DyTed: Disentangling Temporal Invariance and Fluctuations in Dynamic Graph Representation Learning

Kaike Zhang\textsuperscript{1,2}, Qi Cao\textsuperscript{*1}, Gaolin Fang\textsuperscript{4}, Bingbing Xu\textsuperscript{1}, Hongjian Zou\textsuperscript{4}, Huawei Shen\textsuperscript{*1,2}, Xueqi Cheng\textsuperscript{1,3}

\textsuperscript{1} Data Intelligence System Research Center, Institute of Computing Technology, CAS
\textsuperscript{2} University of Chinese Academy of Sciences, Beijing, China
\textsuperscript{3} CAS Key Lab of Network Data Science and Technology, Institute of Computing Technology, CAS
\textsuperscript{4} Tencent Tech.
\{zhangkaike21s, caoqi, xubingbing, shenhuawei, cxq\}@ict.ac.cn
\{glfang, hongjianzou\}@tencent.com

ABSTRACT
Unsupervised representation learning for dynamic graphs has attracted a lot of research attention in recent years. Compared with static graphs, dynamic graphs are the integrative reflection of both the temporal-invariant or stable characteristics of nodes and the dynamic-fluctuate preference changing with time. However, existing dynamic graph representation learning methods generally confound these two types of information into a shared representation space, which may lead to poor explanation, less robustness, and a limited ability when applied to different downstream tasks. Taking the real dynamic graphs of daily capital transactions on Tencent as an example, the learned representation of the state-of-the-art method achieves only 32\% accuracy in predicting temporal-invariant characteristics of users like annual income. In this paper, we introduce a novel temporal invariance-fluctuation disentangled representation learning framework for dynamic graphs, namely DyTed. In particular, we propose a temporal-invariant representation generator and a dynamic-fluctuate representation generator with carefully designed pretext tasks to identify the two types of representations in dynamic graphs. To further enhance the disentanglement or separation, we propose a disentanglement-aware discriminator under an adversarial learning framework. Extensive experiments on Tencent and five commonly used public datasets demonstrate that the different parts of our disentangled representation can achieve state-of-the-art performance on various downstream tasks, as well as be more robust against noise, and is a general framework that can further improve existing methods.

KEYWORDS
Dynamic Graphs, Disentangle Representation Learning, Temporal Invariance, Dynamic Fluctuation

ACM Reference Format:
Kaike Zhang\textsuperscript{1,2}, Qi Cao\textsuperscript{*1}, Gaolin Fang\textsuperscript{4}, Bingbing Xu\textsuperscript{1}, Hongjian Zou\textsuperscript{4}, Huawei Shen\textsuperscript{*1,2}, Xueqi Cheng\textsuperscript{1,3}. 2023. DyTed: Disentangling Temporal Invariance and Fluctuations in Dynamic Graph Representation Learning. In Proceedings of XXXXXXXXXXX (XXX). ACM, New York, NY, USA, 10 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Graph data, which captures the relationships or interactions between entities, is ubiquitous in real world, e.g., social networks [19], citation graphs [39], traffic networks [17], etc. With the abundance of graph data but the expensiveness of training labels, unsupervised graph representation learning has attracted a lot of research attention [3, 6, 43]. It aims to learn a low-dimensional representation of each node in graphs [36], which can benefit various downstream tasks, including node classifications and link predictions. Traditional graph representation learning mainly focuses on static graphs with a fixed set of nodes and edges [10]. However, real-world graphs are generally evolving, where graph structures are dynamically changing with time. How to learn dynamic graph representation becomes an important research problem.

Existing methods for dynamic graph representation learning mainly fall into two categories [28, 37]: continuous-time approaches and discrete-time approaches. The former regards new nodes or edges of dynamic graphs in a streaming manner and integrates the continuous temporal information via point process [31, 41] or temporal random walks [23, 38]. The latter regards the dynamics graphs as a series of snapshots changing at discrete timestamps following a structural-temporal framework that adopts deep graph neural networks [14, 33] to capture structures characteristics and deep temporal neural networks [2, 32] to summarize historical snapshots [27, 28, 37]. Eventually, time-dependent representations have been obtained for each node.

Despite the preliminary success of existing dynamic graph representation learning methods, we found that the performance of the learned representations varies greatly on different downstream tasks. As shown in Figure 1, taking the real dynamic graphs of daily capital transactions between users on Tencent as an example,
the learned user representations are expected to be applied to different downstream tasks, including predicting temporal-invariant or stable characteristics of users (e.g., long-term preference of investment risk, annual income), predicting the dynamic-fluctuate characteristics of users (e.g., the consumption fluctuation of users at a given day), predicting the next user behavior (e.g., the next transaction link), etc. The representation of the state-of-the-art method performs well in predicting the next user behavior (i.e., accuracy 0.92) which is the usually adopted evaluation task, but performs poorly in predicting temporal-invariant or dynamic-fluctuate characteristics (i.e., accuracy 0.32/0.55). There still lacks an effective dynamic graph representation learning method to handle various downstream tasks that require different types of representation.

Recently, disentangled representation learning in various fields has demonstrated that representations that separate the informative factors are an important step toward a better representation learning [20]. Such disentangled representations can be more robust against noise [8], more interpretable [42], and more successfully transferred to downstream tasks [20]. However, due to the non-Euclidean characteristics of graph structure and the complexity of temporal evolution, as well as the lack of guidance for disentanglement, how to learn disentangled dynamic graph representation remains unexplored and challenging.

In this paper, we introduce a novel temporal-invariance-fluctuation disentangled representation learning framework for dynamic graphs, namely DyTed. As pointed out by [20], the unsupervised learning of disentangled representations is impossible without inductive biases. Thus, the most challenging question we need to answer is how to inject our inductive bias into both the models and the data to learn the disentangled representation for dynamic graphs. Without loss of generality, we argue that the dynamic graphs are integrative reflections of both the temporal-invariant or stable characteristics of nodes as well as the dynamic-fluctuate preference changing with time. To effectively identify such two types of information, propose a temporal-invariant representation generator and a dynamic-fluctuate representation generator with carefully designed pretext tasks. To further enhance the disentanglement or separation between temporal-invariant and dynamic-fluctuate representation, we propose a disentanglement-aware discriminator under an adversarial learning framework, which is shown to be equivalent to minimizing the mutual information between temporal-invariant and dynamic-fluctuate representation. As shown in Figure 1, the different parts of the disentangled representation can perform well on various downstream tasks, e.g., gain 0.41 accuracy in predicting users' annual income with temporal-invariant representation, 0.82 accuracy in predicting consumption fluctuation with dynamic-fluctuate representation, and 0.97 accuracy in predicting the next user behavior with the combination of both representations.

Extensive experiments on Tencent and five commonly used public datasets demonstrate that our model achieves state-of-the-art performance on various downstream tasks including node classifications and link predictions. Also, we offer ablation studies to evaluate the importance of each part, conduct noise experiments to demonstrate the model robustness, as well as demonstrate the generality of the proposed framework that can further improve existing methods.

In summary, the main contributions are as follows:

- To the best of our knowledge, we are the first to study and introduce the temporal invariance-fluctuation disentangled representation learning framework for dynamic graphs.
- We propose two representation generators with carefully designed pretext tasks and a disentanglement-aware discriminator under an adversarial learning framework.
- We conduct extensive experiments on real dynamic graphs of daily capital transactions on Tencent, achieving state-of-the-art performance on various downstream tasks.

2 RELATED WORK

This section briefly reviews the research on dynamic graph representation learning and disentangled representation learning.

2.1 Dynamic Graph Representation Learning

Representation learning for dynamic graphs aims to learn time-dependent low-dimensional representations of nodes [36], which can be mainly divided into continuous-time approaches and discrete-time approaches according to the form of dynamic graphs.

Continuous-time approaches treat the dynamics graphs as a flow of nodes or edges annotated with a specific timestamp [22, 30]. To incorporate the temporal information, either temporal random walks are sampled to serve as the context information of nodes [23, 38] or point process are adopted regarding the arrival of nodes/edges as an event [7, 31, 41, 44]. Despite the success of continuous-time approaches, the exact timestamps are sometimes hard to obtain in real scenarios.

1 Tencent is a Chinese internet and technology company, involving social services and financial business (https://www.tencent.com/en-us/)
Another line of research regards the dynamic graphs as a series of snapshots [12, 27, 28], which generally captures the characteristics of these snapshots via the structural and temporal models. Early methods adopt the matrix decomposition to capture the graph structure in each snapshot and regularize the smoothness of the representation of adjacent snapshots [1, 18]. Unfortunately, such matrix decomposition is usually computationally complex [36]. With the development of deep learning, graph neural networks [14] are adopted to capture the structural information while recurrent neural networks or transformer [32] are further utilized to summarize the historical information [27, 28, 34]. To guide the learning of representations, contrastive or predictive tasks are utilized as the pretext task [12, 28, 29, 35], e.g., context-based contrastive learning [28, 29], graph structure reconstruction [4, 12, 37].

However, the above existing methods for dynamic graph representation learning generally confound various factors into a shared representation space, which may lead to less robustness and a limited ability when applied to different downstream tasks.

2.2 Disentangled Representation Learning

Recently, disentangled representation learning has attracted a lot of research attention and achieves great success in many fields [5, 8, 20, 21, 40, 42]. For example, in computer vision, the identity of a face is disentangled from the views or pose information to perform better on image recognition [8, 42]. In natural language generation, the writing style is disentangled from the text content to serve the text-style transfer tasks [13]. In graph neural networks, the factor behind the formation of each edge is disentangled for semi-supervised node classification [21]. As demonstrated in existing research, the disentangle representations are an important step toward a better representation learning [20], which is much closer to human perception and cognition as well as can be more robust, explainable, and transferrable.

However, due to the complexity of graph structure and temporal evolution, how to learn disentangled representation in dynamic graphs remains largely unexplored and challenging.

3 PROBLEM DEFINITION

In this paper, we focus on the dynamic graph that is defined as a series of snapshots $G = \{G^1, G^2, ..., G^T\}$, where $T$ is the total number of snapshots. The snapshot at time $t$, i.e., $G^t = (V^t, E^t)$, is a graph with a node-set $V^t$ and an edge set $E^t \subseteq V^t \times V^t$. We use $\mathcal{A}^t$ to denote the adjacency matrix corresponding to the edge set $E^t$. Note that, as time evolved, there may be both the appearance and disappearance of nodes or edges.

Existing dynamic graph representation learning aims to learn a low-dimensional representation $r^t_v \in \mathbb{R}^d$ for each node $v \in V^t$ at each timestamp $t$, which confound different types of information into a shared representation space.

In this paper, we aim to disentangle the temporal invariance and fluctuation in dynamic graph representation learning, which is formally defined as follows:

**Temporal invariance-fluctuation disentangled representation learning for dynamic graphs.** Given a dynamic graph $G = \{G^1, G^2, ..., G^T\}$, for each node $v \in V$, where $V = \cup_{t=1}^{T} V^t$, we aim to learn a temporal-invariant representation $s_v \in \mathbb{R}^d$ that is independent of time and captures identity or stable characteristics of node $v$, and is independent of time, as well as dynamic-fluctuate representations $d^t_v \in \mathbb{R}^d$, $t = 0, 1, ..., T$, that reflect dynamic-fluctuate preference of node $v$ changing with time. The final disentangled representation $r^t_v \in \mathbb{R}^d$ of node $v$ at timestamp $t$ is the combination of the above two types of representations: $r^t_v = (s_v, d^t_v)$.

4 METHOD

In this section, we present the method, namely DyTed. To effectively identify the two types of information in dynamic graphs, i.e., temporal invariance and dynamic fluctuation, we propose two representation generators with carefully designed pretext tasks and a disentanglement-aware discriminator under an adversarial learning framework. The overview of the framework is shown in Figure 2. In particular, we design a temporal-invariant contrastive learning pretext task to generate temporal-invariant representations of users, shown in Figure 2 (a). Simultaneously, we adopt the structural-proximity contrastive learning pretext task to generate the dynamic-fluctuate representation of users, shown in Figure 2 (b). To further enhance the separation between the temporal-invariant and dynamic-fluctuate representations, we propose a disentanglement-aware discriminator, shown in Figure 2 (c). Next, we introduce each part of the framework in detail.

4.1 Temporal-invariant Representation

The temporal-invariant representation generator aims to identify the stable characteristics of nodes, which is not accessible due to the lack of explicit guidance information. To address this challenge, we consider the fundamental nature behind the temporal-invariant representation, that is, such properties of nodes should be identified as the same in any local temporal clips of the dynamic graphs. Based on this understanding, we design temporal-invariant contrastive learning as the pretext task, together with a temporal-clips sampling and structural-temporal modeling module.

4.1.1 Temporal-clips Sampling. To enforce the generator to extract the temporal-invariant characteristics of nodes in any local temporal clips of dynamic graphs, we first introduce our temporal-clips sampling strategy to construct the positive pair of the temporal-invariant contrastive learning pretext task. It’s worth noting that, if the sampled pair of two temporal clips have a lot of overlapped snapshots, i.e., the two data augmentations in contrastive learning are quite similar, then the generator may exploit this leak to solve the contrastive learning task without really learning the extraction of temporal-invariant representation [9]. As a result, it is critical to adopt a suitable sampling strategy for local temporal clips of dynamic graphs.

In this paper, we design a temporal-clips sampling based on the Bernoulli process as follows:

**Definition 4.1. Bernoulli process for temporal-clips sampling.** Before start, we randomly select a starting timestamp $t$, i.e., $p(t) = 1/T$ where $T$ is the total number of snapshots. Then we include the snapshot of the next timestamp $t+1$ into the temporal clips with probability $p$ and stop the sampling with the probability $1-p$. 
Each decision on whether to include the next timestamp or stop the sampling process is an independent Bernoulli trial.

Following the above sampling process, the probability distribution of the length \( L \) of the sampled temporal clips follows the Geometric distribution, i.e., we have \( Pr(L = k) = p^{k-1}(1 - p) \). This indicates that the longer the temporal clips, the smaller the probability that they are to be sampled.

Then for two sampled temporal clips:

\[
\begin{align*}
\text{TempClip}_{1} &= [G_{t_1}, G_{t_{i_1}}, \ldots, G_{t_{i_k}}], \\
\text{TempClip}_{2} &= [G_{t_1}, G_{t_{j_1}}, \ldots, G_{t_{j_m}}],
\end{align*}
\]

we say that the two temporal clips have overlapped snapshots if there exist \( k_1 \) and \( m_1 \), s.t. \( t_{i_k} = t_{j_1} \leq t_{j_m} \), where \( 0 \leq k_1 \leq k \) and \( 0 \leq m_1 \leq m \).

Let \( X = 1 \) denote that two sampled temporal clips have overlapped snapshots. Otherwise, \( X = 0 \), we have:

**Proposition 4.1.** Following the Bernoulli process for temporal-clips sampling, the probability that any two sampled temporal clips have overlapped snapshots approaches 0 when the total number of snapshots \( T \) approaches infinity.

\[
\lim_{T \to \infty} Pr(X = 1) = 0.
\]

**Proof.** For two sampled temporal clips \( \text{TempClip}_{1} \), \( \text{TempClip}_{2} \), without loss of generality, suppose \( t_{i_1} \leq t_{j_1} \), then the two temporal clips have overlapped snapshots if and only if \( t_{i_1} \leq t_{j_1} \leq t_{i+k} \).

As a result, \( Pr(X = 1) = \sum_{t_{i_1}=1}^{T} \sum_{t_{j_1}=1}^{T} \sum_{k=1}^{k_{i+k}} \frac{1}{T^k} Pr(L = k) = \sum_{t_{i_1}=1}^{T} \sum_{t_{j_1}=1}^{T} \sum_{k=1}^{k_{i+k}} \frac{1}{T^k} \frac{1}{T^k} \frac{1}{T^k} \) when \( T \to \infty \), we have \( \lim_{T \to \infty} Pr(X = 1) = \frac{1}{T^k} \cdot \frac{1}{T^k} = 0 \). 

Compared with the simple sampling strategy that uniformly samples the length of temporal clips, i.e., \( Pr(L = k) = 1/T \) and correspondingly \( \lim_{T \to \infty} Pr(X = 1) = \frac{1}{2} \), our proposed sampling strategy theoretically encourages the non-overlapping of the pair of sampled temporal clips and accordingly encourages the effective learning of the temporal-invariant representation generator under the contrastive learning task.

### 4.1.2 Structural-temporal Model

With the development of deep neural networks, the state-of-the-art backbone model for dynamic graph representation learning generally follows a structural-temporal paradigm. In other words, we adopt deep graph neural networks to capture graph structure characteristics and temporal neural networks to summarize historical snapshots. In this work, as an implementation of our framework, we employ the graph convolutional neural network (GCN) [14] as the structural model and the transformer [32] as the temporal model.

In particular, at each timestamp \( t \), the \( l \)-th layer of GCN takes the adjacency matrix \( A^l \) and the \((l-1)\)-th output embedding \( H^{(l-1)} \) as input. The computational process can be formulated as follows:

\[
H^{(t,l)} = \sigma(\hat{A}^l H^{(t,l-1)}W^{(l)})
\]

where \( \hat{A} \) is the normalization of \( A \): \( \hat{A} = \hat{D}^{-1/2} \hat{A} \hat{D}^{-1/2} \), \( \hat{A} = A + I \), \( \hat{D} = diag(\sum_i \hat{A}_{ii}) \), and the \( \sigma \) is the activation function.

Then we take the output \( H^{(t,L)} \) of GCN in the order of the timestamps as the input of the temporal model. To obtain the representation of the whole temporal clips, we added a trainable vector \( h^0 \) to serve as the output representation. To make the self-attention sensitive to temporal information, we encoded the position of each timestamps[32] together with the multi-head self-attention to capture temporal dependence.

---

**Figure 2:** Overview of DyTedd. (a) Temporal-invariant Representation Generator, which is composed of three parts: temporal-clips sampling, structural-temporal modeling, and temporal-invariant contrastive learning as pretext task. (b) Dynamic-fluctuate representation Generator, composed of structural-temporal modeling and structure-proximity contrastive learning as pretext task. (c) Discriminator: a disentanglement-aware discriminator that enhances the separation between temporal-invariant and dynamic-fluctuate representation.
\[
Z_h = \text{softmax}(\frac{\text{OW}_h^T(K\text{W}_h^K)^T}{\sqrt{d_k}})\text{W}_h^o, \ h = 0, \ldots, H, \tag{4}
\]

\[
Z = \text{concat}(Z_0, Z_1, \ldots, Z_H),
\]

where \(Z_h\) is the output of \(h\)-th head, \(d_k\) is the dimension of the hidden layer, \(H\) is the number of heads, \(K = Q = V = \{H^0, H^{(i, L)}\}\) and \(\text{W}_h^k \in \mathbb{R}^{F \times d_k}, \text{W}_h^o \in \mathbb{R}^{F \times d_o}\) are the corresponding parameters of attention block.

We take the multi-head self-attention layer and the feed-forward network layer as the backbone of the block, and there is layer normalization and residual connection after each layer. The output after stacking several transformer blocks is formulated as follows:

\[
\{s, M\} = \text{TransBlock}(\ldots(\text{TransBlock}(\{H^0, H^{(i, L)}\}))). \tag{5}
\]

The representation of node \(v\) in the temporal clip is denoted as \(s_v\).

4.1.3 Temporal-invariant Contrastive Learning. Given two sampled temporal clips, denoted as TemClip1 = \{\(G_t, G_{t+1}, \ldots, G_{t+k}\)\} and TemClip2 = \{\(G_{t'}, G_{t'+1}, \ldots, G_{t'+k}\)\}, we take the representation of node \(v\) for TemClip1 and TemClip2 as the positive pair of contrastive learning, i.e., \((s_v, s_v')\). We take the representation of node \(v\) for TemClip1 as the negative pair, i.e., \((s_v, s_v^1)\).

To optimize the temporal-invariant representation generator, we used InfoNCE [25] as the contrastive loss to separate the positive pair and negative pair. Such contrastive loss can maximize the mutual information between representations in positive pairs while minimizing the mutual information between representations in negative pairs, ensuring that the extracted representations of the same node in different temporal clips are similar. The loss function is formalized as follows:

\[
L_s = -\mathbb{E}[\log \frac{\exp(\text{sim}(s_v, s_v'))}{\exp(\text{sim}(s_v, s_v^1) + \sum_{v \neq u} \exp(\text{sim}(s_v, s_u^1)))}], \tag{6}
\]

where \(\text{sim}(\cdot)\) is the similarity function.

The final temporal-invariant representation \(s_v\) for node \(v\) is obtained when the temporal clip is equal to the entire dynamic graphs.

4.2 Dynamic-fluctuate Representation

Similar to the temporal-invariant representation generator, we used the structure of GCN plus transformer block in the dynamic-fluctuate representation generator. Note that, the model parameters in the two representation generators are not shared. The temporal-invariant output after the structural-temporal model for user \(v\) at timestamp \(t\) is denoted as \(d_v^t\).

Considering that the graph structure at a given timestamp is the integrative reflection of both the temporal-invariant and dynamic-fluctuate representation, we adopt the commonly used structure-proximity contrastive learning as the pretext task with the combination of temporal-invariant representation \(s_v\) and dynamic-fluctuate representation \(d_v^t\).

In particular, let \(r_v^t = (s_v, d_v^t)\), we take two neighboring nodes \(v\) and \(u\) as the positive pair and two non-neighboring nodes \(v\) and \(w\) as the negative pair. The loss function is:

\[
L_g = -\mathbb{E}[\log \frac{\exp(\text{sim}(r_v^t, r_u^t))}{\exp(\text{sim}(r_v^t, r_w^t) + \sum_{(v,w) \notin E} \exp(\text{sim}(r_v^t, r_u^t)))}]. \tag{7}
\]

4.3 Disentanglement-aware Discriminator

After the temporal-invariant representation generator and dynamic-fluctuate representation generator, we get two parts of representations. To enhance the disentanglement between these two types of information, we propose an adversarial learning framework.

Image that if the temporal-invariant representation \(s_v\) and the dynamic-fluctuate representation \(d_v^t\) of node \(v\) are not well disentangled, in other words, the two representations have some overlap information, then it should easy to tell that \((s_v, d_v^t)\) are coming from the same node while \((s_u, d_u^t)\) are not.

Based on this intuition, we use a simple multi-layer perceptron (MLP) as the disentanglement-aware discriminator \(D\), whose goal is to distinguish correctly whether the temporal-invariant representation and the dynamic-fluctuate representation are coming from the same node. Taking the general framework of generative adversarial networks in f-GAN [24], the adversarial learning between the two representation generators and the discriminator can be written as:

\[
\min_G \max_D V(G, D) = \min_G \max_D \frac{1}{T} \sum_{t=0}^{T} \left[T(D(r_v^t) + f(T(D(z_v^t))))\right], \tag{8}
\]

where \(r_v^t = (s_v, d_v^t)\) is the true sample, \(z_v^t = (s_u, d_u^t), v \neq u\) is the false sample. When \(T(D) = \log(2D)\) and \(f(T) = -\log(2 - e^T)\), the above equivalent to the original GAN [11].

PROPOSITION 4.2. Taking \(T(D) = D, f(T) = e^{T-1}\), denoting the temporal-invariant representation \(s \sim p_s(s)\) and the dynamic-fluctuate representation \(d \sim p_d(d)\), given the optimal discriminator \(D\), then the representation generator is equivalent to minimize the mutual information between temporal-invariant and dynamic-fluctuate representation \(I(s, d)\).

Proof. According to [24], when take \(T(D) = D, f(T) = e^{T-1}\), given the optimal discriminator, the generator is equivalent to minimize the KL divergence between the true data distribution and the generated data distribution, i.e., \(D_{KL}(p_{data}||p_g)\). In our model, \(p_{data} = p(s, d)\) while \(p_g = p(s)p(d)\). In other words, given the optimal discriminator, the representation generators are minimizing \(D_{KL}(p(s, d)||p(s)p(d)) = I(s, d)\). \(\Box\)

To sum up, under the guidance of the disentanglement-aware discriminator, the temporal-invariant and dynamic-fluctuate representation generators are devoted to generating representations that do not contain mutual information as much as possible, achieving the disentanglement of the two types of information.

Finally, the loss function of the discriminator is:

\[
\text{Loss}(D) = -V(G, D) \tag{9}
\]

and the loss function of the generators is:

\[
\text{Loss}(G) = L_s + \alpha L_g + \beta V(G, D) + \frac{1}{2}\lambda \|w\|^2_2 \tag{10}
\]

where \(\alpha, \beta, \lambda\) and \(\lambda\) are the hyperparameters.
We first introduce the datasets and baselines we used as well as the implementation details. The five commonly used datasets of dynamic graphs, including the communication network Enron [15] and UCI [26], Routing network AS733 [16], Citation network HepTh and HepPh [16], together with a real Financial dataset of dynamic graphs on Tencent. The statistics of all datasets are shown in Table 1.

| DATASET  | # Node | # Edge  | # Time steps |
|----------|--------|---------|--------------|
| Enron    | 143    | 22,784  | 16           |
| UCI      | 1,809  | 56,459  | 13           |
| AS733    | 4,648  | 532,230 | 30           |
| HepTh    | 7,576  | 196,463 | 23           |
| HepPh    | 10,404 | 339,556 | 20           |
| Tencent  | 11,623 | 102,464 | 30           |

5 EXPERIMENTS

In this section, we conduct extensive experiments to answer the following research questions (RQs).

- **RQ1:** How does DyTed perform on downstream tasks?
- **RQ2:** Is there any additional benefit of the disentangle representation?
- **RQ3:** Can DyTed framework be applied to existing methods?

5.1 Experimental Setup

We first introduce the datasets and baselines we used as well as the implementation details.

5.1.1 Datasets. In order to evaluate our proposed method, we adopt five commonly used datasets of dynamic graphs, including the communication network Enron [15] and UCI [26], Routing network AS733 [16], Citation network HepTh and HepPh [16], together with a real Financial dataset of dynamic graphs on Tencent. The statistics of all datasets are shown in Table 1.

- **Enron** [15] is an email exchange network, in which each node represents an employee of Enron company, and each edge represents the email interaction between the two employees. We follow the setting in [12] to obtain 16 snapshots.
- **UCI** [26] is a communication network, where links represent messages sent between users on an online social network.
- **AS733** [16] is a routing network. The dataset contains a total of 733 snapshots. We follow the setting in [37] and extract 30 snapshots. To enable all baselines to conduct experiments, we restricted the dataset not to include deleted nodes.
- **HepTh** [16] is a citation network related to high-energy physics theory. Each node represents a paper, and each link represents a reference relationship. We extract 92 months of data from this dataset, forming 23 snapshots. For each reference edge, we set it to exist since the occurrence of the reference.
- **HepPh** [16] is a citation network related to high-energy physics phenomenology. We extract 60 months of data from this dataset, forming 20 snapshots. The other settings are similar to HepTh.
- **Tencent** is the real dynamic graphs of daily capital transactions between users on Tencent after eliminating sensitive information. Each node represents a customer or business, and each edge represents the transaction. The data ranges from April 1, 2020, to April 30, 2020, with 30 snapshots. In addition, each node has five node labels that reflect the temporal-invariant or dynamic-fluctuate characteristics of users, i.e., annual income grade with five classes, age with five classes, asset with five classes, financing risk with three classes, and consumption fluctuation with binary class.

5.1.2 Baseline. We compare the proposed method DyTed with the following five baselines. We select GCN [14] as a representative of static graph representation learning methods, which apply GCN on each snapshot and pooling representation of all snapshots as the final representation of nodes. We further replace the pooling with GRU to capture the temporal information, denoted as GRU-GCN. In addition, we choose three state-of-the-art discrete-time approaches with contrastive or prediction pretext task, i.e., DySAT [28] with the contrastive task, EvolveGCN [27] and HTGN [37] with the predictive task.

5.1.3 Implementation Details. For all baselines and our DyTed, the representation model is trained on the snapshots \(\{G^1, G^2, \ldots, G^T\}\). The total dimension of the representation is set according to the scale of datasets, i.e., 16 for Enron, 32 for UCI, and 64 for other datasets. Note that, we ensure that the dimensions of the representation of each method are equal on the same dataset. The training process stops immediately as long as the loss of pretext task doesn’t decline for 50 consecutive iterations. For our method, we set the sampling probability \(p\) to 0.75 and the minimum sample length to 3. For the hyperparameter in the loss of generators, we set \(\alpha\) to 1.0, \(\beta\) to 0.3 and \(\lambda\) to 5e – 7. Our implementation code is provided through an anonymous link 2.

5.2 Performance on Downstream Tasks

In this section, we answer RQ1, i.e., How does DyTed perform on different downstream tasks? We adopt node classification with temporal-invariant labels, node classification with dynamic-fluctuate labels, and link prediction for the next snapshot as the diverse evaluation tasks. For clarity, we use some abbreviations to denote the different parts of DyTed’s representation as follows:

- **DyTed:** the combination of temporal-invariant and dynamic-fluctuate representation, i.e., \(r_t^e = (s_o, d_t^e), t = 1, 2, \ldots, T\).
- **DyTed-Fluctuate:** the dynamic-fluctuate representation \(d_t^e, t = 1, 2, \ldots, T\).
- **DyTed-Invariant:** the temporal-invariant representation \(s_o\).

5.2.1 Node Classification with Temporal-invariant Labels. In this section, we evaluate the performance of each representation method on node classification. We select GCN [14] as a representative of static graph representation learning methods GCN outperforms other methods, indicating that the existing dynamic graphs representation can not capture the identity or stable characteristics contained in the series of snapshots. As for our disentangled representation, we outperform all baselines. In particular, the temporal-invariant representation, i.e., DyTed-Invariant, earns an average gain of 8.41% in AUC compared to the strongest baseline.

2https://anonymous.4open.science/r/DyTed-pytorch-A164/
Table 2: Node classification with temporal-invariant labels

| Model       | Annual Income | Age | Assets | Financing Risk |
|-------------|---------------|-----|--------|---------------|
|             | micro-F1      | macro-F1 | micro-F1 | macro-F1 | micro-F1      | macro-F1 | micro-F1 | macro-F1 | micro-F1 | macro-F1 | micro-F1 | macro-F1 |
| GCN         | 0.3581        | 0.1719    | 0.1727  | 0.0760   | 0.0957        | 0.0638    | 0.3053  | 0.1785   |           |           |           |           |
| GRU-GCN     | 0.1607        | 0.0954    | 0.0467  | 0.0220   | 0.1093        | 0.0483    | 0.2771  | 0.1929   |           |           |           |           |
| EvolveGCN   | 0.1680        | 0.0947    | 0.0123  | 0.0072   | 0.0739        | 0.0485    | 0.2656  | 0.1412   |           |           |           |           |
| HTGN        | 0.2496        | 0.1310    | 0.0056  | 0.0037   | 0.0043        | 0.0031    | 0.2629  | 0.1768   |           |           |           |           |
| DySAT       | 0.3247        | 0.2044    | 0.1381  | 0.0610   | 0.1286        | 0.0606    | 0.3873  | 0.1879   |           |           |           |           |
| DyTed-Fluctuate | 0.3230     | 0.2397    | 0.2183  | 0.1156   | 0.1600        | 0.0917    | 0.3153  | 0.2338   |           |           |           |           |
| DyTed       | 0.3687        | 0.2629    | 0.2729  | 0.2175   | 0.2175        | 0.1097    | 0.3984  | 0.2730   |           |           |           |           |
| DyTed-Invariant | 0.4069     | 0.3224    | 0.2866  | 0.1839   | 0.2276        | 0.1746    | 0.3914  | 0.3503   |           |           |           |           |

Table 3: Node classification with dynamic-fluctuate labels

| Model | micro-F1 | macro-F1 |
|-------|----------|----------|
| GCN   | 0.4725   | 0.3895   |
| GRU-GCN | 0.5450   | 0.5723   |
| DySAT | 0.5500   | 0.5763   |
| HTGN  | 0.5761   | 0.4776   |
| EvolveGCN | 0.7697   | 0.7678   |
| DyTed-Invariant | 0.5477   | 0.5456   |
| DyTed | 0.7817   | 0.7812   |
| DyTed-Fluctuate | 0.8197   | 0.8191   |

5.2.2 Node Classification with Dynamic-fluctuate Labels. In this subsection, we take the node classification with a dynamic-fluctuate label on Tencent as the downstream evaluation task. Specifically, we take the consumption fluctuation of users as the label, reflecting their occasional consumption preference. Similarly, we employ a single-layer perceptron as the downstream classifier, and separate the train, validation, test according to $0.2 : 0.2 : 0.6$.

From Table 3, we can see that the DyTed-Fluctuate achieves the state-of-the-art performance with 0.8197 micro-F1, while most baselines could only achieve 0.55 micro-F1. It’s worth noting that, the DyTed-Fluctuate performs better than DyTed which combines our two types of representation, which indicate that the irrelevant information may additionally introduce noise and hurt the performance on downstream tasks.

5.2.3 Link Prediction. We use representations of two nodes at snapshot $G^T$ to predict whether they are connected at the snapshot $G^{T+1}$, which is a commonly adopted evaluation task for dynamic graph representation learning. We follow the evaluation method used in [28]: logistic regression as the classifier and the hamard product of representations as the input.

Table 4 shows that the proposed model DyTed achieves the best or comparable performance on all datasets. Since many links in HepPh, HepTh, and AS733 are persistent, i.e., the same edge exists in multiple consecutive snapshots, link prediction on such datasets is too easy to show the difference between methods. In contrast, graphs on Enron and UCI change dramatically with few persistent edges, where our model achieves much better performance. Note that, this task requires both the temporal-invariant and dynamic-fluctuate characteristics of users, making the combination of both representations (DyTed) outperforms any single type of representation (DyTed-Invariant or DyTed-Fluctuate).

5.3 Analysis of Disentangled Representation

This section answers RQ2, i.e., is there any additional benefit of the disentangle representation? We conduct the ablation study and illustrate the benefits of disentanglement from the perspective of robustness and training resources required by downstream tasks. Due to space limitations, we only show the results on UCI or Tencent.

5.3.1 Ablation Study. We verify the role of the main modules of DyTed through an ablation study. We name different versions of DyTed as follows:

- **DyTed-w/o-DA**: DyTed without dynamic-fluctuate representation generator and disentanglement-aware discriminator.
- **DyTed-w/o-TA**: DyTed without temporal-invariant representation generator and disentanglement-aware discriminator.
- **DyTed-w/o-A**: DyTed without disentanglement-aware discriminator (adversarial learning).
- **DyTed-ud**: Replace the Bernoulli process for temporal-clips sampling with a uniform sampling of the length of temporal clips.

As shown in Figure 3, we find it challenging to achieve good results using a temporal-invariant generator alone. The performance...
Table 4: Link prediction on next snapshot

| Model         | Enron AUC | Enron AP | UCI AUC | UCI AP | AS733 AUC | AS733 AP | HepTh AUC | HepTh AP | HepPh AUC | HepPh AP |
|---------------|-----------|----------|---------|--------|-----------|----------|-----------|----------|-----------|----------|
| GCN           | 0.7719    | 0.7622   | 0.6824  | 0.6800 | 0.8104    | 0.8133   | 0.8466    | 0.8521   | 0.8422    | 0.8438   |
| GRU-GCN       | 0.7763    | 0.7751   | 0.7518  | 0.7219 | 0.8223    | 0.8244   | 0.8184    | 0.8261   | 0.8618    | 0.8528   |
| EvolveGCN     | 0.7659    | 0.7681   | 0.7632  | 0.7828 | 0.9376    | 0.9364   | 0.7373    | 0.6651   | 0.9347    | 0.9483   |
| HTGN          | 0.8018    | 0.8138   | 0.7390  | 0.6604 | 0.8768    | 0.8731   | 0.9244    | 0.9163   | 0.9473    | 0.9397   |
| DySAT         | 0.8235    | 0.7760   | 0.7352  | 0.8158 | 0.9499    | 0.9584   | 0.8131    | 0.7547   | 0.9219    | 0.8779   |
| DyTed-Invariant | 0.7517  | 0.7112   | 0.7809  | 0.7699 | 0.8527    | 0.8183   | 0.8752    | 0.8360   | 0.9173    | 0.8940   |
| DyTed-Fluctuate | 0.7984 | 0.7481   | 0.7362  | 0.7103 | 0.8662    | 0.8438   | 0.9243    | 0.8976   | 0.9408    | 0.9215   |
| DyTed         | 0.8869    | 0.8766   | 0.8642  | 0.8693 | 0.9365    | 0.9421   | 0.9569    | 0.9587   | 0.9701    | 0.9700   |

5.3.2 Evaluation of Disentangling Degree. To verify whether the disentanglement-aware discriminator can further disentangle the dynamic-fluctuate representation and the temporal-invariant representation, we use logistic regression to distinguish whether the two types of representations come from the same node (the same idea as the discriminator in Section 4.3). The closer the AUC is to 0.5, the closer the model is to random guesses, i.e., the model cannot judge whether the two types of representations come from the same node. As a result, we use $|AUC - 0.5|$ as the measurement and take the initial and final representation of DyTed and DyTed-w/o-A as inputs. As shown in the left part of Figure 4, the final representation in DyTed achieves the highest disentangling degree of the two parts of representations, i.e., the $|AUC - 0.5|$-score is lower. We also calculate the mutual information between temporal-invariant representation and dynamic-fluctuate representation, in which the same conclusion can be obtained.

5.3.3 Robustness of DyTed. In this section, to evaluate the robustness of DyTed, we add noise to the original data by randomly adding or deleting edges for each snapshot. The ratio $r\%$ of noise refers to the proportion of added or deleted edges to existing edges, which increased from 0% to 50% in steps of 10%. Then we train representation models on the noisy dynamic graphs and test the performance of the obtained representation on the downstream link prediction task without noise. The result in Figure 5 shows that DyTed has strong robustness, whose performance is greatly improved compared with SOTA models.

5.3.4 Demanded Training Resources on Downstream Tasks. This section verifies whether the more disentangled representation requires fewer downstream training resources. We take the multilayer perceptron as the classifier and adjust the number of layers of the classifier to vary the complexity of the downstream model. The left part of Figure 6 shows that when there is a one-layer classifier, the performance of DyTed-w/o-A is much lower than that of DyTed. When the number of layers of the classifier increases to two, the performance of DyTed-w/o-A is greatly improved, while DyTed improves slightly. In other words, it is enough to use a simpler classifier for the downstream tasks with our disentangled representation of DyTed. However, the less disentangled representation of DyTed-w/o-A needs to use a more complex model to achieve the effect of DyTed barely.

In addition, the right part of Figure 6 shows that DyTed can also achieve better results with a low proportion of training samples, while DyTed-w/o-A needs more training samples to catch up with DyTed, demonstrating that a more disentangled representation can even reduce the demand of downstream labels.
We further apply the DyTed framework of disentangling temporal-invariant and dynamic-fluctuate representations to existing methods to show its generality. We select GRU-GCN and EvolveGCN as representatives of existing methods, and the results are shown in Table 5. The DyTed framework further improves the performance of existing methods on all three downstream tasks.

6 CONCLUSION

In this paper, we introduce a novel temporal invariance-fluctuation disentangled representation learning framework for dynamic graphs, namely DyTed. We propose two representation generators with the temporal-invariant and structure-proximity contrastive learning pretext tasks. Moreover, we design a disentanglement-aware discriminator under an adversarial learning framework to further enhance the disentanglement of the two types of representations. Various experiments demonstrate the effectiveness, robustness, and generality of our proposed framework.

REFERENCES

[1] Nahla Mohamed Ahmed, Ling Chen, Yulong Wang, Bin Li, Yun Li, and Wei Liu. 2018. DeepEye: Link prediction in dynamic networks based on non-negative matrix factorization. Big Data Mining and Analytics (2018), 19–33.

[2] Shaojie Bai, Zico Kolter, and Vladlen Koltun. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. In arXiv preprint arXiv:1803.01271.

[3] Yunsheng Bai, Hao Ding, Yang Qiao, Agustin Marinovic, Ken Gu, Ting Chen, and Yong Kong. 2022. Dysat: Deep neural representation learning on dynamic graphs via self-attention and contrastive learning. In Proceedings of the 45th ACM SIGIR Conference on Research and Development in Information Retrieval. 267–277.

[4] Lei Cai, Zhengzheng Chen, Chen Lu, Jiaping Guo, Jingchao Ni, Ding Li, and Haidong Chen. 2021. Structural temporal graph neural networks for anomaly detection in dynamic graphs. In Proceedings of the 30th ACM International Conference on Information and Knowledge Management. 3747–3756.

[5] Jingxiao Cao, Xuefan Liu, Xin Cong, Jie Li, Tingwen Liu, and Bin Wang. 2022. DenseDTR: Learning Disentangled Representations for Cross-Domain Recommendation. In Proceedings of the 45th ACM International SIGIR Conference on Research and Development in Information Retrieval. 891–900.

[6] Xiaoou Chang, Wei Wu, and Qiongxiu Xu. 2015. GraRep: Learning Graph Representations based on Global Structural Information. In Proceedings of the 28th ACM on Conference on Information and Knowledge Management. 891–900.

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Proceedings of the 30th Conference on Neural Information Processing Systems. 4114–4124.

[8] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. In Proceedings of the 5th International Conference on Learning Representations.

[9] Yanbang Wang, Pan Li, Chongyang Bai, V Subrahmanian, and Jure Leskovec. 2020. Dynamic representation learning for dynamic social interaction. In Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 1159–1169.

[10] Guoqiang Xue, Ming Zhong, Jianxin Li, Jia Chen, Chengshuai Zhai, and Ruochen Kong. 2022. Dynamic network embedding survey. Neurocomputing (2022), 212–223.
[37] Menglin Yang, Min Zhou, Marcus Kalander, Zengfeng Huang, and Irwin King. 2021. Discrete-time Temporal Network Embedding via Implicit Hierarchical Learning in Hyperbolic Space. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 1975–1985.

[38] Wenchao Yu, Wei Cheng, Charu C Aggarwal, Kai Zhang, Haifeng Chen, and Wei Wang. 2018. Netwalk: A flexible deep embedding approach for anomaly detection in dynamic networks. In Proceedings of the 24th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2672–2681.

[39] Yizhou Zhang, Guojie Song, Lun Du, Shuwen Yang, and Yilun Jin. 2019. DANE: Domain Adaptive Network Embedding. In Proceedings of the 28th International Joint Conference on Artificial Intelligence. 4362–4368.

[40] Tianxiang Zhao, Xiang Zhang, and Suhang Wang. 2022. Exploring Edge Disentanglement for Node Classification. In Proceedings of the 32nd ACM Web Conference. 1028–1036.

[41] Lekui Zhou, Yang Yang, Xiang Ren, Fei Wu, and Yueting Zhuang. 2018. Dynamic network embedding by modeling triadic closure process. In Proceedings of the 32nd Proceedings of the AAAI Conference on Artificial Intelligence.

[42] Xinqi Zhu, Chang Xu, and Dacheng Tao. 2021. Where and What? Examining Interpretable Disentangled Representations. In Proceedings of the 31st Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 5861–5870.

[43] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2020. Deep graph contrastive representation learning. In arXiv preprint arXiv:2006.04131.

[44] Yuan Zuo, Guannan Liu, Hao Lin, Jia Guo, Xiaoqian Hu, and Junjie Wu. 2018. Embedding temporal network via neighborhood formation. In Proceedings of the 24th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2857–2866.