TE-MDG: TRANSFER ENTROPY FOR MUTATION-AWARE DYNAMIC GRAPH REPRESENTATION LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Dynamic graph representation learning aims to capture the temporal patterns of evolving graphs, which is critical for time-aware applications like social networks and recommendation systems. Most of existing methods typically model long-term dynamics by aggregating the nodes' historical behaviors. However, in real-world dynamic graphs, node features and their interactions often evolve with two types of mutations: Rational Mutations that are consistent with stable evolution patterns and Occasional Mutations that are incidental and irrelevant. Recent advances fail to distinguish different types of mutations due to the absence of a clear standard, causing the meaningful information in mutations to be weakened and hard to extract. To this end, we propose a **T**ransfer **E**ntropy for **M**utation-aware **D**ynamic **G**raph Representation Learning framework (**TE-MDG**), which selectively incorporates different types of mutations into node representations. Specifically, TE-MDG consists of two key components: (i) Evolution Pattern Decoupling, which learns a set of temporal prototypes that capture the stable evolution patterns and serve as a guiding standard for distinguishing rational mutations from occasional ones; and (ii) Mutation Filtering, which uses transfer entropy to measure and strengthen the information gain from temporal prototypes to mutations, thereby encouraging the model to retain rational mutations while filtering out occasional ones. Extensive experiments on multiple real-world benchmark datasets demonstrate that TE-MDG is more effective than state-of-the-art baselines. The code and datasets are anonymously available at https://anonymous.4open.science/r/TE_MDG.

1 INTRODUCTION

Dynamic graphs are widely observed in real-world scenarios (Nguyen et al., 2018; Hamilton et al., 2018; Goyal et al., 2018), such as the continuous evolution of user relationships in social networks (Wu et al., 2019) and the dynamic updates of user-item interactions on e-commerce platforms (Zhou et al., 2018; Pareja et al., 2019). These graphs are characterized by the continuous appearance and disappearance of nodes and edges, together with the temporal variations in their attributes, resulting in highly complex evolving patterns. Dynamic graph representation learning (Zuo et al., 2018; Barros et al., 2021; ENNADIR et al., 2025) aims to learn time-aware node embeddings that effectively capture these evolutionary patterns. Such representations are essential for downstream tasks, including link prediction and node classification.

Existing studies on modeling long-term behavioral evolution of nodes can be broadly categorized into two types: (i) Memory-based Temporal Modeling (Kumar et al., 2019; Trivedi et al., 2019; Jin et al., 2025), which directly models node dynamics by memory states updated with interaction messages to capture long-term evolution patterns. Typically, TGN (Rossi et al., 2020) maintains an updatable memory vector for each node. The memory is updated at each interaction using a message function and a Gated Recurrent Unit. Temporal attention is then applied to aggregate neighbor information and generate time-aware node embeddings. (ii) Sequence-based Temporal Modeling (Xu et al., 2020; Sankar et al., 2020), which directly models historical interaction sequences by self-attention mechanisms to capture rich contextual dependencies and characterize long-term evolution patterns. For instance, DyGFormer (Yu et al., 2023) partitions the historical interaction sequence of nodes into multiple time patches, captures co-occurrence relationships between node pairs via co-occurrence encoding, and employs a Transformer (Vaswani et al., 2017) architecture to model the patch sequence, thereby effectively capturing long-term temporal dynamics.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

However, in real-world scenarios, node behaviors often exhibit mutations such as changes in node interactions and features. In dynamic graphs, there commonly exist stable evolution patterns shared across time and nodes, which characterize the main trends of the system evolution as a guiding standard, thereby naturally distinguishing mutations. **Rational Mutations** that are consistent with these stable evolution patterns, such as interaction shifts that naturally extend from one's prior interests. In contrast, **Occasional Mutations** are mutations that do not follow stable evolution patterns, such as behavior changes triggered by occasional factors unrelated to interests.

Although existing methods have made some progress in modeling the long-term behavioral evolution, they suffer from two key limitations: **(i)** Lack a clear standard to distinguish different types of mutations. Specifically, existing methods indiscriminately encode both rational and occasional mutations into node representations, and without a guiding standard to distinguish them, this inevitably degrades the quality of the learned representations. **(ii)** Inability to selectively capture meaningful information from mutations. Existing methods fail to evaluate which information in mutations is beneficial for node representations. Lacking this selectivity, the model inevitably incorporates irrelevant information, thereby weakening the contribution of rational mutations.

In this paper, we propose **T**ransfer **E**ntropy for **M**utation-aware **D**ynamic **G**raph Representation Learning (**TE-MDG**), a novel framework for accurately modeling long-term behavioral evolution of nodes while implicitly distinguishing the different effects of mutations through selective modeling. It consists of two components: **(i)** Evolution Pattern Decoupling, which learns a set of temporal prototypes to decouple stable evolution patterns from dynamic graphs as a standard for distinguishing mutations; **(ii)** Mutation Filtering, which employs transfer entropy (TE) to measure the information gain between prototypes and local behaviors, and applies the gain to enhance the guidance of the stable evolution patterns on mutations. This enables the model to capture meaningful information from rational mutations. Our main contributions are summarized as follows:

- We propose TE-MDG, a novel framework that decouples stable evolution patterns from dynamic graphs and employs them to distinguish rational and occasional mutations, thereby enabling selective preservation of meaningful information for node representation learning.
- We introduce transfer entropy to measure the information gain from prototypes to local behaviors, and leverage this gain to strengthen the guidance of stable evolution patterns on mutations, which enhances the contribution of rational mutations while suppressing occasional ones.
- Extensive experiments on multiple real-world benchmark datasets demonstrate that TE-MDG outperforms existing state-of-the-art baselines on tasks including link prediction and node classification, validating the empirical superiority of TE-MDG.

2 Preliminary Study

In this section, we introduce the concept of mutations in continuous-time dynamic graph representation learning through a specific case, and then conduct a preliminary experiment to investigate whether mutations exist and how they influence subsequent evolution.

2.1 Mutations in Dynamic Graphs

We consider a continuous-time dynamic graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes and each node $u \in \mathcal{V}$ is associated with a feature vector $\mathbf{x}_u \in \mathbb{R}^{d_n}$. The edge set $\mathcal{E} = \{(u, v, t, \mathbf{e}_{uv}) | u, v \in \mathcal{V}, t \in \mathbb{R}^+, \mathbf{e}_{uv} \in \mathbb{R}^{d_e}\}$ is a timestamp-ordered sequence of events, where u, v are the interacting nodes, t is the timestamp, and $\mathbf{e}_{uv}$ represents the edge feature. The goal of continuous-time dynamic graph representation learning is to learn a time-dependent embedding $\mathbf{z}_u(t) \in \mathbb{R}^d$ for each node u at timestamp t, which captures historical interaction structures and temporal evolution patterns, and supports downstream tasks such as link prediction and node classification.

Existing methods typically assume that node behaviors evolve smoothly over time. However, in real-world scenarios, node behaviors often exhibit short-term local shifts during long-term evolution. Unlike smooth evolution, such shifts may either align with or deviate from stable evolution patterns in dynamic graph. To characterize such shifts , we introduce the concept of ***mutations*** δ. Based on its different effects relative to the stable evolution pattern, we categorize mutations into two types: ***Rational Mutations*** δ_{rat} and ***Occasional Mutations*** δ_{oca}.

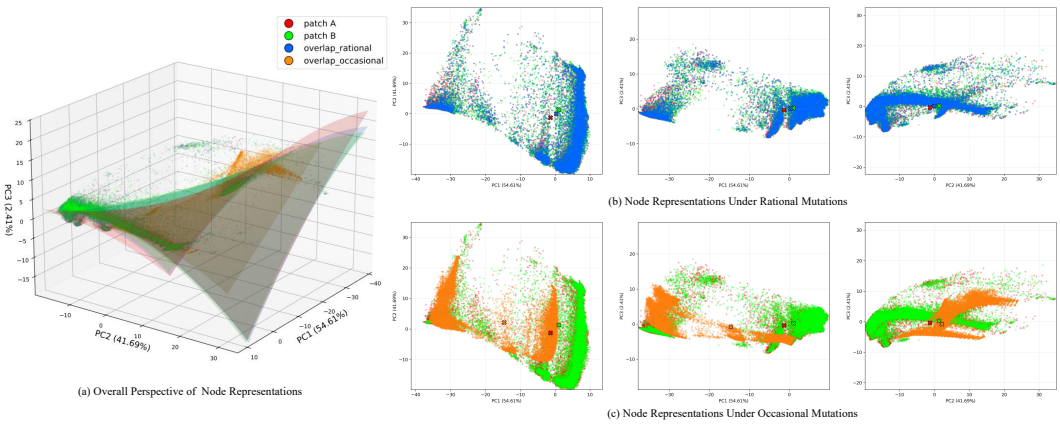


Figure 1: PCA visualization of node representations under rational and occasional mutations. The left panel shows the 3D embedding space, while the right panels present 2D projections.

To illustrate, take user behavior on a short-video platform as an example. A user who prefers the topic of "*home cooking*" may temporarily shift attention to "*air fryer recipes*" as such content becomes popular. Although this appears as a local shift, it remains semantically and temporally consistent with prior interests, thus constituting a rational mutation. In contrast, if the same account suddenly interacts with several "*gaming livestreams*" due to incidental usage by others, such behavior lacks temporal dependency on stable evolution patterns and constitutes an occasional mutation.

2.2 Analysis of Mutations in Representation Space

Building upon the above intuition, we conduct an analysis in the representation space to characterize how different types of mutations perform in node embeddings. For the historical interaction sequence $H_u(t) = \{(u, v_i, t_i, \mathbf{e}_{uv_i}) \mid t_i < t\}$ of node u, we partition it into consecutive temporal segments and select two adjacent ones as patch$_A$ and patch$_B$, representing adjacent stages of temporal evolution. To capture continuous dynamics, we construct an intermediate overlap region that shares the tail of patch$_A$ and the head of patch$_B$. With this setup, we observe the distribution of node representations across patch$_A$, the overlap region, and patch$_B$ to investigate how mutations influence the evolution of node in representation space. The results are shown in Figure 1.

Obs. 1. Rational mutations maintain structural consistency with stable evolution patterns in dynamic graphs. As shown in Fig. 1(b), the representations in the overlap region remain well aligned with those in patch$_A$ and patch$_B$, forming a smooth and continuous transition in the embedding space. This indicates that rational mutations are structurally consistent with the underlying stable evolution patterns in dynamic graphs and conform to the long-term evolution trend.

Obs. 2. Occasional mutations introduce structural inconsistency in the representation space, breaking the continuity of temporal evolution. As shown in Fig. 1(c), the representation of the overlap region become noticeably dispersed and exhibit a significant distributional shift away from those in patch$_A$ and patch$_B$. This suggests that occasional mutations break the continuous temporal evolution of node states, leading to unreliable modeling of evolutionary trends.

Obs. 3. Interestingly, although occasional mutations introduce clear shifts in the overlap region, they do not significantly affect the representations of the subsequent patch$_B$, which remain stable and consistent with the original evolution trend. This suggests that existing methods exhibit a certain degree of robustness to such mutations, as they tend to weaken the contribution of mutations in modeling evolution, thereby limiting their influence on subsequent representations.

Discussion. The above observations suggest that mutations with different structural patterns exhibit distinct behaviors in the embedding space. Although existing dynamic graph representation learning methods achieve good performance in handling such mutations, they typically do not explicitly distinguish between rational and occasional mutations. Instead, they indiscriminately encode both types into node representations without a guiding standard to differentiate them. Moreover, lacking a mechanism to evaluate which information in mutations is beneficial, they tend to weaken the contribution of rational mutations and retain less useful information for evolution modeling. By

decoupling stable evolution patterns from dynamic graphs and introducing transfer entropy to measure and maximize the information gain from global to local, **TE-MDG** distinguishes rational mutations from occasional ones and selectively captures meaningful information from them. This enables the model to preserve the predictive value of rational mutations while filtering out occasional ones.

3 METHODOLOGY

In this section, we present **TE-MDG** in detail, which consists of two components: (i) Evolution Pattern Decoupling, which learns a set of temporal prototypes to decouple stable evolution patterns from local temporal patterns, providing a clear standard for distinguishing rational mutations from occasional ones; (ii) Mutation Filtering, which employs transfer entropy to measure the information gain from global temporal prototypes to local temporal patterns, and maximizes this gain to encourage the model to retain rational mutations while filtering out occasional ones. The overall framework of TE-MDG is illustrated in Figure 2. The details of the method are provided in Appendix B.

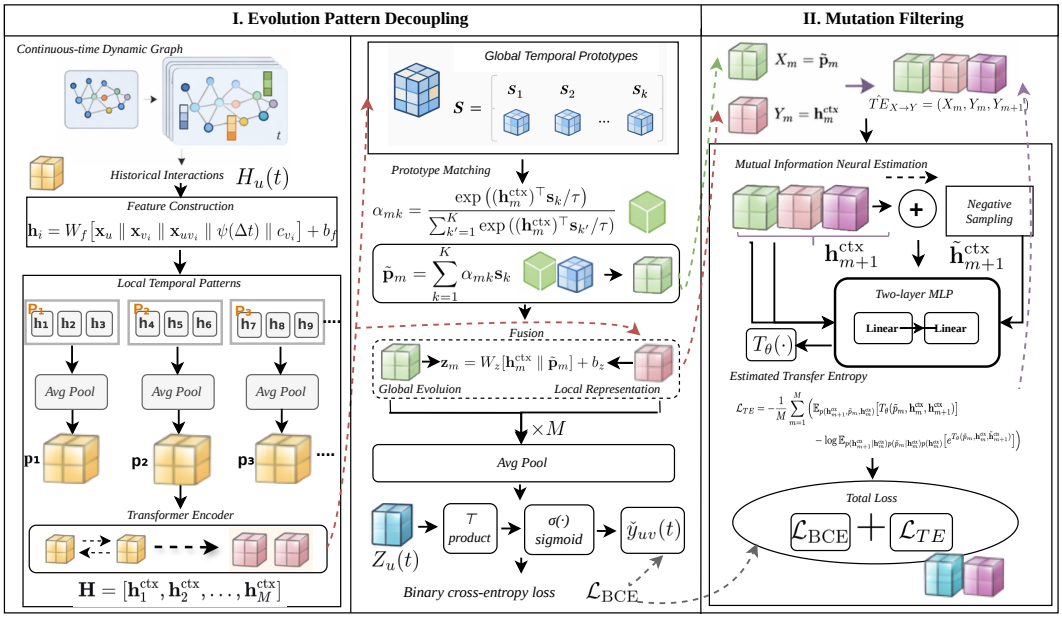


Figure 2: **The Overall Pipeline of TE-MDG.** It consists of two key components: (i) Evolution Pattern Decoupling learns a set of global temporal prototypes to decouple stable evolution patterns as a guiding standard for distinguishing mutations from dynamic graph. (ii) Mutation Filtering introduces transfer entropy as a regularization term to enhance the information gain between global temporal prototypes and local temporal patterns, allowing stable evolution patterns to guide the model in distinguishing different contributions of mutations, thereby achieving selective mutation filtering.

3.1 EVOLUTION PATTERN DECOUPLING

At time t, for the target node u, we sample a neighbor interaction sequence of fixed length $\{(u, v_i, t_i, \mathbf{e}_{uv_i})\}_{i=1}^{L}$ from its historical interaction set $H_u(t)$, where L denotes the maximum sequence length. To comprehensively and effectively characterize both structural and temporal information, we construct a representation vector for each interaction by integrating node features $\mathbf{x}_u$ and $\mathbf{x}_{v_i}$, edge features $\mathbf{e}_{uv_i}$, temporal encoding, and neighborhood structural context $c_{v_i} \in \mathbb{R}^{d_c}$, where the neighborhood structural context denotes the co-occurrence relationships of neighbors within the interaction sequence. Formally, the interaction representation is defined as:

$$\mathbf{h_i} = W_f\left[\mathbf{x}_u \,\|\, \mathbf{x}_{v_i} \,\|\, \mathbf{e}_{uv_i} \,\|\, \psi(t - t_i) \,\|\, \phi(c_{v_i})\right] + b_f, \tag{1}$$

where $\psi(\cdot)$ is a learnable temporal encoding function, $\phi(\cdot)$ is a two-layer perceptron with ReLU activation to encode the neighbor co-occurrence features, and W_f, b_f are learnable parameters. This yields the temporal interaction sequence $\mathcal{H}_u(t) = \{\mathbf{h_i}\}_{i=1}^{L}$. Directly modeling the long temporal interaction sequence is not only computationally expensive but also difficult for capturing long-term

temporal dependencies. To address this, we adopt a chunking mechanism to construct local temporal patterns. Specifically, let $M = \lceil L/p \rceil$ and pad $\mathcal{H}_u(t)$ with zero vectors to length $M \cdot p$, yielding a padded sequence. Then partite it into M non-overlapping subsequences of equal length p, denoted as $P_m = [\mathbf{h}_{(m-1)p+1}, \ldots, \mathbf{h}_{mp}]$ for $m = \{1, \ldots, M\}$. Each subsequence is then aggregated by average pooling to obtain the representation of a local pattern $\mathbf{p}_m = g(P_m) = \frac{1}{p} \sum_{\mathbf{h} \in P_m} \mathbf{h}$, thus obtaining the sequence of local temporal patterns $\mathcal{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_M]$.

To further model the contextual dependencies across local temporal patterns, we feed $\mathcal{P}$ into a multi-layer multi-head Transformer encoder. Through self-attention mechanisms, the model learns the complex and high-order interactions across temporal chunks, yielding rich and context-aware representations of local temporal patterns $\mathbf{H} = \{\mathbf{h}_m^{\text{ctx}}\}_{m=1}^{M}$, where $\mathbf{h}_m^{\text{ctx}}$ is the context-aware embedding of the m-th local pattern after modeling inter-chunk dependencies via self-attention.

Although local temporal patterns can characterize the short-term behavioral dynamics of nodes, long-term evolution trends in dynamic graphs are often governed by a set of stable evolution patterns that occur repeatedly across nodes and time. To model such patterns, we introduce a set of learnable global temporal prototypes $S = \{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_K\}$, where each global temporal prototype represents a stable evolution pattern. For each context-aware local temporal pattern $\mathbf{h}_m^{\text{ctx}}$, by calculating its similarity to all global temporal prototypes S and normalizing them with softmax function, we obtain its corresponding global temporal prototypes representation:

$$\tilde{p}_m = \sum_{k=1}^{K} \alpha_{mk} \mathbf{s}_k, \ \alpha_{mk} = \frac{\exp\left(\mathbf{h}_m^{\text{ctx}\top} \mathbf{s}_k / \tau\right)}{\sum_{k'=1}^{K} \exp\left(\mathbf{h}_m^{\text{ctx}\top} \mathbf{s}_{k'} / \tau\right)}, \tag{2}$$

where α_{mk} denotes the soft assignment weight of the local temporal pattern $\mathbf{h}_m^{\text{ctx}}$ to the global temporal prototype s_k, with τ being a temperature parameter that controls the smoothness of the assignment. These global temporal prototypes serve as shared and learnable parameters across nodes and time, providing a expressive representation space in which local temporal patterns can be represented as combinations of a small set of stable evolution patterns. In this space, the prototypes guide and constrain the local patterns, pushing them to behave consistently with stable evolution patterns.

We concatenate the context-aware local temporal pattern representation $\mathbf{h}_m^{\text{ctx}}$ with its corresponding global prototype representation $\tilde{p}_m$, and obtain a unified and expressive integrated representation of the m-th local temporal pattern via a learnable linear transformation, denoted as:

$$z_m = W_z[\mathbf{h}_m^{\text{ctx}} \| \tilde{p}_m] + b_z, \tag{3}$$

where W_z and b_z are learnable parameters. By average pooling over integrated representations of all local temporal patterns, we obtain the final representation of node u at time t: $z_u(t) = \frac{1}{M} \sum_{m=1}^{M} z_m$. For a node pair (u, v), the interaction probability at time t is defined as:

$$\hat{y}_{uv}(t) = \sigma\left(z_u(t)^\top z_v(t)\right), \tag{4}$$

where $\sigma(\cdot)$ denotes the sigmoid function that outputs a probability score. We adopt Binary Cross Entropy (BCE) as the main task loss for model training, which is defined as:

$$\mathcal{L}_{\text{BCE}} = -\sum_{(u,v,t) \in \mathcal{E}} \log \hat{y}_{uv}(t) - \sum_{(u,v,t) \notin \mathcal{E}} \log(1 - \hat{y}_{uv}(t)). \tag{5}$$

3.2 Mutation Filtering

While the above representation fusion integrates local dynamics and stable evolution patterns into a unified space, relying solely on the prediction loss $\mathcal{L}_{\text{BCE}}$ is insufficient to distinguish the contributions of different types of mutations. To address this, we introduce **transfer entropy** to model the directional influence of global temporal prototypes on local temporal patterns. The reasons for choosing transfer entropy are detailed in the Appendix A.2.

We aim to maximize the additional information that $\tilde{p}_m$ contributes to predicting the future state $\mathbf{h}_{m+1}^{\text{ctx}}$ beyond what is already known from the current state $\mathbf{h}_m^{\text{ctx}}$, thereby encouraging the global temporal prototype to guide the local temporal pattern informatively. This exactly corresponds to

the transfer entropy from $\tilde{p}_m$ to $\mathbf{h}_m^{\text{ctx}}$, which quantifies the information gain from the global temporal prototype for predicting the future state of local temporal pattern and defined as:

$$\text{TE}_{\tilde{p}_m \to \mathbf{h}_m^{\text{ctx}}} = I(\mathbf{h}_{m+1}^{\text{ctx}}; \tilde{p}_m \mid \mathbf{h}_m^{\text{ctx}}) = \mathbb{E}_{p(\mathbf{h}_{m+1}^{\text{ctx}}, \tilde{p}_m, \mathbf{h}_m^{\text{ctx}})} \left[\log \frac{p(\mathbf{h}_{m+1}^{\text{ctx}} \mid \tilde{p}_m, \mathbf{h}_m^{\text{ctx}})}{p(\mathbf{h}_{m+1}^{\text{ctx}} \mid \mathbf{h}_m^{\text{ctx}})} \right], \tag{6}$$

where $I(\cdot;\cdot \mid \cdot)$ denotes the conditional mutual information. To more intuitively understand the different effects of mutations, we model temporal evolution from a latent variable perspective. Specifically, we treat the evolution of local temporal patterns as being driven by two types of mutations: rational mutations δ_{rat} and occasional mutations δ_{oca}. Therefore, the local representation at the next step $\mathbf{h}_{m+1}^{\text{ctx}}$ can be formalized as:

$$\mathbf{h}_{m+1}^{\text{ctx}} \sim p(\cdot \mid \mathbf{h}_m^{\text{ctx}}, \delta_{\text{rat}}^{(m)}, \delta_{\text{oca}}^{(m)}). \tag{7}$$

Since rational mutations are related to the global temporal prototype $\tilde{p}_m$, while occasional mutations are conditionally independent of it, that is:

$$\delta_{\text{rat}}^{(m)} \not\perp \tilde{p}_m \mid \mathbf{h}_m^{\text{ctx}}, \quad \delta_{\text{oca}}^{(m)} \perp \tilde{p}_m \mid \mathbf{h}_m^{\text{ctx}}. \tag{8}$$

Accordingly, the transfer entropy $I(\mathbf{h}_{m+1}^{\text{ctx}}; \tilde{p}_m | \mathbf{h}_m^{\text{ctx}})$ measures the additional information gain provided by introducing the global temporal prototype $\tilde{p}_m$ for predicting the future state $\mathbf{h}_{m+1}^{\text{ctx}}$ given the current local state $\mathbf{h}_m^{\text{ctx}}$. Because rational mutations δ_{rat} are consistent with the evolution trend of the global temporal prototypes, whereas occasional mutations δ_{oca} are irrelevant to it, through the constraint of transfer entropy, the model can focus more on rational mutations consistent with the stable evolution patterns, and be insensitive to occasional ones. The details on how to distinguish the differential contributions of mutations are described in the Appendix B.3.

We adopt the differentiable neural estimator **MINE** (Belghazi et al., 2018) (Mutual Information Neural Estimation) to effectively approximate transfer entropy. The details are shown in Appendix B.2. Let $\widehat{\text{TE}}_{\tilde{p}_m \to \mathbf{h}_m^{\text{ctx}}}$ denote the estimated transfer entropy for the m-th chunk, and define the regularization term as the negative average over all chunks: $\mathcal{L}_{\text{TE}} = -\frac{1}{M} \sum_{m=1}^{M} \widehat{\text{TE}}_{\tilde{p}_m \to \mathbf{h}_m^{\text{ctx}}}$, thereby yielding:

$$\begin{aligned} \mathcal{L}_{TE} = -\frac{1}{M} \sum_{m=1}^{M} \Big(& \mathbb{E}_{p(\mathbf{h}_{m+1}^{\text{ctx}}, \tilde{p}_m, \mathbf{h}_m^{\text{ctx}})} \left[T_\theta(\tilde{p}_m, \mathbf{h}_m^{\text{ctx}}, \mathbf{h}_{m+1}^{\text{ctx}}) \right] \\ & - \log \mathbb{E}_{p(\mathbf{h}_{m+1}^{\text{ctx}} | \mathbf{h}_m^{\text{ctx}}) p(\tilde{p}_m | \mathbf{h}_m^{\text{ctx}}) p(\mathbf{h}_m^{\text{ctx}})} \left[e^{T_\theta(\tilde{p}_m, \mathbf{h}_m^{\text{ctx}}, \tilde{\mathbf{h}}_{m+1}^{\text{ctx}})} \right] \Big), \end{aligned} \tag{9}$$

where $\tilde{\mathbf{h}}_{m+1}^{ctx}$ is a negatively sampled local temporal pattern, and $T_\theta(\cdot)$ is a parameterized statistics network (implemented as a two-layer MLP). Minimizing $\mathcal{L}_{\text{TE}}$ is equivalent to maximizing transfer entropy, which promotes information gain from global temporal prototypes to local temporal patterns and encourages the model to focus on mutations that are predictive of future dynamics, thereby capturing meaningful information from those mutations.

Let λ_{TE} denote the hyperparameter that controls the relative strength of the regularization term to balance the main task loss and the transfer entropy regularization objective. The overall training objective is then formally defined as:

$$\mathcal{L} = \mathcal{L}_{BCE} + \lambda_{TE} \cdot \mathcal{L}_{TE}. \tag{10}$$

4 EXPERIMENT

In this section, we conduct comprehensive experiments to verify the effectiveness of TE-MDG. Our experiments are designed to answer the following key questions in detail. **Q1** (Main Results): How does TE-MDG perform on downstream tasks compared with SOTA dynamic graph representation learning methods? **Q2** (Analysis on Transfer Entropy): Does the transfer entropy increase the information gain from global temporal prototypes to local temporal patterns, thereby providing effective constraints on dynamic evolution? **Q3** (Ablation Study): How does each component of TE-MDG contribute to the performance? More additional experiments are provided in Appendix C.

4.1 EXPERIMENT SETUP

Datasets. We conduct a comprehensive and systematic evaluation of TE-MDG on two widely used benchmarks: the Temporal Graph Benchmark (TGB (Huang et al., 2023)) and the DyGLib (Yu et al., 2023). Specifically, for the link prediction task, we utilize three representative datasets from TGB, including tgbl-wiki, tgbl-review, and tgbl-coin, which feature large-scale continuous-time interactions across diverse domains and complex temporal dynamics. In addition, we employ four classic datasets from DyGLib, including CanParl, Enron, UCI and MOOC, which represent typical temporal interaction scenarios and have been extensively used for evaluating dynamic graph representation learning methods. For the node classification task, we use TGB datasets, including tgbn-trade, tgbn-genre, and tgbn-reddit, which contain diverse node types and attribute features, effectively reflecting the behavioral patterns of different nodes evolving over time. More details of the datasets are provided in Appendix D.1.

Baselines. We compare TE-MDG with SOTA methods, including memory-based models (TGN (Rossi et al., 2020) and JODIE (Kumar et al., 2019)), which maintain dynamic node states to capture temporal evolution; sequence-based methods (TGAT (Xu et al., 2020), CAWN (Wang et al., 2022), GraphMixer (Cong et al., 2023) and DyGFormer (Yu et al., 2023)), which model interaction sequences using time-aware attention mechanisms or Transformer-style architectures to capture temporal dependencies. Additionally, we also compare with DIDA (Zhang et al., 2022), a robust and generalizable model for dynamic graph representation learning. Details are provided in Appendix D.2.

Evaluation Metrics and Implementation Details. We evaluate TE-MDG on two downstream tasks: link prediction and node classification. For link prediction, the goal is to predict future connections based on historical interactions. We adopt commonly used evaluation metrics, including Area Under the ROC Curve (AUC), Average Precision (AP), and Mean Reciprocal Rank (MRR), to assess model performance. For node classification, we employ NDCG@10 (Järvelin & Kekäläinen, 2002) to evaluate the quality of the learned node representations. All implementation details, including model configurations and hyperparameter settings, are provided in Appendix D.4 for reproducibility.

Table 1: Performance (AUC, AP and MRR, %) comparison between our method TE-MDG and SOTA baselines on TGB and DyGLib datasets. Results are reported as mean ± std. The best results in each row are highlighted in **bold**, and the second-best are underlined.

Dataset	Metric	TGN	TGAT	JODIE	CAWN	GraphMixer	DyGFormer	DIDA	TE-MDG(ours)
TGB Datasets									
tgbl-wiki	AUC	73.64±9.77	67.15±7.92	65.25±15.27	91.69±0.62	88.69±1.52	96.89±0.44	81.93±0.48	**97.15±0.35**
	AP	81.10±8.78	76.31±3.87	79.87±10.39	91.86±0.93	91.56±1.44	97.37±0.41	87.50±0.40	**97.64±0.33**
	MRR	28.24±9.22	31.18±2.35	22.50±5.41	70.23±7.91	37.71±6.19	70.88±0.58	39.46±1.81	**73.47±2.03**
tgbl-review	AUC	57.02±4.56	66.43±0.12	71.26±0.09	78.58±0.25	82.13±0.13	71.81±0.56	59.91±0.23	**83.55±1.57**
	AP	65.93±11.27	80.70±2.53	76.95±3.69	90.38±1.45	85.51±2.58	76.10±8.60	56.35±0.22	**93.10±2.84**
	MRR	16.50±10.66	11.73±2.19	15.46±3.25	18.45±0.34	**40.83±2.16**	23.80±0.94	22.87±0.93	35.12±1.07
tgbl-coin	AUC	65.82±2.65	66.83±5.41	75.90±12.23	80.03±4.69	73.38±8.54	57.30±1.06	62.14±3.16	**96.94±0.29**
	AP	82.39±0.34	78.39±0.26	82.49±11.76	93.88±0.49	76.09±0.71	75.78±1.91	67.04±1.91	**98.70±0.13**
	MRR	14.43±6.11	6.89±3.15	29.13±4.65	31.81±0.59	35.80±1.38	26.56±0.39	31.11±2.02	**47.08±0.88**
DyGLib Datasets									
Can·Parl	AUC	73.74±5.90	58.22±7.77	80.35±0.38	66.93±6.43	**80.99±2.39**	79.36±0.42	63.26±0.18	76.37±5.16
	AP	65.92±3.50	54.62±6.60	71.00±0.51	62.12±3.99	70.78±2.12	**72.81±0.68**	62.50±0.09	71.43±3.11
	MRR	16.54±7.57	5.59±2.88	**27.16±2.90**	12.88±1.34	13.46±2.83	17.17±1.78	31.53±2.35	16.77±5.75
Enron	AUC	88.38±3.12	64.94±0.12	82.58±2.54	90.21±0.53	83.35±1.22	92.67±0.08	65.72±2.04	**92.91±0.07**
	AP	85.59±5.46	66.88±1.25	78.88±3.76	90.43±2.56	80.67±4.36	92.04±0.08	62.76±1.94	**92.42±0.04**
	MRR	28.23±5.48	14.73±4.32	13.36±4.79	44.02±6.12	24.91±3.64	43.58±0.50	21.37±1.09	**49.45±0.54**
MOOC	AUC	84.21±2.53	83.00±0.08	83.99±0.29	**85.07±0.12**	82.16±0.54	83.95±0.96	83.84±0.49	84.89±0.63
	AP	81.02±0.25	81.52±0.21	80.31±0.35	**85.29±2.31**	80.74±0.15	84.15±0.52	82.49±0.40	84.80±0.54
	MRR	27.85±0.12	27.59±1.53	20.29±0.36	33.62±0.79	26.56±0.45	31.35±0.14	17.12±0.78	**33.82±0.76**
UCI	AUC	92.31±0.96	76.98±0.49	89.24±0.44	94.79±0.15	91.70±0.63	94.94±0.12	69.59±1.89	**96.54±0.24**
	AP	92.47±1.03	78.20±0.51	86.93±0.72	95.98±0.11	93.29±0.53	96.07±0.07	66.31±2.13	**97.65±0.12**
	MRR	46.61±4.08	31.63±0.23	17.94±6.22	77.11±0.07	59.43±1.24	77.04±0.15	10.65±0.88	**79.23±0.21**

4.2 MAIN RESULTS (Q1)

To answer Q1, we evaluate TE-MDG against SOTA dynamic graph representation learning methods on multiple datasets. The main results of the comparison for link prediction and node classification are reported in Table 1 and Appendix C.1. We summarize the key observations as follows.

Obs. 1. TE-MDG achieves consistently strong performance across both tasks, with both link prediction and node classification achieving the best performance in the vast majority of datasets. For link prediction, on the tgbl-coin, it achieves 96.94% AUC and 98.7% AP, significantly outperforming the strongest baseline CAWN by 16.91% and 4.82%, respectively. Even in the few datasets where it does not achieve the top score, its performance remains highly competitive without noticeable degradation. Overall, these results indicate that TE-MDG learns more discriminative and generalizable representations, leading to strong performance across different downstream tasks.

Obs. 2. TE-MDG exhibits more significant advantages in complex temporal dynamic scenarios. We observe that on datasets with strong temporal dynamics, such as tgbl-coin and tgbl-review, TE-MDG shows particularly notable performance gains. These datasets often involve frequent mutations, where methods that rely primarily on local history often struggle to effectively model long-term dependencies. TE-MDG decouples stable evolution patterns as a clear standard via global temporal prototypes and enhances information flow between global and local patterns through transfer entropy, thereby suppressing occasional mutations while preserving rational evolution patterns, enabling more accurate modeling of long-term evolution. On relatively stable datasets, although the performance improvements are smaller, TE-MDG still maintains a consistent advantage, demonstrating remarkable stability and adaptability across diverse temporal dynamic patterns.

4.3 Analysis on Transfer Entropy (Q2)

To answer Q2, we analyze the evolution of transfer entropy during training. Transfer entropy measures the information gain from global temporal prototypes to local temporal patterns, allowing stable evolution patterns to guide the model in distinguishing the different contributions of mutations. A higher value indicates a stronger constraint on mutations, encouraging the model to focus on mutations consistent with stable evolution patterns in the dynamic graph while suppressing the influence of occasional mutations. Therefore, examining the dynamics of transfer entropy during training provides a way to evaluate whether it effectively regularizes the temporal evolution of node representations. Specifically, we visualize the evolution of transfer entropy throughout training, as shown in Figure 3.

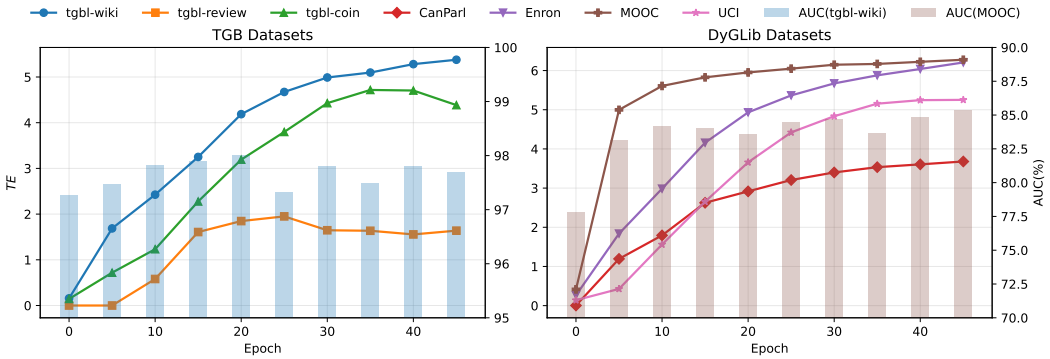


Figure 3: Variation trend of transfer entropy (TE) during training on TGB and DyGLib datasets. We report the value of TE (on the left axis) and AUC (on the right axis) for MOOC and tgbl-wiki.

Obs. 3. Transfer entropy consistently increases during training, indicating its effective regularization role. On most datasets, transfer entropy gradually increases and eventually stabilizes, suggesting that the model progressively strengthens the information gain from global temporal prototypes to local temporal patterns, thereby enhancing its ability to constrain mutations in different types. For example, on the MOOC dataset, the rapid increase of transfer entropy in early training is accompanied by a significant improvement in AUC, indicating that enhanced information transmission can be effectively translated into performance gains. Furthermore, the magnitude of the increase varies between datasets. On tgbl-review, transfer entropy grows more slowly with mild fluctuations, reflecting that occasional mutations make it difficult for global temporal prototypes to capture stable evolution patterns. In contrast, on datasets with more stable temporal structures, such as tgbl-wiki and MOOC, transfer entropy exhibits a more clear and smoother increase. Overall, the consistent upward trend of transfer entropy across datasets aligns with the performance improvements reported in Table 1, validating the effectiveness of the proposed regularization mechanism.

4.4 Ablation Study (Q3)

To analyze the contribution of each component in TE-MDG, we design two ablation settings:

- **w/o** TE (without transfer entropy): We remove the transfer entropy regularization by setting its weight to zero (i.e., $\lambda_{TE} = 0$) while keeping the global temporal prototypes. This setting verifies how the model performs without the constraint that enhances information gain;
- **w/o** S (without global temporal prototypes): We remove the global temporal prototypes, which also disables the transfer entropy (as it depends on the prototypes). The model then relies solely on local historical information without the guidance and constraints from stable evolution patterns, thus failing to extract meaningful information from mutations.

The ablation results for link prediction and node classification are in Table 2 and Appendix C.2.2.

Table 2: Ablation study of TE-MDG components on the link prediction task across TGB and DyGLib datasets. Δ denotes the performance drop relative to the full TE-MDG model. Δ corresponds to the decrease in AUC, while Δ corresponds to the decrease in AP.

Model	Metric	TGB Datasets			Average		DyGLib Datasets				Average	
		tgbl-wiki	tgbl-review	tgbl-coin	All	Δ	Can·Parl	Enron	MOOC	UCI	All	Δ
w/o TE	AUC	$97.17_{\pm 0.21}$	$82.24_{\pm 1.16}$	$96.88_{\pm 0.36}$	92.10	↓0.45	$66.73_{\pm 3.19}$	$89.34_{\pm 3.10}$	$81.74_{\pm 0.20}$	$94.03_{\pm 0.31}$	82.96	↓4.72
	AP	$97.55_{\pm 0.32}$	$90.85_{\pm 0.31}$	$98.68_{\pm 0.16}$	95.69	↓0.79	$62.51_{\pm 2.14}$	$89.27_{\pm 2.33}$	$82.35_{\pm 0.34}$	$95.54_{\pm 0.11}$	82.42	↓4.16
w/o S	AUC	$96.89_{\pm 0.44}$	$71.81_{\pm 0.56}$	$57.47_{\pm 0.86}$	75.39	↓17.16	$74.06_{\pm 3.28}$	$89.85_{\pm 3.51}$	$82.11_{\pm 0.60}$	$94.94_{\pm 0.12}$	85.24	↓2.44
	AP	$97.37_{\pm 0.41}$	$76.10_{\pm 8.60}$	$79.64_{\pm 0.15}$	84.37	↓12.11	$68.50_{\pm 2.39}$	$89.79_{\pm 2.63}$	$82.47_{\pm 0.76}$	$96.07_{\pm 0.07}$	84.21	↓2.37
TE-MDG	AUC	$97.15_{\pm 0.35}$	$83.55_{\pm 1.57}$	$96.94_{\pm 0.29}$	**92.55**	–	$76.37_{\pm 5.16}$	$92.91_{\pm 0.07}$	$84.89_{\pm 0.63}$	$96.54_{\pm 0.24}$	**87.68**	–
	AP	$97.64_{\pm 0.33}$	$93.10_{\pm 2.84}$	$98.70_{\pm 0.13}$	**96.48**	–	$71.43_{\pm 3.11}$	$92.42_{\pm 0.04}$	$84.80_{\pm 0.54}$	$97.65_{\pm 0.12}$	**86.58**	–

Obs. 4. The transfer entropy TE **and the global temporal prototypes** S **are indispensable.** Overall, the full model achieves the best performance on all datasets and removing any component leads to performance degradation. This effect is particularly clear on TGB datasets, where the average AUC and AP drop by as much as 17.16% and 12.11%. These results indicate that transfer entropy and global temporal prototypes play distinct yet complementary roles: the global temporal prototypes capture stable evolution patterns and serve as a guiding standard for distinguishing mutations, based on this standard, the transfer entropy measures and maximizes the information gain from prototypes to mutations, enabling the model to retain rational mutations and filter out occasional ones.

Obs. 5. Different components play different roles under different data distributions. A clear contrast emerges when comparing TGB and DyGLib datasets. On TGB datasets, removing the global temporal prototypes leads to a much larger performance drop than removing TE (e.g., average AUC drop of 17.16% vs. 0.45%), suggesting that modeling stable evolution patterns is critical in complex scenarios with long temporal spans. In contrast, on DyGLib datasets, removing TE results in a comparatively larger performance drop than removing the global temporal prototypes (e.g., average AUC drop of 4.72% vs. 2.44%). This suggests that in smaller and relatively stable settings, where the impact of mutations is more pronounced and thus the model relies more on the TE objective to enforce effective information transmission. While the dominant role of each component varies across data distributions, both contribute substantially to the overall performance.

5 Conclusion

In this paper, we argue that existing dynamic graph representation learning methods lack a clear standard to distinguish mutations, and selectively retain their meaningful information. We address this limitation with TE-MDG, which incorporates different mutations into node representations. TE-MDG first learns a set of temporal prototypes that capture stable evolution patterns and serve as a guiding standard for distinguishing mutations. Then it employs transfer entropy to measure and maximize the information gain from these prototypes to local behaviors, enabling the model to selectively retain meaningful information from rational mutations while filtering out occasional ones. Extensive experiments on multiple benchmarks demonstrate that TE-MDG consistently outperforms state-of-the-art baselines on link prediction and node classification. Ablation studies further validate that the global temporal prototypes and transfer entropy regularization play distinct yet complementary roles.

AI USE STATEMENT

We used generative AI tools only for language polishing, grammar checking, and improving the clarity and readability of the manuscript. We did not use generative AI tools to formulate research questions or hypotheses, develop the methodology or theoretical framework. We reviewed and verified all AI-assisted edits. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

We have reviewed the ICLR Code of Ethics and have ensured full compliance with its guidelines, particularly in terms of data privacy, transparency, and responsible computing practices. The datasets used in the experiments include those in Appendix D.1 which are publicly available and widely used.

REPRODUCIBILITY STATEMENT

To facilitate replication and extension of our work, we provide comprehensive resources: the code is given in Appendix A.1, dataset and preprocessing details in Appendix D.1, and complete derivations and theoretical proofs in Appendix B. With the complete set of code, data, preprocessing details, and theoretical derivations provided, the reproducibility of our work is fully ensured.

REFERENCES

Lionel Barnett, Adam B. Barrett, and Anil K. Seth. Granger causality and transfer entropy are equivalent for gaussian variables. *Phys. Rev. Lett.*, 103:238701, Dec 2009.

Claudio D. T. Barros, Matheus R. F. Mendonça, Alex B. Vieira, and Artur Ziviani. A survey on embedding dynamic graphs. *ACM Comput. Surv.*, 55(1), November 2021. ISSN 0360-0300.

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *International conference on machine learning*, pp. 531–540. PMLR, 2018.

Dong Chen, Shuai Zheng, Muhao Xu, Zhenfeng Zhu, and Yao Zhao. Signn: A spike-induced graph neural network for dynamic graph representation learning, 2024.

Katrina Chen, Mingbin Feng, and Tony S. Wirjanto. Multivariate time series anomaly detection via dynamic graph forecasting, 2023.

Su Chen, Xiaohua Qi, Xixun Lin, Yanmin Shang, Xiaolin Xu, and Yangxi Li. Deep graph neural point process for learning temporal interactive networks, 2025.

Weilin Cong, Si Zhang, Jian Kang, Baichuan Yuan, Hao Wu, Xin Zhou, and Hanghang Tong. Do we really need complicated model architectures for temporal networks?, 2023.

Xinhua Dong, Hui Zhang, Hongmu Han, and Zhigang Xu. Dpgad: Structure-aware dual-path attention graph node anomaly detection. *Symmetry*, 17(9), 2025. ISSN 2073-8994.

Sofiane ENNADIR, Gabriela Zarzar Gandler, Filip Cornell, Lele Cao, Oleg Smirnov, Tianze Wang, Levente Zólyomi, Björn Brinne, and Sahar Asadi. Expressivity of representation learning on continuous-time dynamic graphs: An information-flow centric review. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. Survey Certification.

Palash Goyal, Nitin Kamra, Xinran He, and Yan Liu. Dyngem: Deep embedding method for dynamic graphs, 2018.

Tony Gracious and Ambedkar Dukkipati. Deep representation learning for forecasting recursive and multi-relational events in temporal networks, 2024.

Jianhao Guo, Siliang Tang, Juncheng Li, Kaihang Pan, and Lingfei Wu. Rustgraph: Robust anomaly detection in dynamic graphs by jointly learning structural-temporal dependency. *IEEE Transactions on Knowledge and Data Engineering*, 36(7):3472–3485, 2024.

William L. Hamilton, Rex Ying, and Jure Leskovec. Representation learning on graphs: Methods and applications, 2018.

Jing He, Mian Wang, Zhen Zhang, Lin Chen, and YunYun Dong. Graph representation learning method using diffusion sampling for dynamic graph anomaly detection: J. he et al. *Knowledge and Information Systems*, pp. 1–24, 2025.

Katerina Hlaváčková-Schindler, Milan Paluš, Martin Vejmelka, and Joydeep Bhattacharya. Causality detection based on information-theoretic approaches in time series analysis. *Physics Reports*, 441 (1):1–46, 2007. ISSN 0370-1573.

Changqin Huang, Binghang Yu, Chengling Gao, Yaxin Tu, Fan Jiang, and Xiaodi Huang. Structural-temporal mining for motif-level anomaly detection in dynamic graphs. *Know.-Based Syst.*, 325(C), September 2025. ISSN 0950-7051.

Shenyang Huang, Farimah Poursafaei, Jacob Danovitch, Matthias Fey, Weihua Hu, Emanuele Rossi, Jure Leskovec, Michael Bronstein, Guillaume Rabusseau, and Reihaneh Rabbany. Temporal graph benchmark for machine learning on temporal graphs. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

Md Abrar Jahin, Taufikur Rahman Fuad, Jay Pujara, and Craig Knoblock. Chronospike: An adaptive spiking graph neural network for dynamic graphs, 2026.

Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, October 2002. ISSN 1046-8188.

Seunghun Jeong, Huioon Kim, Young Ho Kim, Chang-Soo Park, Hyoyoung Jung, and Hong Kook Kim. Spatiotemporal anomaly detection in distributed acoustic sensing using a graphdiffusion model. *Sensors*, 25(16), 2025. ISSN 1424-8220.

Zhigang Jin, Renjun Su, and Hao Zhang. Memory-based walk-enhanced dynamic graph neural network for temporal graph representation learning. *Neurocomputing*, 633:129759, 2025. ISSN 0925-2312.

Hyeonsoo Jo, Jongha Lee, Fanchen Bu, and Kijung Shin. Tiger: Self-supervised purification for time-evolving graphs, 2025.

Damjan Kalajdzievski, Ximeng Mao, Pascal Fortier-Poisson, and Guillaume Lajoie. Transfer entropy bottleneck: Learning sequence to sequence information transfer, 2023.

Seyed Mehran Kazemi, Rishab Goel, Kshitij Jain, Ivan Kobyzev, Akshay Sethi, Peter Forsyth, and Pascal Poupart. Representation learning for dynamic graphs: a survey. *J. Mach. Learn. Res.*, 21(1), January 2020. ISSN 1532-4435.

Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, pp. 1269–1278, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362016.

Chao Li, Runshuo Liu, Jinhu Fu, Zhongying Zhao, Hua Duan, and Qingtian Zeng. Dgnn-mn: Dynamic graph neural network via memory regenerate and neighbor propagation: C. li et al. *Applied Intelligence*, 54(19):9253–9268, 2024.

Guanghua Liu, Chenlong Wang, Zhiguo Gong, Jia Zhang, Shuqi Tang, and Huan Wang. Hypergraph-driven anomaly detection in dynamic noisy graphs. *IEEE Transactions on Information Forensics and Security*, 20:9848–9863, 2025.

Yixin Liu, Shirui Pan, Yu Guang Wang, Fei Xiong, Liang Wang, Qingfeng Chen, and Vincent CS Lee. Anomaly detection in dynamic graphs via transformer. *IEEE Transactions on Knowledge and Data Engineering*, 35(12):12081–12094, December 2023. ISSN 2326-3865.

Omer Luxembourg, Dor Tsur, and Haim Permuter. Treet: Transfer entropy estimation via transformers. *IEEE Access*, 2025.

Adrian Moldovan, Angel Caţaron, and Răzvan Andonie. Information plane analysis visualization in deep learning via transfer entropy. In *2023 27th International Conference Information Visualisation (IV)*, pp. 278–285. IEEE, July 2023.

Adrian Moldovan, Angel Cataron, and Răzvan Andonie. Transfer entropy in graph convolutional neural networks. In *2024 28th International Conference Information Visualisation (IV)*, pp. 1–7, 2024.

Sebastien Montella, Lina M. Rojas Barahona, and Johannes Heinecke. Hyperbolic temporal knowledge graph embeddings with relational and time curvatures. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 3296–3308, Online, August 2021. Association for Computational Linguistics.

Simon Pedro Galeano Munoz, Mustapha Bounoua, Giulio Franzese, Pietro Michiardi, and Maurizio Filippone. Tende: Transfer entropy neural diffusion estimation, 2025.

Giang Hoang Nguyen, John Boaz Lee, Ryan A. Rossi, Nesreen K. Ahmed, Eunyee Koh, and Sungchul Kim. Continuous-time dynamic network embeddings. In *Companion Proceedings of the The Web Conference 2018*, WWW '18, pp. 969–976, Republic and Canton of Geneva, CHE, 2018. International World Wide Web Conferences Steering Committee. ISBN 9781450356404.

Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao B. Schardl, and Charles E. Leiserson. Evolvegcn: Evolving graph convolutional networks for dynamic graphs, 2019.

Megha Ashok Patil, Sunil Kumar, and Sandeep Kumar. GraphDrift-net: a dynamic graph-based framework for concept drift detection in short unstructured text streams. *European Physical Journal Plus*, 140(9):884, September 2025.

Jie Peng, Zhewei Wei, and Yuhang Ye. Tidformer: Exploiting temporal and interactive dynamics makes A great dynamic graph transformer. *CoRR*, abs/2506.00431, 2025.

Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal graph networks for deep learning on dynamic graphs, 2020.

Ali Royat, Seyed Mohamad Moghadas, Lesley De Cruz, and Adrian Munteanu. Gintrip: Interpretable temporal graph regression using information bottleneck and prototype-based method, 2025.

Jakob Runge, Peer Nowack, Marlene Kretschmer, Seth Flaxman, and Dino Sejdinovic. Detecting and quantifying causal associations in large nonlinear time series datasets. *Science Advances*, 5 (11), November 2019. ISSN 2375-2548.

Aravind Sankar, Yanhong Wu, Liang Gou, Wei Zhang, and Hao 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*, WSDM '20, pp. 519–527, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450368223.

Thomas Schreiber. Measuring information transfer. *Phys. Rev. Lett.*, 85:461–464, Jul 2000.

Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information, 2017.

Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pp. 1–5, 2015.

Rakshit S. Trivedi, Mehrdad Farajtabar, and Prasenjeet Biswal. Dyrep: Learning representations over dynamic graphs. In *International Conference on Learning Representations*, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. 2017.

Raul Vicente, Michael Wibral, Michael Lindner, and Gordon Pipa. Transfer entropy–a model-free measure of effective connectivity for the neurosciences. *J. Comput. Neurosci.*, 30(1):45–67, February 2011. ISSN 0929-5313.

Haohui Wang, Yuzhen Mao, Yujun Yan, Yaoqing Yang, Jianhui Sun, Kevin Choi, Balaji Veeramani, Alison Hu, Edward Bowen, Tyler Cody, and Dawei Zhou. Evolunet: advancing dynamic non-iid transfer learning on graphs. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024a.

Huan Wang, Junyang Chen, Yirui Wu, Victor C. M. Leung, and Di Wang. Epm: Evolutionary perception method for anomaly detection in noisy dynamic graphs. *IEEE Transactions on Knowledge and Data Engineering*, 37(7):4035–4048, 2025a.

Jihong Wang, Yuxin Bai, Chunqiang Zhu, Hao Qian, Ziqi Liu, and Minnan Luo. Dynamic graph representation learning with disentangled information bottleneck. *Neural Networks*, 194:108056, 2026. ISSN 0893-6080.

Jimin Wang. Dynamic social network anomalous behavior detection based on spatiotemporal multi-view graph attention fusion network. *Frontiers in Physics*, Volume 14 - 2026, 2026. ISSN 2296-424X.

Lu Wang, Xiaofu Chang, Shuang Li, Yunfei Chu, Hui Li, Wei Zhang, Xiaofeng He, Le Song, Jingren Zhou, and Hongxia Yang. Tcl: Transformer-based dynamic graph modelling via contrastive learning. *ArXiv*, abs/2105.07944, 2021.

Qianqian Wang, Xiujun Wang, and Tao Tao. Adaptive neighbors graph convolutional networks with transfer entropy optimization. In *2024 Cross Strait Radio Science and Wireless Technology Conference (CSRSWTC)*, pp. 1–3, 2024b.

Yanbang Wang, Yen-Yu Chang, Yunyu Liu, Jure Leskovec, and Pan Li. Inductive representation learning in temporal networks via causal anonymous walks, 2022.

Zhe Wang, Sheng Zhou, Jiawei Chen, Zhen Zhang, Binbin Hu, Yan Feng, and Chun Chen. Dynamic graph transformer with correlated spatial-temporal positional encoding. In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining*, 2025b.

Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. Session-based recommendation with graph neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):346–353, July 2019. ISSN 2159-5399.

Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs, 2020.

Long Xu, Zhiqiang Pan, Honghui Chen, and Shen Wang. Mtdyg: Multi-scale transformers with continuous time dynamic graph model for link prediction. *Knowledge-Based Systems*, 300:112245, 2024. ISSN 0950-7051.

Yuanyuan Xu, Wenjie Zhang, Xiwei Xu, Binghao Li, and Ying Zhang. Scalable and effective temporal graph representation learning with hyperbolic geometry. *IEEE Transactions on Neural Networks and Learning Systems*, 36(4):6080–6094, 2025. doi: 10.1109/TNNLS.2024.3394161.

Changtian Ying, Qi Li, Chen Wang, and Donghua Yu. Dnrhp: Temporal network representation learning via hawkes point process. *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–7, 2025.

Peng You, Xinxin Wang, Peng Chen, Lei Wu, Juan Chen, Xi Li, Shengke Zeng, and Huangyining Gao. Dt-dgsl: dynamic transformer using denoising graph structure learning for iot time series anomaly detection. *The Journal of Supercomputing*, 81, 07 2025.

Le Yu, Leilei Sun, Bowen Du, and Weifeng Lv. Towards better dynamic graph learning: New architecture and unified library. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Haonan Yuan, Qingyun Sun, Xingcheng Fu, Cheng Ji, and Jianxin Li. Dynamic graph information bottleneck, 2024.

Zirui Yuan, Minglai Shao, and Qiben Yan. Motif-level anomaly detection in dynamic graphs. *IEEE Transactions on Information Forensics and Security*, 18:2870–2882, 2023.

Siwei Zhang, Yun Xiong, Yao Zhang, Yiheng Sun, Xi Chen, Yizhu Jiao, and Yangyong Zhu. Rdgsl: Dynamic graph representation learning with structure learning. In *Proceedings of the 32nd ACM international conference on information and knowledge management*, pp. 3174–3183, 2023.

Zeyang Zhang, Xin Wang, Ziwei Zhang, Haoyang Li, Zhou Qin, and Wenwu Zhu. Dynamic graph neural networks under spatio-temporal distribution shift. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 6074–6089. Curran Associates, Inc., 2022.

Lekui Zhou, Yang Yang, Xiang Ren, Fei Wu, and Yueting Zhuang. Dynamic network embedding by modeling triadic closure process. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press, 2018. ISBN 978-1-57735-800-8.

Ming Zhou and Jie Lu. Continuous graph learning-based self-adaptation for multi-stream concept drift. *IEEE Transactions on Cybernetics*, 55(8):3760–3773, 2025.

Ming Zhou, Jie Lu, Pengqian Lu, and Guangquan Zhang. Dynamic graph regularization for multi-stream concept drift self-adaptation. *IEEE Transactions on Knowledge and Data Engineering*, 36 (11):6016–6028, 2024.

Yuan Zuo, Guannan Liu, Hao Lin, Jia Guo, Xiaoqian Hu, and Junjie Wu. Embedding temporal network via neighborhood formation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, pp. 2857–2866, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450355520.

Contents of Appendix

A Frequently Asked Questions (FAQs) . . . 16

B Method Details . . . 19

B.1 Transfer Entropy in Dynamic Graphs . . . 19

B.2 Neural Estimation for Transfer Entropy via MINE . . . 20

B.3 Distinction of Mutations . . . 21

C Additional Experiments . . . 21

C.1 Additional Experiments on Node Classification . . . 21

C.2 Additional Ablation Study . . . 22

C.2.1 Supplement to Ablation Study on Link Prediction . . . 22

C.2.2 Ablation Study on Node Classification . . . 22

C.3 Hyperparameter Sensitivity Experiments . . . 23

C.3.1 The Weight of TE Loss . . . 23

C.3.2 The Number of Global Prototypes . . . 24

D Implementation Details . . . 24

D.1 Benchmark Datasets . . . 24

D.2 Baseline Details . . . 26

D.3 Evaluation Metrics . . . 27

D.4 TE-MDG Setup . . . 28

E Related Works . . . 28

E.1 Continuous-time Dynamic Graph Representation Learning . . . 28

E.2 Transfer Entropy . . . 29

E.3 Modeling Changes in Dynamic Graphs . . . 29

F Limitations . . . 29

G Broader Impacts . . . 30

A Frequently Asked Questions (FAQs)

A.1 Code Availability

To promote transparency and reproducibility, we have uploaded our code into the Anonymous GitHub Repository at https://anonymous.4open.science/r/TE_MDG. The repository includes the source code of TE-MDG and scripts for data preprocessing, model training and evaluation. Additionally, all datasets used in our experiments are also provided or linked within the repository, ensuring that researchers have full access to the resources required to reproduce and extend our work.

A.2 What is transfer entropy? Why transfer entropy?

Transfer entropy (Schreiber, 2000) (TE), introduced by Thomas Schreiber, is a non-parametric information-theoretic statistic that quantifies the directed, asymmetric information flow between two stochastic processes and serves as a core statistical tool for identifying the direction of information flow in time series. TE can be viewed as a form of conditional mutual information, measuring how much additional predictive information the source process X provides about the future state of the target process Y, beyond what is already contained in the target's own history alone.

Formally, given two stochastic processes X and Y, the transfer entropy from X to Y is defined as:

$$\mathrm{TE}_{X\to Y} = I(Y_{t+1}; X_t \mid Y_t). \tag{11}$$

Equivalently, it can be written as:

$$\mathrm{TE}_{X\to Y} = \mathbb{E}_{p(Y_{t+1},X_t,Y_t)}\left[\log \frac{p(Y_{t+1} \mid X_t, Y_t)}{p(Y_{t+1} \mid Y_t)}\right], \tag{12}$$

where $p(Y_{t+1} \mid Y_t)$ predicts the future state using only the current state Y_t of the target, and $p(Y_{t+1} \mid X_t, Y_t)$ represents the enhanced prediction after incorporating information from the external source X. Transfer entropy quantifies the expected log-ratio of predictive power gained by introducing X, thus serving as a measure of predictive information gain in a temporal conditioning framework.

A natural question that arises is why not directly use only **Mutual Information** (MI) or other entropy-based constraints. We answer this question from the following key aspects:

Mutual information is symmetric and ignores temporal directionality. The standard mutual information is inherently symmetric, i.e., $I(X;Y) = I(Y;X)$. This symmetry means it can only measure statistical dependence between variables, but simply cannot specific distinguish the direction of information flow. In dynamic graphs, future states are often driven by historical states, yet mutual information still cannot capture such asymmetric temporal dependencies.

Transfer entropy aligns with our modeling assumptions. In this paper, we propose that local temporal evolution is driven by two types of change: mutations that are consistent with stable evolution patterns, and occasional mutations that are not. Transfer entropy naturally captures the following mechanism: (i) if the mutation is rational, the global temporal prototype provides strong predictive information about its future state, yielding high transfer entropy; (ii) if the mutation is occasional, the global temporal prototype offers little additional predictive gain, resulting in low transfer entropy. Therefore, by maximizing TE, the model is guided to focus on mutations that are consistent with the stable evolution patterns, while suppressing occasional mutations . This cannot be achieved by standard mutual information or other symmetric entropy regularizers.

A.3 Why can TE-MDG effectively distinguish the different contributions of mutations?

TE-MDG can effectively distinguish the different contributions of mutations due to the synergy between two key designs. First, the evolution pattern decoupling module constructs a structured representation space that separates stable evolution patterns from dynamic graphs, providing a standard to constrain subsequent evolution. Second, the transfer entropy enforces a predictive constraint, selectively prioritizing mutations that carry meaningful information. They enable selective modeling of different behavior patterns at both the representation level and the dynamic level.

Specifically, the evolution pattern decoupling module maps local temporal pattern representations into a representation space spanned by global temporal prototypes, allowing each local state to be expressed as a weighted combination of a small number of prototypes. This design implicitly assumes that long-term evolutionary processes are dominated by a limited set of stable evolution patterns, thereby forming structural constraints in the representation space. In this space, mutations that are consistent with the stable evolution patterns can often be captured by these prototypes, showing good alignment with the prototype space. In contrast, irregular perturbations are difficult to be explained by these stable patterns and typically manifest as deviations from the prototype space. Therefore, this module introduces a structural inductive bias at the representation level, encouraging the model to prioritize patterns with evolutionary consistency while downplaying occasional mutations.

However, structural alignment alone is insufficient to determine whether a mutation is actually meaningful for future evolution. To address this, we further introduce a transfer entropy regularization term $I(\mathbf{h}_{m+1}^{\text{ctx}}; \tilde{p}_m \mid \mathbf{h}_m^{\text{ctx}})$, which characterizes whether the introduction of the global temporal prototype representation $\tilde{p}_m$ can provide additional predictive information for the future state $\mathbf{h}_{m+1}^{\text{ctx}}$ given the current local state $\mathbf{h}_m^{\text{ctx}}$. This conditional formulation is crucial, as it effectively eliminates the confusion caused by the strong temporal dependence between $\mathbf{h}_m^{\text{ctx}}$ and $\mathbf{h}_{m+1}^{\text{ctx}}$, thereby accurately characterizing the true contribution of stable evolution patterns to future dynamics. Consequently, only those mutations that are both aligned with the prototype structure and predictive of the future state yield high transfer entropy and are thus reinforced during optimization, whereas mutations that are structurally inconsistent or non-predictive will be naturally suppressed.

In summary, TE-MDG does not explicitly distinguish mutation types for modeling. Instead, it establishes a learning mechanism in which only mutations that are truly consistent with stable evolution patterns and carry predictive value are preserved and strengthened, while irrelevant perturbations are suppressed. This fundamentally explains why the model can effectively capture meaningful information from rational mutations while remaining robust to occasional mutations.

A.4 Computational Complexity Analysis

We analyze the computational complexity of TE-MDG by decomposing training into several stages, as shown in Algorithm 1. Let B the batch size, L the sequence length, p the patch size, $M = \lceil L/p \rceil$ the number of patches, K the number of global prototypes, and D the embedding dimension.

Stage 1. In the neighborhood sequence construction stage, for each node, we sample L historical neighbors and construct interaction features. The sampling operation incurs a linear cost $O(L)$, while feature construction involves concatenation followed by a linear projection in D-dimensional space, resulting in $O(LD)$. Therefore, the total complexity per node is:

$$O(L + LD) = O(LD), \tag{13}$$

and for a batch:

$$O(BLD). \tag{14}$$

Stage 2. In the local temporal patterns modeling stage, after partitioning the sequence into M patches, patch construction and average pooling require linear aggregation over L elements, yielding $O(BLD)$. The dominant computation arises from the Transformer encoder. For each layer, multi-head self-attention computes pairwise interactions between all M patches, leading to $O(M^2D)$, while the feed-forward network applies two linear transformations to each patch, yielding $O(MD^2)$. Thus, a single layer has complexity $O(M^2D + MD^2)$, and stacking T layers yields:

$$O(BLD + BT(M^2D + MD^2)). \tag{15}$$

Stage 3. In the global prototype assignment stage, for each patch representation, we compute similarity scores with K prototypes via dot products. Each dot product between two D-dimensional vectors costs $O(D)$, leading to an overall $O(KD)$ computational cost per patch. The subsequent softmax normalization introduces $O(K)$ cost, and the weighted aggregation of prototypes requires another $O(KD)$. Combining these steps, the total per-patch cost remains $O(KD)$, resulting in:

$$O(BMKD). \tag{16}$$

Stage 4. For each patch, we concatenate local and global representations (dimension $2D$) and apply a linear projection to dimension D. This corresponds to a dense matrix multiplication of size $D \times 2D$,

Algorithm 1 Overall pipeline of TE-MDG for link prediction

Require: Training graph $\mathcal{G}$; node/edge features $\mathbf{x}$; sequence length L; patch size p; prototypes S; batch size B; hyperparameters λ_{TE}, k
Ensure: Updated model parameters Θ; Node embeddings $\mathbf{z}_u$, $\mathbf{z}_v$; Link prediction scores $\hat{y}_{uv}$

for each mini-batch $\mathcal{B} = \{(u, v, t)\}_{i=1}^{B}$ **do**
  Stage 1: Neighbor Sequence Construction
  $\mathcal{H}_u \leftarrow \textsc{Sample}(H_u(t), L)$ ▷ Sample L historical neighbors
  $\mathbf{h}_i \leftarrow W_f[\mathbf{x}_u \| \mathbf{x}_{v_i} \| \mathbf{x}_{uv_i} \| \psi(t - t_i) \| c_{v_i}]$ ▷ Interaction encoding
  $\mathcal{H}_u = [\mathbf{h}_1, \ldots, \mathbf{h}_L]$ ▷ Temporal sequence for node u
  Stage 2: Local Temporal Patterns
  $\{P_m\}_{m=1}^{M} \leftarrow \textsc{Chunk}(\mathcal{H}_u, p)$ ▷ Partition into $M = \lceil L/p \rceil$ patches
  $\mathbf{p}_m \leftarrow \frac{1}{|P_m|} \sum_{\mathbf{h} \in P_m} \mathbf{h}$ ▷ Average pool to get local patterns
  $\mathbf{H}^{\text{ctx}} \leftarrow \textsc{Transformer}(\{\mathbf{p}_m\})$ ▷ Multi-head self-attention
  Stage 3: Global Temporal Prototypes Assignment
  $\alpha_{mk} \leftarrow \frac{\exp(\mathbf{h}_m^{\text{ctx}\top} \mathbf{s}_k / \tau)}{\sum_{k'} \exp(\mathbf{h}_m^{\text{ctx}\top} \mathbf{s}_{k'} / \tau)}$ ▷ Soft assignment to k prototypes
  $\tilde{\mathbf{p}}_m \leftarrow \sum_{k=1}^{K} \alpha_{mk} \mathbf{s}_k$ ▷ Global pattern representation
  Stage 4: Representation Fusion
  $\mathbf{z}_m \leftarrow W_z[\mathbf{h}_m^{\text{ctx}} \| \tilde{\mathbf{p}}_m] + b_z$ ▷ Integrate local and global
  $\mathbf{z}_u, \mathbf{z}_v \leftarrow \frac{1}{M} \sum_{m=1}^{M} \mathbf{z}_m$ ▷ Final node embedding
  Stage 5: Link Prediction Loss
  $\hat{y}_{uv} \leftarrow \sigma(\mathbf{z}_u^{\top} \mathbf{z}_v)$ ▷ Predict interaction probability
  $\mathcal{L}_{\text{BCE}} \leftarrow -\sum_{(u,v,t) \in \mathcal{B}} \log \hat{y}_{uv} - \sum_{\text{neg}} \log(1 - \hat{y}_{uv'})$ ▷ Binary cross entropy
  Stage 6: Transfer Entropy Regularization
  if $\lambda_{\text{TE}} > 0$ **then**
    Construct triplets $(\tilde{\mathbf{p}}_m, \mathbf{h}_m^{\text{ctx}}, \mathbf{h}_{m+1}^{\text{ctx}})$
    $\text{TE} \leftarrow \widehat{I}(\mathbf{h}_{m+1}^{\text{ctx}}; \tilde{\mathbf{p}}_m \mid \mathbf{h}_m^{\text{ctx}})$ ▷ Transfer entropy between global prototype and local pattern
    $\widehat{\text{TE}} \leftarrow \mathbb{E}_{\mathbb{P}}[T_\theta(\cdot)] - \log \mathbb{E}_{\mathbb{P} \times \mathbb{Q}}[e^{T_\theta(\cdot)}]$ ▷ MINE estimator
    $\mathcal{L}_{\text{TE}} \leftarrow -\widehat{\text{TE}}$ ▷ Maximize transfer entropy
  else
    $\mathcal{L}_{\text{TE}} \leftarrow 0$
  end if
  Stage 7: Joint Optimization
  $\mathcal{L} \leftarrow \mathcal{L}_{\text{BCE}} + \lambda_{\text{TE}} \cdot \mathcal{L}_{\text{TE}}$ ▷ Total objective
  $\Theta \leftarrow \Theta - \eta \nabla_\Theta \mathcal{L}$ ▷ Gradient descent update
end for

yielding $O(D^2)$ per patch. Aggregating over all patches and nodes gives:

$$O(BMD^2). \tag{17}$$

Stage 5. The link prediction stage typically only involves simple vector inner product calculations, with a computational complexity of $O(D)$ per calculation. The overall complexity is $\mathcal{C}_{stage5} = O(BD)$, which can be safely and entirely neglected compared to other stages.

Stage 6. In the transfer entropy regularization stage, we construct a triplet based on adjacent patches and estimate it via the MINE network. If the width of the discriminative network is of the same order as D, the complexity of a single forward propagation is $O(D^2)$, leading to the overall complexity:

$$O(BMD^2). \tag{18}$$

Finally, by combining the complexities, the computational complexity of the training process as:

$$O(BLD + BTM^2D + BTMD^2 + BMKD + BMD^2). \tag{19}$$

Substituting $L = pM$ and ignoring lower-order terms can be simplified to:

$$O(BTM^2D + B(T+1)MD^2 + BMKD). \tag{20}$$

In practice, the quadratic self-attention term $O(BTM^2D)$ dominates. Therefore, TE-MDG has the same asymptotic complexity as Transformer-based models such as DyGFormer, while the transfer entropy module introduces only linear overhead, without changing the overall complexity order.

We further validate the time complexity analysis by comparing the training time of TE-MDG with multiple representative baselines on multiple datasets. The results are presented in Table 3. Overall, TE-MDG introduces only moderate time overhead compared to existing Transformer-based methods. Specifically, compared to DyGFormer, TE-MDG exhibits consistent but limited increases in training time across all datasets. For instance, on the *tgbl-wiki*, the training time increases from 37.62s to 41.13s, and on the *MOOC*, it increases from 191.42s to 221.68s. Similar trends are observed on other datasets, indicating that the additional overhead introduced by the modules is stable and controllable. This additional overhead primarily stems from the global prototype assignment and transfer entropy regularization, which exhibit linear computational complexity related to the number of patches M, consistent with the theoretical complexity analysis mentioned above.

Compared to lightweight models such as JODIE and GraphMixer, TE-MDG has a relatively longer training time due to its Transformer-based temporal modeling architecture. However, such models often trade off modeling capacity for higher efficiency, resulting in relatively weaker overall performance. In contrast, TE-MDG introduces only limited additional computational overhead while maintaining strong expressiveness, achieving a favorable balance between performance and efficiency.

Table 3: Training time per epoch (seconds) comparison across different models on multiple datasets.

Model	tgbl-wiki	tgbl-review	tgbl-coin	Can·Parl	Enron	MOOC	UCI
TGN	21.39±1.49	13.06±0.17	32.51±1.01	47.44±0.60	68.33±0.89	252.57±2.19	40.50±0.51
TGAT	15.86±0.82	6.87±0.28	19.44±0.47	30.91±1.59	50.08±1.58	161.25±1.53	24.06±1.36
CAWN	23.65±0.16	11.84±0.37	26.55±0.03	36.47±1.96	56.88±6.87	162.08±2.24	27.92±5.46
GraphMixer	8.18±0.37	4.00±0.61	9.38±0.23	21.76±7.72	13.27±0.61	52.73±4.08	17.68±1.64
DyGFormer	37.62±1.30	15.37±0.23	46.67±0.48	38.68±0.39	59.45±2.56	191.42±3.77	29.88±0.29
JODIE	3.67±0.70	4.11±0.30	12.33±0.35	1.92±0.24	3.13±0.37	29.04±1.57	2.08±0.31
TE-MDG (ours)	41.13±0.59	18.49±0.11	49.56±1.18	42.65±1.24	64.13±0.80	221.68±9.77	32.14±0.26

B Method Details

B.1 Transfer Entropy in Dynamic Graphs

In the main paper, we construct a global temporal prototype representation $\tilde{p}_m$ and a local temporal representation $\mathbf{h}_m^{\text{ctx}}$ for each local temporal pattern. To enforce the guiding role of prototypes over local temporal patterns, we introduce the transfer entropy to quantify information gain from $\tilde{p}_m$ to $\mathbf{h}_m^{\text{ctx}}$. In this subsection, we provide a formal information-theoretic derivation of the transfer entropy.

Under the standard local Markov assumption, the evolution of a local representation is modeled as

$$p(\mathbf{h}_{m+1}^{\text{ctx}} \mid \mathbf{h}_m^{\text{ctx}}), \tag{21}$$

which implies that future states depend solely on its own history. However, in dynamic graphs, node behaviors should always follow global stable evolution patterns and avoid introducing irrelevant information. Using only its own local information may easily cause the representation to deviate from these stable patterns. Therefore, we introduce a global temporal prototype to serve as a guide, ensuring that local evolution stays closely aligned with the global regularities.

To incorporate global information, we extend the conditional distribution:

$$p(\mathbf{h}_{m+1}^{\text{ctx}} \mid \mathbf{h}_m^{\text{ctx}}, \tilde{p}_m). \tag{22}$$

The additional information provided by $\tilde{p}_m$ can be measured as the reduction in uncertainty:

$$H(\mathbf{h}_{m+1}^{\text{ctx}} \mid \mathbf{h}_m^{\text{ctx}}) - H(\mathbf{h}_{m+1}^{\text{ctx}} \mid \mathbf{h}_m^{\text{ctx}}, \tilde{p}_m). \tag{23}$$

For continuous variables, the conditional entropy is defined as:

$$H(\mathbf{h}_{m+1}^{\text{ctx}} \mid \mathbf{h}_m^{\text{ctx}}) = -\mathbb{E}_{p(\mathbf{h}_{m+1}^{\text{ctx}}, \mathbf{h}_m^{\text{ctx}})} \left[\log p(\mathbf{h}_{m+1}^{\text{ctx}} \mid \mathbf{h}_m^{\text{ctx}})\right], \tag{24}$$

$$H(\mathbf{h}_{m+1}^{\text{ctx}} \mid \mathbf{h}_m^{\text{ctx}}, \tilde{p}_m) = -\mathbb{E}_{p(\mathbf{h}_{m+1}^{\text{ctx}}, \mathbf{h}_m^{\text{ctx}}, \tilde{p}_m)} \left[\log p(\mathbf{h}_{m+1}^{\text{ctx}} \mid \mathbf{h}_m^{\text{ctx}}, \tilde{p}_m)\right]. \tag{25}$$

This difference is equivalent to the conditional mutual information:

$$I(\mathbf{h}^{\text{ctx}}_{m+1}; \tilde{p}_m \mid \mathbf{h}^{\text{ctx}}_m) = H(\mathbf{h}^{\text{ctx}}_{m+1} \mid \mathbf{h}^{\text{ctx}}_m) - H(\mathbf{h}^{\text{ctx}}_{m+1} \mid \mathbf{h}^{\text{ctx}}_m, \tilde{p}_m), \tag{26}$$

Using the marginalization relationship $p(\mathbf{h}^{\text{ctx}}_{m+1}, \mathbf{h}^{\text{ctx}}_m) = \int p(\mathbf{h}^{\text{ctx}}_{m+1}, \mathbf{h}^{\text{ctx}}_m, \tilde{p}_m)\, d\tilde{p}_m$, the above equation can be mathematically unified into a joint distribution form:

$$I(\mathbf{h}^{\text{ctx}}_{m+1}; \tilde{p}_m \mid \mathbf{h}^{\text{ctx}}_m) = \mathbb{E}_{p(\mathbf{h}^{\text{ctx}}_{m+1}, \tilde{p}_m, \mathbf{h}^{\text{ctx}}_m)} \left[\log \frac{p(\mathbf{h}^{\text{ctx}}_{m+1} \mid \tilde{p}_m, \mathbf{h}^{\text{ctx}}_m)}{p(\mathbf{h}^{\text{ctx}}_{m+1} \mid \mathbf{h}^{\text{ctx}}_m)} \right]. \tag{27}$$

By explicitly treating $\tilde{p}_m$ as an external driving variable, the above conditional mutual information directly corresponds exactly to the transfer entropy from $\tilde{p}_m$ to $\mathbf{h}^{\text{ctx}}_m$:

$$\text{TE}_{\tilde{p}_m \to \mathbf{h}^{\text{ctx}}_m} = I(\mathbf{h}^{\text{ctx}}_{m+1}; \tilde{p}_m \mid \mathbf{h}^{\text{ctx}}_m). \tag{28}$$

B.2 Neural Estimation for Transfer Entropy via MINE

Although transfer entropy can be expressed as conditional mutual information, its computation relies on the true joint distribution $p(\mathbf{h}^{\text{ctx}}_{m+1}, \tilde{p}_m, \mathbf{h}^{\text{ctx}}_m)$ and the conditional distribution $p(\mathbf{h}^{\text{ctx}}_{m+1} \mid \tilde{p}_m, \mathbf{h}^{\text{ctx}}_m)$, which are difficult to obtain directly in practice. Therefore, we introduce Mutual Information Neural Estimation (MINE) (Belghazi et al., 2018) to obtain a learnable approximation for transfer entropy.

The conditional mutual information can be written as a conditional KL divergence:

$$I(\mathbf{h}^{\text{ctx}}_{m+1}; \tilde{p}_m \mid \mathbf{h}^{\text{ctx}}_m) = \mathbb{E}_{p(\mathbf{h}^{\text{ctx}}_m)} \left[D_{\text{KL}} \left(p(\mathbf{h}^{\text{ctx}}_{m+1}, \tilde{p}_m \mid \mathbf{h}^{\text{ctx}}_m) \parallel p(\mathbf{h}^{\text{ctx}}_{m+1} \mid \mathbf{h}^{\text{ctx}}_m) p(\tilde{p}_m \mid \mathbf{h}^{\text{ctx}}_m) \right) \right]. \tag{29}$$

Define the joint and product distributions:

$$P = p(\mathbf{h}^{\text{ctx}}_{m+1}, \tilde{p}_m, \mathbf{h}^{\text{ctx}}_m), \quad Q = p(\mathbf{h}^{\text{ctx}}_{m+1} \mid \mathbf{h}^{\text{ctx}}_m) p(\tilde{p}_m \mid \mathbf{h}^{\text{ctx}}_m) p(\mathbf{h}^{\text{ctx}}_m). \tag{30}$$

Hence, the conditional mutual information can be expressed as:

$$I(\mathbf{h}^{\text{ctx}}_{m+1}; \tilde{p}_m \mid \mathbf{h}^{\text{ctx}}_m) = D_{\text{KL}}(P \,\|\, Q). \tag{31}$$

Following the Donsker–Varadhan representation, we have:

$$D_{\text{KL}}(P \,\|\, Q) \geq \mathbb{E}_P[T_\theta] - \log \mathbb{E}_Q[e^{T_\theta}]. \tag{32}$$

Here, T_θ is a neural network parameterized by θ that takes the triplet $(\mathbf{h}^{\text{ctx}}_{m+1}), \tilde{p}_m, \mathbf{h}^{\text{ctx}}_m$ and outputs a scalar. Thus, the transfer entropy admits the lower bound:

$$\text{TE}_{\tilde{p}_m \to \mathbf{h}^{\text{ctx}}_m} \geq \mathbb{E}_P \left[T_\theta(\mathbf{h}^{\text{ctx}}_{m+1}, \tilde{p}_m, \mathbf{h}^{\text{ctx}}_m) \right] - \log \mathbb{E}_Q \left[e^{T_\theta(\mathbf{h}^{\text{ctx}}_{m+1}, \tilde{p}_m, \mathbf{h}^{\text{ctx}}_m)} \right]. \tag{33}$$

In practice, the first term is directly estimated using observed samples from the joint distribution, while the second term is efficiently approximated via negative sampling by shuffling samples within mini-batches to approximate Q. The resulting neural estimator is:

$$\widehat{\text{TE}}_{\tilde{p}_m \to \mathbf{h}^{\text{ctx}}_m} = \mathbb{E}_P[T_\theta] - \log \mathbb{E}_Q[e^{T_\theta}]. \tag{34}$$

Maximizing this objective encourages stronger information flow from global temporal patterns to local evolution, thereby improving the model's ability to capture consistent dynamic behaviors.

Theoretical Guarantee. According to the Donsker–Varadhan variational characterization of the KL divergence, the proposed neural estimator $\widehat{\text{TE}}_{\tilde{p}_m \to \mathbf{h}^{\text{ctx}}_m}$ is a valid lower bound of the true transfer entropy. Moreover, when the statistic network has sufficient expressive capacity and reaches the optimum, the bound is tight, i.e., the estimator recovers the exact transfer entropy.

B.3 Distinction of Mutations

To formalize how the model distinguishes different types of mutations, we analyze the evolution of local temporal representations from a latent-variable perspective. Let $\{\mathbf{h}_m^{\text{ctx}}\}_{m=1}^{M}$ denote the sequence of context-aware local temporal representations. We introduce two latent random variables, $\delta_{rat}^{(m)}$ which captures rational mutations that are structurally consistent variations aligned with stable evolution patterns and $\delta_{\text{oca}}^{(m)}$ which captures occasional mutations that are stochastic perturbations unrelated to historical patterns. The transition dynamics of the representation are abstracted as:

$$\mathbf{h}_{m+1}^{\text{ctx}} \sim p\big(\cdot \mid \mathbf{h}_m^{\text{ctx}}, \delta_{\text{rat}}^{(m)}, \delta_{\text{oca}}^{(m)}\big), \tag{35}$$

which factorizes the influences on the next state without committing to a specific generative form. We further assume that rational mutations are conditionally dependent on the stable evolution patterns, while occasional mutations are conditionally independent, i.e.,

$$\delta_{\text{rat}}^{(m)} \not\perp \tilde{p}_m \mid \mathbf{h}_m^{\text{ctx}}, \qquad \delta_{\text{oca}}^{(m)} \perp \tilde{p}_m \mid \mathbf{h}_m^{\text{ctx}}, \tag{36}$$

where $\tilde{p}_m$ denotes the global temporal prototype representation. Under these assumptions, consider the transfer entropy $I(\mathbf{h}_{m+1}^{\text{ctx}}; \tilde{p}_m \mid \mathbf{h}_m^{\text{ctx}})$, which quantifies the reduction in uncertainty of the future state given access to the global temporal prototype. Since $\mathbf{h}_{m+1}^{\text{ctx}}$ is jointly determined by $(\delta_{\text{rat}}^{(m)}, \delta_{\text{oca}}^{(m)})$, applying the chain rule of mutual information yields:

$$I(\mathbf{h}_{m+1}^{\text{ctx}}; \tilde{p}_m \mid \mathbf{h}_m^{\text{ctx}}) \le I(\delta_{\text{rat}}^{(m)}, \delta_{\text{oca}}^{(m)}; \tilde{p}_m \mid \mathbf{h}_m^{\text{ctx}}). \tag{37}$$

Assuming $\delta_{\text{rat}}^{(m)}$ and $\delta_{\text{oca}}^{(m)}$ are independent given $\mathbf{h}_m^{\text{ctx}}$, the right-hand side decomposes as:

$$I(\delta_{\text{rat}}^{(m)}, \delta_{\text{oca}}^{(m)}; \tilde{p}_m \mid \mathbf{h}_m^{\text{ctx}}) = I(\delta_{\text{rat}}^{(m)}; \tilde{p}_m \mid \mathbf{h}_m^{\text{ctx}}) + I(\delta_{\text{oca}}^{(m)}; \tilde{p}_m \mid \mathbf{h}_m^{\text{ctx}}). \tag{38}$$

Under the conditional independence assumption, $I(\delta_{\text{oca}}^{(m)}; \tilde{p}_m \mid \mathbf{h}_m^{\text{ctx}}) = 0$, leading to approximation:

$$I(\mathbf{h}_{m+1}^{\text{ctx}}; \tilde{p}_m \mid \mathbf{h}_m^{\text{ctx}}) \approx I(\delta_{\text{rat}}^{(m)}; \tilde{p}_m \mid \mathbf{h}_m^{\text{ctx}}). \tag{39}$$

This indicates that the transfer entropy primarily captures the specific component of evolution that is aligned with global temporal prototypes, i.e., the information carried by rational mutations, while being inherently insensitive to occasional mutations by design.

Consequently, maximizing transfer entropy $\text{TE} = I(\mathbf{h}_{m+1}^{\text{ctx}}; \tilde{p}_m \mid \mathbf{h}_m^{\text{ctx}})$ encourages the model to enhance the predictability of δ_{rat} while suppressing variations induced by δ_{oca}. In practice, we do not model these latent variables; instead, by optimizing a neural estimator of TE, we implicitly steer the representation learning process toward preserving mutations consistent with stable evolution patterns, thereby achieving a effective distinction between different types of mutations.

Table 4: Node classification performance (NDCG@10, %) comparison between TE-MDG and SOTA baselines on TGB datasets. The best results are highlighted in **bold**, and the second-best are underlined. Δ denotes the performance gap compared with TE-MDG on the average score.

Dataset	TGN	TGAT	CAWN	GraphMixer	DyGFormer	JODIE	DyREP	TCL	TE-MDG(ours)
tgbn-trade	37.73±0.68	37.29±0.21	37.22±0.04	37.34±0.06	**38.61±0.61**	37.53±0.46	37.17±0.08	37.20±0.03	38.38±0.11
tgbn-genre	32.65±0.33	34.21±0.27	34.92±0.08	34.91±0.05	35.10±0.41	35.12±0.31	33.82±0.31	33.45±0.12	**37.91±0.05**
tgbn-reddit	**33.08±0.59**	31.22±0.06	31.24±0.04	31.20±0.03	31.28±0.12	30.56±0.03	30.49±0.11	31.08±0.04	31.42±0.03
Average	34.49	34.24	34.46	34.48	34.99	34.40	33.83	33.91	**35.90**
Δ	↑1.41	↑1.66	↑1.44	↑1.42	↑0.91	↑1.50	↑2.07	↑1.99	–

C Additional Experiments

C.1 Additional Experiments on Node Classification

To further evaluate the effectiveness and generalization of TE-MDG, we conduct additional experiments on the node classification task. The results are summarized in Table 4.

Obs. 6. TE-MDG outperforms all baselines in average performance, demonstrating strong effectiveness. Specifically, TE-MDG achieves the highest average NDCG@10 across three datasets, reaching 35.90%, an improvement of 0.91% compared to the strongest baseline DyGFormer. On tgbn-genre, the performance improvement is particularly prominent, with a 2.79% increase compared to the optimal baseline. In contrast, on tgbn-trade and tgbn-reddit, although TE-MDG does not achieve the best performance, it maintains competitiveness with a smaller gap compared to the strongest baseline. These results indicate that TE-MDG is capable of learning stable and effective representation learning across different tasks and dynamic graph scenarios.

C.2 Additional Ablation Study

To further validate the necessity of components in TE-MDG, we conduct comprehensive ablation studies. Specifically, we provide: (i) additional results on the link prediction task, including MRR metrics (Table 5) and visualizations of AUC/AP (Figure 4); (ii) ablation results on the node classification task (Table 6). These supplementary experiments consistently support the key findings in the main paper: the transfer entropy regularization and global temporal prototypes play complementary and indispensable roles, and their relative importance varies across data distributions and tasks.

Table 5: **Ablation study** of TE-MDG components on the link prediction task (MRR, %). Each entry reports mean ± std, with the relative change (Δ) compared to the full model shown below.

Model	TGB Datasets			DyGLib Datasets			
	tgbl-wiki	tgbl-review	tgbl-oin	Can·Parl	Enron	MOOC	UCI
w/o TE	74.22± 0.55	33.44± 0.99	47.08± 1.08	17.36± 2.08	43.87± 1.64	30.76± 0.55	78.22± 1.19
	↑0.75	↓1.68	-	↑0.59	↓5.58	↓3.06	↓1.01
w/o S	70.88± 0.58	23.80± 0.94	38.50± 1.73	15.93± 1.47	45.51± 1.95	31.16± 0.19	77.04± 0.15
	↓2.59	↓11.32	↓8.58	↓0.84	↓3.94	↓2.66	↓2.19
TE-MDG	73.47± 2.03	35.12± 1.07	47.08± 0.88	16.77± 5.75	49.45± 0.54	33.82± 0.76	79.23± 0.21

C.2.1 Supplement to Ablation Study on Link Prediction

Obs. 7. The complementary roles of transfer entropy and global temporal prototypes are further validated on the MRR metric, while the visualizations clearly highlight the differential contributions of each component across data distributions. On most datasets, the full model achieves the highest or comparable MRR. Removing global temporal prototypes (w/o S) leads to significant performance degradation on all datasets, especially on TGB datasets, with a decrease of 11.32% on tgbl-review and 8.58% on tgbl-coin. Removing only the transfer entropy regularization (w/o TE) also causes significant performance drop on DyGLib datasets such as Enron, MOOC, UCI decreasing by 5.58%, 3.06%, and 1.01%, respectively, while the impact on TGB datasets is minor or even slightly improved (e.g., a 0.75% increase on tgbl-wiki), but this improvement is within the standard deviation range. Observing Figure 4, it is found that on TGB datasets which have long temporal spans and complex dynamics, the performance degradation caused by w/o S is much greater than that caused by w/o TE; while on DyGLib datasets which are smaller in scale and relatively stable, w/o TE leads to a larger drop, highlighting that the dominant role of each component varies with data distribution. These results are consistent with the findings in the main paper: the two modules play different but complementary roles, and their relative importance depends on data characteristics.

C.2.2 Ablation Study on Node Classification

Obs. 8. The indispensability and complementarity of the two components are further validated on the node classification task. As shown in Table 6, removing either component generally leads to performance degradation, confirming that both contribute to effective node representation learning. Specifically, on tgbn-trade, removing the global temporal prototype (w/o S) results in a larger performance drop (8.90%) than removing the transfer entropy regularization (w/o TE, 5.70%), indicating that modeling global temporal prototypes plays a more significant role in this dataset setting. In contrast, on tgbn-reddit, removing TE even leads to a slight performance increase of 1.60%, while removing S results in only a marginal change of 0.14%. This suggests that on certain datasets, the introduction of transfer entropy regularization may not yield substantial performance

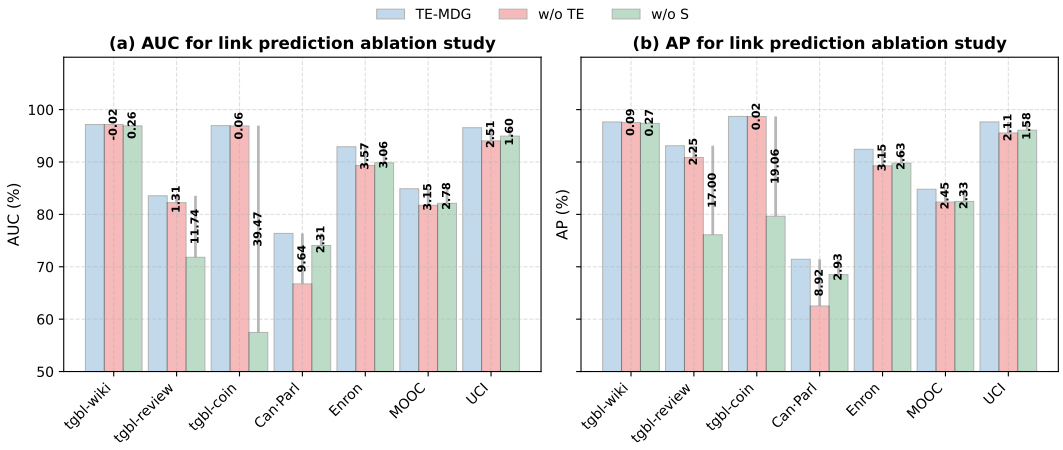


Figure 4: AUC (a) and AP (b) bar charts for the link prediction ablation study. The full TE-MDG model is compared against two ablated variants: w/o TE (without transfer entropy regularization) and w/o S (without global temporal prototypes) across seven datasets.

gains, yet the overall stability of the model remains unaffected. Overall, these results indicate that the two modules play distinct roles in the model: global temporal prototypes capture stable structures across time scales, while the transfer entropy regularization enforces local temporal consistency. Their relative contributions vary across different datasets, but both are indispensable.

Table 6: Ablation study on node classification (NDCG@10, %). Δ denotes the performance change.

Model	tgbn-trade		tgbn-genre		tgbn-reddit	
	NDCG@10	Δ	NDCG@10	Δ	NDCG@10	Δ
w/o TE	$32.68_{\pm 2.88}$	↓5.70	$34.84_{\pm 0.03}$	↓3.07	$33.02_{\pm 0.23}$	↑1.60
w/o S	$29.48_{\pm 3.32}$	↓8.90	$34.97_{\pm 0.11}$	↓2.94	$31.28_{\pm 0.12}$	↓0.14
TE-MDG (full)	$\mathbf{38.38}_{\pm 0.11}$	–	$\mathbf{37.91}_{\pm 0.05}$	–	$\mathbf{31.42}_{\pm 0.03}$	–

C.3 Hyperparameter Sensitivity Experiments

We perform sensitivity analysis on two key hyperparameters in TE-MDG: the weight of transfer entropy regularization λ_{TE} and the number of global temporal prototypes k.

C.3.1 The Weight of TE Loss

We comprehensively evaluate the hyperparameter λ_{TE} with values $\{0.01, 0.05, 0.1\}$, and the corresponding results are shown in Table 7. The main observations are as follows.

Obs. 9. The performance of TE-MDG is insensitive to variations in λ_{TE}, demonstrating remarkably good robustness and stability. When λ_{TE} increases from 0.01 to 0.1, the observed performance change of most datasets is relatively limited. For example, on tgbl-wiki and tgbl-coin, the change in AUC/AP is typically within 0.5%, and the corresponding fluctuation of MRR remain similarly small. This indicates that the transfer entropy regularization term does not introduce additional instability and can smoothly integrate into the overall training objective.

Obs. 10. Smaller λ_{TE} typically leads to better performance, essentially mitigating the scale mismatch problem between different loss terms. On most datasets, λ_{TE} =0.01 achieves optimal or near optimal results. For example, on tgbl-review and MOOC, compared to λ_{TE} =0.1, AUC increases by 1.64% and 5.72%, respectively. This phenomenon is closely related to the numerical scale of the loss term. As shown in Figure 3, $\mathcal{L}_{\text{TE}}$ typically lies in the interval [3,6], while $\mathcal{L}_{\text{BCE}}$ is relatively small. When λ_{TE} is large, the transfer entropy term dominates the total loss, thereby weakening

Table 7: **Hyperparameter sensitivity analysis of λ_{TE} and number of prototypes k on AUC, AP, and MRR (%)**. Results are reported as mean ± std. Best results are highlighted in **bold**.

Dataset	Metric	λ_{TE}			k			
		0.01	0.05	0.1	20	10	5	2
TGB Datasets								
tgbl-wiki	AUC	**97.15±0.35**	97.02±0.29	96.87±0.92	96.60±1.07	97.05±0.17	96.47±0.70	**97.15±0.35**
	AP	97.64±0.33	**97.66±0.20**	97.09±1.22	**97.69±0.24**	97.64±0.32	97.56±0.05	97.64±0.33
	MRR	73.47±2.03	**74.07±0.70**	73.57±1.30	**74.63±0.90**	74.26±0.20	74.48±0.91	73.47±2.03
tgbl-review	AUC	**83.55±1.57**	82.93±0.63	81.91±1.12	82.58±6.04	**85.05±2.35**	80.27±4.39	83.55±1.57
	AP	**93.10±2.84**	92.96±2.65	91.25±0.08	90.69±2.54	**93.51±2.40**	89.32±2.53	93.10±2.84
	MRR	**35.12±1.07**	31.79±1.58	32.16±2.70	**36.78±2.00**	33.29±1.18	31.86±3.82	35.12±1.07
tgbl-coin	AUC	96.94±0.29	96.37±0.85	**97.12±0.12**	**97.03±0.09**	96.97±0.21	96.95±0.16	96.94±0.29
	AP	98.70±0.13	98.55±0.29	**98.72±0.09**	**98.72±0.09**	98.70±0.09	98.69±0.11	98.70±0.13
	MRR	47.08±0.88	**47.62±0.90**	46.93±0.66	**47.10±0.32**	46.85±0.28	46.80±0.50	47.08±0.88
DyGLib Datasets								
Can·Parl	AUC	**76.37±5.16**	73.29±0.22	72.43±2.35	76.21±2.45	**77.99±1.95**	75.38±4.34	76.37±5.16
	AP	**71.43±3.11**	68.98±0.40	68.79±1.07	70.71±2.52	**72.43±1.18**	70.23±3.19	71.43±3.11
	MRR	16.77±5.75	**18.64±2.84**	15.10±4.17	**17.65±3.61**	14.19±0.50	16.88±0.87	16.77±5.75
Enron	AUC	**92.91±0.07**	90.20±3.95	90.10±4.62	92.70±0.08	92.83±0.12	92.58±0.26	**92.91±0.07**
	AP	**92.42±0.04**	90.14±2.91	90.04±3.75	92.14±0.06	92.28±0.15	92.08±0.28	**92.42±0.04**
	MRR	**49.45±0.54**	46.52±2.76	46.71±4.06	48.55±0.74	48.66±1.09	48.71±1.13	**49.45±0.54**
MOOC	AUC	**84.89±0.63**	79.56±3.54	79.17±3.43	**85.09±0.26**	82.96±4.30	84.71±0.11	84.89±0.63
	AP	**84.80±0.54**	80.69±2.84	80.32±2.74	**85.08±0.24**	83.37±3.38	84.80±0.04	84.80±0.54
	MRR	33.82±0.76	**34.74±0.47**	33.66±0.74	**34.31±0.24**	34.08±1.11	34.01±0.90	33.82±0.76
UCI	AUC	**96.54±0.24**	94.12±0.03	94.03±0.05	94.25±0.09	94.25±0.24	94.26±0.02	**96.54±0.24**
	AP	**97.65±0.12**	95.46±0.02	95.41±0.03	95.58±0.06	95.60±0.12	95.59±0.03	**97.65±0.12**
	MRR	**79.23±0.21**	76.84±0.21	76.31±0.13	77.27±0.57	77.09±0.56	77.19±0.09	**79.23±0.21**

$\mathcal{L}_{\text{BCE}}$ on model parameter updates leads to an imbalance in optimization objectives. In contrast, a smaller λ_{TE} can achieve better performance by introducing global temporal prototype constraints while maintaining the effectiveness of the supervised signal.

C.3.2 The Number of Global Prototypes

In practice, we consider $k \in \{2, 5, 10, 20\}$. The corresponding results are reported in Table 7.

Obs. 11. TE-MDG is robust to k, and a small number of prototypes is sufficient to capture the dominant temporal structures. Within the range $k \in \{2, 5, 10, 20\}$, the model maintains stable performance across datasets. In particular, $k = 2$ achieves the best results on most datasets. For example, on UCI, $k = 2$ improves MRR to 79.23%, compared to around 77% for other values of k, while maintaining strong AUC and AP. This indicates that a small number of global temporal prototypes is sufficient to effectively capture the dominant evolution patterns. Furthermore, we observe that on certain datasets (e.g., MOOC), larger values such as $k = 20$ yield the best performance. This is closely related to the underlying interaction patterns of the dataset. Specifically, MOOC models many-to-many interactions between students and course content, where courses typically span multiple knowledge domains and exhibit high heterogeneity. As a result, a larger number of global temporal prototypes is required to adequately capture the diverse temporal dynamics.

D Implementation Details

D.1 Benchmark Datasets

We evaluate TE-MDG on ten datasets across two benchmarks, including TGB and DyGLib. The detailed statistics of these datasets are presented in Table 8 and Table 9.

TGB (Huang et al., 2023) ***(Temporal Graph Benchmark)*** is a standardized benchmark for dynamic graph representation learning, providing large-scale temporal graph datasets from diverse real-world domains along with unified evaluation protocols. It supports tasks such as link prediction and node

Table 8: Statistics of link prediction datasets

Datasets	tgbl-wiki	tgbl-review	tgbl-coin	Can·Parl	Enron	MOOC	UCI
Domains	Social	Interaction	Finance	Politics	Social	Interaction	Social
Nodes	9,227	352,637	638,486	734	184	7,144	1,899
Edges	157,474	4,873,540	22,809,486	74,478	125,235	411,749	59,835
Steps	152,757	6,865	1,295,720	14	22,632	345,600	58,911

classification, and emphasizes reproducibility and realistic settings. All datasets are split into training, validation, and test sets in chronological order to ensure temporal consistency.

For the link prediction task, we use the following datasets:

• **tgbl-wiki:** a temporal interaction network derived from Wikipedia editing activities, where nodes represent users or pages, and edges denote timestamped edit events with associated textual features. The task is to accurately predict which page a user will edit at a given time.

• **tgbl-review:** an Amazon product review network spanning from 1997 to 2018, where nodes represent users or items, and edges denote timestamped review interactions. The task is to accurately predict which item a user will review at a given time.

• **tgbl-coin:** a cryptocurrency transaction network, where nodes correspond to wallet addresses and edges represent timestamped transfers between addresses. The task is to predict the destination address a given source address will interact with at a specific time.

For the node classification task, we use datasets:

Table 9: Statistics of node classification datasets

Datasets	tgbn-trade	tgbn-genre	tgbn-reddit
Domains	Interaction	Interaction	Interaction
Nodes	255	1,505	11,766
Edges	468,245	17,858,395	27,174,118
Steps	32	133,758	21,889,537

• **tgbn-trade:** a trade network based on agricultural exchanges between countries from 1986 to 2016. Nodes represent countries, and edges denote the total trade value between country pairs. The task is to predict the distribution of a country's future trade values with other countries.

• **tgbn-genre:** a user–genre interaction network constructed from music listening behavior, where nodes represent users and music genres, and edges denote timestamped listening events. The task is to rank the genres a user is most likely to listen to in the following week.

• **tgbn-reddit:** a user–subreddit interaction network from Reddit, where nodes represent users and subreddits, and edges indicate posting activities. The task is to rank the subreddits a user is most likely to engage with in the next week.

DyGLib (Yu et al., 2023) is a widely used library for dynamic graph representation learning that integrates multiple benchmark datasets with a unified data processing and evaluation pipeline. We select the following four representative datasets for link prediction:

• **Can·Parl:** a dynamic political network that records interactions between Canadian Members of Parliament (MPs) from 2006 to 2019. Nodes represent MPs, and edges are formed when two MPs both vote "yes" on the same bill.

• **Enron:** an email communication network capturing interactions among employees of the Enron corporation over three years.

• **MOOC:** a bipartite interaction network from online learning platforms, where nodes represent students and course content units, and edges denote access events.

• **UCI:** an online communication network where nodes represent university students and edges correspond to message exchanges.

Sample strategy. Due to the substantial computational and memory overhead of training on large-scale TGB datasets, we adopt a structure and time-aware sampling strategy to construct a compact subgraph. The sampling procedure preserves (i) high-importance nodes, (ii) global connectivity, and (iii) temporal distribution, thereby maintaining the evolution dynamics of the original graph.

Algorithm 2 Structure and Temporal-Aware Edge Sampling

Input: Edge set $\mathcal{E}$ with timestamps $\mathcal{T}$, ratio r, degree threshold δ, bridge ratio α, bins B
Output: Sampled edge indices $\mathcal{E}_s$

```
Compute node degrees d(v) from E and define core nodes V_c = {v | d(v) > δ}
Extract core edges E_c = {(u, v) ∈ E | u, v ∈ V_c}
Construct auxiliary graph G_c using a subset of E_c and identify bridge edges E_b
Set sampling budget N_s = ⌊r|E_c|⌋, with bridge portion N_b = ⌊αN_s⌋ and temporal portion
    N_t = N_s − N_b
Sample bridge edges E_b^s:
if |E_b| ≥ N_b then
    uniformly sample N_b edges from E_b
else
    take all E_b and supplement with top-k edges ranked by betweenness centrality
end if
Sort E_c by timestamps and partition into B temporal bins {B_i}_{i=1}^B
Sample temporal edges E_t^s by allocating N_t proportionally across bins, excluding E_b^s
Merge sampled edges E_s = E_b^s ∪ E_t^s and reorder by timestamps
return E_s
```

Formally, let $\mathcal{E}$ denote the full set of temporal edges, and $\mathcal{E}_s \subseteq \mathcal{E}$ denote the sampled subset. Our goal is to construct $\mathcal{E}_s$ such that it preserves key structural and temporal properties of the original graph. This can be viewed as the following constrained optimization problem:

$$\mathcal{E}_s = \arg \max_{\mathcal{E}' \subseteq \mathcal{E},\, |\mathcal{E}'| \leq r|\mathcal{E}|} \Big(\lambda_1 \mathcal{C}(\mathcal{E}') + \lambda_2 \mathcal{S}(\mathcal{E}') + \lambda_3 \mathcal{T}(\mathcal{E}') \Big), \tag{40}$$

where $\mathcal{C}(\cdot)$ measures global connectivity, $\mathcal{S}(\cdot)$ captures structural importance (e.g., degree-based node significance), and $\mathcal{T}(\cdot)$ denotes temporal coverage over the entire time span. The specific implementation is shown in Algorithm 2.

The proposed sampling strategy approximates the original dynamic graph from three complementary aspects. First, degree-based filtering identifies structurally important nodes, ensuring that high-activity regions are preserved. Second, prioritizing bridge edges (and high-betweenness edges when necessary) maintains global connectivity and prevents graph fragmentation. Third, temporal stratified sampling enforces balanced coverage across the entire time span, avoiding bias toward dense periods and preserving long-term temporal dependencies.

D.2 Baseline Details

We compare our method with representative dynamic graph learning baselines. In this section, we provide detailed descriptions of each baseline used in our comparison.

JODIE (Kumar et al., 2019) is a recurrent neural network-based model that captures the co-evolution of user and item embeddings through temporal interactions. It employs coupled RNNs to update node representations upon each interaction and introduces a temporal projection operator to estimate future embeddings. As a sequence-based method, JODIE models dynamic graphs as streams of interaction events.

TGAT (Xu et al., 2020) extends graph attention networks to continuous-time dynamic graphs. It introduces a functional time encoding mechanism based on Fourier features and incorporates temporal information into the attention computation. By aggregating time-aware neighbor information, TGAT produces time-dependent node embeddings in an inductive manner.

TGN (Rossi et al., 2020) is a general framework for temporal graph networks that leverages node-level memory modules to store historical information. It consists of four components: message generation, memory update, memory storage, and embedding computation. Messages derived from interactions are used to update node memories via recurrent units, enabling effective modeling of temporal dependencies.

CAWN (Wang et al., 2022) captures high-order structural and temporal dependencies via temporal random walks. It introduces Causal Anonymous Walks (CAWs), which encode structural patterns

independent of node identities. These walks are then processed by sequence models to generate node representations, allowing CAWN to effectively model motif-level temporal structures.

GraphMixer (Cong et al., 2023) adopts an MLP-Mixer architecture for dynamic graph representation learning. Instead of relying on attention or recurrence, it separately mixes information across the token (neighbor) dimension and the feature dimension using multilayer perceptrons. This design achieves competitive performance with improved computational efficiency.

DyGFormer (Yu et al., 2023) is a transformer-based model that represents temporal interactions as sequences of patches. By segmenting interaction histories into temporal patches and applying self-attention over them, DyGFormer captures long-range temporal dependencies. It has demonstrated strong performance on multiple dynamic graph benchmarks.

DyREP (Trivedi et al., 2019) is a dynamic graph representation learning framework that jointly models temporal point processes and node embeddings. It introduces a temporal attention mechanism to capture the intensity of future interactions and updates node representations upon each event. It effectively disentangles structural and temporal dynamics, enabling prediction of both link occurrences and timestamps.

TCL (Wang et al., 2021) introduces a topology-aware Transformer framework for dynamic graph representation learning, where temporal interactions are encoded through a self-attention mechanism augmented with structural biases to capture graph topology. The model adopts a two-stream encoder that separately models temporal dynamics and structural dependencies, enabling more effective integration of time-evolving patterns and local graph context. To further enhance representation quality, TCL employs a contrastive learning objective that maximizes the agreement between different temporal views of the same node while distinguishing them from negative samples. This design allows TCL to learn robust and expressive node embeddings for downstream tasks such as link prediction and node classification.

DIDA (Zhang et al., 2022) is designed to handle spatio-temporal distribution shifts by leveraging invariant patterns. It introduces a disentangled spatio-temporal attention network to separate variant and invariant structures and features, a spatio-temporal intervention mechanism that samples and reassembles variant patterns across neighborhoods and timestamps to construct interventional distributions, and an invariance regularization that minimizes prediction variance across these distributions, thereby forcing the model to rely on stable invariant patterns for robust predictions under distribution shifts.

D.3 Evaluation Metrics

To comprehensively evaluate the performance of TE-MDG on different tasks, we employ multiple metrics. Specifically, for the link prediction task, where the model needs to predict whether a new edge will form between nodes u and v at future time step t, we use Area Under the ROC Curve (AUC), Average Precision (AP), and Mean Reciprocal Rank (MRR) to measure the model's ranking and discriminative ability. For the node classification task, where the model assigns category labels or behavior types to nodes, we use Normalized Discounted Cumulative Gain (NDCG) to assess the ranking quality of the model's top predictions.

Area Under the ROC Curve (AUC) measures the model's overall ranking ability over positive and negative samples. For link prediction, it reflects whether the model can correctly distinguish edges that will form $e^+ \in E_t$ (positive samples) from those that will not form $e^- \notin E_t$ (negative samples) at future time step t. It is defined as:

$$AUC = \mathbb{P}(f(u, v, t) \text{ for } e^+ > f(u', v', t) \text{ for } e^-), \tag{41}$$

where $f(u, v, t)$ denotes the model's predicted score for node pair (u, v) at time t. A higher AUC indicates stronger ranking ability over the entire set of future edges.

Average Precision (AP) is the area under the precision-recall curve, emphasizing the importance of ranking positive samples at the top. In our context, it measures whether the model can place the most likely edges at the front. It is defined as:

$$\text{AP} = \frac{1}{R} \sum_{k=1}^{N} P(k) \cdot \text{Rel}(k), \tag{42}$$

where R is the total number of relevant samples, $P(k)$ is the precision at cutoff k, and $\text{Rel}(k) \in \{0, 1\}$ indicates whether the k-th predicted edge is a positive sample.

Mean Reciprocal Rank (MRR) focuses on the position of the first positive sample in the ranking for each query. In dynamic graph link prediction, the query set $\mathcal{Q}$ can be defined as the candidate edge set E_t^{cand} at future time step t. MRR measures whether the model can return the most likely future edge as early as possible. It is defined as:

$$\text{MRR} = \frac{1}{|\mathcal{Q}|} \sum_{(u,v,t)\in\mathcal{Q}} \frac{1}{\text{rank}_{(u,v,t)}}, \tag{43}$$

where $\text{rank}_{(u,v,t)}$ is the position of the first true edge $(u, v) \in E_t$ in the model's predicted ranking. A higher MRR indicates that the model can quickly pinpoint the most important future edge.

Normalized Discounted Cumulative Gain (NDCG) measures the quality of the top-k predictions, considering both relevance and position importance. NDCG@k evaluates whether the model correctly classifies the most important nodes among its top-k predictions. It is defined as:

$$\text{NDCG@}k = \frac{\text{DCG@}k}{\text{IDCG@}k}, \quad \text{where DCG@}k = \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{\log_2(i+1)}, \tag{44}$$

and IDCG@k is the ideal DCG value. Here, rel_i denotes the relevance label of the i-th node in the prediction list. This metric applies a logarithmic discount to ranking positions, giving higher contributions to top-ranked nodes, thus reflecting the model's accuracy in classifying critical nodes.

D.4 TE-MDG Setup

We introduce two key hyperparameters, λ_{TE} and k, which control the weight of the transfer entropy regularization term and the number of global temporal prototypes, respectively. Specifically, λ_{TE} balances the contribution between the primary task objective and the transfer entropy constraint, while k determines the capacity of the model to capture the global evolution skeleton.

In all experiments, to ensure fairness and rationality, we uniformly set λ_{TE}=0.01 and k=2. These values are selected based on preliminary validation, yielding stable and competitive performance across different datasets. In particular, a relatively small λ_{TE} introduces effective directional regularization without disrupting the optimization of the main objective, while a small k is sufficient to capture the dominant global evolution patterns, avoiding overfitting caused by redundant prototypes.

We further evaluate the sensitivity of the model to these hyperparameters in the Appendix C.3 , where the results demonstrate that the performance remains stable within a reasonable range, indicating good robustness of the chosen configuration. All the above experiments are conducted on an NVIDIA GeForce RTX 3090 GPU with 24GB of memory.

E Related Works

E.1 Continuous-time Dynamic Graph Representation Learning

Continuous-time dynamic graphs (CTDGs) (Kazemi et al., 2020; Barros et al., 2021; ENNADIR et al., 2025) represent interactions with precise real-valued timestamps, enabling fine-grained modeling of dynamics. Existing CTDG methods can be broadly categorized into three types. Memory-based methods maintain evolving node states to capture long-term dependencies, as exemplified by DyRep (Trivedi et al., 2019), JODIE (Kumar et al., 2019), and TGN (Rossi et al., 2020), with recent extensions incorporating message passing into memory updates (Li et al., 2024). Sequence-based methods model historical interactions directly as temporal sequences using attention or MLP-based architectures, including TGAT (Xu et al., 2020), GraphMixer (Cong et al., 2023), and DyGFormer (Yu et al., 2023), along with subsequent improvements on multi-scale modeling and temporal encoding (Xu et al., 2024; Peng et al., 2025; Wang et al., 2025b; Jahin et al., 2026). Hybrid approaches combine these paradigms to address their individual limitations, such as STGN_h (Xu et al., 2025), temporal point process-based methods (Chen et al., 2025; Ying et al., 2025; Gracious & Dukkipati,

2024), and hyperbolic models like Hercules (Montella et al., 2021). Despite their success, these methods generally treat historical interactions uniformly, lacking mechanisms to distinguish stable evolution patterns from transient changes, which motivates our proposed TE-MDG framework.

E.2 Transfer Entropy

Transfer entropy (TE), introduced by Schreiber (Schreiber, 2000), is an information-theoretic measure that quantifies directional information flow between variables and can capture causal or predictive relationships. It has been widely used in neuroscience (Vicente et al., 2011; Munoz et al., 2025; Runge et al., 2019), finance (Barnett et al., 2009), and complex systems analysis (Hlaváčková-Schindler et al., 2007). Recent advances in differentiable neural estimation of TE have enabled its integration into deep learning frameworks. Belghazi et al. (Belghazi et al., 2018) proposed MINE, which estimates mutual information using neural networks based on the Donsker-Varadhan representation. Luxembourg et al. (Luxembourg et al., 2025) introduced TREET, an attention-based method for TE estimation in stationary processes. Galeano Munoz et al. (Munoz et al., 2025) proposed TENDE, which leverages diffusion models for flexible TE estimation. Information-theoretic measures have also been used to analyze representation learning (Tishby & Zaslavsky, 2015; Shwartz-Ziv & Tishby, 2017) and understand information flow in neural networks (Moldovan et al., 2023). In graph learning, Moldovan et al. (Moldovan et al., 2024) first introduced TE into graph convolutional networks to measure directional information flow and alleviate over-smoothing. Wang et al. (Wang et al., 2024b) proposed ANGCN, which uses TE to adaptively weight neighbors in heterogeneous graphs. Another related line of work is the information bottleneck theory. Kalajdzievski et al. (Kalajdzievski et al., 2023) proposed the transfer entropy bottleneck (TEB) to constrain information flow in sequence-to-sequence models. However, existing works have not incorporated transfer entropy as a regularization constraint in continuous-time dynamic graph representation learning to distinguish the heterogeneous contributions of local mutations to long-term evolution. To the best of our knowledge, TE-MDG is the first framework that uses TE to constrain directional information flow and information gain between global temporal prototypes and local temporal patterns.

E.3 Modeling Changes in Dynamic Graphs

In real-world dynamic graphs, node behaviors often involve both natural evolution and occasional changes, and distinguishing predictive changes from random noise remains a key challenge. Existing studies on modeling such changes can be broadly grouped into several directions. Dynamic graph anomaly detection methods focus on identifying abnormal nodes or edges, such as GDDyAD (He et al., 2025), RustGraph (Guo et al., 2024), and HADF (Liu et al., 2025), with subsequent works improving robustness through multi-view modeling, diffusion mechanisms, and Transformer-based designs (Wang, 2026; Chen et al., 2023; Dong et al., 2025), while other approaches further explore motif-level and spatio-temporal modeling (Huang et al., 2025; Yuan et al., 2023; Liu et al., 2023). Graph purification methods instead aim to remove noisy edges directly, as explored by TiGer (Jo et al., 2025), EPM Wang et al. (2025a), and GraphDiffusion (Jeong et al., 2025). Another line of work focuses on robust temporal graph learning, enhancing model stability under noisy or incomplete data through approaches such as EvoluNet (Wang et al., 2024a), RDGSL (Zhang et al., 2023), and DGIB (Yuan et al., 2024), with extensions like DDGIB (Wang et al., 2026) and related methods such as GraphFlow (You et al., 2025) further improving robustness via information bottleneck and neighbor selection mechanisms. In parallel, concept drift and evolutionary pattern modeling methods aim to adapt to distribution shifts and capture long-term patterns, including GraphDrift-net (Patil et al., 2025), CGLM (Zhou & Lu, 2025), and MSGR (Zhou et al., 2024), while other works focus on pattern discovery, such as SiGNN (Chen et al., 2024) and GINTRIP (Royat et al., 2025). Despite their differences, these methods generally treat changes as noise or fail to distinguish their heterogeneous roles, without explicitly preserving changes that are predictive of future dynamics.

F Limitations

Our work focuses solely on mutations occurring in dynamic graphs, such as changes in node interactions or features. It cannot be directly applied to other data modalities, such as time series from audio signals, where mutations have different characteristics. In the future, we would adapt

our mutation filtering approach to handle non-graph sequential data by redefining stable evolution patterns and transfer entropy.

G BROADER IMPACTS

This paper aims to advance the field of dynamic graph representation learning by proposing a novel framework that selectively acquires information that is beneficial for representation learning. Our TE-MDG framework effectively distinguishes the differential contributions of mutations during representation learning, enabling the model to focus on rational mutations while filtering out occasional ones. This capability has positive social implications by improving the trustworthiness and transparency of machine learning applications in critical domains such as financial fraud detection, user behavior analysis, and social network monitoring. Furthermore, we anticipate that our work will not pose direct social or ethical negative impacts.