Temporal Graph Pattern Machine

Yijun Ma [1 *] **Zehong Wang** [1 *] **Weixiang Sun** [1] **Yanfang Ye** [1]

Abstract

Temporal graph learning is pivotal for deciphering dynamic systems, where the core challenge lies in explicitly modeling the underlying evolving patterns that govern network transformation. However, prevailing methods are predominantly task-centric and rely on restrictive assumptions—such as short-term dependency modeling, static neighborhood semantics, and retrospective time usage. These constraints hinder the discovery of transferable temporal evolution mechanisms. To address this, we propose the Temporal Graph Pattern Machine (TGPM), a foundation framework that shifts the focus toward directly learning generalized evolving patterns. TGPM conceptualizes each interaction as an *interaction patch* synthesized via temporally-biased random walks, thereby capturing multi-scale structural semantics and long-range dependencies that extend beyond immediate neighborhoods. These patches are processed by a Transformer-based backbone designed to capture global temporal regularities while adapting to context-specific interaction dynamics. To further empower the model, we introduce a suite of self-supervised pre-training tasks—specifically masked token modeling and next-time prediction—to explicitly encode the fundamental laws of network evolution. Extensive experiments on temporal link prediction and temporal node classification show that TGPM consistently ranks among the top-performing methods, demonstrating exceptional cross-domain transferability. Our code has been released in https://github.com/antman9914/TGPM.

[1]University of Notre Dame. Correspondence to: Zehong Wang <zwang43@nd.edu>, Yanfang Ye <yye7@nd.edu>.

The Workshop on Graph Foundation Models at the 43rd International Conference on Machine Learning (ICML 2026), Seoul, South Korea. Copyright 2026 by the author(s).

1. Introduction

Real-world complex systems are inherently temporal: system states evolve continuously as events unfold, and future behaviors emerge as a consequence of historical dynamics (Kossinets & Watts, 2006; Cai et al., 2024). In such systems, events are neither independent nor random; instead, they are jointly shaped by historical context, interactions among entities, and temporal factors such as periodicity, burstiness, and other time-dependent conditions (Liguori et al., 2025). As a result, temporal systems exhibit structured evolution patterns that depend on both historical trajectories and relational contexts (Holme & Saramäki, 2012). Temporal graphs (Michail, 2016) provide a principled representation for modeling these systems by encoding entities as nodes and time-stamped interactions as temporal edges, thereby unifying relational structure with temporal dynamics. Modeling how temporal graphs evolve, and how future events are generated from historical and structural context, has consequently become a central problem in machine learning and data mining (Longa et al., 2023).

Although temporal graphs provide a principled modeling method for temporal systems, learning from them remains fundamentally challenging. The difficulty arises not primarily from data scale (Zheng et al., 2025) or computational complexity (Wang et al., 2024b), but from uncovering the underlying mechanisms that govern how temporal systems evolve (Pan et al., 2025b). To accurately understand and predict future behaviors in a temporal environment, models must move beyond optimizing predictive performance for individual downstream tasks and instead capture how events are generated through the interplay of historical context, relational structure, and temporal dynamics (Huang et al., 2025). From this perspective, temporal graph learning can be viewed as a form of *mechanism learning*, whose objective is to infer the latent processes driving system evolution rather than to fit task-specific correlations.

Despite their practical effectiveness, most existing temporal graph learning methods typically rely on several modeling assumptions that are fundamentally misaligned with the objective of this mechanism. *(1) Static neighborhood semantics assumption.* Most methods model temporal graph evolution primarily through one-hop neighborhood dynamics (Yu et al., 2023; Wu et al., 2024; Tian et al., 2024) .

However, neighborhood semantics in temporal systems are inherently non-stationary (Layne et al., 2023): the functional role and behavioral patterns of the same neighbor may change substantially as the relational structure evolves. Compressing such semantic dynamics into fixed temporal embeddings and sampling procedures therefore constrains a model's ability to learn evolution patterns (Feng et al., 2026; Wang et al., 2021c) that generalize across different graph instances and temporal regimes. *(2) Short-term dependency assumption.* Many approaches assume that recent neighbor interactions are sufficient to characterize a node's current state, and consequently restrict representation learning to local time windows or recent-neighbor sampling (Wang et al., 2021d; Xu et al., 2020; Cong et al., 2023; Yu et al., 2023). Although computationally efficient, this design choice systematically biases models toward short-term signals (Wang et al., 2021c) and limits their ability to capture long-term dependencies that often govern the evolution of real-world systems. *(3) Retrospective temporal modeling assumption.* In most existing methods, temporal modeling is predominantly retrospective (Yu et al., 2023; Wu et al., 2024), which means that temporal information is incorporated only through post-hoc conditioning on past events, where temporal signals are treated as auxiliary annotations of historical events. Such temporal annotations are exploited via time decay (Nguyen et al., 2018) and attention weighting (Xu et al., 2020) to modulate the influence of past events, but they are not treated as an explicit modeling target. Consequently, while these models can identify which historical events are relevant, they often fail to accurately characterize when future events will occur or to capture the temporal structures underlying system evolution. Together, these assumptions suggest that the central challenge of temporal graph learning lies not in improving performance on individual prediction tasks, but in understanding the underlying processes by which temporal systems evolve (Huang et al., 2025; Pan et al., 2025b).

To overcome these challenges, effective temporal graph modeling requires explicitly learning the evolving patterns that govern system dynamics. Such patterns correspond to generalizable generative mechanisms that characterize how historical dependencies, structural conditions, and temporal regularities jointly shape future events (Huang et al., 2025). Importantly, these evolution patterns are not tied to specific prediction tasks or individual graph instances; instead, they capture system-level regularities of temporal dynamics that recur across different domains and time horizons.

Motivated by these insights, we propose **Temporal Graph Pattern Machine (TGPM)**, a pattern-centric modeling framework that explicitly treats evolving patterns as the primary modeling target. Instead of learning task-specific representations tied to individual nodes or events, TGPM aims to capture generalizable temporal evolution mechanisms from temporal graphs. To this end, we design a modeling paradigm in which temporal contexts are represented as sequences of evolving patterns that jointly encode structural relationships, historical dependencies, and temporal information (C1). TGPM is trained using self-supervised objectives that encourage the model to capture temporal dependencies at multiple time scales (C2) and to explicitly reason about future event timing (C3). As a result, the learned representations support robust generalization across different tasks, time periods, and graph domains.

Our main contributions are summarized as follows: (1) **Pattern-centric modeling paradigm.** We propose a pattern-centric perspective for temporal graph learning that explicitly targets the modeling of evolving patterns governing system dynamics. This paradigm aims to capture generalizable temporal evolution regularities across different graphs, tasks, and time periods; (2) **Temporal Graph Pattern Machine (TGPM).** We introduce TGPM, a new temporal graph modeling framework that represents temporal contexts as sequences of interaction patterns constructed via temporally biased random walks and encodes them using a Transformer-based architecture. TGPM is trained with self-supervised objectives that jointly model multi-scale temporal dependencies and future event timing, enabling the learning of generalizable representations; (3) **Extensive empirical evaluation.** We conduct comprehensive experiments on temporal link prediction and temporal node classification over multiple real-world temporal graph benchmarks. The results demonstrate that TGPM consistently achieves the best overall average ranking and exhibits robust generalization across different scenarios.

2. Temporal Graph Pattern Machine

A temporal graph is denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X}, \mathbf{E})$, where $\mathcal{V}$ is the set of nodes with $|\mathcal{V}| = N$, and $\mathcal{E}$ is a sequence of chronologically ordered interactions $\{(u_1, v_1, t_1), (u_2, v_2, t_2), \ldots, (u_E, v_E, t_E)\}$ satisfying $0 \leq t_1 \leq t_2 \leq \ldots \leq t_E$. For each interaction, $u_i, v_i \in \mathcal{V}$ denote the source and destination nodes at timestamp t_i. $\mathbf{X}$ and $\mathbf{E}$ are the node and edge feature matrices, where each node $v \in \mathcal{V}$ is associated with a feature vector $\mathbf{x} \in \mathbb{R}^{d_n}$ and each edge $e \in \mathcal{E}$ with a feature vector $\mathbf{e} \in \mathbb{R}^{d_e}$.

As illustrated in Figure 1, TGPM models temporal contexts as sequences of *interaction patches*, each derived from evolving local substructures to capture dynamic neighborhood semantics. These interaction patches are then encoded and contextualized to form temporal representations, which are further optimized through self-supervised pre-training objectives, including masked token modeling and next time prediction, to capture multi-scale temporal dependencies and future event dynamics.

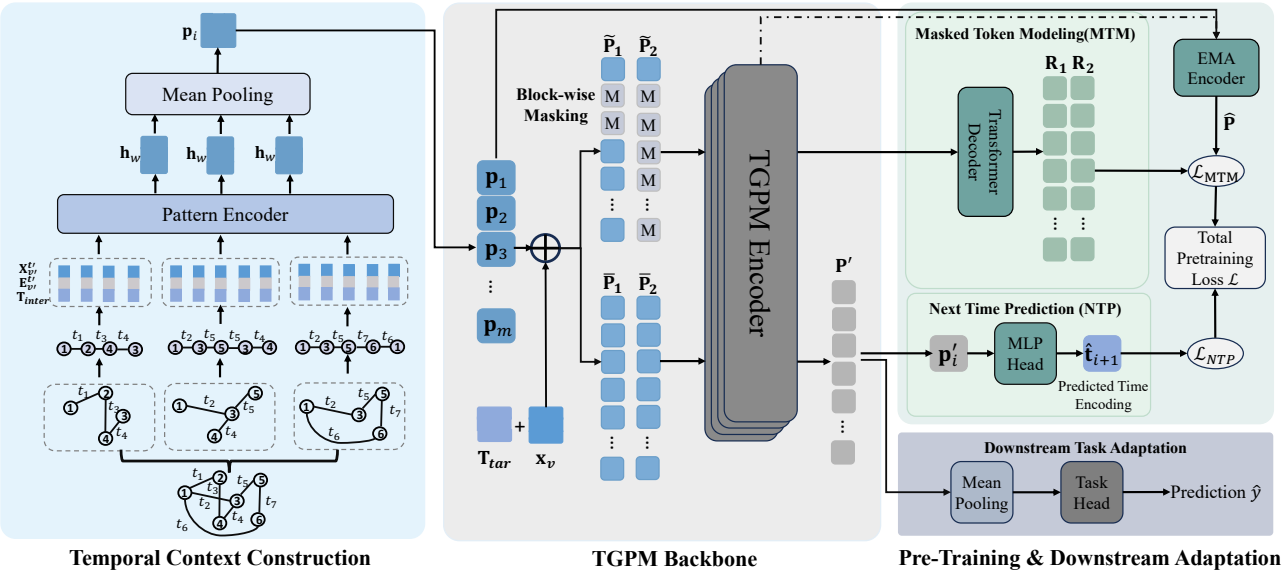


Figure 1. **Overview of TGPM**. (a) Temporal context of an interaction is represented by an interaction patch aggregated from a set of temporally biased random walks. (b) Transformer-based TGPM encoder adapt local structural and temporal semantics to target-specific temporal context, capturing global temporal regularities. (c) MTM and NTP pre-training enforce TGPM to learn from multi-scale temporal dynamics and the temporal rhythms of evolving patterns.

2.1. Temporal Context Representation

In common practice of temporal context construction (Rossi et al., 2020a; Yu et al., 2023) , given a target timestamp t, the temporal context of a target node v is represented as a sequence of first-hop interactions, denoted by $\mathcal{S}_v^t = \{(v, v', t') \mid t' < t\}$. Such representations are inherently node-centric and rely on temporal encodings to summarize past interactions. However, the semantic content of an interaction is not solely determined by the participating nodes and timestamps, but also by the evolving structural context in which it occurs (Layne et al., 2023). Compressing this information into node-level temporal features therefore leads to substantial loss of structural and semantic dynamics.

To overcome static neighborhood semantics assumption, we propose a pattern-centric modeling paradigm in which each interaction is represented by an *interaction patch*. Considering that temporal substructures can naturally reflect how local graph substructures and historical interactions co-evolve over time, we design interaction patches to summarize the evolving structural and temporal context associated with a specific interaction by aggregating multiple temporally contextualized substructures. Individual substructure captures only partial view of the evolving semantics, while aggregating multiple complementary substructures into an interaction patch can provide more stable and semantically meaningful representations of how interaction semantics evolve over time. Interaction patch serves as the basic semantic unit for modeling temporal contexts in TGPM and enables the representation of interactions beyond simple node-level histories.

Sampling Substructures via Temporal Random Walks. A key challenge in constructing interaction patches lies in how to systematically tokenize evolving relational structures into sequences of substructures. Direct enumeration or matching of subgraphs is computationally prohibitive (Sun et al., 2012; Zhao et al., 2025) and requires a predefined substructure vocabulary (Wang et al., 2024c). However, a suitable tokenization mechanism should remain efficient and scalable as temporal graphs evolve, and meanwhile preserve diverse substructures and temporal dependencies. Temporal random walk sampling naturally satisfies these principles by approximating substructure matching through scalable time-aware stochastic traversal. As a result, we propose to construct interaction patches using *temporally biased random walks*. These walks provide a scalable approximation to atomic evolution patterns while bypassing the need for explicit substructure enumeration (Nguyen et al., 2018). Moreover, they naturally encode both historical dependencies and structural relations within a unified sequential representation.

For each interaction (v, v', t'), we generate a set of contextualized walks rooted at node v'. To avoid data leakage, ensure pattern diversity, and preserve contextual dependency, the sampling process follows two principles: (1) *Retrospective sampling*: all sampled edges are considered as undirected edges and must have timestamps earlier than t'; and (2) *Temporal recency bias*: edges closer to t' are assigned higher sampling priority, following the time-decay assumption widely adopted in dynamic processes (Hawkes, 1971; Kalman, 1960). Unlike traditional chronologically

monotonic temporal walks (Wang et al., 2021d; Nguyen et al., 2018), we relax strict causal ordering and allow more complex temporal dependencies.

Formally, let $\mathcal{T} : \mathcal{E} \to \mathbb{R}^+$ denote the edge timestamp mapping function. Given an anchor interaction (v, v', t'), a temporally biased random walk w of length L is defined as a node sequence $w = (v_0, v_1, \ldots, v_L)$ with $v_0 = v'$, generated by the following Markov process:

$$\eta(u, v) = \begin{cases} \exp(t' - \mathcal{T}(u, v)), & (u, v) \in \mathcal{E},\ \mathcal{T}(u, v) < t' \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

$$P(v_{i+1} \mid v_0, v_1, \ldots, v_i) = \frac{\eta(v_i, v_{i+1})}{\sum_{u \in \mathcal{V}} \eta(v_i, u)}. \tag{2}$$

This transition rule prioritizes temporally close edges while allowing non-monotonic traversal of historical interactions.

Proposition 2.1. *Given a temporal random walk $w = (v_0, v_1, \ldots, v_L)$ with $\mathcal{T}(v_i, v_{i+1}) < t', i = 0, 1, \ldots, L-1$ and a causal walk $w_c = (u_0, u_1, \ldots, u_L)$ with $\mathcal{T}(u_{i-1}, u_i) < \mathcal{T}(u_i, u_{i+1}) < t', i = 1, 2, \ldots, L-1$, w can capture more complex evolving semantics than w_c.*

It is proved in Appendix C.1 that temporally non-monotonic random walks are more expressive than causally monotonic walks, able to encode temporal structures that are fundamentally inaccessible under causal constraints. This justify Proposition 2.1 and our proposed sampling principles.

Constructing Patch Embeddings via Aggregating Substructures. For each interaction (v, v', t'), we sample k temporally biased random walks (i.e., substructures) to construct its contextualized evolution patterns. For a walk w, we retrieve the corresponding edge sequence $w_e = ((v, v'), (v', v_1), \ldots, (v_{L-1}, v_L))$ and timestamp sequence $w_t = (t', \mathcal{T}(v', v_1), \ldots, \mathcal{T}(v_{L-1}, v_L))$. Based on w and w_e, we obtain the node and edge feature matrices $\mathbf{X}_{v'}^{t'} \in \mathbb{R}^{L \times d_n}$ and $\mathbf{E}_{v'}^{t'} \in \mathbb{R}^{L \times d_e}$ from $\mathbf{X}$ and $\mathbf{E}$, respectively. To encode temporal information, we adopt sinusoidal time-interval embeddings (Xu et al., 2020; Yu et al., 2023). Using t' as the anchor time, each timestamp t_i in w_t is converted to an interaction-relative interval $\Delta t = t' - t_i$ and encoded by time encoder T_{enc} as

$$T_{enc}(\Delta t) = \sqrt{\frac{1}{d_t}} \big[\cos(\omega_1 \Delta t), \sin(\omega_1 \Delta t), \ldots, \cos(\omega_{d_t} \Delta t), \sin(\omega_{d_t} \Delta t)\big], \tag{3}$$

where d_t is the encoding dimension and $\{\omega_i\}$ are trainable parameters. The resulting matrix is denoted by $\mathbf{T}_{inter} \in \mathbb{R}^{L \times d_t}$. The concatenated representation $\bar{\mathbf{H}}_w = [\mathbf{X}_{v'}^{t'} \| \mathbf{E}_{v'}^{t'} \| \mathbf{T}_{inter}]$ is projected via a linear layer and encoded by a Transformer encoder f:

$$\mathbf{h}_w = f(\mathbf{H}_w), \quad \mathbf{H}_w = \mathbf{W}\bar{\mathbf{H}}_w + \mathbf{b}, \tag{4}$$

where $\mathbf{h}_w$ denotes the pattern embedding of walk w.

Finally, the interaction patch embedding $\mathbf{p}$ for (v, v', t') is obtained by aggregating the embeddings of its k sampled walks via mean pooling: $\mathbf{p} = \frac{1}{k} \sum_w \mathbf{h}_w$. This interaction patch serves as a pattern-centric representation that summarizes the evolving structural and temporal semantics of the interaction and forms the basic input token for the TGPM backbone encoder.

2.2. TGPM Backbone

Given a target node v at time t, its temporal context is represented as a sequence of interaction patch embeddings $\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_m]$, where $m = |\mathcal{S}_v^t|$ and each $\mathbf{p}_i$ corresponds to the interaction patch of $(v, v_i', t_i') \in \mathcal{S}_v^t$. While interaction patches capture local structural and temporal semantics, they are invariant to absolute time shifts and lack target-specific temporal localization. To contextualize these patches, we adopt a Transformer backbone that models their temporal dependencies and relevance to the target node. We construct an *interaction token sequence*

$$\bar{\mathbf{P}} = [\mathbf{P} \,\|\, \mathbf{T}_{tar} \,\|\, \mathbf{x}_v],$$

where $\mathbf{T}_{tar} \in \mathbb{R}^{m \times d_t}$ denotes the target-relative time interval encodings derived by $T_{enc}(\cdot)$ with $\Delta t = t - t_i'$ as input, and $\mathbf{x}_v$ is the feature vector of node v.

Given $\bar{\mathbf{P}}$, a Transformer layer computes

$$\mathbf{Q} = \bar{\mathbf{P}}\mathbf{W}_Q, \quad \mathbf{K} = \bar{\mathbf{P}}\mathbf{W}_K, \quad \mathbf{V} = \bar{\mathbf{P}}\mathbf{W}_V, \tag{5}$$

$$\mathrm{Attn}(\bar{\mathbf{P}}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_{out}}}\right)\mathbf{V}, \tag{6}$$

$$\mathbf{P}' = \mathrm{FFN}\big(\bar{\mathbf{P}} + \mathrm{Attn}(\bar{\mathbf{P}})\big), \tag{7}$$

where $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V$ are trainable matrices and d_{out} is the output dimension. We employ multi-head attention (Vaswani et al., 2017) and stack multiple layers. The output of the final layer is $\mathbf{P}' = [\mathbf{p}_1', \mathbf{p}_2', \ldots, \mathbf{p}_m']$, which represents the contextual embeddings of interaction patches and is used for pre-training and downstream tasks.

2.3. TGPM Pre-Training

Although the proposed pattern-centric representation alleviates the limitations of node-centric temporal modeling, naive task-centric training still tends to over-emphasize short-term correlations and treats time only implicitly. This behavior hinders the learning of generalizable temporal evolution mechanisms. To address these issues, we adopt self-supervised pre-training to decouple representation learning from downstream supervision and to explicitly model temporal dependencies at multiple time scales. Specifically, we design two complementary objectives: (1) *Masked Token Modeling (MTM)* to capture semantic dependencies

among interaction patterns across different temporal ranges and go beyond short-term correlations; (2) *Next Time Prediction (NTP)* to explicitly model when future interactions occur and model temporal dynamics from both retrospective and prospective perspectives. Together, they encourage the model to learn both *what* evolves and *when* it evolves.

Masked Token Modeling (MTM) for Learning Long-/Short-Horizon Dependency. Given an interaction token sequence $\bar{\mathbf{P}} = [\bar{\mathbf{p}}_1, \bar{\mathbf{p}}_2, \ldots, \bar{\mathbf{p}}_m]$, MTM randomly masks blocks of consecutive tokens and trains the model to reconstruct the masked tokens from the visible context. This follows the conditional factorization principle of masked language modeling $p(\bar{\mathbf{p}}_1, \ldots, \bar{\mathbf{p}}_m) = p(\bar{\mathbf{p}}_{/M}) \prod_{i \in M} p(\bar{\mathbf{p}}_i \mid \bar{\mathbf{p}}_{/M})$ (Devlin et al., 2019), where M denotes the masked positions and $\bar{\mathbf{p}}_{/M}$ the visible tokens.

To encourage dependency learning at multiple temporal scales (e.g., short-term or long-term), we adopt block-wise masking. Given a masking budget n_m and the number of blocks $b \geq 1$, we sample b non-overlapping blocks of size $\lceil n_m/b \rceil$. Larger blocks require long-range dependency modeling, while smaller blocks emphasize short-term dependency modeling. Formally, we define the set of masked position $\mathcal{M}$ as

$$\mathcal{M} = \bigcup_{i=1}^{b} \{s_i, s_i + 1, \ldots, s_i + \lceil n_m/b \rceil - 1\} \tag{8}$$

where s_i is the index of $\bar{\mathbf{p}}_i$. To define reconstruction targets, we employ an exponential moving average encoder f_{EMA} to generate stable semantic targets $\hat{\mathbf{p}}_i = f_{\text{EMA}}(\{w \mid v_0 = v'_i\})$. The visible tokens are augmented with learnable mask tokens to form $\tilde{\mathbf{P}}$, which is passed through the TGPM encoder and a Transformer decoder to obtain reconstructed token $\mathbf{r}_i$ in SimMIM fashion (Xie et al., 2022). The loss for block size b is defined as

$$\mathcal{L}_{\text{MTM}}(b) = \frac{1}{m} \sum_{i=1}^{m} \text{is_masked}(\bar{\mathbf{p}}_i) \cdot \|\mathbf{r}_i - \text{sg}[\hat{\mathbf{p}}_i]\|_2^2, \tag{9}$$

where $\text{is_masked}(\cdot)$ is an indicator function for masked position, and $\text{sg}[\cdot]$ denotes the stop-gradient operator. The final MTM objective aggregates losses over a predefined set of block sizes B:

$$\mathcal{L}_{\text{MTM}} = \sum_{b \in B} \mathcal{L}_{\text{MTM}}(b). \tag{10}$$

Proposition 2.2. *Under block-wise masked token modeling with masking block size b, the optimal reconstruction must exploit contextual dependencies beyond certain dependency range r.*

We justify Proposition 2.2 in Appendix C.2 that block-wise masking is not merely a heuristic design, but an information-theoretic necessity for enforcing multi-scale temporal dependency learning. The masking block size directly controls the minimum temporal horizon that the model is forced to reason over.

Next Time Prediction (NTP) for Prospective Temporal Modeling. While MTM focuses on modeling semantic dependencies among interaction patterns, it does not explicitly constrain the temporal dimension. To model temporal evolution directly, we introduce the NTP task. Given the embeddings from the TGPM encoder $\mathbf{P}' = [\mathbf{p}'_1, \ldots, \mathbf{p}'_m]$, a two-layer MLP head f_{NTP} predicts the target-relative time interval encoding of the next interaction:

$$p(\mathbf{t}_1, \ldots, \mathbf{t}_m) = \prod_{i=1}^{m} p(\mathbf{t}_i \mid \mathbf{t}_{<i}, \bar{\mathbf{p}}_{<i}), \tag{11}$$

where $\mathbf{t}_i$ is the encoding corresponding to $\mathbf{p}_i$. The training objective is

$$\mathcal{L}_{\text{NTP}} = \frac{1}{m-1} \sum_{i=1}^{m-1} \|f_{\text{NTP}}(\mathbf{p}'_i) - \mathbf{t}_{i+1}\|. \tag{12}$$

The NTP encourages TGPM to infer when the next interaction is likely to occur conditioned on the observed patterns. Specifically, it allows TGPM to encode the temporal granularity of evolution, such as the typical gaps between successive interaction patterns, as well as frequency-related signals, reflecting how often interactions involving certain evolving patterns occur. Thus, the model learns to associate different evolving patterns with distinct temporal rhythms.

Joint training and downstream usage. The overall pre-training objective is $\mathcal{L} = \mathcal{L}_{\text{MTM}} + \mathcal{L}_{\text{NTP}}$. After pre-training, the decoder is discarded and a task-specific prediction head is attached to the TGPM encoder. For downstream tasks, we apply mean pooling over the contextualized sequence $\mathbf{P}'$ and compute predictions as $\hat{\mathbf{y}} = \text{Head}\big(\frac{1}{m} \sum_{i=1}^{m} \mathbf{p}'_i\big)$.

3. Experiments

3.1. Experimental Settings

We evaluate the effectiveness of our proposed TGPM on two downstream tasks. For temporal link prediction, we adopt three datasets from DTGB (Zhang et al., 2024), including Enron, ICEWS1819 and Googlemap CT. Following prior work, evaluation is conducted under both transductive and inductive settings (Yu et al., 2023; Cong et al., 2023). The transductive setting aims to predict future links between nodes within training set, and the inductive setting predicts future links on unseen nodes. For each link, one negative sample is generated for classification. For temporal node classification, we adopt two widely used datasets, Wikipedia and Reddit (Kumar et al., 2019). These datasets are featured by rich attributes, spanning various domains. We adopt the same dataset split strategy for all these datasets, which is

Table 1. **Performance and average ranking (A.R.) comparison over transductive and inductive temporal link prediction.** Results are reported on Enron, ICEWS1819 and Googlemap CT, with the best and sub-best performance highlighted in **Boldface** and Underline. TGPM achieves the strongest A.R., with larger gains on datasets with rich temporal dynamics.

Methods	*Transductive Link Prediction*			*Inductive Link Prediction*			A.R.
	Enron	ICEWS1819	Googlemap CT	Enron	ICEWS1819	Googlemap CT	
TGAT	95.30±0.54	98.73±0.03	79.11±0.63	84.51±1.92	96.06±0.07	73.41±1.07	3.8
GraphMixer	95.31±0.19	98.68±0.01	79.23±0.04	83.45±1.00	95.89±0.07	72.43±0.18	5.5
DyGFormer	95.30±0.16	98.86±0.04	77.46±0.53	86.42±0.34	**96.60±0.16**	70.97±0.36	3.8
PT-DGNN	95.55±0.13	98.81±0.07	77.35±0.76	**86.64±0.89**	96.07±0.31	71.50±1.31	4.0
DDGCL	94.77±0.52	98.65±0.03	78.62±0.10	84.47±1.62	95.89±0.16	72.45±0.33	5.0
CPDG	93.97±0.62	97.54±0.04	76.52±0.58	80.23±1.36	92.04±0.08	70.16±0.98	7.7
TGPM w/o Pretrain	**96.06±0.15**	98.90±0.04	78.55±0.51	86.13±1.24	96.38±0.12	72.35±0.37	3.2
TGPM	94.68±0.18	**98.94±0.03**	**80.51±0.44**	81.64±1.16	96.39±0.18	**74.32±0.34**	**3.0**

Table 2. **Performance and average ranking (A.R.) comparison over temporal node classification.** The best and sub-best results are highlighted in **Boldface** and Underline. TGPM achieves the strongest A.R..

Methods	Wikipedia	Reddit	A.R.
TGAT	80.03±1.43	50.96±3.85	6.5
GraphMixer	85.47±0.79	51.98±1.78	4.5
DyGFormer	77.93±11.79	53.35±1.78	6.0
PT-DGNN	81.61±1.29	49.32±1.82	6.0
DDGCL	78.04±3.95	53.14±3.24	6.0
CPDG	80.47±0.44	54.99±2.40	4.0
TGPM w/o Pretrain	86.83±1.08	55.21±2.17	2.0
TGPM	**87.40±0.70**	**55.54±1.22**	**1.0**

70/15/15 train/val/test chronological split. ROC-AUC score is used as the evaluation metric for both downstream tasks.

We compare our TGPM with supervised methods (TGAT (Xu et al., 2020), GraphMixer (Cong et al., 2023), DyGFormer (Yu et al., 2023)) and self-supervised learning methods (PT-DGNN (Chen et al., 2022b), DDGCL (Tian et al., 2021), CPDG (Bei et al., 2023)) targeting temporal graphs. For TGPM, we report the results both before (TGPM w/o Pretrain) and after pre-training (TGPM). More details are provided in Appendix A.

3.2. Main Results

Temporal Link Prediction. Table 1 summarizes temporal link prediction results under both transductive and inductive settings. Pre-training is conducted on each individual dataset. Overall, TGPM achieves the best average ranking of 3.0 among all selected methods, with best and sub-best performance in most datasets under both transductive and inductive setting. Notably, on Googlemap CT with more temporal dynamics, our proposed pre-training strategies can bring significant performance gain (80.51 vs. 78.55), highlighting its ability to capture complex evolving patterns. In contrast, on datasets with significant temporal burstiness, TGPM already performs well without pre-training due to sufficient temporal context information, but our pre-training strategies are much less beneficial and may even lead to trivial solution, reflecting the inadequacy of our method in handling temporal burstiness.

Temporal Node Classification. Table 2 summarizes temporal node classification results. Pretrained TGPM and TGPM trained from scratch achieve the best and sub-best performance on both datasets. Given that both Wikipedia and Reddit are featured by rich temporal dynamics, the consistent performance gain demonstrate the strength of TGPM in capturing complex evolving patterns.

Cross-Domain Transferability. We evaluate the cross-domain transferability of TGPM in Table 3, measuring generalization under distribution shifts across domains. In this setting, we train models on a source graph and evaluate on target graphs without further adaptation. We adopt ICEWS1819 and Googlemap CT as source graphs respectively and treat the remaining graphs as target graphs. Self-supervised learning baselines (PT-DGNN, DDGCL, CPDG) are adopted for comparison. TGPM achieves the best transfer results, showing significant positive gains over all target graphs. We attribute this to TGPM's ability to learn transferable evolving patterns, enabling generalization across significantly different domains, while time-shifting and structural proximity based self-supervised learning remains sensitive to subtle shifts of temporal dynamics.

Cross-Task Transferability. We measure the cross-task transferability of TGPM, evaluating whether the learned evolving patterns in TGPM can be effectively generalized to diverse feature spaces and downstream tasks. In this setting, we pre-train TGPM on link prediction datasets ICEWS1819 and Googlemap CT respectively, and further adapt to temporal node classification via fine-tuning. We report the in-domain pre-trained results of all self-supervised methods and cross-task fine-tuned results of TGPM in Table 4, where TGPM (ICEWS) and TGPM (Googlemap) denote the respective source datasets. TGPM transferred from both

Table 3. **Comparison with self-supervised methods over cross-domain transferability.** TGPM achieves the strongest performance compared to prior works across all datasets.

Methods	*Transductive Link Prediction*					*Inductive Link Prediction*				
	ICEWS1819		Googlemap CT		A.R.	ICEWS1819		Googlemap CT		A.R.
	Enron	Googlemap CT	Enron	ICEWS1819		Enron	Googlemap CT	Enron	ICEWS1819	
PT-DGNN	90.83±1.22	53.79±6.25	65.79±5.58	66.99±10.53	3.0	71.06±3.29	48.74±5.12	49.24±4.92	51.20±7.08	3.5
DDGCL	91.40±1.30	52.42±4.93	66.66±13.68	56.71±17.33	2.8	72.97±3.70	49.38±3.68	58.42±5.78	46.05±10.41	3.0
CPDG	89.18±0.70	51.96±2.27	75.74±2.58	66.12±6.45	3.3	65.89±2.98	50.81±2.33	67.72±5.04	53.04±6.34	2.5
TGPM	**92.51±0.53**	**56.21±6.96**	**87.06±5.37**	**88.92±5.42**	**1.0**	**78.64±1.18**	**53.09±4.56**	**77.01±3.29**	**79.84±2.43**	**1.0**

Table 4. **Cross-task transferability of TGPM compared with in-domain pre-training.** Transferred TGPM achieves competitive and even superior temporal node classification performance compared to in-domain pre-trained model.

Methods	Wikipedia	Reddit	A.R.
PT-DGNN	81.61±1.29	49.32±1.82	5.0
DDGCL	78.04±3.95	53.14±3.24	5.5
CPDG	80.47±0.44	54.99±2.40	4.0
TGPM	**87.40±0.70**	55.54±1.22	**1.5**
TGPM (ICEWS)	85.64±0.80	54.99±2.78	3.0
TGPM (Googlemap)	86.76±0.91	**61.16±2.59**	**1.5**

Table 5. **Ablation of model components and alternative solutions in TGPM on Googlemap CT and Wikipedia.**

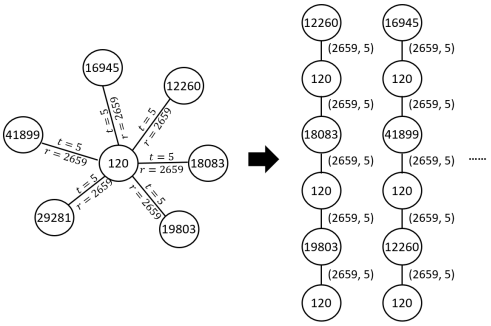
Methods	Googlemap CT		Wikipedia
	Transductive	Inductive	
Full Method	**80.51**	**74.32**	**87.40**
TGPM w/o ntp	79.59	73.25	84.63
TGPM w/o ltm	80.44	74.28	86.47
TGPM w/o stm	79.31	72.44	84.38
TGPM w/ cp	79.58	73.07	87.00
TGPM w/ rm	79.39	72.62	87.25

temporal link prediction datasets achieves competitive and even superior performance on both Wikipedia and Reddit, demonstrating TGPM's ability in capturing generalizable evolving patterns regardless of feature and temporal heterogeneity. Besides, TGPM pre-trained on Googlemap CT shows stronger cross-task generalizability, indicating that TGPM can better leverage its advantages on datasets with complex temporal dynamics.

3.3. Discussion

Ablation Study. We conduct two branches of ablation studies on Googlemap CT and Wikipedia: (1) Main ablation: we remove next time prediction (w/o NTP), long-term block masking (w/o LTM) and short-term block masking (w/o STM) respectively; (2) Alternative solution discussion: we first replace temporal random walk with chronologically monotonic causal path (w/ CP), and then replace short-/long-term masking with random masking (w/ RM). Table 5 presents the results of ablation studies. For our proposed components, NTP and short-term masking significantly contribute to evolving pattern mining, while the impact of long-term masking is domain-specific. This suggests that temporal evolving mechanisms and short-range dependencies are critical for understanding generalizable evolving patterns. Based on the results of alternative solutions, both causal paths and conventional random masking based generative pre-training consistently yield inferior performance, indicating the importance of pattern ordering relaxation and block-based dependency modeling.

(a) Statistics of temporal fan-out and edge feature homogeneity across datasets.



(b) Example of temporal burstiness on the derived temporally biased random walks starting from node 120 in Enron.

Figure 2. **Failure Case Analysis in Enron,** which has large-scale homogeneous temporal burstiness, leading to highly similar sampled patterns and performance degradation of pretraining.

Failure Case Analysis. In this section, we analyze the homogeneous temporal burstiness phenomenon in Enron and explain why TGPM fails in such situation. We design two metrics to measure the extent of temporal burstiness: *(1) fan-out* fo and *(2) feature homogeneity score* $homo$, which

measure the scale of temporal burstiness and the homogeneity of broadcasting behavior, respectively. Given a starting node u and timestamp t, fo and $homo$ can be defined as $fo(u,t) = |\{v|(u,v,t) \in \mathcal{E}\}|$, $homo(u,t) = \frac{\sum_q \binom{n_q(u,t)}{2}}{\binom{|\mathcal{S}_u^t|}{2}}$ where q is the index of edge feature, $n_q(u,t)$ denotes the number of edges in $\mathcal{S}_u^t$ whose feature index equals q, and $\sum_q n_q(u,t) = |\mathcal{S}_u^t|$. Higher fo and $homo$ refers to larger-scale homogeneous broadcasting behavior, or more concurrent edges with the same features. Figure 2(a) reports the maximum fo, the 95th percentile of fo and the average $homo$ ($fo_{\max}, fo_{95}, homo_{\text{avg}}$) of all the three benchmarks. Enron exhibits the largest scale of temporal burstiness, and the concurrent edges are homogeneous, corresponding to bulk emails in email network. As illustrated in Figure 5(b), this phenomenon produces highly similar interaction patches: half of sampled node features and all edge and temporal features are shared across sampled walks. As a result, masked patches can be trivially reconstructed from visible ones, leading to the collapse of MTM and NTP pre-training, preventing TGPM from learning meaningful temporal patterns. One promising direction is to aggregate homogeneous concurrent edges into meta-patterns before pre-training.

It is worth noting that while temporal burstiness is a well-studied property of real-world temporal networks (Barabasi, 2005), the failure mode identified here additionally requires large-scale feature homogeneity among concurrent edges, which is highly domain-specific. Real-world temporal graphs usually exhibit diverse edge features even during bursty periods, as concurrent interactions typically originate from independent user behaviors. Therefore, while this failure case represents a meaningful limitation of TGPM, it does not reflect a fundamental gap.

4. Related Work

Temporal Graph Representation Learning. Representation learning on temporal graphs has been widely studied (Feng et al., 2025; Longa et al., 2023; Kazemi et al., 2020). Recently, continuous-time methods have emerged to directly learn on the whole event sequences of dynamic graphs with temporal random walks (Nguyen et al., 2018; Wang et al., 2021d; Jin et al., 2022), temporal point process (Trivedi et al., 2019; Wen & Fang, 2022), memory networks (Kumar et al., 2019; Ma et al., 2020; Rossi et al., 2020b; Wang et al., 2021b; Luo & Li, 2022) and temporal graph neural networks (Xu et al., 2020). To efficiently learn from long-term interaction-wise dependencies, most state-of-the-art methods turn to sequential models, especially Transformer-based models (Wang et al., 2021a; Cong et al., 2023; Yu et al., 2023; Wu et al., 2024; Pan et al., 2025a; Wu et al., 2025; Peng et al., 2025). Although these methods have shown empirical success, they are mostly task-centric methods suffering from limited receptive field, retrospective temporal modeling and inability to capture evolving semantics. Our proposed TGPM can overcome these challenges.

Temporal Graph Pre-Training. Extended from static graph pre-training methods (Hu et al., 2020; Zhao et al., 2021; 2023; Qian et al.; Ju et al., 2022; Liu et al., 2023; Wang et al., 2023; 2025e), continuous-time temporal graph pre-training methods aim to capture inherent properties of temporal graphs via various self-supervised signals. Commonly used pre-training strategies include structural and temporal contrastive learning (Bei et al., 2023; Tian et al., 2021), masked attribute reconstruction (Chen et al., 2022b), curvature-aware self-contrastive learning (Sun et al., 2022) and graph prompting (Chen et al., 2024b; Yu et al., 2025; Thapaliya et al., 2025; Wang et al., 2025a). However, their focus on time-shifting invariant properties and low-level signal reconstruction misaligned with generalizable evolution pattern extraction. In contrast, TGPM introduces a pattern-centric pre-training framework targeting multi-scale temporal dynamics and prospective temporal modeling, which is able to learn from complex evolving patterns and unlock better generalizability.

Graph Tokenization. Graph pattern tokenization is a long-standing challenge due to the non-Euclidean nature of graphs. Early methods adopt graph kernels (Yanardag & Vishwanathan, 2015) and anonymous random walks (Ivanov & Burnaev, 2018) to extract graph patterns. Recent approaches propose to tokenize graphs into sequences of substructures, such as subgraphs (He et al., 2023), multisets (Baek et al., 2021) and multi-hop neighborhood features (Chen et al., 2023; 2024a). To further overcome task heterogeneity in graph learning, task-agnostic tokenization via message passing (Wang et al., 2024a; Chen et al., 2022a; Zhao et al., 2022), tree structures (Wang et al., 2024c; 2025b) and random walks (Wang et al., 2025c;d) are proposed. However, these methods are inherently designed for static graphs, which emphasizes the learning of substructure correlations. In contrast, temporal graph learning focuses on evolution mechanism learning.

5. Conclusion

We propose TGPM, a pattern-centric temporal graph learning framework that models generalizable temporal evolution mechanisms instead of task-specific correlations. By representing interactions as patches constructed via temporally biased random walks, TGPM captures evolving structural semantics and long-term dependencies. A Transformer backbone captures global invariances and local temporal patterns, aided by self-supervised objectives for learning multi-scale dynamics. Extensive experiments demonstrate

TGPM's strong transferability and scalability. We also observe performance degradation under large-scale homogeneous temporal burstiness, indicating important directions for future work.

References

Baek, J., Kang, M., and Hwang, S. J. Accurate learning of graph representations with graph multiset pooling. In *International Conference on Learning Representations*, 2021.

Barabasi, A.-L. The origin of bursts and heavy tails in human dynamics. *Nature*, 435(7039):207–211, 2005.

Bei, Y., Xu, H., Zhou, S., Chi, H., Wang, H., Zhang, M., Li, Z., and Bu, J. Cpdg: A contrastive pre-training method for dynamic graph neural networks, 2023.

Cai, L., Mao, X., Zhou, Y., Long, Z., Wu, C., and Lan, M. A survey on temporal knowledge graph: Representation learning and applications. *arXiv preprint arXiv:2403.04782*, 2024.

Chen, D., O'Bray, L., and Borgwardt, K. Structure-aware transformer for graph representation learning. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, Proceedings of Machine Learning Research, 2022a.

Chen, J., Gao, K., Li, G., and He, K. NAGphormer: A tokenized graph transformer for node classification in large graphs. In *The Eleventh International Conference on Learning Representations*, 2023.

Chen, J., Liu, H., Hopcroft, J. E., and He, K. Leveraging contrastive learning for enhanced node representations in tokenized graph transformers. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024a.

Chen, K.-j., Zhang, J., Jiang, L., Wang, Y., and Dai, Y. Pre-training on dynamic graph neural networks. *Neurocomputing*, 500:679–687, 2022b.

Chen, X., Zhang, S., Xiong, Y., Wu, X., Zhang, J., Sun, X., Zhang, Y., Zhao, F., and Kang, Y. Prompt learning on temporal interaction graphs, 2024b.

Cong, W., Zhang, S., Kang, J., Yuan, B., Wu, H., Zhou, X., Tong, H., and Mahdavi, M. Do we really need complicated model architectures for temporal networks? In *The Eleventh International Conference on Learning Representations*, 2023.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.

Feng, Z., Wang, R., Wang, T., Song, M., Wu, S., and He, S. A comprehensive survey of dynamic graph neural networks: Models, frameworks, benchmarks, experiments and challenges. *IEEE Transactions on Knowledge and Data Engineering*, 2025.

Feng, Z., Wang, R., Wang, T., Song, M., Wu, S., and He, S. A comprehensive survey of dynamic graph neural networks: Models, frameworks, benchmarks, experiments and challenges. *IEEE Transactions on Knowledge and Data Engineering*, 38(1):26–46, 2026.

Hawkes, A. G. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, 1971.

He, X., Hooi, B., Laurent, T., Perold, A., LeCun, Y., and Bresson, X. A generalization of vit/mlp-mixer to graphs. In *International Conference on Machine Learning*, pp. 12724–12745. PMLR, 2023.

Holme, P. and Saramäki, J. Temporal networks. *Physics Reports*, 519(3):97–125, 2012. Temporal Networks.

Hu, W., Liu, B., Gomes, J., Zitnik, M., Liang, P., Pande, V., and Leskovec, J. Strategies for pre-training graph neural networks. In *International Conference on Learning Representations*, 2020.

Huang, X., Chow, W., Zhu, Y., Wang, Y., Chai, Z., Wang, C., Chen, L., and Yang, Y. Enhancing cross-domain link prediction via evolution process modeling. In *Proceedings of the ACM on Web Conference 2025*, WWW '25, pp. 2158–2171, 2025.

Ivanov, S. and Burnaev, E. Anonymous walk embeddings. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2186–2195, 10–15 Jul 2018.

Jin, M., Li, Y.-F., and Pan, S. Neural temporal walks: Motif-aware representation learning on continuous-time dynamic graphs. *Advances in Neural Information Processing Systems*, 35:19874–19886, 2022.

Ju, M., Hou, S., Fan, Y., Zhao, J., Ye, Y., and Zhao, L. Adaptive kernel graph neural network. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pp. 7051–7058, 2022.

Kalman, R. E. A new approach to linear filtering and prediction problems. 1960.

Kazemi, S. M., Goel, R., Jain, K., Kobyzev, I., Sethi, A., Forsyth, P., and Poupart, P. Representation learning for dynamic graphs: A survey. *Journal of Machine Learning Research*, 21(70):1–73, 2020.

Kossinets, G. and Watts, D. J. Empirical analysis of an evolving social network. *science*, 311(5757):88–90, 2006.

Kumar, S., Zhang, X., and Leskovec, J. 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, 2019.

Layne, J., Carpenter, J., Serra, E., and Gullo, F. Temporal sir-gn: Efficient and effective structural representation learning for temporal graphs. *Proc. VLDB Endow.*, 16(9): 2075–2089, May 2023.

Liguori, A., Caroprese, L., Minici, M., Veloso, B., Spinnato, F., Nanni, M., Manco, G., and Gama, J. Modeling events and interactions through temporal processes: A survey. *Neurocomputing*, 653:131191, 2025.

Liu, Y., Jin, M., Pan, S., Zhou, C., Zheng, Y., Xia, F., and Yu, P. S. Graph self-supervised learning: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 35 (6):5879–5900, 2023.

Longa, A., Lachi, V., Santin, G., Bianchini, M., Lepri, B., Lio, P., Scarselli, F., and Passerini, A. Graph neural networks for temporal graphs: State of the art, open challenges, and opportunities. *arXiv preprint arXiv:2302.01018*, 2023.

Luo, Y. and Li, P. Neighborhood-aware scalable temporal network representation learning. In *Learning on Graphs Conference*, pp. 1–1, 2022.

Ma, Y., Guo, Z., Ren, Z., Tang, J., and Yin, D. Streaming graph neural networks. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*, pp. 719–728, 2020.

Michail, O. An introduction to temporal graphs: An algorithmic perspective*. *Internet Mathematics*, 12(4):239–280, 2016.

Nguyen, G. H., Lee, J. B., Rossi, R. A., Ahmed, N. K., Koh, E., and Kim, S. Continuous-time dynamic network embeddings. In *Companion proceedings of the the web conference 2018*, pp. 969–976, 2018.

Pan, Z., Gao, C., Cai, F., Chen, H., and Li, Y. Light dynamic graph learning on temporal networks. *ACM Transactions on Information Systems*, 2025a.

Pan, Z., Gao, C., Cai, F., Chen, W., Zhang, X., Chen, H., and Li, Y. On the cross-graph transferability of dynamic link prediction. In *Proceedings of the ACM on Web Conference 2025*, WWW '25, pp. 4101–4110, 2025b.

Peng, J., Wei, Z., and Ye, Y. Tidformer: Exploiting temporal and interactive dynamics makes a great dynamic graph transformer. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 2245–2256, 2025.

Qian, Y., Zhang, C., Zhang, Y., Wen, Q., Ye, Y., and Zhang, C. Co-modality graph contrastive learning for imbalanced node classification. *Advances in Neural Information Processing Systems*, 35:15862–15874.

Rossi, E., Chamberlain, B., Frasca, F., Eynard, D., Monti, F., and Bronstein, M. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637*, 2020a.

Rossi, E., Chamberlain, B., Frasca, F., Eynard, D., Monti, F., and Bronstein, M. Temporal graph networks for deep learning on dynamic graphs. In *ICML 2020 Workshop on Graph Representation Learning*, 2020b.

Sun, L., Ye, J., Peng, H., and Yu, P. S. A self-supervised riemannian gnn with time varying curvature for temporal graph learning. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, CIKM '22, pp. 1827–1836, 2022.

Sun, Z., Wang, H., Wang, H., Shao, B., and Li, J. Efficient subgraph matching on billion node graphs. *Proc. VLDB Endow.*, 5(9):788–799, 2012.

Thapaliya, S., Wang, Z., Li, J., Li, Z., Ye, Y., and Zhang, C. Semantic refinement with llms for graph representations. *arXiv preprint arXiv:2512.21106*, 2025.

Tian, S., Wu, R., Shi, L., Zhu, L., and Xiong, T. Self-supervised representation learning on dynamic graphs. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, pp. 1814–1823, 2021.

Tian, Y., Qi, Y., and Guo, F. Freedyg: Frequency enhanced continuous-time dynamic graph model for link prediction. In *The twelfth international conference on learning representations*, 2024.

Trivedi, R., Farajtabar, M., Biswal, P., and Zha, H. Dyrep: Learning representations over dynamic graphs. In *International conference on learning representations*, 2019.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Wang, C., Tsepa, O., Ma, J., and Wang, B. Graph-mamba: Towards long-range graph sequence modeling with selective state spaces. *arXiv preprint arXiv:2402.00789*, 2024a.

Wang, L., Chang, X., Li, S., Chu, Y., Li, H., Zhang, W., He, X., Song, L., Zhou, J., and Yang, H. Tcl: Transformer-based dynamic graph modelling via contrastive learning. *arXiv preprint arXiv:2105.07944*, 2021a.

Wang, R., Zhao, W., Sun, D., Mendis, C., and Abdelzaher, T. Towards efficient temporal graph learning: Algorithms, frameworks, and tools. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, CIKM '24, pp. 5530–5533, 2024b.

Wang, X., Lyu, D., Li, M., Xia, Y., Yang, Q., Wang, X., Wang, X., Cui, P., Yang, Y., Sun, B., et al. Apan: Asynchronous propagation attention network for real-time temporal graph embedding. In *Proceedings of the 2021 international conference on management of data*, pp. 2628–2638, 2021b.

Wang, Y., Cai, Y., Liang, Y., Ding, H., Wang, C., and Hooi, B. Time-aware neighbor sampling for temporal graph networks, 2021c.

Wang, Y., Chang, Y.-Y., Liu, Y., Leskovec, J., and Li, P. Inductive representation learning in temporal networks via causal anonymous walks. In *International Conference on Learning Representations*, 2021d.

Wang, Z., Li, Q., Yu, D., Han, X., Gao, X.-Z., and Shen, S. Heterogeneous graph contrastive multi-view learning. In *Proceedings of the 2023 SIAM international conference on data mining (SDM)*, pp. 136–144. SIAM, 2023.

Wang, Z., Zhang, Z., Chawla, N. V., Zhang, C., and Ye, Y. Gft: Graph foundation model with transferable tree vocabulary. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024c.

Wang, Z., Liu, S., Zhang, Z., Ma, T., Zhang, C., and Ye, Y. Can llms convert graphs to text-attributed graphs? In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 1412–1432, 2025a.

Wang, Z., Zhang, Z., Ma, T., Chawla, N. V., Zhang, C., and Ye, Y. Towards graph foundation models: Learning generalities across graphs via task-trees. In *Forty-second International Conference on Machine Learning*, 2025b.

Wang, Z., Zhang, Z., Ma, T., Chawla, N. V., Zhang, C., and Ye, Y. Beyond message passing: Neural graph pattern machine. In *Forty-second International Conference on Machine Learning*, 2025c.

Wang, Z., Zhang, Z., Ma, T., Zhang, C., and Ye, Y. Generative graph pattern machine. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025d.

Wang, Z., Zhang, Z., Zhang, C., and Ye, Y. Training mlps on graphs without supervision. In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining*, pp. 697–706, 2025e.

Wen, Z. and Fang, Y. Trend: Temporal event and node dynamics for graph representation learning. In *Proceedings of the ACM web conference 2022*, pp. 1159–1169, 2022.

Wu, Y., Fang, Y., and Liao, L. On the feasibility of simple transformer for dynamic graph modeling. In *Proceedings of the ACM Web Conference 2024*, pp. 870–880, 2024.

Wu, Y., Liao, L., and Fang, Y. Retrieval augmented generation for dynamic graph modeling. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1434–1443, 2025.

Xie, Z., Zhang, Z., Cao, Y., Lin, Y., Bao, J., Yao, Z., Dai, Q., and Hu, H. Simmim: a simple framework for masked image modeling. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9643–9653, 2022.

Xu, D., Ruan, C., Korpeoglu, E., Kumar, S., and Achan, K. Inductive representation learning on temporal graphs. In *International Conference on Learning Representations*, 2020.

Yanardag, P. and Vishwanathan, S. Deep graph kernels. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pp. 1365–1374, 2015.

Yu, L., Sun, L., Du, B., and Lv, W. Towards better dynamic graph learning: New architecture and unified library. *Advances in Neural Information Processing Systems*, 36: 67686–67700, 2023.

Yu, X., Liu, Z., Zhang, X., and Fang, Y. Node-time conditional prompt learning in dynamic graphs. In *The Thirteenth International Conference on Learning Representations*, 2025.

Zhang, J., Chen, J., Yang, M., Feng, A., Liang, S., Shao, J., and Ying, R. Dtgb: a comprehensive benchmark for dynamic text-attributed graphs. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, 2024.

Zhao, J., Wen, Q., Sun, S., Ye, Y., and Zhang, C. Multi-view self-supervised heterogeneous graph embedding. In *Joint European conference on machine learning and knowledge discovery in databases*, pp. 319–334. Springer, 2021.

Zhao, J., Wen, Q., Ju, M., Zhang, C., and Ye, Y. Self-supervised graph structure refinement for graph neural

networks. In *Proceedings of the sixteenth ACM international conference on web search and data mining*, pp. 159–167, 2023.

Zhao, J., Wang, Z., Liao, Y., Zhang, C., and Ye, Y. Controllable graph generation with diffusion models via inference-time tree search guidance. *arXiv preprint arXiv:2510.10402*, 2025.

Zhao, L., Jin, W., Akoglu, L., and Shah, N. From stars to subgraphs: Uplifting any GNN with local structure awareness. In *International Conference on Learning Representations*, 2022.

Zheng, Y., Yi, L., and Wei, Z. A survey of dynamic graph neural networks. *Frontiers of Computer Science*, 19: 196323–, 2025.

A. Additional Experimental Details

A.1. Datasets

For temporal link prediction, we evaluate all the selected methods on three semantically-enriched temporal graph benchmarks from the Dynamic Text-Attributed Graph Benchmark (DTGB) (Zhang et al., 2024): Enron, ICEWS1819, and Googlemap CT. These datasets cover diverse application domains and exhibit substantially different temporal characteristics, enabling a comprehensive evaluation of both in-domain performance and cross-domain generalization ability. Enron is an email communication network where nodes represent email users and temporal edges correspond to email exchanges with timestamps. This dataset is characterized by strong temporal burstiness, where a large number of interactions may occur concurrently at the same timestamp, often sharing highly homogeneous edge features. ICEWS1819 is a temporal knowledge graph derived from the Integrated Crisis Early Warning System (ICEWS), covering international political events from 2018 to 2019. Nodes represent entities, and edges correspond to event interactions annotated with rich semantic features and timestamps. Compared to Enron, ICEWS1819 contains fewer concurrent interactions per timestamp and exhibits clearer temporal evolution patterns. Googlemap CT is a temporal interaction graph collected from a recommender system, where nodes represent users or items and edges denote time-stamped behaviors. This dataset exhibits complex long-term temporal dynamics, making it well-suited for evaluating models' ability to capture multi-scale temporal dependencies.

For temporal node classification, the evaluation involves two widely used benchmarks from (Kumar et al., 2019): Wikipedia and Reddit, which are both bipartite temporal interaction graphs recorded in Unix timestamps. In both datasets, nodes represent users and items. Items refer to Wikipedia pages and subreddits, while temporal edges correspond to page edits on Wikipedia and posts on Reddit respectively. The node classification task is to predict whether a user will be banned at a given time point.

Table 6. Dataset Statistics

Dataset	Nodes	Edges	Unique Steps	Dim of Node/Edge
Enron	42,711	797,907	1,006	768/768
ICEWS1819	31,796	1,100,071	730	768/768
Googlemap CT	111,168	1,380,623	55,521	768/768
Wikipedia	9,227	157,474	152,757	–/172
Reddit	10,984	672,447	669,065	–/172

A.2. Baselines

Supervised methods. TGAT (Xu et al., 2020) employs temporal attention mechanisms to aggregate time-aware neighborhood information and models temporal interactions using continuous time encodings. GraphMixer (Cong et al., 2023) replaces message passing paradigm with time-aware MLPMixer to efficiently capture temporal dependencies in interaction sequences. DyGFormer (Yu et al., 2023) is a Transformer-based model that represents historical interactions as token sequences and stacks self-attention layers to model long-range temporal dependencies.

Self-supervised methods. PT-DGNN (Chen et al., 2022b) adopts masked attribute reconstruction as a pre-training strategy for dynamic graph neural networks. DDGCL (Tian et al., 2021) performs self-supervised contrastive learning on temporal graphs by contrasting node representations across temporally adjacent views. CPDG (Bei et al., 2023) introduces contrastive pre-training based on temporal and structural augmentations to improve generalization.

A.3. Implementation Details

Most experiments are conducted on Linux servers equipped with four Nvidia A40 GPUs. The models are implemented by PyTorch 2.4.0 and PyTorch Cluster 2.1.2, with CUDA 12.1 and Python 3.9. The hyperparameter settings are summarized in Table 7. For optimization, we use the AdamW optimizer with weight decay and set the number of epochs as 10 for pre-training and 20 for fine-tuning. Full fine-tuning is conducted after pre-training, while TGPM backbone and task-specific decoder use different learning rate, with 1e-5 and 1e-4 respectively. We use a two-layer multi-layer perceptron as the prediction head for temporal link prediction to take the concatenated representations of two input nodes and return the probability of the given link. Temporal link prediction experiments are conducted five times with different random seeds,

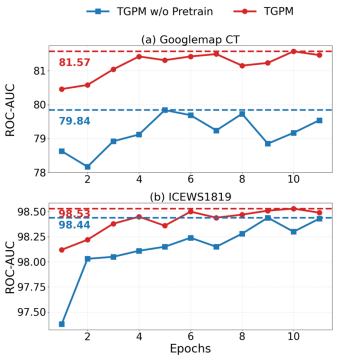


Figure 3. **Convergence curve comparison between training from scratch and fine-tuning the pre-trained model.**

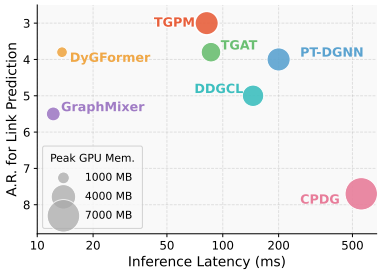


Figure 4. **Computational Efficiency Analysis.** Sizes of points indicate the peak GPU memory usage. TGPM maintains competitive training efficiency, significantly more efficient than self-supervised baselines.

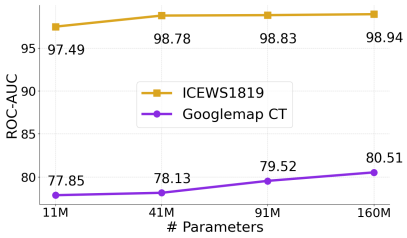


Figure 5. **Parameter Scaling Analysis.** Increasing model parameters can consistently enhance performance on transductive settings, showing the great potential of TGPM acting as the scalable backbone in temporal graph learning.

while temporal node classification experiments are conducted for three times.

Table 7. Hyperparameter Settings

Hyperparameter	Value	Hyperparameter	Value
Batch Size	128	Dropout	0.1
Hidden Dimension	768	Weight Decay	0.05
Time Encoding Dimension	100	Pre-Train Learning Rate	0.0001
Number of Heads	12	Fine-Tune Learning Rate	0.00001
Number of Encoder Layers	2	Pattern Size L	6
Number of Decoder Layers	1	Pattern Per Interaction k	8
EMA Momumtum	0.99	Masking Block Size $\mathcal{B}$	[6, 24]
EMA Update Every	10	Maximum Sampled Interaction $\lvert\mathcal{S}_v^t\rvert$	32

B. Supplementary Experimental Results

B.1. Convergence Curve Analysis

Figure 3 compares the training dynamics of TGPM trained from scratch versus those finetuned from a pre-trained TGPM checkpoint. We make comparisons over transductive temporal link prediction on Googlemap CT and ICEWS1819, where Googlemap CT has abundant temporal dynamics, while ICEWS1819 has fewer unique timestamps. Although pre-training facilitates performance improvement in both datasets, pre-training on datasets with more complex temporal dynamics like Googlemap CT leads to faster convergence and more significant performance gain, indicating that our proposed pre-training strategy can better capture complex temporal dependencies within rich temporal dynamics, improving in-domain generalization. As expected, accuracy improves with additional training epochs in both datasets.

B.2. Computational Efficiency

We evaluate computational efficiency by measuring average per-batch inference latency and peak GPU memory usage on Wikipedia. For supervised methods and self-supervised methods, we report the statistics during training and pre-training respectively. As shown in Figure 4, TGPM achieves competitive inference efficiency. Compared to self-supervised baselines, TGPM is substantially faster and more memory-efficient. These results demonstrate that TGPM achieves satisfying trade-off between predictive performance and inference efficiency.

B.3. Parameter Scaling

Built on Transformer architecture, TGPM can naturally scale to larger model sizes by stacking more Transformer layers or extending to larger hidden dimension. As illustrated in Figure 5, increasing model parameters can consistently enhance

performance on selected benchmarks. However, for datasets with abundant temporal dynamics like Googlemap CT, the performance gain from increased parameter is more promising. In contrast, for datasets with more concurrent edges, TGPM struggles to scale due to its disadvantage in processing temporal burstiness.

C. Proofs

C.1. Temporal Random Walk Expressiveness

Definition C.1 (Temporal Random Walk Expressiveness)**.** In this section, we formally establish the expressive advantage of non-monotonic temporal walks over monotonic walks. Our proof proceeds in four steps: (1) we define monotonic and non-monotonic temporal walks under the framework established in Section X; (2) we prove that non-monotonic walks are strictly more expressive by constructing a temporal graph where the inclusion is proper; (3) we quantify this expressiveness gap using the temporal reachability gap metric; and (4) we introduce temporal disorder as a measure to characterize the specific patterns captured by non-monotonic walks but missed by monotonic walks.

Given a temporal graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X}, \mathbf{E})$ and anchor interaction (v, v', t'), we define:

- **Monotonic temporal walk**: A node sequence $w = (v_0, v_1, \ldots, v_L)$ where $v_0 = v'$ and timestamps satisfy $\mathcal{T}(v_{i-1}, v_i) < \mathcal{T}(v_i, v_{i+1}) < t'$ for all valid i.
- **Non-monotonic temporal walk**: A node sequence $w = (v_0, v_1, \ldots, v_L)$ where $v_0 = v'$ and timestamps only satisfy $\mathcal{T}(v_i, v_{i+1}) < t'$ for all i.

Let $\mathcal{W}^{t'}_{\text{mono}}(v')$ and $\mathcal{W}^{t'}_{\text{non}}(v')$ denote the sets of reachable node sequences under monotonic and non-monotonic constraints, respectively.

Proposition C.2 (Expressiveness of Non-Monotonic Temporal Walks)**.** *For any temporal graph $\mathcal{G}$ and anchor interaction (v, v', t'):*

(i) $\mathcal{W}^{t'}_{\text{mono}}(v') \subseteq \mathcal{W}^{t'}_{\text{non}}(v')$, i.e., non-monotonic walks are at least as expressive as monotonic walks.

(ii) There exists a class of temporal graphs $\mathcal{G}^*$ such that $\mathcal{W}^{t'}_{\text{mono}}(v') \subsetneq \mathcal{W}^{t'}_{\text{non}}(v')$, i.e., the inclusion is strict.

(iii) Non-monotonic walks can capture composite temporal motifs *that are fundamentally inaccessible to monotonic walks.*

Proof. **Part (i):** By definition, any monotonic temporal walk satisfies $\mathcal{T}(v_{i-1}, v_i) < \mathcal{T}(v_i, v_{i+1}) < t'$, which immediately implies $\mathcal{T}(v_i, v_{i+1}) < t'$. Thus, every monotonic walk is also a valid non-monotonic walk, establishing $\mathcal{W}^{t'}_{\text{mono}}(v') \subseteq \mathcal{W}^{t'}_{\text{non}}(v')$.

Part (ii): We construct a temporal graph $\mathcal{G}^* = (\mathcal{V}^*, \mathcal{E}^*)$ where the inclusion is strict. Let $\mathcal{V}^* = \{A, B, C, D\}$ and consider the following temporal edges with anchor time $t' = 10$:

$$\mathcal{E}^* = \{(A, B, t = 8), (B, C, t = 3), (C, D, t = 6)\}$$

Starting from node B (i.e., $v' = B$), consider the walk $w = (B, C, D)$:

- **Non-monotonic**: Both edges satisfy the constraint $\mathcal{T}(e) < t' = 10$. Specifically, $\mathcal{T}(B, C) = 3 < 10$ and $\mathcal{T}(C, D) = 6 < 10$. Thus, $w \in \mathcal{W}^{t'}_{\text{non}}(B)$.
- **Monotonic**: The walk requires $\mathcal{T}(B, C) < \mathcal{T}(C, D)$, i.e., $3 < 6$, which holds. However, we also need the walk to originate from the anchor context. If we extend to walk (A, B, C, D) starting from the anchor edge $(A, B, t = 8)$, monotonicity requires $\mathcal{T}(A, B) < \mathcal{T}(B, C)$, i.e., $8 < 3$, which is false.

More directly, consider anchor $(A, B, t' = 10)$ and the walk $w = (B, C)$. For monotonic walks rooted at this anchor, we require that subsequent edges have timestamps greater than the most recent edge in the path. Since we start from B with implicit context timestamp 8 (from edge (A, B)), reaching C requires $\mathcal{T}(B, C) > 8$, but $\mathcal{T}(B, C) = 3 < 8$. Thus $(B, C) \notin \mathcal{W}^{t'}_{\text{mono}}(B)$ under anchor-aware monotonicity, while $(B, C) \in \mathcal{W}^{t'}_{\text{non}}(B)$.

Part (iii): We formally characterize the class of patterns accessible only to non-monotonic walks. Define a *composite temporal motif* as a subgraph pattern where node relationships span multiple non-consecutive time periods.

Definition C.3 (Temporal Reachability Gap)**.** For anchor (v, v', t'), the *temporal reachability gap* is defined as:

$$\Delta_{\mathcal{G}}(v', t') = |\mathcal{W}_{non}^{t'}(v')| - |\mathcal{W}_{mono}^{t'}(v')| \tag{13}$$

The gap $\Delta_{\mathcal{G}}(v', t') > 0$ whenever the temporal graph contains *temporally interleaved structures*, defined as node pairs (u, w) reachable from v' through paths where edge timestamps do not form a monotonic sequence. Such structures arise naturally in real-world temporal graphs:

- **Rekindled relationships**: User B interacted with C in the past ($t = 3$), then with A recently ($t = 8$). Monotonic walks from B after the (A, B) interaction cannot revisit the earlier B-C relationship.
- **Periodic behaviors**: Seasonal patterns where interactions recur at non-consecutive intervals.
- **Multi-scale dynamics**: Long-term structural relationships that persist across short-term activity bursts.

To provide a finer-grained characterization of the expressiveness gap beyond mere cardinality, we introduce a quantitative measure of temporal non-monotonicity: Formally, let $\pi = (e_1, e_2, \ldots, e_k)$ be a path of temporal edges. Define the *temporal disorder* of π as:

$$\text{disorder}(\pi) = \sum_{i=1}^{k-1} \mathbf{1}[\mathcal{T}(e_i) > \mathcal{T}(e_{i+1})] \tag{14}$$

Monotonic walks are restricted to paths with $\text{disorder}(\pi) = 0$, while non-monotonic walks can traverse paths with $\text{disorder}(\pi) \geq 0$. The additional patterns captured by non-monotonic walks precisely correspond to paths with positive temporal disorder, which encode historically relevant but temporally non-sequential relationships. □

Remark C.4. The expressiveness gap has practical implications for temporal graph learning. Many real-world phenomena exhibit *temporal non-monotonicity*: a user's current behavior may be better predicted by combining recent activity with historically distant but semantically relevant interactions, rather than strictly recent causal chains. Our temporally biased random walk with exponential decay weighting (Eq. 1-2) provides a principled mechanism to sample from $\mathcal{W}_{non}^{t'}(v')$ while still prioritizing temporally proximate edges.

C.2. Theoretical Analysis of Multi-Scale Masking

We provide an information-theoretic justification for our multi-scale masking strategy. The key insight is that **block size determines the range of temporal dependencies that can be effectively learned**: small blocks capture short-term patterns but lose long-term structure, while large blocks capture long-term patterns but suffer from information loss for short-term dependencies. This motivates a multi-scale approach.

Definition C.5 (r-Markov Dependency)**.** A token sequence $\bar{\mathbf{P}} = [\bar{\mathbf{p}}_1, \ldots, \bar{\mathbf{p}}_m]$ satisfies *r-Markov dependency* if for all i:

$$\bar{\mathbf{p}}_i \perp\!\!\!\perp \bar{\mathbf{P}}_{\{j:|j-i|>r\}} \mid \bar{\mathbf{P}}_{\{j:0<|j-i|\leq r\}}. \tag{15}$$

The *dependency range* $r^*(\bar{\mathbf{P}})$ is the minimal such r. We say $\bar{\mathbf{P}}$ has *multi-scale dependency* $(r_1, \ldots, r_K)$ if it decomposes as $\bar{\mathbf{p}}_i = f(\mathbf{u}_i^{(1)}, \ldots, \mathbf{u}_i^{(K)}, \boldsymbol{\epsilon}_i)$ where each $\mathbf{u}^{(k)}$ satisfies r_k-Markov dependency independently.

Definition C.6 (Predictive Information)**.** We use $I(X; Y)$ to denote the mutual information between random variables X and Y. For a contiguous block mask $M_b = \{a, \ldots, a + b - 1\}$, the *predictive information* is $\mathcal{I}(M_b) = I(\bar{\mathbf{P}}_{M_b}; \bar{\mathbf{P}}_{M_b^c})$.

The *r-local predictive information* is

$$\mathcal{I}_r(M_b) = I(\bar{\mathbf{P}}_{M_b}; \bar{\mathbf{P}}_{M_b^c \cap \mathcal{N}_r(M_b)})$$

Where $\mathcal{N}_r(M_b) = \{j : \min_{i \in M_b} |j - i| \leq r\}$ is the r-neighborhood of M_b. This measures information accessible within range r—if $\mathcal{I}_r(M_b) < \mathcal{I}(M_b)$, then reconstruction requires looking beyond range r.

Finally, define the *minimal reconstruction range*:

$$R^*(M_b) = \min\{r : \mathcal{I}_r(M_b) = \mathcal{I}(M_b)\} \tag{16}$$

This is the minimum context range needed for full predictive information.

Proposition C.7 (Block Size Determines Learnable Dependency Scale)**.** *For a sequence with multi-scale dependency $(r_1, \ldots, r_K)$ where $r_1 < \cdots < r_K$, and block mask M_b of size b:*

(i) ***Block size lower-bounds required range****: $R^*(M_b) \geq b$. Large blocks require long-range context for reconstruction.*

(ii) ***Scale mismatch causes information loss****:*

- *If $b \gg r_k$ (block too large for scale k): Dependencies at scale r_k contribute only $O(r_k/b)$ fraction of reconstruction signal, causing* short-term information loss.
- *If $b \ll r_k$ (block too small for scale k): The model cannot access context at range r_k needed to capture scale-k patterns, causing* long-term information loss.
- *Optimal learning for scale k occurs when $b \approx r_k$.*

(iii) ***Single block size is suboptimal****: No single block size b can simultaneously satisfy $b \approx r_k$ for all $k \in \{1, \ldots, K\}$ when dependency scales are well-separated ($r_{k+1}/r_k \gg 1$).*

Proof. **Part (i)**: Consider the interior tokens of M_b, specifically those at positions i with $\min_{j \in M_b^c} |i - j| = \lceil b/2 \rceil$ (center of the block). To reconstruct $\bar{\mathbf{p}}_i$, the model must access information from tokens at distance $\geq \lceil b/2 \rceil$. Even under 1-Markov dependency, information must propagate across this distance, requiring $R^*(M_b) \geq b/2$. For general r-Markov dependencies, the argument extends to $R^*(M_b) \geq b$.

Part (ii): We analyze each mismatch scenario:

Case 1: Block too large ($b \gg r_k$). Consider scale-k component $\mathbf{u}^{(k)}$ with r_k-Markov dependency. For an interior token $\bar{\mathbf{p}}_i$ at distance $d > r_k$ from block boundaries, the scale-k information about $\bar{\mathbf{p}}_i$ is conditionally independent of visible context given the r_k-neighborhood. Specifically:

$$I(\mathbf{u}_i^{(k)}; \bar{\mathbf{P}}_{M_b^c}) \approx 0 \quad \text{for } \min_{j \in M_b^c} |i - j| > r_k \tag{17}$$

Only tokens within distance r_k of boundaries (approximately $2r_k$ tokens total) have nonzero scale-k information from context. Thus:

$$\frac{\text{Scale-}k \text{ learnable signal}}{\text{Total block size}} \approx \frac{2r_k}{b} \to 0 \text{ as } b/r_k \to \infty \tag{18}$$

This means **large blocks cannot effectively learn short-range dependencies**—most masked tokens have no learnable signal for fine-grained patterns.

Case 2: Block too small ($b \ll r_k$). For scale-k patterns with dependency range r_k, full predictive information requires:

$$\mathcal{I}(M_b) = \mathcal{I}_{r_k}(M_b) > \mathcal{I}_r(M_b) \quad \forall r < r_k \tag{19}$$

However, if the model architecture or training procedure only accesses context within range $r_{\text{eff}} < r_k$ (e.g., due to limited attention span or locality bias), then the model cannot capture scale-k information. Specifically, the learnable signal is:

$$\mathcal{I}_{\text{learnable}}(M_b) \leq \mathcal{I}_{\min(r_{\text{eff}}, r_k)}(M_b) < \mathcal{I}(M_b) \quad \text{when } r_{\text{eff}} < r_k \tag{20}$$

Moreover, even with sufficient architectural capacity, small blocks provide weak training signal for long-range patterns because the block size b itself limits the "observable distance" during masked prediction. This means **small blocks cannot effectively learn long-range dependencies**.

Optimal matching: When $b \approx r_k$, (1) interior tokens are within range r_k of boundaries, enabling information flow, and (2) the block size matches the characteristic scale, maximizing learnable signal.

Part (iii): Suppose dependency scales satisfy $r_{k+1}/r_k \geq c > 1$ for some constant c (well-separated scales). For a single block size b:

- If $b \approx r_j$ for some j, then for $k < j$: $b/r_k \geq c^{j-k} \gg 1$ (too large), and for $k > j$: $r_k/b \geq c^{k-j} \gg 1$ (too small).
- From part (ii), scales $k \neq j$ suffer information loss of order $O(1/c^{|j-k|})$.

Therefore, no single b can efficiently capture all scales when $K > 1$ and scales are separated. □

Corollary C.8 (Multi-Scale MTM Optimality)**.** *To learn representations sufficient for prediction at all dependency scales* $(r_1, \ldots, r_K)$*:*

1. *A set of block sizes* $B = \{b_1, \ldots, b_K\}$ *with* $b_k \approx r_k$ *is* ***necessary****.*
2. *The combined objective* $\mathcal{L}_{MTM} = \sum_{b \in B} \mathcal{L}_{MTM}(b)$ *provides training signal at each scale, enabling the model to learn multi-scale dependencies that single-scale masking cannot capture.*

Remark C.9. The proof establishes a complete information-theoretic characterization of the scale-matching problem in masked token modeling. The core mechanism is that block size b creates an information bottleneck: Part (i) shows that reconstructing interior tokens requires context range $R^*(M_b) \geq b$, establishing a lower bound on the dependency scale accessible to a given block size; Part (ii) quantifies the cost of scale mismatch—when $b \gg r_k$, only $O(r_k/b)$ of tokens carry learnable signal for scale-k patterns (short-term information loss), while when $b \ll r_k$, the model cannot access sufficient context to capture scale-k dependencies (long-term information loss); Part (iii) proves this is unavoidable for any single block size when dependency scales are well-separated, as the mismatch penalty degrades exponentially at $O(1/c^{|j-k|})$ for scales $k \neq j$. Together, these results show that multi-scale MTM is not merely an empirical improvement but an information-theoretic necessity: each block size $b_k \approx r_k$ provides an independent learning channel for scale-k dependencies, which is the only way to avoid fundamental information loss across the full spectrum of dependency scales present in sequential data.