathcal{F}^{i-1}$ on $\mathcal{G}^{i-1}$ is denoted as $\mathcal{F}^{\mathcal{C}}_{i-1,i-1}$. Consequently, the HIS-FORGETTING evaluation can be calculated by $\mathcal{F}^{\mathcal{C}}_{i-1,i-1} - \mathcal{F}^{\mathcal{H}}_{i,i-1}$.

(iii) FULL-ACCURACY. It evaluates the global vertex embeddings on an offline test set ($\mathcal{G}_{test}$), which is a static graph derived from the dynamic network. After training CLDNE on the last graph snapshot, it measures the performance of the final mapping function $\mathcal{F}$ on $\mathcal{G}_{test}$ by calculating three evaluation metrics.

Implementation and hyper-parameters. All settings of CLDNE are implemented with PyTorch (GPU with two 3090 graphics cards). We set the number of layers to 2, the learning rate to $1e^{-3}$, and the dropout to 0.5. The embedding dimensions are set to 128 for all experiments. The batch size is set to 100 and the epoch is set to 500. We employ $\alpha = 1e^{-2}$, $\beta = 5$, $\gamma = 1e^{-2}$ and $\delta = 1e^{-4}$ in the objective function.

We also introduce three tunable parameters in CLDNE: (i) the historical window length τ to determine the available historical networks in experience replay. (ii) the replay ratio Γ to control the limitations of sampled links in SGAE. (iii) the k-hops neighbourhoods to summarize an abstract of historical networks. We tune these parameters from $\tau = \{1,2,all\}$,[1] $\Gamma = \{0.2,0.4,0.6,0.8,1.0\}$, and $k = \{1,2,3,4\}$. In CLDNE, we set them to $\tau = all$, $\Gamma = 1.0$, and $k = 2$.

[1] denote all historical networks are used to construct the summary graph in the experience replay buffer.

Baselines and skyline. We compare CLDNE with DynGEM [7], Inc_SAT [8], EvolveGCN [36], OnlineGNN [30], ContinualGNN [30], dyngraph2vecAE [37], TWP-GAE [13], ROLAND [14], SpikeNet [38], and Ada-DyGNN [21] on two tasks with the offline test setting. All parameters of these baselines are consistent with their released source code. Besides, we design three variants of CLDNE as baselines, i.e., FinetuneGAE, DistillGAE, and ReplayGAE. The FinetuneGAE setting is the lower bound without any continual learning mechanism. The DistillGAE and ReplayGAE settings replay the historical features from the model and data perspectives, respectively. Last, we provide a skyline setting of CLDNE, i.e., RetrainGAE, which is jointly trained on all data in $\{\mathcal{G}^0,\ldots,\mathcal{G}^t\}$ at time step t.

6. Results and discussions

6.1. Overall performance

To demonstrate the overall performance of CLDNE after training on the last graph snapshot, we take the global vertex embeddings of different methods to carry out the link prediction task on three dynamic networks. The FULL-ACCURACY under the offline test manner is summarized in Table 2.

We can observe that CLDNE outperforms the baseline methods in most situations, especially in the wiki-RfA-4t network. There are several reasons for this superiority. Firstly, as opposed to parameter updating techniques such as DynGEM and OnlineGNN, CLDNE incorporates historical data into the training process as new graph data arrives. This approach ensures robustness against changes in the data representation. Secondly, unlike the experience replay methods like ContinualGNN and dyngraph2vecAE, CLDNE condenses the available historical data into a high-order abstraction graph. This abstract graph serves as prior knowledge for predicting possible links in the future. Lastly, when compared to streaming GNNs such as EvolveGCN and ROLAND, CLDNE explores new patterns in incremental data by reconstructing the adjacency matrix instead of simply aggregating neighbour information. This unique approach enables CLDNE to effectively capture global topological features while considering local information as well. Although SpikeNet is better than CLDNE in the DBLP-8t network, it requires the supervisory signals of vertices to train the model, so they are limited to the expansion on the other two networks.

Furthermore, upon comparing CLDNE with its three variants, we can draw two-fold conclusions. On the one hand, when the knowledge distillation module is reduced (ReplayGAE), the performance of CLDNE only exhibits a slight decline. This suggests that the static features learned in the base model have minimal influence on the learning of new patterns. On the other hand, when the experience replay buffer is reduced (DistillGAE and FinetuneGAE), the performance of CLDNE experiences a significant decrease. This highlights the crucial importance of replaying high-order historical topology in CLDNE. Additionally, we also assess the performance of CLDNE and its variants on the test set at each time step t. The results are shown in Fig. 4. The superior performance of CLDNE at each time step emphasizes the complementary and essential nature of the combined modules in capturing

Table 2
The FULL-ACCURACY of the link prediction task in the offline test manner.

METHODS	wiki-RfA-4t			DBLP-8t			ML-10M-6t		
	$\bar{A}\bar{C}C$	$\bar{A}\bar{U}C$	Ma-$\bar{F}_1$	$\bar{A}\bar{C}C$	$\bar{A}\bar{U}C$	Ma-$\bar{F}_1$	$\bar{A}\bar{C}C$	$\bar{A}\bar{U}C$	Ma-$\bar{F}_1$
DynGEM	0.5377	0.6001	0.5595	0.5633	0.6075	0.6287	0.5837	0.5921	0.5619
OnlineGNN	0.5479	0.6063	0.5538	0.5591	0.5934	0.6193	0.5713	0.5929	0.5817
Inc_SAT	0.6273	0.6789	0.6393	0.5785	0.6123	0.6311	0.6377	0.6721	0.6719
EvolveGCN	0.6400	0.6869	0.6097	0.5711	0.6023	0.5961	0.6317	0.6533	0.6570
ContinualGNN	0.6537	0.7089	0.6675	0.5841	0.6274	0.6253	0.6337	0.6521	0.6637
dyngraph2vecAE	0.6381	0.6939	0.6193	0.5733	0.6081	0.6001	0.6279	0.6413	0.6520
TWP-GAE	0.6577	0.7101	0.6685	0.5803	0.6237	0.6279	0.6451	0.6693	0.6699
ROLAND	0.6507	0.7036	0.6676	0.5791	0.6253	0.6261	0.6517	0.6633	0.6670
SpikeNet	–	–	–	**0.6231**	**0.6793**	0.6303	–	–	–
Ada-DyGNN	0.6064	0.6329	0.6831	0.6048	0.6244	0.5788	0.6092	0.6613	**0.7053**
CLDNE*	**0.6613**	**0.7313**	**0.6892**	0.5982	0.6435	**0.6377**	**0.6720**	**0.6940**	0.6907
FinetuneGAE	0.5344	0.5723	0.2955	0.5641	0.5932	0.6182	0.5534	0.5568	0.3614
DistillGAE	0.5383	0.5724	0.3044	0.5654	0.5965	0.6215	0.5491	0.5583	0.3218
ReplayGAE	0.6609	0.7298	0.6855	0.5970	0.6349	0.6276	0.6628	0.6884	0.6754
RetrainGAE (skyline)	*0.8555*	*0.9139*	*0.8472*	*0.7629*	*0.8157*	*0.7702*	*0.7719*	*0.8413*	*0.7634*

The results of CLDNE are obtained based on the parameter set $\{\tau = all, \Gamma = 1.0, k = 2\}$. "$*$" denotes the $p\text{-}value < 0.01$ when performing T-test on CLDNE with the best baseline. "–" denotes N/A because of the lack of vertex labels.

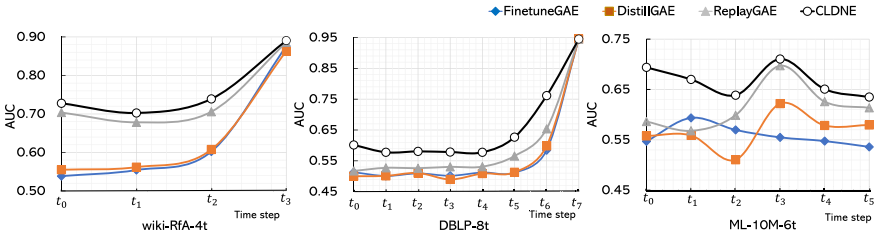

Fig. 4. The AUC values in FULL-ACCURACY on different time steps t_i.

Table 3
Results of different methods on the node classification task.

METHODS	Cora-14t		DBLP-8t	
	Macro-F_1	Micro-F_1	Macro-F_1	Micro-F_1
DynGEM	0.5776	0.6856	0.4727	0.5268
OnlineGNN	0.5176	0.6996	0.4599	0.5125
Inc_SAT	0.5917	0.6912	0.5012	0.5337
EvolveGCN	0.5917	0.6912	0.5012	0.5337
ContinualGNN	0.5833	0.7956	0.4636	0.5149
dyngraph2vecAE	0.6036	0.7985	0.4979	0.5625
TWP-GAE	0.6147	0.7993	0.5303	0.5510
ROLAND	0.5901	0.7078	0.4727	0.5268
SpikeNet	**0.7612**	0.8081	0.5652	0.5953
Ada-DyGNN	0.7404	0.8005	0.5711	0.5979
CLDNE	0.6394	**0.8101**	**0.5862**	**0.6025**
FinetuneGAE	0.5892	0.7070	0.4674	0.5168
DistillGAE	0.5909	0.7208	0.4688	0.5237
ReplayGAE	0.6101	0.7859	0.5176	0.5526
RetrainGAE(skyline)	*0.6418*	*0.8137*	*0.6444*	*0.6526*

network features. This observation further strengthens the notion that the integration of three modules enhances the effectiveness of CLDNE.

To showcase the scalability of CLDNE, we assess the performance of global vertex embeddings learned by various methods on the node classification task. In this regard, we employ the decoder utilized in ContinualGNN for predicting vertex labels. The Macro-F_1 and Micro-F_1 scores are summarized in Table 3. The results demonstrate that CLDNE delivers promising outcomes on two widely used dynamic networks. Specifically, CLDNE is comparable with the advanced supervised GNN framework for the Micro-F_1 evaluation in the Cora-14t network. Most importantly, CLDNE outperforms all methods in the DBLP-8t network. These findings elucidate that CLDNE is not only effective in exploring vertex similarity features for link prediction but also proficient in learning discriminative vertex features for node classification.

6.2. Overcome catastrophic forgetting

To show the effectiveness of CLDNE in striking the balance between learning new patterns and retaining historical knowledge, we report the INC-LEARNING and HIS-FORGETTING in the online test manner in Table 4, from which we conclude the following three-fold observations:

(i) CLDNE outperforms all baselines in terms of HIS-FORGETTING. This result indicates that at each time step, CLDNE effectively preserves previous patterns from historical data while simultaneously acquiring new patterns from the incremental graph snapshot. In the best-case scenario, CLDNE achieves a notable consolidation of approximately 5.9% ACC, 5.2% AUC, and 2.8% macro-F_1 for historical knowledge. This can be attributed to CLDNE's ability to learn the new streaming function by integrating high-order historical data and incremental data, which ensures the stability of historical features in the new function. (ii) CLDNE demonstrates comparable performance to other benchmark methods in INC-LEARNING. While in some cases the performance of CLDNE slightly falls short of the optimal benchmark setting, overall performance is either comparable or slightly better. This suggests that the introduction of historical data and static features has a relatively minor negative impact on CLDNE's ability to learn new patterns. (iii) CLDNE significantly reduces training time by approximately 80%. Compared to the skyline setting (RetrainGAE), CLDNE saves about 80% of training time in learning the new streaming function at each time step. Additionally, the training data volume for CLDNE is approximately six times smaller than that of RetrainGAE. Furthermore, we observe that RetrainGAE is not consistently effective in learning new patterns. For instance, in the case of DBLP-8t, approximately 34% of test edges have already appeared in the training set, simulating potential future author cooperation. In such scenarios, RetrainGAE fails to achieve better INC-LEARNING than CLDNE, indicating that the unguided introduction of historical topology might interfere with the learning of new patterns.

To demonstrate the superior ability of CLDNE in retaining historical knowledge compared to other baselines, we also present the performance of CLDNE on the latest historical snapshot (time $t-1$)

Table 4
The HIS-FORGETTING and INC-LEARNING of the link prediction task in the online test manner.

	METHODS	HIS-FORGETTING ↓			INC-LEARNING ↑			Complexity ↓	
		$\bar{ACC}$	$\bar{AUC}$	Ma-$\bar{F}_1$	$\bar{ACC}$	$\bar{AUC}$	Ma-$\bar{F}_1$	Avg./Tot. Time (s)	DataSize
wiki-RfA-4t	DynGEM	0.1632	0.2029	0.2203	0.7702	0.8458	0.7517	0.46/229.67	-K
	Inc_SAT	0.1633	0.1905	0.2026	0.7858	0.8527	0.7602	0.57/285.63	-K
	ContinualGNN	0.1363	0.1251	0.1106	0.7833	0.8497	0.7575	0.60/300.63	-K
	dyngraph2vecAE	0.1325	0.1277	0.1035	0.7793	0.8458	0.7597	0.71/354.78	-K
	TWP-GAE	0.1258	0.1229	0.0993	0.7887	0.8505	0.7662	0.48/237.72	-K
	CLDNE	**0.1176**	**0.1152**	**0.0776**	0.7924∗	0.8535∗	0.7735∗	**0.82/407.81** ▽	**561K** ▽
	FinetuneGAE	0.1818	0.2171	0.2537	0.7807	0.8509	0.7601	0.44/220.25	374K
	DistillGAE	0.1693	0.2134	0.2526	0.7767	0.8466	0.7532	0.44/220.62	374K
	ReplayGAE	0.1371	0.1297	0.0895	**0.7970**	**0.8749**	**0.7777**	0.80/400.31	561K
	RetrainGAE (skyline)	*0.0293+*	*0.0007+*	*0.0380+*	*0.8380*	*0.9000*	*0.8275*	*4.19/2095.00* ▲	*1.51M* ▲
DBLP-8t	DynGEM	0.2858	0.2633	0.2993	0.9103	0.9525	0.9031	3.19/1593.49	-M
	Inc_SAT	0.2813	0.2575	0.2935	0.9197	0.9593	0.9172	3.47/1734.48	-M
	ContinualGNN	0.2391	0.2218	0.2713	0.9100	0.9597	0.9134	3.70/1851.32	-M
	dyngraph2vecAE	0.2301	0.2189	0.2607	0.9104	0.9600	0.9102	4.50/2249.31	-M
	TWP-GAE	0.2279	0.2205	0.2583	0.9187	0.9617	0.9119	3.22/1610.35	-M
	CLDNE	**0.2247**	**0.2121**	**0.2538**	**0.9238**∗	**0.9644**∗	**0.9220**∗	**4.07/2034.07** ▽	**1.92M** ▽
	FinetuneGAE	0.3150	0.3165	0.3229	0.9034	0.9509	0.9006	3.19/1595.12	1.71M
	DistillGAE	0.3257	0.3175	0.3261	0.9062	0.9521	0.9033	3.19/1595.48	1.71M
	ReplayGAE	0.2301	0.2247	0.2622	0.9214	0.9596	0.9199	4.05/2026.83	1.92M
	RetrainGAE (skyline)	*0.0008+*	*0.0115+*	*0.0073+*	*0.8603*	*0.9357*	*0.8520*	*17.22/8607.85* ▲	*6.36M* ▲
ML-10M-6t	DynGEM	0.0087	0.0219	0.0858	0.5273	0.5345	0.5037	0.60/298.45	-K
	Inc_SAT	0.0073	0.0179	0.0595	0.5558	0.5607	0.5362	0.62/311.07	-K
	ContinualGNN	0.0311+	0.0303+	0.0007	0.5485	0.5532	0.5234	0.57/287.85	-K
	dyngraph2vecAE	0.0267+	0.0319+	0.0034+	0.5429	0.5553	0.5427	0.66/330.67	-K
	TWP-GAE	0.0359+	0.0327+	0.0113+	0.5416	0.5571	0.5382	0.55/278.31	-K
	CLDNE	**0.0590+**	**0.0520+**	**0.0281+**	**0.5673**∗	0.5597∗	**0.5535**∗	**0.74/367.50** ▽	**290K** ▽
	FinetuneGAE	0.0378	0.0653	0.1146	0.5497	0.5552	0.5000	0.59/296.85	116K
	DistillGAE	0.0033	0.0144	0.0926	0.5221	0.5321	0.4751	0.59/297.13	116K
	ReplayGAE	0.0011	0.0052	0.0162	0.5605	**0.5616**	0.5532	0.73/367.35	290K
	RetrainGAE (skyline)	*0.0603+*	*0.0875+*	*0.0519+*	*0.6532*	*0.6380*	*0.6418*	*3.37/1682.61* ▲	*608K* ▲

The results of CLDNE are obtained based on the parameter set $\{\tau = all, \Gamma = 1.0, k = 2\}$. "∗" denotes the *p-value* < 0.05 when performing T-test on CLDNE with the best baseline. The upper bound results are italicized. ▲ denotes the highest complexity, and ▽ denotes the complexity of CLDNE. "+" denotes the new DNE model outperforms the old DNE model on the historical data.

after training on the incremental graph snapshot at time t. Fig. 5 showcases the results for three variants of CLDNE as well as the compared baselines. The outcomes reveal that the "Increment" curves of different models exhibit similar trends within a dynamic network. However, the "History" curves of CLDNE consistently surpass those of other baselines. This observation indicates that CLDNE performs exceptionally well on historical data even after training on the new graph snapshot. Consequently, it verifies the robustness of the historical exemplar knowledge against changes in the data representation [39].

6.3. Different incremental modes

In practical scenarios, there are dynamic networks like Cora-14t that only introduce new links at each time step. These complex situations pose high demands on the adaptability of DNE models. Thus, in this section, we examine the performance of CLDNE on dynamic networks with different incremental modes.

First, we define the dynamic network like Cora-14t as the **Closed Dynamic Network**. A closed dynamic network consists of the base network $\mathcal{G}^0$ and several incremental networks $\mathcal{G}^i$. The base network $\mathcal{G}^0$ contains all vertices $\mathcal{V}$ and partially edges $\mathcal{E}^0 \subseteq \mathcal{V} \times \mathcal{V}$. Each incremental state $\mathcal{G}^i$ consists of partially vertices $\mathcal{V}^i \subseteq \mathcal{V}$ and new edges $\mathcal{E}^i$.

In addition to Cora-14t, we introduce a new network called wiki-RfA-4t-edge based on the concept of the closed dynamic network. In this network, we consolidate all vertices into a base network and sample a suitable number of links to construct its topological structure. The remaining links that are not sampled are then divided into three incremental network states and progressively streamed into the training pipeline.

We conduct the link prediction task on these two networks and summarize the FuLL-ACCURACY of different methods in Table 5. It is evident from the results that CLDNE consistently achieves the highest performance by effectively preserving the global patterns of the networks. Moreover, the closed dynamic network can be regarded as an expansion of the base network's topology, where the added edges enhance the topological connections among existing vertices. This finding further supports that CLDNE exhibits superior adaptability to the structural evolution in dynamic networks compared to existing methods.

Furthermore, we also evaluate the extent of knowledge forgetting in CLDNE within closed dynamic networks by assessing the vertex

Table 5
The FULL-ACCURACY on the link prediction task in the closed dynamic networks.

METHODS	Cora-14t			wiki-RfA-4t-edge		
	$\bar{ACC}$	$\bar{AUC}$	Ma-$\bar{F}_1$	$\bar{ACC}$	$\bar{AUC}$	Ma-$\bar{F}_1$
DynGEM	0.6895	0.7037	0.6057	0.7974	0.8805	0.7798
OnlineGNN	0.5725	0.6568	0.5733	0.7836	0.8593	0.7687
Inc_SAT	0.6931	0.7105	0.6113	0.8020	0.8897	0.8030
EvolveGCN	0.6917	0.7079	0.6207	0.8112	0.8911	0.8132
ContinualGNN	0.6895	0.7037	0.6057	0.7974	0.8805	0.7798
dyngraph2vecAE	0.7028	0.7341	0.6285	0.8304	0.8972	0.8154
TWP-GAE	0.6997	0.7231	0.6287	0.8073	0.8935	0.8102
ROLAND	**0.7147**	0.7414	0.6155	**0.8329**	**0.9101**	0.8282
SpikeNet	0.7007	0.7437	**0.7017**	–	–	–
Ada-DyGNN	0.6883	0.6667	0.6841	0.8163	0.8791	0.8197
CLDNE*	0.7059	**0.7452**	0.6378	0.8286	0.9023	**0.8307**
FinetuneGAE	0.4992	0.5067	0.0360	0.7964	0.8716	0.7703
DistillGAE	0.5101	0.5044	0.0442	0.8077	0.8850	0.7839
ReplayGAE	0.7020	0.7232	0.6255	0.8135	0.8945	0.7948
RetrainGAE(skyline)	*0.7236*	*0.7630*	*0.6941*	*0.8494*	*0.9128*	*0.8402*

The results are obtained based on the parameter set $\{\tau = all, \Gamma = 1.0, h = 2\}$. "∗" denotes the *p-value* < 0.05 when performing T-test on CLDNE with the best baseline. "–" denotes N/A because of the lack of vertex labels.

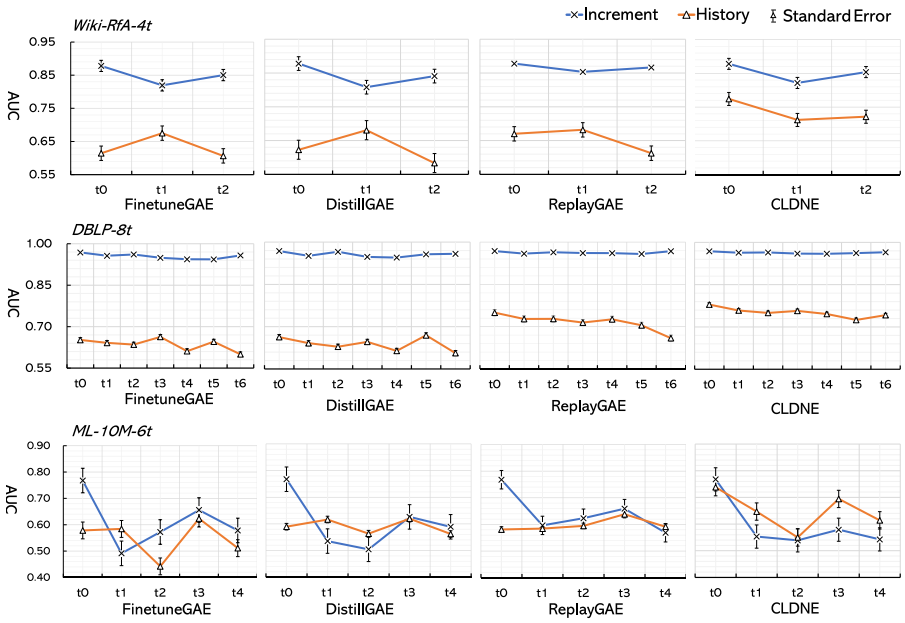

Fig. 5. Performance on the latest historical graph snapshot after training on the incremental graph. "Increment" means to evaluate the new model trained at time t on the test set $\mathcal{G}^t_{test}$. "History" means to evaluate the new model trained at time $t+1$ on the test set $\mathcal{G}^t_{test}$.

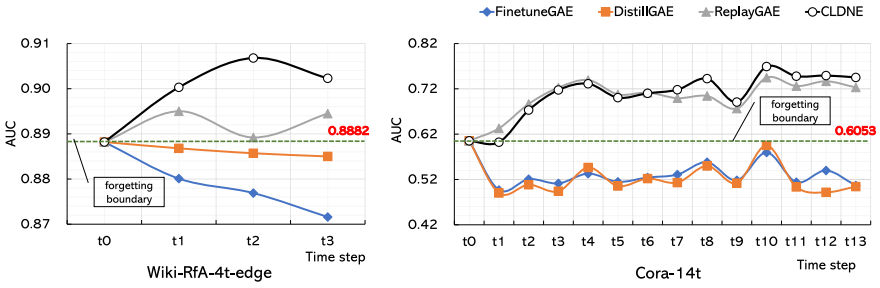

Fig. 6. Knowledge forgetting degree of different methods on two closed dynamic networks. The red dotted line indicates the forgetting boundary. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

embeddings obtained at different time steps on the unified offline test set. The AUC values for CLDNE and its three variants are shown in Fig. 6, with the performance of vertex embeddings trained on the base network serving as the forgetting boundary. Upon analysing the curves of different methods, we observe that the replay-based approaches (CLDNE and ReplayGAE) do not demonstrate knowledge forgetting as new networks are introduced into the training pipeline. This can be attributed to the incorporation of historical links during model training on the incremental network. It also indicates that in dynamic streaming scenarios with progressively dense topological structures, the experience replay strategy is the preferred solution for mitigating the issue of catastrophic forgetting.

6.4. Ablation experiments

6.4.1. Parameter selection

To illustrate the reason for setting three tunable parameters to replay the historical knowledge, we execute CLDNE with different parameters on wiki-RfA-4t. The results are displayed in Figs. 7 and 8.

Fig. 7 illustrates the performance of CLDNE across various historical windows (τ) and replay ratios (Γ), with optimal results observed at $\tau = \text{all}, \Gamma = 1$. This configuration underscores the efficacy of replaying all links from the entirety of available historical networks. The tendency of CLDNE's performance reveals a greater impact of Γ compared to τ. Specifically, with different values of τ, the INC-LEARNING remains relatively stable, fluctuating between $(-0.001, 0.05)$, and the FULL-ACCURACY exhibits minor variations within the range $(0, 0.01)$. In contrast, variations in Γ introduce substantial instability in CLDNE, with volatility in ACC and AUC in INC-LEARNING reaching up to 13.22% and 9.75%, respectively. These findings suggest a practical implication: extracting high-order topological information from a limited set of historical networks suffices, thus obviating the need for extensive storage space for all historical data. Fig. 8 explores the impact of extracting k-order historical links on CLDNE, holding the other parameters constant at $\tau = \text{all}, \Gamma = 1$. It is evident that $k = 2$ significantly enhances CLDNE's ability to assimilate incremental knowledge, corroborating theories on local structural evolution, where common-neighbourhood dynamics are pivotal in the development of local structures [40].

6.4.2. Loss components analysis

To examine the effectiveness of each term in Eq. (13), we conduct ablation experiments on Wiki-RfA-4t to demonstrate the roles of different loss components. The experimental results are summarized in Table 6. We can observe that each component in the objective function contributes to improving the performance of CLDNE, especially the $\mathcal{O}_{2nd}$. In addition, the $\mathcal{O}_{1st}$ makes up for the lack of $\mathcal{O}_{2nd}$ in learning immediate topological features of vertices. And the $\mathcal{O}_{kd}$ helps CLDNE retain fine-grained static features of historical vertices.

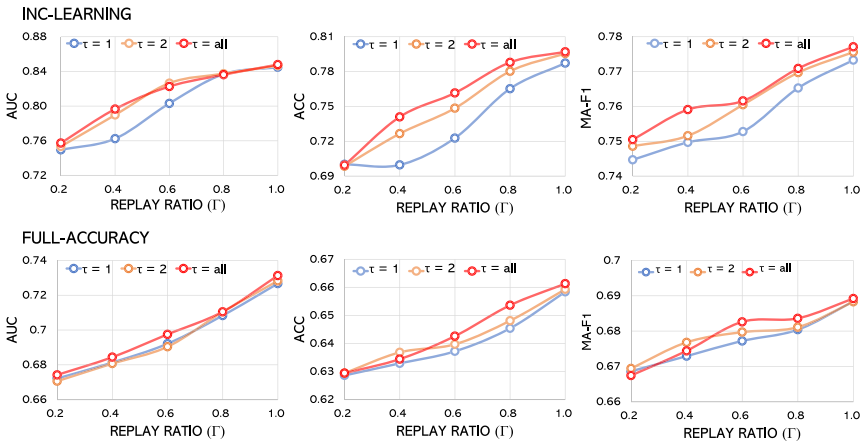

Fig. 7. Performance of CLDNE under different $\tau \in \{1, 2, all\}$ and $\Gamma \in \{0.2, 0.4, 0.6, 0.8, 1.0\}$. We set $\tau = all$ and $\Gamma = 1.0$ in CLDNE.

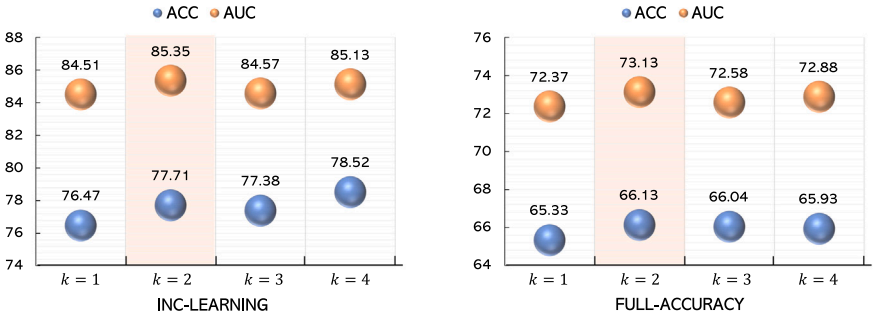

Fig. 8. Performance of CLDNE under k-order topological graph, $k = \{1, 2, 3, 4\}$.

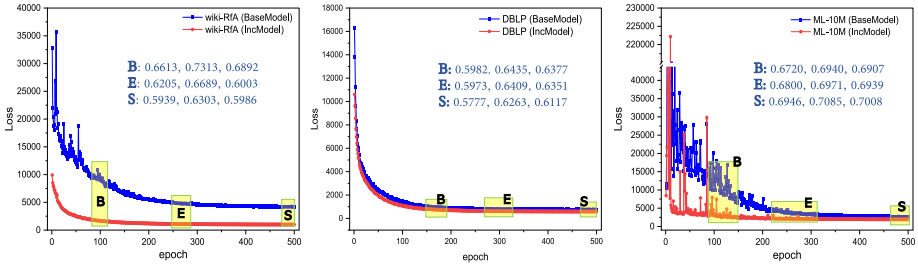

Fig. 9. The loss convergence process of CLDNE on three incremental networks. B is the "*best model*" selected through the validation set. E is "*early stop*" when loss fluctuates within $[-100, 100]$. S is "*stop training*" when the epoch reaches a fixed number (here is 500).

Table 6
Ablation experiment of different loss components on Wiki-RfA-4t.

	INC-LEARNING			FULL-ACCURACY		
	$\bar{\text{ACC}}$	$\bar{\text{AUC}}$	$\bar{\text{F}_1}$	$\bar{\text{ACC}}$	$\bar{\text{AUC}}$	$\bar{\text{F}_1}$
$\mathcal{O}_{cldne}$	**0.7924**	**0.8535**	**0.7735**	**0.6613**	**0.7313**	**0.6892**
$-\mathcal{O}_{kd}$	0.7864	0.8467	0.7668	0.6507	0.7079	0.6628
$-\mathcal{O}_{1st}$	0.7915	0.8504	0.7719	0.6596	0.7243	0.6821
$-\mathcal{O}_{2nd}$	0.6338	0.6102	0.5420	0.5534	0.5760	0.4504

6.4.3. Training strategy

To validate the effectiveness of the training protocol outlined in Section 5.2, we conduct a comparison of FULL-ACCURACY achieved by CLDNE trained with different strategies. We explore three different strategies, namely best model (B), early stop (E), and stop training (S), to identify optimal parameters for CLDNE. The experimental results are summarized in Fig. 9, which indicates that the B strategy outperforms the other two strategies. However, it is important to acknowledge the significance of the early stop and stop training strategies, as the stop training strategy yields higher results on the ML-10M-6t network. This can be attributed to the fact that the stop-training strategy is more robust in cases where there are wide fluctuations in the loss function.

Additionally, it is also evident that the loss values on three dynamic networks exhibit steady fluctuations within a bounded range after training 500 epochs. This indicates that CLDNE is capable of effectively fitting the training data of both the base network state and all incremental graph snapshots. A noteworthy observation is that the loss of IncModel maintains synchronous changes with that of BaseModel, underscoring the compatibility of CLDNE with the parameters of the base model. This compatibility enables CLDNE to fully retain the static features learned from the historical networks.

7. Conclusions

In this study, we explore the challenge of continual learning within the context of dynamic network embedding. We introduce CLDNE, a

novel off-task continual learning framework designed to derive generalized representations of dynamic networks. The framework incorporates a streaming graph auto-encoder that captures global patterns from incremental graph data while employing an experience replay buffer to manage high-order historical information. Extensive experimental evaluations demonstrate the proficiency of CLDNE in assimilating new patterns and retaining previously learned information. Despite these advancements, the issue of catastrophic forgetting remains a significant hurdle in dynamic network embedding. Currently, CLDNE is limited to handling data streams based on anchor vertices over time. In future work, we plan to enhance the framework by integrating more sophisticated experience replay mechanisms and a knowledge distillation loss, aimed at preserving static historical features inductively.

CRediT authorship contribution statement

Zhizheng Wang: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Project administration, Resources, Validation, Visualization, Writing – original draft, Writing – review & editing. **Yuanyuan Sun:** Funding acquisition, Supervision, Writing – review & editing, Project administration. **Xiaokun Zhang:** Validation, Visualization, Writing – original draft, Writing – review & editing. **Bo Xu:** Supervision, Validation, Writing – original draft, Writing – review & editing. **Zhihao Yang:** Funding acquisition, Supervision, Writing – review & editing. **Hongfei Lin:** Funding acquisition, Project administration, Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work is supported by the National Key Research and Development Program of China (No. 2022YFC3301801), and the Fundamental Research Funds for the Central Universities, China (No. DUT22ZD205).

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.patcog.2024.111093.

Data availability

Data will be made available on request.

References

[1] D. Cheng, F. Yang, S. Xiang, J. Liu, Financial time series forecasting with multi-modality graph neural network, Pattern Recognit. 121 (2022) 108218.
[2] M. Li, L. Zhang, L. Cui, L. Bai, Z. Li, X. Wu, BLoG: Bootstrapped graph representation learning with local and global regularization for recommendation, Pattern Recognit. (2023) 109874.
[3] S. Wu, F. Sun, W. Zhang, X. Xie, B. Cui, Graph neural networks in recommender systems: a survey, ACM Comput. Surv. 55 (5) (2022) 1–37.
[4] P. Mei, Y.H. Zhao, Dynamic network link prediction with node representation learning from graph convolutional networks, Sci. Rep. 14 (1) (2024) 538.
[5] Multi-objective optimization algorithm based on characteristics fusion of dynamic social networks for community discovery, Inf. Fusion 79 (2022) 110–123.
[6] M. McCloskey, N.J. Cohen, Catastrophic interference in connectionist networks: The sequential learning problem, in: Psychology of Learning and Motivation, Vol. 24, 1989, pp. 109–165.
[7] P. Goyal, N. Kamra, X. He, Y. Liu, DynGEM: Deep embedding method for dynamic graphs, 2018, arXiv e-prints, arXiv–1805.
[8] A. Sankar, Y. Wu, L. Gou, W. Zhang, H. Yang, Dysat: Deep neural representation learning on dynamic graphs via self-attention networks, in: Proceedings of the 13th International Conference on Web Search and Data Mining, 2020, pp. 519–527.
[9] P. Bielak, K. Tagowski, M. Falkiewicz, T. Kajdanowicz, N.V. Chawla, FILDNE: A framework for incremental learning of dynamic networks embeddings, Knowl.-Based Syst. 236 (2022) 107453.
[10] Z. Liu, C. Huang, Y. Yu, P. Song, B. Fan, J. Dong, Dynamic representation learning for large-scale attributed networks, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 1005–1014.
[11] L. Wang, X. Zhang, H. Su, J. Zhu, A comprehensive survey of continual learning: theory, method and application, IEEE Trans. Pattern Anal. Mach. Intell. (2024) 5362–5383.
[12] J. Hurtado, D. Salvati, R. Semola, M. Bosio, V. Lomonaco, Continual learning for predictive maintenance: Overview and challenges, Intell. Syst. Appl. 19 (2023) 200251.
[13] H. Liu, Y. Yang, X. Wang, Overcoming catastrophic forgetting in graph neural networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35, 2021, pp. 8653–8661.
[14] J. You, T. Du, J. Leskovec, ROLAND: graph learning framework for dynamic graphs, in: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2022, pp. 2358–2366.
[15] C. Hou, H. Zhang, S. He, K. Tang, Glodyne: Global topology preserving dynamic network embedding, IEEE Trans. Knowl. Data Eng. (2020).
[16] C. Ji, T. Zhao, Q. Sun, X. Fu, J. Li, Higher-order memory guided temporal random walk for dynamic heterogeneous network embedding, Pattern Recognit. 143 (2023) 109766.
[17] Y. Lu, X. Wang, C. Shi, P.S. Yu, Y. Ye, Temporal network embedding with micro- and macro-dynamics, in: Proceedings of the 28th ACM International Conference on Information & Knowledge Management, 2019, pp. 469–478.
[18] H. Huang, Z. Fang, X. Wang, Y. Miao, H. Jin, Motif-preserving temporal network embedding, in: IJCAI, 2020, pp. 1237–1243.
[19] P. Bao, J. Li, R. Yan, Z. Liu, Dynamic graph contrastive learning via maximize temporal consistency, Pattern Recognit. 148 (2024) 110144.
[20] N. Yin, M. Wang, Z. Chen, G. De Masi, H. Xiong, B. Gu, Dynamic spiking graph neural networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 38, 2024, pp. 16495–16503.
[21] H. Li, C. Li, K. Feng, Y. Yuan, G. Wang, H. Zha, Robust knowledge adaptation for dynamic graph neural networks, IEEE Trans. Knowl. Data Eng. (2024) 1–14.
[22] G. Zhang, Z. Hu, G. Wen, J. Ma, X. Zhu, Dynamic graph convolutional networks by semi-supervised contrastive learning, Pattern Recognit. 139 (2023) 109486.
[23] Y. Xu, Y. Zhang, W. Guo, H. Guo, R. Tang, M. Coates, Graphsail: Graph structure aware incremental learning for recommender systems, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 2861–2868.
[24] F. Mi, X. Lin, B. Faltings, Ader: Adaptively distilled exemplar replay towards continual learning for session-based recommendation, in: Fourteenth ACM Conference on Recommender Systems, 2020, pp. 408–413.
[25] J. Rajasegaran, S. Khan, M. Hayat, F.S. Khan, M. Shah, Itaml: An incremental task-agnostic meta-learning approach, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 13588–13597.
[26] L. Li, E. Piccoli, A. Cossu, D. Bacciu, V. Lomonaco, Calibration of continual learning models, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 4160–4169.
[27] M. Perini, G. Ramponi, P. Carbone, V. Kalavri, Learning on streaming graphs with experience replay, in: Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing, 2022, pp. 470–478.
[28] J. Wang, W. Zhu, G. Song, L. Wang, Streaming graph neural networks with generative replay, in: Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2022, pp. 1878–1888.
[29] A. Mallya, S. Lazebnik, Packnet: Adding multiple tasks to a single network by iterative pruning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7765–7773.
[30] J. Wang, G. Song, Y. Wu, L. Wang, Streaming graph neural networks via continual learning, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 1515–1524.
[31] D. Goswami, Y. Liu, B.o. Twardowski, J. van de Weijer, FeCAM: Exploiting the heterogeneity of class distributions in exemplar-free continual learning, in: A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, S. Levine (Eds.), Advances in Neural Information Processing Systems, Vol. 36, Curran Associates, Inc., 2023, pp. 6582–6595.
[32] J. Liu, W. Ke, P. Wang, Z. Shang, J. Gao, G. Li, K. Ji, Y. Liu, Towards continual knowledge graph embedding via incremental distillation, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 38, 2024, pp. 8759–8768.
[33] H. Tian, R. Zafarani, Exploiting common neighbor graph for link prediction, in: CIKM, 2020, pp. 3333–3336.
[34] G. Salha, R. Hennequin, M. Vazirgiannis, Simple and effective graph autoencoders with one-hop linear models, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, 2020, pp. 319–334.

[35] D. Wang, P. Cui, W. Zhu, Structural deep network embedding, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 1225–1234.
[36] A. Pareja, G. Domeniconi, J. Chen, T. Ma, T. Suzumura, H. Kanezashi, T. Kaler, T. Schardl, C. Leiserson, Evolvegcn: Evolving graph convolutional networks for dynamic graphs, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 5363–5370.
[37] P. Goyal, S.R. Chhetri, A. Canedo, Dyngraph2vec: Capturing network dynamics using dynamic graph representation learning, Knowl.-Based Syst. 187 (2020) 104816.
[38] J. Li, Z. Yu, Z. Zhu, L. Chen, Q. Yu, Z. Zheng, S. Tian, R. Wu, C. Meng, Scaling up dynamic graph representation learning via spiking neural networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 37, 2023, pp. 8588–8596.
[39] R. Kemker, C. Kanan, FearNet: Brain-inspired model for incremental learning, in: International Conference on Learning Representations, 2018.
[40] C. Ai-Xiang, F. Yan, S. Ming-Sheng, C. Duan-Bing, Z. Tao, Emergence of local structures in complex network: common neighborhood drives the network evolution, Acta Phys. Sin. 60 (3) (2011).

Zhizheng Wang is a postdoc fellowship researcher at NIH, USA. He received his M.S. and Ph.D. degrees from the Dalian University of Technology in 2018 and 2023. His research interests include Graph Data Mining, Representation Learning, and Knowledge Graph.

Yuanyuan Sun received the B.E., M.S., and Ph.D. degrees from Dalian University of Technology, Dalian, China, in 2000, 2003, and 2009, respectively. She is currently a professor in the College of Computer Science of the Dalian University of Technology. Her research interests are Natural Language Processing, Nonlinear Theory and Applications, and Machine Learning.

Xiaokun Zhang received the BSc and M.S. degrees from the Dalian University of Technology, China. He is currently pursuing the PhD degree with the School of Computer Science and Technology, Dalian University of Technology, China. His research interests include data mining and information retrieval, mainly focusing on intelligent recommender systems.

Bo Xu received the B.Sc. and Ph.D. degrees from the Dalian University of Technology, China, in 2011 and 2018, respectively, where he is currently an associated professor with the school of computer science and technology. His current research interests include information retrieval and natural language processing.

Zhihao Yang received a Ph.D. degree in computer science from the Dalian University of Technology, China, in 2008. He is currently a professor at the School of Computer Science and Technology, Dalian University of Technology. He has published more than 20 research papers on topics in biomedical literature data mining. His current research interests include biomedical literature data mining and information retrieval.

Hongfei Lin received a BSc degree from Northeastern Normal University in 1983, an MSc degree from the Dalian University of Technology in 1992, and the Ph.D. degree from Northeastern University in 2000. He is currently a professor at the School of Computer Science and Technology at the Dalian University of Technology. He has published more than 100 research papers in various journals, conferences, and books. His research interests include information retrieval, text mining for biomedical literature, biomedical hypothesis generation, information extraction from huge biomedical resources, and learning-to-rank. He is the director of the Information Retrieval Lab. at the Dalian University of Technology.