Improving Temporal Link Prediction via Temporal Walk Matrix Projection

Xiaodong Lu
CCSE Lab, Beihang University
Beijing, China
xiaodonglu@buaa.edu.cn

Leilei Sun *
CCSE Lab, Beihang University
Beijing, China
leileisun@buaa.edu.cn

Tongyu Zhu
CCSE Lab, Beihang University
Beijing, China
zhutongyu@buaa.edu.cn

Weifeng Lv
CCSE Lab, Beihang University
Beijing, China
lwf@buaa.edu.cn

Abstract

Temporal link prediction, aiming at predicting future interactions among entities based on historical interactions, is crucial for a series of real-world applications. Although previous methods have demonstrated the importance of relative encodings for effective temporal link prediction, computational efficiency remains a major concern in constructing these encodings. Moreover, existing relative encodings are usually constructed based on structural connectivity, where temporal information is seldom considered. To address the aforementioned issues, we first analyze existing relative encodings and unify them as a function of temporal walk matrices. This unification establishes a connection between relative encodings and temporal walk matrices, providing a more principled way for analyzing and designing relative encodings. Based on this analysis, we propose a new temporal graph neural network called TPNet, which introduces a temporal walk matrix that incorporates the time decay effect to simultaneously consider both temporal and structural information. Moreover, TPNet designs a random feature propagation mechanism with theoretical guarantees to implicitly maintain the temporal walk matrices, which improves the computation and storage efficiency. Experimental results on 13 benchmark datasets verify the effectiveness and efficiency of TPNet, where TPNet outperforms other baselines on most datasets and achieves a maximum speedup of $33.3\times$ compared to the SOTA baseline. Our code can be found at https://github.com/lxd99/TPNet.

1 Intorduction

Many real-world dynamic systems can be abstracted as a temporal graph [1], where entities and interactions among them are denoted as nodes and edges with timestamps respectively. Temporal link prediction, aiming at predicting future interactions based on historical interactions, is a fundamental task for temporal graph learning, which can not only help us understand the evolution pattern of the temporal graph but also is crucial for a series of real-world tasks such as recommendations for online platforms [2, 3] and information diffusion prediction [4, 5].

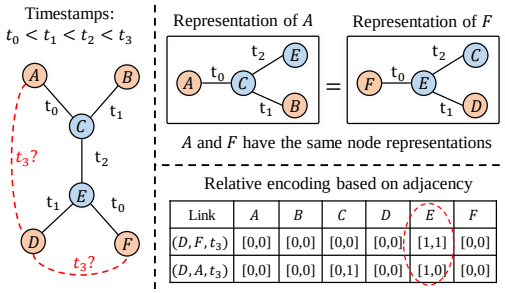
Relative encodings have become an indispensable module for effective temporal link prediction [6–9] where, without them, node representations computed independently by neighbor aggregation will fail

*Corresponding Author.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

 https://doi.org/10.52202/079017-4482

to capture the pairwise information. As the toy example shown in Figure 1, A and F will have the same node representation due to sharing the same local structure. Thus it can not be determined whether D will interact with A or F at t_3 according to their representations. However, by assigning nodes with relative encodings (i.e., additional node features) specific to the target link before computing the node representation, we can highlight the importance of each node and guide the representation learning process to extract pairwise information. For example, in Figure 1, we can infer from the relative encoding of E (in red circle) that D is more likely to interact with F than with A since D and F share a common neighbor, E (detailed discussed in Section 2.2). Although achieving remarkable success, injecting pairwise information based on relative encodings is still far from satisfactory. 1) **On Concept**, existing ways of constructing relative encodings are fragmented as they are derived from different heuristics. There still lacks a unified view on the connections between different relative encodings, which may allow a more principled way to design new relative encodings. 2) **On Method Design**, most existing relative encodings are constructed based on structural connectivity between nodes (e.g., whether two nodes are neighbors), while the temporal information is ignored. 3) **On Computation**, existing methods for relative encodings are inefficient, which usually involve time-consuming graph query operations and need to re-extract the relative encoding from scratch for each target link, making them even become the main computational bottleneck for some methods (shown in Section 4.2).



Link	A	B	C	D	E	F
(D, F, t_3)	[0,0]	[0,0]	[0,0]	[0,0]	[1,1]	[0,0]
(D, A, t_3)	[0,0]	[0,0]	[0,1]	[0,0]	[1,0]	[0,0]

Figure 1: Without relative encodings, the learned node representations fail to capture the correlation between nodes. (For each link (u, v, t), the relative encoding here for a node w is $[g(u, w), g(v, w)]$, where $g(u, w) = 1$ if there is an interaction between u and w before t, otherwise $g(u, w) = 0$.)

To tackle the above issues, this paper makes the following three technical contributions. 1) **A Unified View (Concept)**. We analyze the construction of existing relative encodings and find that they can be uniformly viewed as a function of temporal random walk matrices, where different relative encodings essentially correspond to the construction of a series of temporal walk matrices based on temporal walk weighting. The presented function provides a unified view to analyze existing methods and may allow a more principal way to design new relative encodings. 2) **A Effective and Efficient Method (Method Design and Computation)**. Based on our analysis, we propose a Time decay matrix Projection-based graph neural Network for temporal link prediction, named TPNet for short. TPNet introduces a new temporal walk matrix that incorporates the time decay effect of the temporal graph, simultaneously considering both temporal and structural information. Moreover, TPNet encodes the temporal walk matrix into a series of node representations by random feature propagation, which can be efficiently updated when the graph structure changes and is storage-efficient. Importantly, such node representations can be shared by different link likelihood computation processes to decode the pairwise information, reducing the redundancy computation of re-extracting the relative encodings and avoiding time-consuming graph query operations. 3) **Theoretical and Empirical Analysis.** We conduct a theoretical analysis of the node representations obtained through random feature propagation. Our analysis demonstrates that these representations stem from the random projection of the proposed temporal walk matrix while preserving the inner product of different rows of the matrix. Moreover, we discuss the conditions under which random feature propagation can be applied to improve the computational and storage efficiency of other temporal walk matrices and provide the corresponding propagation mechanisms for temporal walk matrices of existing methods. Empirically, we conduct experiments on 13 benchmark datasets to verify the effectiveness and efficiency of the proposed method, where TPNet outperforms other baselines on most datasets and achieves a maximum speedup of $33.3\times$ compared to the SOTA baseline. Detailed ablation studies also verify the effectiveness of the proposed submodules.

2 Preliminary

In this section, we will first formally define some important concepts in this paper and then give a unified formulation of existing relative encodings.

2.1 Definitons

Definition 1 (Temporal Graph)**.** A temporal graph can be considered as a sequence of non-decreasing chronological interactions $\mathcal{G} = [(\{u_1, v_1\}, t_1), (\{u_2, v_2\}, t_2), \cdots]$ with $0 \leq t_1 \leq t_2 \leq \cdots$, where $(\{u_i, v_i\}, t_i)$ can be considered as a undirected link between u_i and v_i with timestamp t_i. Each node u can be associated with node feature $\boldsymbol{x}_u \in \mathbb{R}^{d_N}$, and each interaction $(\{u, v\}, t)$ has link feature $\boldsymbol{e}^t_{u,v} \in \mathbb{R}^{d_E}$, where d_N and d_E denote the dimensions of the node feature and link feature.

Definition 2 (Temporal Link Prediction)**.** The interaction sequence reflects the graph dynamics, and thus the ability of a model to capture the evolution pattern of a dynamic graph can be evaluated by how accurately it predicts the future interactions based on historical interactions. Formally, given the interactions before t (i.e., $\{(\{u', v'\}, t') | t' < t\}$) and two nodes u, v, the temporal link prediction task aims to predict whether there will be an interaction between u and v at t.

Definition 3 (K-hop Subgraph)**.** We use the notation $\mathcal{G}(t) = (\mathcal{V}(t), \mathcal{E}(t))$ to denote the graph snapshot at t, where $\mathcal{E}(t)$ includes all the interactions that happen before t and $\mathcal{V}(t)$ includes all the nodes appear in $\mathcal{E}(t)$. Besides, we defined the k-hop subgraph of node u as $\mathcal{G}^k_u(t) = (\mathcal{V}^k_u(t), \mathcal{E}^k_u(t)$, where $\mathcal{V}^k_u(t) \subset \mathcal{N}(t)$ is the set of nodes whose shortest path distance to u is less than k on $\mathcal{G}(t)$ and $\mathcal{E}^k_u(t) \subset \mathcal{E}(t)$ is the set of interactions between $\mathcal{V}^k_u(t)$.

Definition 4 (Temporal Walk)**.** A k-step temporal walk W on $\mathcal{G}(t)$ is a sequence of node-time pair with decreased temporal order [6], which can be denoted as $W = [(w_0, t_0), \cdots, (w_k, t_k)]$ with $t = t_0 > t_1 > \cdots > t_k$ and $(\{w_i, w_{i+1}\}, t_{i+1})$ is an edge on $\mathcal{G}(t)$ for $0 \leq i \leq k-1$. Figure 2 shows a visual illustration of a temporal walk. Here, we stipulate the first timestamp t_0 as the current time t to avoid undefinedness of t_0. We use the notation $\mathcal{M}^k_{u,v}(t)$ to denote the set of all k-step temporal walks from u to v on $\mathcal{G}(t)$. Specially, $\mathcal{M}^0_{u,w}(t) = \{[(u, t)]\}$ if $u = w$ and $\mathcal{M}^0_{u,w}(t) = \emptyset$ otherwise. When there is no ambiguity, we replace $\mathcal{M}^k_{u,v}(t)$ with $\mathcal{M}^k_{u,v}$.

2.2 Relative Encoding for Dynamic Link Prediction

2.2.1 Unified Framework

Given a future link (u, v, t) to be predicted, existing temporal link prediction methods (referred as node-wise methods) usually first learn the node representations $\boldsymbol{h}_u(t)$ and $\boldsymbol{h}_v(t)$ independently, which are obtained by encoding their k-hop subgraphs,

$$\boldsymbol{h}_u(t) = f_{\text{enc}}(\mathcal{G}^k_u(t), \boldsymbol{X}^N_{u,k}, \boldsymbol{X}^E_{u,k}), \quad \boldsymbol{h}_v(t) = f_{\text{enc}}(\mathcal{G}^k_v(t), \boldsymbol{X}^N_{v,k}, \boldsymbol{X}^E_{v,k}), \tag{1}$$

where $\boldsymbol{X}^N_{u,k}$ and $\boldsymbol{X}^E_{u,k}$ are the features of nodes and edges in $\mathcal{G}^k_u(t)$ respectively [2]. The $f_{\text{enc}}(\cdot)$ here can be any encoding function that maps a graph into a representation such as a k-layer GNN with a pooling layer Then the link likelihood $p^t_{u,v}$ is given $p^t_{u,v} = f_{\text{dec}}(\boldsymbol{h}_u(t), \boldsymbol{h}_v(t))$. The $f_{\text{dec}}(\cdot)$ is a decoding function that maps the node representations into link likelihood such as an MLP with a Sigmoid output layer. Detailed discussion about existing methods can be found in Appendix F.2.

As mentioned in the Section 1, learning representations independently might fail to capture the pairwise information of the given link. Thus recent methods (referred as link-wise methods) inject the pairwise information by constructing a relative encoding $\boldsymbol{r}^{w|(u,v)}$ for each node $w \in \mathcal{V}^k_u(t) \cup \mathcal{V}^k_v(t)$ as an additional node feature (Detailed construction way for different methods will be introduced in Section 2.2.2). Then Equation (1) will be changed into

$$\boldsymbol{h}_u(t) = f_{\text{enc}}(\mathcal{G}^k_u(t), \boldsymbol{X}^N_{u,k} \oplus \boldsymbol{X}^R_{u,k}, \boldsymbol{X}^E_{u,k}), \quad \boldsymbol{h}_v(t) = f_{\text{enc}}(\mathcal{G}^k_v(t), \boldsymbol{X}^N_{v,k} \oplus \boldsymbol{X}^R_{v,k}, \boldsymbol{X}^E_{v,k}), \tag{2}$$

where $\boldsymbol{X}^R_{u,k}$ is the relative encodings for nodes in $\mathcal{G}^k_u(t)$ and $\boldsymbol{X}^N_{u,k} \oplus \boldsymbol{X}^R_{u,k}$ indicate the new node features obtained by concatenating $\boldsymbol{X}^N_{u,k}$ and $\boldsymbol{X}^R_{u,k}$. Intuitively, the relative encoding $\boldsymbol{r}^{w|(u,v)}$ for each node w reflects its importance to predicted link (u, v, t), which can guide the representation learning process to extract the pairwise information specific to the predicted link from the subgraph. For the detailed construction way, the relative encoding $\boldsymbol{r}^{w|(u,v)}$ is the concatenation of two similarity features $\boldsymbol{r}^{w|u}$ and $\boldsymbol{r}^{w|v}$, where $\boldsymbol{r}^{w|u}/\boldsymbol{r}^{w|v}$ encodes some form of similarity between u/v and w (e.g., the number of k-step temporal walks from u to w). Although the designs of similarity feature $\boldsymbol{r}^{w|u}$

[2]For memory-based methods (i.e, TGN), $\boldsymbol{X}^N_{u,k}$ represents the concatenation of node memories and features.

https://doi.org/10.52202/079017-4482

for different methods are induced from different heuristics, we find that they can be unified in a function about the temporal walk matrix, which is

$$\boldsymbol{r}^{w|u} = g([A^{(0)}_{u,w}(t), A^{(1)}_{u,w}(t), \cdots, A^{(k)}_{u,w}(t)]), \quad A^{(i)}_{u,w}(t) = \sum_{W \in \mathcal{M}^{i}_{u,w}} s(W) \text{ for } 0 \leq i \leq k. \quad (3)$$

The $s(\cdot)$ here is a score function that maps a temporal walk to a scalar and $\boldsymbol{A}^{(i)}(t)$ denotes an i-hop temporal walk matrix whose each entry $A^{(i)}_{u,w}$ is the sum of the score of all i-step temporal walks from u to w. $g(\cdot)$ is a function applied on the vector of $[A^{(0)}_{u,w}(t), A^{(1)}_{u,w}(t), \cdots, A^{(k)}_{u,w}(t)] \in \mathbb{R}^{k+1}$ to extract similarity feature. The above equation shows that each relative encoding implies a construction of the temporal walk matrix based on weighting each temporal walk (i.e., $s(\cdot)$). Next, we will analyze existing relative encodings and show how they can be represented in the form of Equation (3).

2.2.2 Analysis of Existing Methods

Our analysis focuses on the four representative link-wise methods DyGFormer, PINT, NAT, and CAW, which cover all the link-wise methods in the benchmark of dynamic link prediction [9].

DyGFormer [9]. The $\boldsymbol{r}^{w|u}$ of DyGFormer is a one-dimensional vector representing the number of links between w and u. For DyGFormer, we can first set the $s(\cdot)$ to be a one-const function (i.e., $s(W) = 1$ in for any W), which will make $A^{(k)}_{u,w}$ be the number of the k-step temporal walks from u to w. Then, setting $g(\cdot)$ to be a function that selects the second number of a vector (i.e., $g([x_0, x_1, .., x_k]) = x_1$) will make Equation (3) generate the similarity feature of DyGFormer.

PINT [8]. The $\boldsymbol{r}^{w|u}$ of PINT is a $(k+1)$-dimensional vector, where $r^{w|u}_i$ is the number of $(i-1)$-step temporal walks from u to w for $1 \leq i \leq k+1$. Setting $s(\cdot)$ and $g(\cdot)$ can be set to a one-const function and an identity function respectively will make Equation (3) outputs the similarity feature of PINT.

NAT [7]. NAT maintains a series of fix-sized hash maps $s^{(0)}_u, ..., s^{(k)}_u$ and generates the $\boldsymbol{r}^{w|u}$ based on the hash maps. As proved in Appendix A.1, if the size of the hash maps becomes infinite, the $\boldsymbol{r}^{w|u}$ is equivalent to a $(k+1)$-dimensional vector, where, for $1 \leq i \leq k+1$, $r^{w|u}_i = 1$ if the shortest temporal walk from u to w is less than i; otherwise, $r^{w|u}_i = 0$. Let $h(\cdot)$ be a binary function where $h(x) = 1$ if $x > 0$ and $h(x) = 0$ otherwise. Setting the $s(\cdot)$ to be a one-const function and $g(\cdot)$ to a function of $g([x_0, x_1, ..., x_k]) = [h(\sum^0_{i=0} x_i), h(\sum^1_{i=0} x_i), ..., h(\sum^k_{i=0} x_i)]$ can make the Equation (3) generate the similarity feature of NAT.

CAWN [6]. The similarity feature $\boldsymbol{r}^{w|u}$ of CAWN reflects the probability of a node w being visited when performing a temporal walk from u. Specifically, CAWN first samples a set of temporal walks beginning from u based on a causal sampling strategy. Then, for each node w, the similarity feature $\boldsymbol{r}^{w|u}$ is extracted based on its occurrence in the sampled walks. As proved in Appendix A.2, the similarity feature $\boldsymbol{r}^{w|u}$ is the estimation of a $(k+1)$-dimensional vector, where, for $1 \leq i \leq k+1$, $r^{w|u}_i$ is the probability of visiting w through a $(i-1)$-step temporal walk matrix based on the sampling strategy of CAWN, which can be represented as $r^{w|u}_i = \sum_{W \in \mathcal{M}^{i-1}_{u,w}} s'(W)$. The value $s'(W)$ for a given temporal walk $W = [(w_0, t_0), (w_1, t_1), ..., (w_k, t_k)]$ is defined as $\prod^{k-1}_{i=0} \frac{\exp(-\alpha(t_i - t_{i+1}))}{\sum_{(\{w', w\}, t') \in \mathcal{E}_{w_i, t_i}} \exp(-\alpha(t_i - t'))}$, where α is hyperparameter to control the sampling process, $\mathcal{E}_{w_i, t_i} = \{(\{w', w\}, t') | t' < t_i\}$ is the set of interactions attached to w_i before t_i, and $\frac{\exp(-\alpha(t_i - t_{i+1}))}{\sum_{(\{w', w\}, t') \in \mathcal{E}_{w_i, t_i}}}$ can be considered as the probability of moving from (w_i, t_i) to (w_{i+1}, t_{i+1}) in the sampling process. Setting $s(\cdot)$ to $s'(\cdot)$ and $g(\cdot)$ to be an identity function can make Equation (3) generate the similarity feature of CAWN.

Conclusion. According to the above analysis, we can conclude that (i) Equation (3) provides a unified view to consider the injection of pairwise information from the construction of temporal walk matrix, where different relative encodings (implicitly or explicitly) correspond to a kind of temporal walk matrix. (ii) Examining existing relative encodings from the unified view reveals their limitations. First, the relative encodings of existing methods (except CAWN) ignore the temporal information carried by each temporal walk, where their score function $s(\cdot)$ always yield 1 and the entries of the temporal walk matrix is just the count of the temporal walks. Next, although CAWN considers temporal information, they estimate the temporal walk matrix from the sampled temporal

Algorithm 1: Node Representation Maintaining ($\mathcal{G}$,λ,k,n,d_R)

Initialize $\boldsymbol{H}^{(0)}, \boldsymbol{H}^{(1)}, ..., \boldsymbol{H}^{(k)} \in \mathbb{R}^{n \times d_R}$ as zero matrix ;
Fill $\boldsymbol{H}^{(0)}$ with entries independently drawn from $\mathcal{N}(0, \frac{1}{d_R})$;
for $(u, v, t) \in \mathcal{G}$ **do**
  for $l = k$ **to** 1 **do**
    $\boldsymbol{H}_u^{(l)} = \boldsymbol{H}_u^{(l)} + e^{\lambda t} * \boldsymbol{H}_v^{(l-1)}$;
    $\boldsymbol{H}_v^{(l)} = \boldsymbol{H}_v^{(l)} + e^{\lambda t} * \boldsymbol{H}_u^{(l-1)}$;
Return $\boldsymbol{H}^{(0)}, \boldsymbol{H}^{(1)}, .., \boldsymbol{H}^{(k)}$;

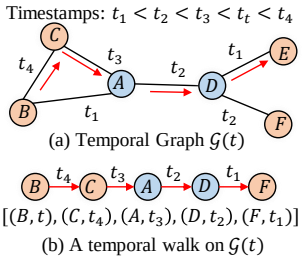


Figure 2: A illustration of the temporal walk.

walks, which needs time-consuming graph sampling and may introduce large estimation errors. In the next section, we will present a new temporal walk matrix to simultaneously consider the temporal and structural information and show how to efficiently maintain the proposed temporal walk matrix.

3 Methodology

TPNet mainly consists of two modules: Node Representation Maintaining (NRM) and Link Likelihood Computing (LLC). The NRM is responsible for encoding the pairwise information, which maintains a series of node representations. Such representations will be updated when new interaction occurs and can be used to decode the temporal walk information between two nodes. The LLC module is a prediction module, which utilizes the maintained node representations and auxiliary information (e.g., like features) to compute the likelihood of the link to be predicted.

3.1 Node Representation Maintaining

Temporal Matrix Construction. Based on the Equation (3), designing a temporal walk matrix relies on the definition of the score function $s(\cdot)$, where the element of a temporal walk matrix is $A_{u,v}^{(k)}(t) = \sum_{W \in M_{u,v}^k} s(W)$. Unlike most previous methods that only count the number of temporal walks, we consider the temporal information carried by a temporal walk in $s(\cdot)$ to simultaneously consider the temporal and structural information. Formally, let t be the current time, given a temporal walk $W = [(w_0, t_0), (w_1, t_1), .., (w_k, t_k)]$, the value of the score function is $s(W) = \prod_{i=1}^{k} e^{-\lambda(t-t_i)}$, where $\lambda > 0$ is a hyperparameter to control the time decay weight. As the current time t goes on, for each interaction $(\{w_i, w_{i+1}\}, t_{i+1})$ in the temporal walk W, its weight $e^{-\lambda(t-t_i)}$ will decay exponentially. The design of the score function is motivated by the widely observed time decay effect [1, 10] on the temporal graph, where the importance of interactions will decay as time goes on, benefiting better modeling the graph evolution patterns. In the following part of Section 3, the notation of $s(\cdot)$ and $\boldsymbol{A}^{(k)}(t)$ will specifically refer to the score function and temporal walk matrix proposed in this part. Besides, for $\boldsymbol{A}^{(0)}(t)$, we stipulate it as an identity matrix constantly.

Node Representation Maintaining. Directly computing the temporal walk matrices is impractical since we need to enumerate the temporal walks and each matrix needs expensive $O(n \times n)$ space complexity to store. Thus, we implicitly maintain the temporal walk matrices by maintaining a series of node representations $\boldsymbol{H}^{(0)}(t), \boldsymbol{H}^{(1)}(t), ..., \boldsymbol{H}^{(k)}(t) \in \mathbb{R}^{n \times d_R}$, where $d_R \ll n$ is the dimension of node representations and $\boldsymbol{H}_u^{(l)}(t) \in \mathbb{R}^{d_R}$ encodes the information about the l-step temporal walks beginning from u for $0 \leq l \leq k$. The node representations will be updated when a new interaction occurs. Specifically, we first construct a random feature matrix $\boldsymbol{P} \in \mathbb{R}^{n \times d_R}$, where each entry of $\boldsymbol{P}$ is independently drawn from Gaussian distribution with mean 0 and variance $\frac{1}{d_R}$. Then we initialize $\boldsymbol{H}^{(0)}(t)$ as $\boldsymbol{P}$ and $\boldsymbol{H}^{(1)}(t), \boldsymbol{H}^{(2)}(t), ..., \boldsymbol{H}^{(k)}(t)$ as zero matrix. When a new interaction (u, v, t) occurs, for $1 \leq l \leq k$, we update the representations of u and v by

$$\boldsymbol{H}_u^{(l)}(t^+) = \boldsymbol{H}_u^{(l)}(t) + e^{\lambda t} * \boldsymbol{H}_v^{(l-1)}(t), \quad \boldsymbol{H}_v^{(l)}(t^+) = \boldsymbol{H}_v^{(l)}(t) + e^{\lambda t} * \boldsymbol{H}_u^{(l-1)}(t), \tag{4}$$

where the t^+ denotes the time right after t. A pseudocode for maintaining the node representations is shown in Algorithm 1. The maintaining mechanism here can be considered as a random feature propagation mechanism on the temporal graph, where we initialize the zero layer's representation

https://doi.org/10.52202/079017-4482

of each node as a random feature and repeatedly propagate these features among nodes from the low layer to the high layer as interactions continuously appear. The following theorem shows the relationship between the node representations and the temporal walk matrices.

Theorem 1. *If any two interactions on temporal graph $\mathcal{G}$ have different timestamps, the obtained representations $\boldsymbol{H}^{(0)}(t), \boldsymbol{H}^{(1)}(t), ..., \boldsymbol{H}^{(k)}(t)$ satisfy $e^{-\lambda lt} * \boldsymbol{H}^{(l)}(t) = \boldsymbol{A}^{(l)}(t)\boldsymbol{P}$ for $0 \leq l \leq k$.*

For simplicity, we assume the timestamps of the interaction are different, and we show how to update the representations when multiple interactions have the same timestamps in Appendix C. We leave the proof in the Appendix B.1. The above theorem shows that the obtained node representation is the projection (i.e. linear transformation) of the temporal walk matrices, where the transform matrix is the initial random feature matrix $\boldsymbol{P}$. The next theorem shows that the node representations preserve the inner product of the temporal walk matrices.

Theorem 2. *Given any $\epsilon \in (0, 1)$, let $\| \cdot \|_2$ denote the L2 norm, $c_{u,v}^{l_1,l_2}$ denote $\frac{1}{2}(\|\boldsymbol{A}_u^{(l_1)}(t)\|_2^2 + \|\boldsymbol{A}_v^{(l_2)}(t)\|_2^2)$, and $\bar{\boldsymbol{H}}^{(l)}(t)$ denote $e^{-\lambda lt} * \boldsymbol{H}^{(l)}(t)$, if dimension d_R of node representations satisfy $d_R \geq \frac{24}{\epsilon^2}\log(4^{1/3}(k+1)n)$, then for any $1 \leq u, v \leq n$, and $0 \leq l_1, l_2 \leq k$, we have*

$$\mathbb{P}\left(\left|\langle \bar{\boldsymbol{H}}_u^{(l_1)}(t), \bar{\boldsymbol{H}}_v^{(l_2)}(t)\rangle - \langle \boldsymbol{A}_u^{(l_1)}(t), \boldsymbol{A}_v^{(l_2)}(t)\rangle\right| \leq \epsilon c_{u,v}^{l_1,l_2}\right) \geq 1 - \frac{1}{(k+1)n}, \tag{5}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product and $|\cdot|$ denotes taking absolute value.

We leave the proof in Appendix B.2. The above theorem shows that the inner product of the node representations is approximately equal to the inner product of the temporal walk matrices (i.e., $\langle \bar{\boldsymbol{H}}_u^{(l_1)}(t), \bar{\boldsymbol{H}}_v^{(l_2)}(t)\rangle \approx \langle \boldsymbol{A}_u^{(l_1)}(t)), \boldsymbol{A}_v^{(l_2)}(t)\rangle$). Specifically, given any error rate ϵ, if the dimension of the node representations satisfies a certain condition, the difference between $\langle \bar{\boldsymbol{H}}_u^{(l_1)}(t), \bar{\boldsymbol{H}}_v^{(l_2)}(t)\rangle$ and $\langle \boldsymbol{A}_u^{(l_1)}(t)), \boldsymbol{A}_v^{(l_2)}(t)\rangle$ for any u, v, l_1, l_2 will be less than $\epsilon c_{u,v}^{l_1,l_2}$ with high probability ($\geq 1 - \frac{1}{(k+1)n}$). The error rate can approach 0 infinitely and thus the inner product of the node representations can approach that of temporal walk matrices infinitely, at the cost of increasing the dimension d_R. As we will see in Section 4.3, a small dimension ($\ll n$) in practice is enabled to make the inner product of node representations a good estimation of that of temporal walk matrices. Additionally, due to the i-th row of $\boldsymbol{A}^{(0)}(t)$ being the one-hot encoding of i (since it is an identity matrix), we have $\langle \boldsymbol{A}_u^{(l)}(t), \boldsymbol{A}_w^{(0)}(t)\rangle = A_{u,w}^{(l)}(t)$. Thus, we can obtain $[A_{u,w}^{(0)}(t), \cdots, A_{u,w}^{(k)}(t)]$ in Equation (3) by calculating the inner product between all layers of u's representation and the zero layer of w's representation, expressed as $[\langle \bar{\boldsymbol{H}}_u^{(0)}(t), \bar{\boldsymbol{H}}_w^{(0)}(t)\rangle, \cdots, \langle \bar{\boldsymbol{H}}_u^{(k)}(t), \bar{\boldsymbol{H}}_w^{(0)}(t)\rangle]$.

Remark. Compared to directly computing the temporal walk matrices $\boldsymbol{A}^{(0)}(t), .., \boldsymbol{A}^{(k)}(t)$, which needs to enumerate the temporal walks and $O((k+1)n^2)$ space complexity to store, maintaining the node representations largely improve the computation and storage efficiency, which only needs $O((k+1)nd_R)$ space complexity to store and $O(kd_R)$ time complexity to update when a new interaction occurs. Actually, Theorem 2 is the direct result of Theorem 1 based on Johnson-Lindenstrauss Lemma [11], where the random projection can preserve the inner product and norm [12]. Notably, the method for implicitly maintaining temporal walk matrices via random feature propagation can be extended to other types of temporal walk matrices, provided they meet a specific condition (i.e., the updating function of the temporal walk matrix can be written as the linear combination of its rows). This condition is not restrictive, and all the temporal walk matrices discussed in Section 2 fulfill it. We show their propagation mechanism and related discussion in Appendix D. In conclusion, the unified function in Equation (3), combined with methods of implicitly maintaining the temporal walk matrices, provides a new way to design a more effective and efficient way to inject pairwise information.

3.2 Link Likelihood Computing

Given an interaction (u, v, t) to be predicted, we compute its happening likelihood based on the obtained node representations and auxiliary features. Specifically, we first decode a pairwise feature $\boldsymbol{f}_{u,v}(t)$ from the node representations obtained in Section 3.1. Then we compute the node embeddings $\boldsymbol{h}_u(t)$ and $\boldsymbol{h}_v(t)$ for node u and v respectively based on their historical interactions. Finally, we give the link likelihood based on $\boldsymbol{h}_u(t), \boldsymbol{h}_v(t), \boldsymbol{f}_{u,v}(t)$. For notation simplicity, we omit the suffix of $\boldsymbol{h}_u(t), \boldsymbol{h}_v(t), \boldsymbol{f}_{u,v}(t)$ and denote them as $\boldsymbol{h}_u, \boldsymbol{h}_v, \boldsymbol{f}_{u,v}$ in the following part.

Pairwise Feature Decoding. Although we can obtain the $(k+1)$-dimensional feature in Equation (3) by calculating the inner product between the zero-layer representation of one node and all layers of the other node's representation, this method does not consider the correlation between all layers of both nodes. Therefore, we use representations from all layers to decode the pairwise information. Specifically, we first take the node representation of u and v from different layers, which can be denoted as $\boldsymbol{F}_* = [e^{-\lambda 0 t}\boldsymbol{H}_*^{(0)}, ..., e^{-\lambda k t}\boldsymbol{H}_*^{(k)}] \in \mathbb{R}^{(k+1)\times d_R}$ with * could be u or v. Then we concatenate them together to get $\boldsymbol{F}_{u,v} = [\boldsymbol{F}_u, \boldsymbol{F}_v] \in \mathbb{R}^{2(k+1)\times d_R}$ and obtain the raw pairwise feature $\tilde{\boldsymbol{f}}_{u,v}$ by computing the inner product among different rows of $\boldsymbol{F}_{u,v}$, which is $\tilde{\boldsymbol{f}}_{u,v} = \text{flat}(\boldsymbol{F}_{u,v}\boldsymbol{F}_{u,v}^T)$ with flat$(\cdot)$ means flatten a matrix of $\mathbb{R}^{2(k+1)\times 2(k+1)}$ into a vector of $\mathbb{R}^{4(k+1)^2}$. Finally, we feed the raw pairwise feature $\tilde{\boldsymbol{f}}_{u,v}$ into an MLP to get the pairwise feature $\boldsymbol{f}_{u,v}$, which is $\boldsymbol{f}_{u,v} = \text{MLP}(\log(\text{ReLU}(\tilde{\boldsymbol{f}}_{u,v}) + 1))$. The ReLU$(\cdot)$ here is used to reduce estimation error, where the inner product of temporal walk matrices should be larger than zero and we thus set it to be zero if the inner product of the node representations is negative. The $\log(\cdot)$ is used to scale the raw pairwise feature, where the range of the inner product between different layers varies greatly and the $+1$ is the shift term to avoid the undefined value of $\log(0)$. We will see in Section 4.3 that these two operations can improve the training stability.

Auxiliary Feature Learning. The auxiliary features such as link features also provide rich information for modeling the evolution patterns of the temporal graph. In this part, we follow the previous methods [13, 9] and consider the auxiliary feature learning as a sequential learning problem. Specifically, for node u, we take its recent m interactions $S_u = [(\{u, w_1\}, t_1), ..., (\{u, w_m\}, t_m)]$ before t and learn node embedding $\boldsymbol{h}_u$ from this sequence. We first fetch the node features $\boldsymbol{X}_{u,N} = [\boldsymbol{x}_{w_1}, ..., \boldsymbol{x}_{w_m}] \in \mathbb{R}^{m\times d_N}$ and edge features $\boldsymbol{X}_{u,E} = [\boldsymbol{e}_{u,w_1}^{t_1}, .., \boldsymbol{e}_{u,w_1}^{t_m}] \in \mathbb{R}^{m\times d_E}$. For timestamps, we map the timestamps into temporal features $\boldsymbol{X}_{u,T} = [\phi(t-t_1), .., \phi(t-t_n)] \in \mathbb{R}^{m\times d_T}$ like [14], where $\phi(\Delta t) = [\cos(w_1\Delta t), .., \cos(w_{d_T}\Delta t)]$ is a time encoding function to learn the periodic temporal pattern. Besides, we construct a relative encoding sequence $\boldsymbol{X}_{u,F} = [\boldsymbol{f}_{u,w_1} \oplus \boldsymbol{f}_{v,w_1}, ..., \boldsymbol{f}_{u,w_m} \oplus \boldsymbol{f}_{v,w_m}] \in \mathbb{R}^{m\times 8(k+1)^2}$ to inject the pairwise features, where $\boldsymbol{f}_{u,w_m}$ denote the pairwise feature of u and w_m and $\oplus$ is the concatenation operation. After obtaining the above feature sequence, we concatenate them together and feed it into an MLP to get the final feature sequence $\boldsymbol{Z}_u^{(0)} = \text{MLP}([\boldsymbol{X}_{u,N}, \boldsymbol{X}_{u,E}, \boldsymbol{X}_{u,T}, \boldsymbol{X}_{u,F}]) \in \mathbb{R}^{m\times d}$. Subsequently, we stack l layers of MLP-Mixer [15] to capture the temporal and structural dependencies within the feature sequence, which is

$$\begin{aligned}\tilde{\boldsymbol{Z}}_u^{(l)} &= \boldsymbol{Z}_u^{(l-1)} + \boldsymbol{W}_1^{(l)}\text{GeLU}(\boldsymbol{W}_2^{(l)}\text{LayerNorm}(\boldsymbol{Z}_u^{(l-1)})) \\ \boldsymbol{Z}_u^{(l)} &= \tilde{\boldsymbol{Z}}_u^{(l)} + \boldsymbol{W}_3^{(l)}\text{GeLU}(\boldsymbol{W}_4^{(l)}\text{LayerNorm}(\tilde{\boldsymbol{Z}}_u^{(l)})).\end{aligned} \tag{6}$$

Finally, we get the node embedding by mean pooling $\boldsymbol{h}_u = \text{MEAN}(\boldsymbol{Z}_u^{(l)})$. The procedure to get node embedding $\boldsymbol{h}_v$ is similar and for the node that does not have m interactions, we pad the feature sequence with zero. Then, the likelihood of the link (u, v, t) is given by $p_{u,v}^t = \text{MLP}([\boldsymbol{h}_u, \boldsymbol{h}_v, \boldsymbol{f}_{u,v}])$, where MLP$(\cdot)$ is a 2-layer MLP model with Sigmoid activation function in its output layer.

4 Experiments

4.1 Experimental Settings

Datasets and Baselines. We conduct experiments on 13 benchmark datasets for temporal link prediction, which are Wikipedia, Reddit, MOOC, LastFM, Enron, Social Evo., UCI, Flights, Can. Parl., US Legis., UN Trade, UN Vote and Contact. Eleven popular temporal graph learning methods are selected as baselines including JODIE, DyRep, TGAT, TGN, CAWN, EdgeBank, TCL, GraphMixer, NAT, PINT, and DyGFormer. Details about datasets and baselines can be found in Appendix F.

Task Settings. The task settings strictly follow [9]. Specifically, we conduct experiments under two settings: 1) the transductive setting that predicts links between nodes that have been seen during training and 2) the inductive setting that predicts links between nodes that are not seen during training. Three different negative sampling strategies introduced by [16] are used to sample the negative links and the Average Precision (AP) and Area Under the Receiver Operating Characteristic Curve (AUC-ROC) are adopted as the evaluation metrics. For dataset splitting, we chronologically split

 https://doi.org/10.52202/079017-4482

Table 1: Transductive results for different baselines under the random negative sampling strategy. **blod** and underline highlight the best and second best result respectively.

Metric	Dataset	JODIE	DyRep	TGAT	TGN	CAWN	EdgeBank	TCL	GraphMixer	NAT	PINT	DyGFormer	TPNet
AP	Wikipedia	96.50±0.14	94.86±0.06	96.94±0.06	98.45±0.06	98.76±0.03	90.37±0.00	96.47±0.16	97.25±0.03	98.03±0.07	98.45±0.04	99.03±0.02	**99.32±0.03**
	Reddit	98.31±0.14	98.22±0.04	98.52±0.02	98.63±0.06	99.11±0.01	94.86±0.00	97.53±0.02	97.31±0.01	99.13±0.10	99.15±0.02	99.22±0.01	**99.27±0.00**
	MOOC	80.23±2.44	81.97±0.49	85.84±0.15	89.15±1.60	80.15±0.25	57.97±0.00	82.38±0.24	82.78±0.15	85.88±0.55	88.08±0.86	87.52±0.49	**96.39±0.09**
	LastFM	70.85±2.13	71.92±2.21	73.42±0.21	77.07±3.97	86.99±0.06	79.29±0.00	67.27±2.16	75.61±0.24	88.02±1.94	89.66±1.81	93.00±0.12	**94.50±0.08**
	Enron	84.77±0.30	82.38±3.36	71.12±0.97	86.53±1.11	89.56±0.09	83.53±0.00	79.70±0.71	82.25±0.16	90.60±0.66	92.20±0.15	92.47±0.12	**92.90±0.17**
	Social Evo.	89.89±0.55	88.87±0.30	93.16±0.17	93.57±0.17	84.96±0.09	74.95±0.00	93.13±0.16	93.37±0.07	88.92±3.45	94.42±0.03	**94.73±0.01**	**94.73±0.02**
	UCI	89.43±1.09	65.14±2.30	79.63±0.70	92.34±1.04	95.18±0.06	76.20±0.00	89.57±1.63	93.25±0.57	93.40±0.26	96.45±0.11	95.79±0.17	**97.35±0.04**
	Flights	95.60±1.73	95.29±0.72	94.03±0.18	97.95±0.14	98.51±0.01	89.35±0.00	91.23±0.02	90.99±0.05	98.57±0.12	98.80±0.02	98.91±0.01	**98.93±0.02**
	Can. Parl.	69.26±0.31	66.54±2.76	70.73±0.72	70.88±2.34	69.82±2.34	64.55±0.00	68.67±2.67	77.04±0.46	79.72±1.76	68.36±1.43	**97.36±0.45**	90.28±0.37
	US Legis.	75.05±1.52	75.34±0.39	68.52±3.16	75.99±0.58	70.58±0.48	58.39±0.00	69.59±0.48	70.74±1.02	78.71±0.87	74.85±0.97	71.11±0.59	**80.58±0.23**
	UN Trade	64.94±0.31	63.21±0.93	61.47±0.18	65.03±1.37	65.39±0.12	60.41±0.00	62.21±0.03	62.61±0.27	73.95±1.16	70.20±0.58	66.46±1.29	**87.24±0.65**
	UN Vote	63.91±0.81	62.81±0.80	52.21±0.98	65.72±2.17	52.84±0.10	58.49±0.00	51.90±0.30	52.11±0.16	70.45±0.68	66.25±0.78	55.55±0.42	**75.12±0.29**
	Contact	95.31±1.33	95.98±0.15	96.28±0.09	96.89±0.56	90.26±0.28	92.58±0.00	92.44±0.12	91.92±0.03	97.39±0.22	98.64±0.02	98.29±0.01	**98.66±0.01**
	Avg. Rank	7.85	8.77	8.54	5.00	7.00	10.54	9.77	8.31	4.15	3.69	3.15	**1.08**
AUC	Wikipedia	96.33±0.07	94.37±0.09	96.67±0.07	98.37±0.07	98.54±0.04	90.78±0.00	95.84±0.18	96.92±0.03	97.75±0.11	98.16±0.06	98.91±0.02	**99.30±0.02**
	Reddit	98.31±0.05	98.17±0.05	98.47±0.02	98.60±0.06	99.01±0.01	95.37±0.00	97.42±0.02	97.17±0.02	99.09±0.10	99.09±0.03	99.15±0.01	**99.22±0.00**
	MOOC	83.81±2.09	85.03±0.58	87.11±0.19	91.21±1.15	80.38±0.26	60.86±0.00	83.12±0.18	84.01±0.17	87.42±0.58	90.55±0.43	87.91±0.58	**97.17±0.08**
	LastFM	70.49±1.66	71.16±1.89	71.59±0.18	78.47±2.94	85.92±0.10	83.77±0.00	64.06±1.16	73.53±0.12	86.92±2.72	89.28±1.63	93.05±0.10	**94.39±0.04**
	Enron	87.96±0.52	84.89±3.00	68.89±1.10	88.32±0.99	90.45±0.14	87.05±0.00	75.74±0.72	84.38±0.21	91.68±0.83	92.87±0.34	93.33±0.13	**93.98±0.26**
	Social Evo.	92.05±0.46	90.76±0.21	94.76±0.16	95.39±0.17	87.34±0.08	81.60±0.00	94.84±0.17	95.23±0.07	90.84±3.72	96.16±0.02	96.30±0.01	**96.43±0.02**
	UCI	90.44±0.49	68.77±2.34	78.53±0.74	92.03±1.13	93.87±0.08	77.30±0.00	87.82±1.36	91.81±0.67	92.31±0.37	95.57±0.16	94.49±0.26	**96.79±0.05**
	Flights	96.21±1.42	95.95±0.62	94.13±0.17	98.22±0.13	98.45±0.01	90.23±0.00	91.21±0.02	91.13±0.01	98.69±0.10	98.89±0.02	98.93±0.01	**99.00±0.02**
	Can. Parl.	78.21±0.23	73.35±3.67	75.69±0.78	76.99±1.80	75.70±3.27	64.14±0.00	72.46±3.23	83.17±0.53	84.04±1.13	77.96±1.46	**97.76±0.41**	92.05±0.34
	US Legis.	82.85±1.07	82.28±0.32	75.84±1.99	83.34±0.43	77.16±0.39	62.57±0.00	76.27±0.63	76.96±0.79	85.36±0.52	82.10±0.85	77.90±0.58	**86.49±0.18**
	UN Trade	69.62±0.44	67.44±0.83	64.01±0.12	69.10±1.67	68.54±0.18	66.75±0.00	64.72±0.05	65.52±0.51	77.61±1.36	74.87±0.53	70.20±1.44	**89.17±0.46**
	UN Vote	68.53±0.95	67.18±1.04	52.83±1.12	69.71±2.65	53.09±0.22	62.97±0.00	51.88±0.36	52.46±0.27	75.32±0.63	70.69±1.02	57.12±0.62	**79.88±0.30**
	Contact	96.66±0.89	96.48±0.14	96.95±0.08	97.54±0.35	89.99±0.34	94.34±0.00	94.15±0.09	93.94±0.02	97.79±0.16	98.90±0.02	98.53±0.01	**98.91±0.01**
	Avg. Rank	7.15	8.69	8.92	5.08	7.15	10.31	10.15	8.62	4.08	3.46	3.23	**1.08**

each dataset with $70\%/15\%/15\%$ for training/validating/testing. The training and testing pipeline is the same as that in [9].

Model Configuration. For TPNet, the layer l of node representations, the number of recent interactions m, and dimension d_R of the node representations are set to 3, 20 and $10 * \log(2E)$, where E is the number of the interactions. We find the best time decay weight λ via grid search within a range of 10^{-4} to 10^{-7}. Specifically, the best λ for Contact is 10^{-4}, the best λ for MOOC, Can. Parl. and UN Vote is 10^{-5}, the best λ for Wikipedia, Reddit, Enron, Social Evo., Flights and US Legis. is 10^{-6}, the best λ for LastFM, UCI and UN Trade is 10^{-7}.

Implementatoin Details. For baselines, we use the implementation of DyGLib [9], which is a unified temporal graph learning library, and has tuned the best hyperparameters for each baseline. For baselines that are not included in DyGLib (i.e., NAT and PINT), we use their official implementation and find the best hyperparameters via grid search. Experiments are conducted on a Ubuntu server, whose CPU and GPU devices are one Intel(R) Xeon(R) Gold 6226R CPU @ 2.9GHz with 64 CPU cores and four GeForce RTX 3090 GPUs with 24 GB memory respectively. We run each experiment five times and report the average.

4.2 Performance and Efficiency Comparison

Performance comparison among baselines. The performance of TPNet and baselines is shown in Table 1. Due to space limit, Table 1 only shows the results under the transductive setting with random negative sampling strategy and more similar results can be found in Appendix G.1. As shown in Table 1, TPNet achieves the best performance among all the methods on most datasets, verifying its effectiveness. Besides, the link-wise methods (CAWN, NAT, PINT, and DyGFormer) perform better than the node-wise methods, indicating the importance of injecting the pairwise feature. Compared to the baselines, TPNet simultaneously considers the temporal and structural correlations between nodes and encodes the temporal walk matrix into node representations with theoretical guarantees, which contributes to its superior performance.

Efficiency Analysis. We compare the relative inference time of different methods to TPNet to evaluate their efficiency. The results on LastFM and MOOC are shown in Figure 4 and more results can be found in Appendix G.2. As shown in Figure 4, TPNet not only achieves the best performance but is also more efficient than other link-wise methods, where TPNet achieves a $33\times$ and $71.5\times$

Table 2: Ablation study results, where N/A indicates the numerical overflow error.

Datasets	TPNet	w/o NR	w/o Time	w/o Scale
MOOC	$96.39_{\pm 0.09}$	$83.21_{\pm 0.23}$	$94.62_{\pm 0.34}$	$63.04_{\pm 0.95}$
LastFM	$94.50_{\pm 0.08}$	$76.52_{\pm 0.41}$	$94.30_{\pm 0.03}$	N/A
Enron	$92.90_{\pm 0.17}$	$83.23_{\pm 0.13}$	$92.85_{\pm 0.17}$	N/A
UCI	$97.35_{\pm 0.04}$	$88.70_{\pm 2.53}$	$97.19_{\pm 0.10}$	$73.13_{\pm 2.57}$
US Legis.	$80.58_{\pm 0.23}$	$69.47_{\pm 1.33}$	$71.83_{\pm 0.52}$	$70.44_{\pm 1.97}$
UN Trade	$87.24_{\pm 0.65}$	$62.56_{\pm 0.32}$	$65.98_{\pm 0.45}$	$56.58_{\pm 1.08}$
UN Vote	$75.12_{\pm 0.29}$	$52.58_{\pm 0.66}$	$54.80_{\pm 0.28}$	$53.20_{\pm 1.39}$

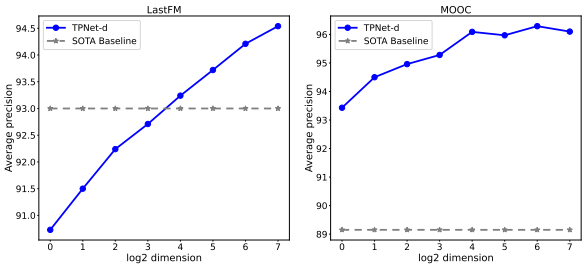


Figure 3: Influence of node representation dimension.

speedups compared to the SOTA baseline DyGFormer and CAWN respectively on LastFM. The CAWN and DyGFormer models utilize time-consuming graph queries (e.g., temporal walk sampling) to construct relative encodings, which constitute the main computational bottleneck and consume over 70% of the running time. In contrast, TPNet caches historical interactions into node representations and constructs pairwise features based on these representations, thereby enhancing efficiency.

Scalability Analysis. To further verify the scalability of TPNet, we generated a series of random graphs with an average degree fixed at 100 and the number of edges varying from 1e5 to 1e8. Figure 4 shows the change of running time and GPU memory of TPNet, where the growth of the running time and GPU memory is close to and less than the linear growth curve respectively, showing the good scalability of TPNet. In contrast, PINT explicitly stores the temporal walk matrices and encounters out-of-memory error when the edge number reaches 1e7 (shown inAppendix G.3), verifying the impracticability of explicitly storing temporal walk matrices.

4.3 Ablation Study

Proposed Components. To verify the effectiveness of the proposed components in TPNet, we compare TPNet with the following variants: 1) w/o NR that remove the node representations and the corresponding features that decoded from them. 2) w/o Time that only considers the structural information by setting the time decay weight λ to be zero. 3) w/o Scale that remove the $\log(\cdot)$ and $\mathrm{ReLU}(\cdot)$ in the pairwise feature decoding. As shown in Table 2, there is a dramatic performance drop of w/o NR, which shows that the pairwise information carried by the node representations plays a vital role in the performance of TPNet. There is also an obvious performance drop of w/o Time, which confirms the necessity of incorporating temporal information in temporal walk matrix construction. Besides, the unreasonable poor performance of the w/o Scale is due to the various distribution of node representations across different layers, where, without scaling the raw pairwise features, the training will be unstable, and numerical overflow errors may even occur on some datasets. Further details on the distribution of node representations from different layers are provided in Appendix G.5.

Dimension Change. To verify the influence of the node representation dimension. We vary the dimensions from 1 to 128 and report the performance of TPNet (denoted as TPNet-d). As shown in Figure 3, the required dimension of node representations is small, where only 1-dimensional and 16-dimensional node representations can achieve satisfactory performance on MOOC and LastFM respectively. For different datasets, we observe that the average degree may be a main influence of the required dimension, where sparse graphs (like MOOC and Wikipedia) only need a small dimension, and dense graphs (like LastFM and Enron) may require a larger dimension. Empirically, setting the dimension to be $10 * \log(2E)$ is enough to achieve satisfactory performance on all datasets, where E is the number of edges. Results on more datasets can be found in Appendix G.4.

5 Related Work

Temporal link prediction [17] aims at predicting future interactions based on historical topology, which is crucial for a series of real-world applications [2, 3, 18]. Earlier methods considered the temporal graph as a series of graph snapshots that are sampled at regularly-spaced timestamps [19, 20], which will lose the fine-grained temporal information due to ignoring the temporal orders of

https://doi.org/10.52202/079017-4482

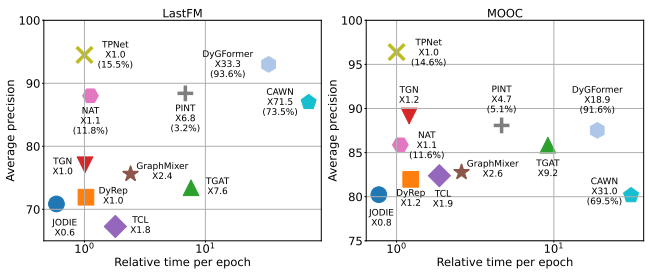


Figure 4: Relative running time of different methods. The proportion of construction relative encoding time to all running time is marked in brackets for link-wise methods.

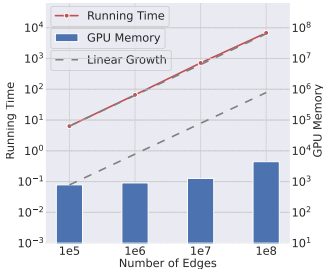


Figure 5: Scalability analysis of TPNet on synthetic datasets with different sizes.

interactions in a graph snapshot. Recently, some continuous-time temporal graph learning methods have been proposed [21, 3, 14, 13], which consider the temporal graph as a sequence of interactions with irregular time intervals to fully capture the graph dynamics. For example, TGN [21] maintained a dynamic memory vector for each node and generated node representations by aggregating memory vectors via temporal graph attention to capture the evolution pattern of the temporal graph. However, capturing pairwise information by merely aggregating representations of neighboring nodes [22] is challenging. To address this issue, the link-wise method was proposed, which constructs relative encodings as additional node features to inject the pairwise information into the representation learning process [6, 7, 9, 8]. For example, Souza et al. [8] proposed a relative encoding based on temporal walk counting and theoretically showed that constructing the relative encodings can improve the expressive power of models in distinguishing different links. Despite these advances, existing ways to construct the relative encodings are still far from satisfactory, where computation efficiency is a main concern and temporal information is seldom considered. In this paper, we unify existing relative encodings into a function of temporal walk matrices and explore encoding the pairwise information effectively and efficiently by temporal walk matrix projection.

6 Limitation

One limitation of our method is that the matrix construction approach requires manual predefined settings. Different networks may necessitate distinct construction methods, potentially leading to additional human effort in experimenting with various approaches. For instance, the proposed temporal walk matrix that incorporates the time decay effect may not be optimal for networks characterized by long-term dependencies. Developing an adaptive matrix construction technique will be an interesting direction for future research.

7 Conclusion

In this paper, we study the problem of pairwise information injection for temporal link prediction. We unify existing construction ways of relative encodings into a unified function, which reveals a connection between the relative encoding and temporal walk matrix. Then we propose a new temporal link prediction model, TPNet, to address the computational inefficiencies and the ignorance of temporal information in previous methods. TPNet introduces a new temporal walk matrix to simultaneously consider the temporal and structural information and a random feature propagation mechanism to maintain the temporal walk matrices efficiently. Theoretically, TPNet preserves the inner product of the maintained temporal walk matrices and empirically outperforms other link-wise methods in both effectiveness and efficiency. An interesting future direction may be designing an adaptive feature propagation mechanism.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62272023 and No. 62276015).

References

[1] Petter Holme and Jari Saramäki. Temporal networks. *Physics reports*, 519(3):97–125, 2012.

[2] Ziwei Fan, Zhiwei Liu, Jiawei Zhang, Yun Xiong, Lei Zheng, and Philip S. Yu. Continuous-time sequential recommendation with temporal graph collaborative transformer. In *International Conference on Information and Knowledge Management*, pages 433–442. ACM, 2021.

[3] Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In *International Conference on Knowledge Discovery & Data Mining*, pages 1269–1278. ACM, 2019.

[4] Fan Zhou, Xovee Xu, Goce Trajcevski, and Kunpeng Zhang. A survey of information cascade analysis: Models, predictions, and recent advances. *ACM Comput. Surv.*, 54(2):27:1–27:36, 2022.

[5] Xiaodong Lu, Shuo Ji, Le Yu, Leilei Sun, Bowen Du, and Tongyu Zhu. Continuous-time graph learning for cascade popularity prediction. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pages 2224–2232, 2023.

[6] Yanbang Wang, Yen-Yu Chang, Yunyu Liu, Jure Leskovec, and Pan Li. Inductive representation learning in temporal networks via causal anonymous walks. In *9th International Conference on Learning Representations*, 2021.

[7] Yuhong Luo and Pan Li. Neighborhood-aware scalable temporal network representation learning. In *Learning on Graphs Conference*, 2022.

[8] Amauri H. Souza, Diego Mesquita, Samuel Kaski, and Vikas K. Garg. Provably expressive temporal graph networks. In *Advances in Neural Information Processing Systems*, 2022.

[9] Le Yu, Leilei Sun, Bowen Du, and Weifeng Lv. Towards better dynamic graph learning: New architecture and unified library. In *Advances in Neural Information Processing Systems*, 2023.

[10] Giang Hoang Nguyen, John Boaz Lee, Ryan A. Rossi, Nesreen K. Ahmed, Eunyee Koh, and Sungchul Kim. Continuous-time dynamic network embeddings. In *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon , France, April 23-27, 2018*, 2018.

[11] D Sivakumar. Algorithmic derandomization via complexity theory. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 619–626, 2002.

[12] Santosh S Vempala. *The random projection method*, volume 65. American Mathematical Soc., 2005.

[13] Weilin Cong, Si Zhang, Jian Kang, Baichuan Yuan, Hao Wu, Xin Zhou, Hanghang Tong, and Mehrdad Mahdavi. Do we really need complicated model architectures for temporal networks? In *The Eleventh International Conference on Learning Representations*, 2023.

[14] Da Xu, Chuanwei Ruan, Evren Körpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs. In *8th International Conference on Learning Representations*. OpenReview.net, 2020.

[15] Ilya O. Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, and Alexey Dosovitskiy. Mlp-mixer: An all-mlp architecture for vision. In *Advances in Neural Information Processing Systems*, 2021.

[16] Farimah Poursafaei, Shenyang Huang, Kellin Pelrine, and Reihaneh Rabbany. Towards better evaluation for dynamic link prediction. In *Advances in Neural Information Processing Systems*, 2022.

[17] Meng Qin and Dit-Yan Yeung. Temporal link prediction: A unified framework, taxonomy, and review. *ACM Comput. Surv.*, 56(4):89:1–89:40, 2024. doi: 10.1145/3625820. URL `https://doi.org/10.1145/3625820`.

 https://doi.org/10.52202/079017-4482

[18] Le Yu, Zihang Liu, Leilei Sun, Bowen Du, Chuanren Liu, and Weifeng Lv. Continuous-time user preference modelling for temporal sets prediction. *IEEE Trans. Knowl. Data Eng.*, 36(4): 1475–1488, 2024.

[19] Aravind Sankar, Yanhong Wu, Liang Gou, Wei Zhang, and Hao Yang. Dysat: Deep neural representation learning on dynamic graphs via self-attention networks. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, pages 519–527. ACM, 2020.

[20] 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. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.

[21] Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael M. Bronstein. Temporal graph networks for deep learning on dynamic graphs. *CoRR*, abs/2006.10637, 2020.

[22] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[23] Roman Vershynin. High-dimensional probability. *University of California, Irvine*, 10:11, 2020.

[24] Moses Charikar. Similarity estimation techniques from rounding algorithms. In John H. Reif, editor, *Proceedings on 34th Annual ACM Symposium on Theory of Computing*, 2002.

[25] Ping Li, Trevor Hastie, and Kenneth Ward Church. Very sparse random projections. In *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 20-23, 2006*, 2006.

[26] John Wright and Yi Ma. *High-dimensional data analysis with low-dimensional models: Principles, computation, and applications*. Cambridge University Press, 2022.

[27] James W Pennebaker, Martha E Francis, and Roger J Booth. Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*, 71(2001):2001, 2001.

[28] Rakshit Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. Dyrep: Learning representations over dynamic graphs. In *International Conference on Learning Representations*, 2019.

[29] Peter J Diggle. Spatio-temporal point processes: methods and applications. *Monographs on Statistics and Applied Probability*, 107:1, 2006.

[30] Walter Rudin. *Fourier analysis on groups*. Courier Dover Publications, 2017.

[31] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

[32] 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. *CoRR*, abs/2105.07944, 2021.

[33] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? In *Annual Conference on Neural Information Processing Systems*, 2021.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Annual Conference on Neural Information Processing Systems*, 2017.

A Analysis of Existing Relative Encodings

A.1 NAT

In Algorithm 1 of NAT, it maintains a series of hash maps $s_u^{(0)}, s_u^{(1)}, .., s_u^{(k)}$ for each node u, which are sets of node ids. Next, we will first prove the statement holds if the size of the hash maps is infinite.

Statement. $s^{(i)}$ is a set of nodes that u can approach through a temporal walk whose length is less than $i+1$.

Initially, the $s_u^{(0)}$ is set as $\{u\}$ and $s_u^{(1)}, .., s_u^{(k)}$ are set as empty set. The statement holds. When a new interaction $(\{u, v\}, t)$ occurs and if the size of the hash maps is infinite, for $1 \leq i \leq k$, the updating function of hash maps can be represented as

$$\bar{s}_u^{(i)} = s_u^{(i)} \cup s_v^{(i-1)}, \qquad \bar{s}_v^{(i)} = s_v^{(i)} \cup s_u^{(i-1)}, \tag{7}$$

where $\bar{s}_u^{(i)}$ and $s_u^{(i)}$ denote the hash map after and before adding the interaction respectively. Then if the timestamp of $(\{u, v\}, t)$ is larger than that of previous interactions, the newly generated temporal walks beginning from u must first visit v through $(\{u, v\}, t)$ (proved in Section B.1), thus the newly added nodes that u can visit through a temporal walk with length less than $i+1$ must belong to $s_v^{(i-1)}$. So $\bar{s}_u^{(i)}$ will contain all the nodes that u can approach through a temporal walk with length less than $i+1$ after adding $(\{u, v\}, t)$. The statement holds.

For a node w, its similarity feature $\boldsymbol{r}^{w|u}$ is a $(k+1)$-dimensional vector, where for $1 \leq i \leq k+1$, $r_i^{w|u} = 1$ if $w \in s_u^{(i-1)}$ and $r_i^{w|u} = 0$ otherwise. Considering that the above statement holds, the similarity feature $\boldsymbol{r}^{w|u}$ is equivalent to a $(k+1)$-dimensional vector, where $r_i^{w|u} = 1$ if the shortest temporal walk from u to w is less than i; otherwise, $r_i^{w|u} = 0$.

Algorithm 2: Temporal Walk Extraction ($\mathcal{G}(t)$, α, k, u)

Initialize W to be $\{(u, t)\}$;
for i *from* 1 *to* k **do**
  $(w_{\mathrm{p}}, t_{\mathrm{p}}) \leftarrow$ the last (node, time) pair in W;
  Sample one $(\{w_p, w'\}, t') \in \mathcal{E}_{w_{\mathrm{p}}, t_{\mathrm{p}}}$ with prob. $\propto$ $\exp(-\alpha(t_{\mathrm{p}} - t'))$;
  $W_i \leftarrow W_i \oplus (w', t')$;
Return W;

A.2 CAWN

For constructing $\boldsymbol{r}^{w|u}$, CAWN first repeatedly samples m temporal walks of length k begging at u according to the sampling strategy in Algorithm 2. Then $\boldsymbol{r}^{w|u}$ is set to be a $(k+1)$-dimensional vector, where $r_i^{w|u}$ will be the number of walks whose i-th visited node is w for $1 \leq i \leq k+1$, which can be represented as,

$$r_i^{w|u} = \sum_{j=1}^{m} \mathbf{1}_{W_j[i][0]=w}, \tag{8}$$

where W_j is the j-th sampled walks and $\mathbf{1}_{W_j[i][0]=w}$ is 1 if $W_j[i][0] = w$; otherwise, $\mathbf{1}_{W_j[i][0]=w}$ is 0. Due to each temporal walk being sampled independently, $\mathbf{1}_{W_1[i][0]=w}, \cdots, \mathbf{1}_{W_m[i][0]=w}$ are m independent and identically distributed Bernoulli random variables $\mathrm{Ber}(\mu_w^{i-1})$, where μ_w^{i-1} is the probability of reaching w from u after performing a $(i-1)$-step temporal walk according to Algorithm 2. And according to the strong law of large numbers (Theorem 1.3.1 of [23]), the mean of these random variables (i.e., $\frac{1}{m}\sum_{j=1}^{m} \mathbf{1}_{W_j[i][0]=w} \iff \frac{1}{m} r_i^{w|u}$) will coverage to the mean as the number of sampled walks $m \to \infty$. The mean of the $\mathrm{Ber}(\mu_w^{i-1})$ is μ_w^{i-1}. Expand all $(i-1)$-step temporal walks from u to w, we have

$$\mu_w^{i-1} = \sum_{W \in \mathcal{M}_{u,w}^{i-1}} f(W), \tag{9}$$

https://doi.org/10.52202/079017-4482

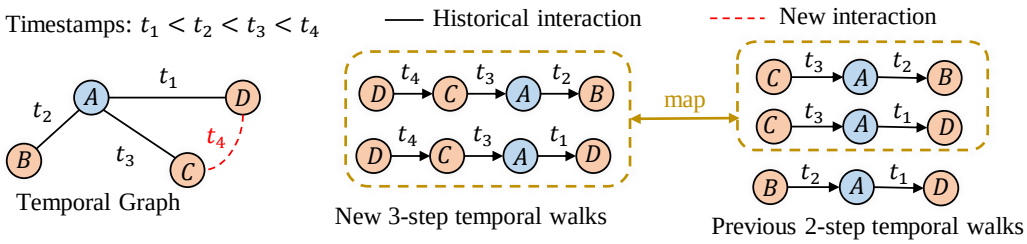


Figure 6: Illustration of the newly generated 3-step temporal walks beginning from D after adding a new interaction $(\{C, D\}, t_4)$ There is a one-to-one map between $\Delta M^3_{D,*}$ and $M^2_{C,*}(t_4)$.

where $f(\cdot)$ is the sampled probability of a walk according to Algorithm 2. For Algorithm 2, if we are current at (w_i, t_i), then the probability of moving to (w_{i+1}, t_{i+1}) through an interaction $(\{w_i, w_{i+1}\}, t_{i+1})$ is proportional to $\exp(-\alpha(t_i - t_{i+1}))$, which is $\frac{\exp(-\alpha(t_i - t_{i+1}))}{\sum_{(w',t')\in\mathcal{E}_{w_i,t_i}} \exp(-\alpha(t_i - t'))}$. Thus the probability of a temporal walk $W = [(w_0, t_0), (w_1, t_1), .., (w_k, t_k)]$ is

$$f(W) = \prod_{i=0}^{k-1} \frac{\exp(-\alpha(t_i - t_{i+1}))}{\sum_{(\{w',w\},t')\in\mathcal{E}_{w_i,t_i}} \exp(-\alpha(t_i - t'))} \tag{10}$$

which is the same as the score function $s'(\cdot)$ of CAWN we shown in Section 2.2.2. Thus, $r_i^{w|u}$ (multiplied with a const $\frac{1}{m}$) is the same as $\sum_{W\in\mathcal{M}^{i-1}_{u,w}} s'(W)$.

B Proofs

B.1 Proof of Theorem 1

The equation in Theorem 1 can be rewritten as $\boldsymbol{H}^{(l)}(t) = e^{\lambda l t} * \boldsymbol{A}^{(l)}(t)\boldsymbol{P}$. We can consider $e^{\lambda l t} * \boldsymbol{A}^{(l)}(t)$ as a new temporal walk matrix whose score function for a temporal walk $[(w_0, t_0), (w_1, t_1), ..., (w_l, t_l)]$ at time t is $e^{\lambda l t} * s(W)$. Expanding it, we have

$$e^{\lambda l t} * s(W) = e^{\lambda l t} * \prod_{j=1}^{l} e^{-\lambda(t-t_j)} = \prod_{j=1}^{l} e^{-\lambda(t-t_j)} * e^{\lambda t} = \prod_{j=1}^{l} e^{\lambda t_j}. \tag{11}$$

We use the notation $\bar{\boldsymbol{A}}^{(l)}(t)$ to denote $e^{\lambda l t} * \boldsymbol{A}^{(l)}(t)$ and $\bar{s}(W)$ to denote $e^{\lambda l t} * s(W)$ for a l-step temporal walk in the following proof. The original problem is transformed into proving that $\boldsymbol{H}^{(l)}(t) = \bar{\boldsymbol{A}}^{(l)}(t)\boldsymbol{P}$. Note that for a given temporal walk $[(w_0, t_0), (w_1, t_1), ..., (w_l, t_l)]$, each term of $\bar{s}(W)$ (i.e., $\prod_{j=1}^{l} e^{\lambda t_j}$) will no change as time t goes on. Thus the temporal walk matrices $\bar{\boldsymbol{A}}^{(l)}(t)$ will only change when a new interaction occurs. Next, we inspect how the $\bar{\boldsymbol{A}}^{(l)}(t)$ changes when a new interaction (u, v, t) occurs. For each element $\bar{A}^{(l)}_{i,j}(t)$, its change is caused by the newly generated l-step temporal walks from i to j after adding the interaction (u, v, t), which can be written as

$$A^{(l)}_{i,j}(t^+) = A^{(l)}_{i,j}(t) + \sum_{W\in\Delta M^l_{i,j}} \bar{s}(W), \tag{12}$$

where t^+ denotes the timestamps right after t and $\Delta M^l_{i,j}$ denote the newly generated l-step temporal walks from i to j. According to the definition of the temporal walk, the new l-step temporal walks must begin from u or v. Because if there is a temporal walk that does not begin from u or v, it can be represented as $[(w_0, t_0), .., (w_i, t_i), (u, t_{i+1}), (v, t), .., (w_l, t_l))]$ or $[(w_0, t_0), .., (w_i, t_i), (v, t_{i+1}), (u, t), .., (w_l, t_l))]$, which means that there is an interaction $(\{w_i, u\}, t_{i+1})$ or $(\{w_i, v\}, t_{i+1})$ whose timestamp t_i is larger than t. (Since the timestamps of a temporal walk are decreasing). This is impossible since (u, v, t) is a newly happened interaction. Thus only the u-th row and v-th row of the $\bar{\boldsymbol{A}}^{(l)}(t)$ will change. Besides for each newly generated temporal walk from u, it must be $[(u, t), (v, t_1), .., (w_l, t_l))]$, where $[(v, t_1), .., (w_l, t_l))]$ corresponds to a $(l-1)$-step temporal walk beginning from v. And for any $(l-1)$-step temporal walk beginning

from v, we can add a prefix (u,t) to make it become a l-step temporal walk beginning from u. Thus for any $1 \leq i \leq n$, there is a one-to-one mapping between $\Delta M_{u,i}^{l}$ and $M_{v,i}^{l-1}(t)$ with $M_{v,i}^{l-1}(t)$ denote the set of $(l-1)$-th temporal walks from v to i before t, and we can rewritten Equation (12) as

$$A_{u,i}^{(l)}(t^+) = A_{u,i}^{(l)}(t) + \sum_{W \in M_{v,i}^{l-1}(t)} e^{-\lambda t} * \bar{s}(W) = A_{u,i}^{(l)}(t) + e^{-\lambda t} * A_{v,i}^{(l-1)}(t), \tag{13}$$

The updating function for $\bar{A}_{v,i}^{(l)}(t)$ is also similar. We give a visual illustration of the new temporal walks in Figure 6 for better understanding. Finally, writing the update formula in vector form, for any $1 \leq l \leq k$ we have

$$\bar{\boldsymbol{A}}_{u}^{(l)}(t^+) = \bar{\boldsymbol{A}}_{u}^{(l)}(t) + e^{\lambda t} * \bar{\boldsymbol{A}}_{v}^{(l-1)}(t), \quad \bar{\boldsymbol{A}}_{v}^{(l)}(t^+) = \bar{\boldsymbol{A}}_{v}^{(l)}(t) + e^{\lambda t} * \bar{\boldsymbol{A}}_{u}^{(l-1)}(t), \tag{14}$$

which is the same as Equation (14)! Thus if $\boldsymbol{H}^{(l)}(t) = \bar{\boldsymbol{A}}^{(l)}(t)\boldsymbol{P}$ for $0 \leq l \leq k$, after adding a new interaction (u,v,t), for $0 \leq l \leq k$, $\boldsymbol{H}^{(l)}(t^+) = \bar{\boldsymbol{A}}^{(l)}(t^+)\boldsymbol{P}$ still holds. Because we have

$$\boldsymbol{H}_{u}^{(l)}(t^+) = \boldsymbol{H}_{u}^{(l)}(t) + e^{\lambda t} * \boldsymbol{H}_{v}^{(l-1)}(t) = (\bar{\boldsymbol{A}}_{u}^{(l)}(t) + e^{\lambda t} * \bar{\boldsymbol{A}}_{v}^{(l-1)}(t))\boldsymbol{P} = \bar{\boldsymbol{A}}_{u}^{(l)}(t^+)\boldsymbol{P} \tag{15}$$

Note that the equation in Equation (1) holds at initialization, thus the equation always holds and the theorem is proved.

B.2 Proof of Theorem 2

The Theorem 2 can be considered as a special case of the Johnson-Lindenstrauss Lemma [11], where the random projection [12, 24, 25] can preserve the norm and inner product. For the convenience of readers lacking relevant background, we follow [26] to provide the proof under the given conditions of this paper. We begin the proof by the following lemma.

Lemma 1 ((Lemma 3.18 of [26])). *Let $\boldsymbol{x} \in \mathbb{R}^d$ be a d-dimensional random vector whose entries are independent $\mathcal{N}(0, \frac{1}{d})$. Then for any $\epsilon \in [0,1]$,*

$$\mathbb{P}\left(\left|\|\boldsymbol{x}\|_2^2 - 1\right| > \epsilon\right) \leq 2\exp(\frac{-\epsilon^2 d}{8}) \tag{16}$$

The following part can be divided into 1) We first give two corollaries and their proofs. 2) Then we give the proof of Theorem 2 based on the corollaries.

B.2.1 Two Corolarries

Based on the above lemma, we can get the following corollaries.

Corollary 1. *Given any $\epsilon \in (0,1)$, $\boldsymbol{x} \in \mathbb{R}^m$. Let $\boldsymbol{P} \in \mathbb{R}^{d \times m}$ be a random matrix whose entries are independent $\mathcal{N}(0, \frac{1}{d})$, if $d \geq \frac{8}{\epsilon^2}\log(\frac{1}{\delta})$, we have*

$$\mathbb{P}\left((1-\epsilon)\|\boldsymbol{x}\|_2^2 \leq \|\boldsymbol{P}\boldsymbol{x}\|_2^2 \leq (1+\epsilon)\|\boldsymbol{x}\|_2^2\right) \geq 1 - 2\delta \tag{17}$$

Proof. For the above corollary, we have

$$\begin{aligned} &\mathbb{P}\left((1-\epsilon)\|\boldsymbol{x}\|_2^2 \leq \|\boldsymbol{P}\boldsymbol{x}\|_2^2 \leq (1+\epsilon)\|\boldsymbol{x}\|_2^2\right) \geq 1 - 2\delta \\ \iff &\mathbb{P}\left(\left|\frac{\|\boldsymbol{P}\boldsymbol{x}\|_2^2}{\|\boldsymbol{x}\|_2^2} - 1\right| \leq \epsilon\right) \geq 1 - 2\delta \\ \iff &\mathbb{P}\left(\left|\frac{\|\boldsymbol{P}\boldsymbol{x}\|_2^2}{\|\boldsymbol{x}\|_2^2} - 1\right| > \epsilon\right) \leq 2\delta \end{aligned} \tag{18}$$

Note that each entry of $\boldsymbol{P}$ is an independent $\mathcal{N}(0, \frac{1}{d})$, thus $\boldsymbol{P}\boldsymbol{x}$ is a random vector whose each entry $(\boldsymbol{P}\boldsymbol{x})_k = \sum_{i=1}^{m} P_{k,i} * x_i$ is an independent $\mathcal{N}(0, \frac{\|\boldsymbol{x}\|_2^2}{d})$ and each entry of $\frac{\boldsymbol{P}\boldsymbol{x}}{\|\boldsymbol{x}\|_2}$ is an independent $\mathcal{N}(0, \frac{1}{d})$. Substituting it to Lemma 1 and taking $d \geq \frac{8}{\epsilon^2}\log(\frac{1}{\delta})$, we have

$$\begin{aligned} \mathbb{P}\left(\left|\frac{\|\boldsymbol{P}\boldsymbol{x}\|_2^2}{\|\boldsymbol{x}\|_2^2} - 1\right| > \epsilon\right) &\leq 2\exp(\frac{-\epsilon^2 d}{8}) \\ &\leq 2\exp(\frac{-\epsilon^2}{8} * \frac{8}{\epsilon^2} * \log(\frac{1}{\delta})) \leq 2\delta \end{aligned} \tag{19}$$

Based on Corollary 1, we can get the following corollary.

 https://doi.org/10.52202/079017-4482

Corollary 2. *Given any n m-dimensional vectors $\boldsymbol{x}_1, ..., \boldsymbol{x}_n$, $\epsilon \in (0,1)$, let $\boldsymbol{P} \in \mathbb{R}^{d\times m}$ be a random matrix whose entries are independent $\mathcal{N}(0, \frac{1}{d})$, if $d \geq \frac{24}{\epsilon^2}\log(4^{1/3}n)$, for any $1 \leq i, j \leq n$, we have*

$$\mathbb{P}\left(|\langle \boldsymbol{P}\boldsymbol{x_i}, \boldsymbol{P}\boldsymbol{x_j}\rangle - \langle \boldsymbol{x}_i, \boldsymbol{x}_j\rangle| \leq \frac{\epsilon}{2}(\|\boldsymbol{x}_i\|_2^2 + \|\boldsymbol{x}_j\|_2^2)\right) \geq 1 - \frac{1}{n} \tag{20}$$

let $E_{i,j}^+$ and $E_{i,j}^-$ denote the event of $\{\big|\|\boldsymbol{P}(\boldsymbol{x_i} + \boldsymbol{x_j})\|_2^2 - \|\boldsymbol{x_i} + \boldsymbol{x_j}\|_2^2\big| \leq \epsilon\|\boldsymbol{x_i} + \boldsymbol{x_j}\|_2^2\}$ and $\{\big|\|\boldsymbol{P}(\boldsymbol{x_i} - \boldsymbol{x_j})\|_2^2 - \|\boldsymbol{x_i} - \boldsymbol{x_j}\|_2^2\big| \leq \epsilon\|\boldsymbol{x_i} - \boldsymbol{x_j}\|_2^2\}$ respectively. Since $\boldsymbol{x}_i + \boldsymbol{x_j}$ and $\boldsymbol{x}_i - \boldsymbol{x_j}$ can also be consider two m-dimensional vectors. Thus take it and $d \geq \frac{24}{\epsilon^2}\log(4^{1/3}n)$ into Corollary 1, we have $\mathbb{P}(E_{i,j}^+) \geq 1 - \frac{1}{2n^3}$ and $\mathbb{P}(E_{i,j}^-) \geq 1 - \frac{1}{2n^3}$. Then let $C_{i,j} = E_{i,j}^+ \cap E_{i,j}^-$ denote that $E_{i,j}^+$ and $E_{i,j}^-$ hold simultaneously, according to the union bound, we have

$$\mathbb{P}(C_{i,j}) = 1 - \mathbb{P}(\overline{E_{i,j}^+} \cup \overline{E_{i,j}^-}) \geq 1 - (\mathbb{P}(\overline{E_{i,j}^+}) + \mathbb{P}(\overline{E_{i,j}^-})) \geq 1 - \frac{1}{n^3}, \tag{21}$$

where $\overline{E_{i,j}^+}$ denote that $E_{i,j}^+$ does not hold and $\overline{E_{i,j}^+} \cup \overline{E_{i,j}^-}$ denotes that $\overline{E_{i,j}^+}$ or $\overline{E_{i,j}^-}$ holds. According to the union bound, we further have

$$\mathbb{P}(\cap_{i,j}C_{i,j}) = 1 - \mathbb{P}(\cup_{i,j}\overline{C_{i,j}}) \geq 1 - \sum_{i,j}\mathbb{P}(\overline{C_{i,j}}) \geq 1 - n^2 * \frac{1}{n^3} \geq 1 - \frac{1}{n}, \tag{22}$$

where $\cap_{i,j}C_{i,j}$ denote that $C_{i,j}$ holds for any i, j and $\cup_{i,j}\overline{C_{i,j}}$ denote that there exist i, j that $\overline{C_{i,j}}$ holds. If $C_{i,j}$ holds, we have

$$(1-\epsilon)\|\boldsymbol{x_i} + \boldsymbol{x_j}\|_2^2 \leq \|\boldsymbol{P}(\boldsymbol{x_i} + \boldsymbol{x_j})\|_2^2 \leq (1+\epsilon)\|\boldsymbol{x_i} + \boldsymbol{x_j}\|_2^2 \tag{23}$$

$$(1-\epsilon)\|\boldsymbol{x_i} - \boldsymbol{x_j}\|_2^2 \leq \|\boldsymbol{P}(\boldsymbol{x_i} - \boldsymbol{x_j})\|_2^2 \leq (1+\epsilon)\|\boldsymbol{x_i} - \boldsymbol{x_j}\|_2^2 \tag{24}$$

Multiplying Equation (24) with -1 and adding it to (23), we have

$$|\langle \boldsymbol{P}\boldsymbol{x_i}, \boldsymbol{P}\boldsymbol{x_j}\rangle - \langle \boldsymbol{x}_i, \boldsymbol{x_j}\rangle| \leq \frac{\epsilon}{2}(\|\boldsymbol{x_i}\|_2^2 + \|\boldsymbol{x_j}\|_2^2) \tag{25}$$

Thus we have

$$\mathbb{P}(\cap_{i,j}C_{i,j}) \geq 1 - \frac{1}{n} \implies \mathbb{P}(|\langle \boldsymbol{P}\boldsymbol{x_i}, \boldsymbol{P}\boldsymbol{x_j}\rangle - \langle \boldsymbol{x_i}, \boldsymbol{x_j}\rangle| \leq \frac{\epsilon}{2}(\|\boldsymbol{x_i}\|_2^2 + \|\boldsymbol{x_j}\|_2^2)) \geq 1 - \frac{1}{n} \tag{26}$$

holds for any $1 \leq i, j \leq n$.

B.2.2 Proof based on Corollaries

The matrix of $\boldsymbol{A}^{(l)}(t)$ in Theorem 2 can be considered as n vectors with n dimensions, where each row of $\boldsymbol{A}^{(l)}(t)$ is a n-dimensional vector. Similarly, we can consider $\boldsymbol{A}^{(0)}(t), \cdots, \boldsymbol{A}^{(k)}(t)$ together as $n(k+1)$ vectors with n dimensions. Considering that $\boldsymbol{P} \in \mathbb{R}^{n\times d_R}$ is a random matrix where each entry is an independent $\mathcal{N}(0, \frac{1}{d_R})$ and $e^{-\lambda l t} * \boldsymbol{H}^{(l)}(t)$ is the projection of $\boldsymbol{A}^{(l)}(t)$, substitute it into Corollary 2 and taking the number of vectors as $(k+1)n$, we can get Theorem 2.

C Batch Updating Mechanism

For the situation where multiple interactions happen simultaneously, we can first compute each interaction's contribution independently and sum them together to update the node representations. The maintaining mechanism is shown in Algorithm 3, where we packed the interactions that happen at the same time into a set $\mathcal{B}$ and sum the independent contribution of each interaction in $\mathcal{B}$ into $\Delta\boldsymbol{H}^{(1)}, \cdots, \Delta\boldsymbol{H}^{(k)}$. For simplicity, we initialize $\Delta\boldsymbol{H}^{(1)}, \cdots, \Delta\boldsymbol{H}^{(k)}$ each time and it can be replaced by some efficient implementation such as scatter_add operation in pytorch.

Algorithm 3: Batch Node Representation Maintaining ($\mathcal{G}$,λ,k,n,d_R)

Initialize $\boldsymbol{H}^{(0)}, \boldsymbol{H}^{(1)}, ..., \boldsymbol{H}^{(k)} \in \mathbb{R}^{n \times d_R}$ as zero matrix ;
Fill $\boldsymbol{H}^{(0)}$ with entries independently drawn from $\mathcal{N}(0, \frac{1}{d_R})$;
for $\mathcal{B} \in \mathcal{G}$ **do**
    Initialize $\Delta\boldsymbol{H}^{(1)}, ..., \Delta\boldsymbol{H}^{(k)} \in \mathbb{R}^{n \times d_R}$ as zero matrix ;
    for $(u, v, t) \in \mathcal{B}$ **do**
        for $l = k$ **to** 1 **do**
            $\Delta\boldsymbol{H}_u^{(l)} = \Delta\boldsymbol{H}_u^{(l)} + e^{\lambda t} * \boldsymbol{H}_v^{(l-1)}$;
            $\Delta\boldsymbol{H}_v^{(l)} = \Delta\boldsymbol{H}_v^{(l)} + e^{\lambda t} * \boldsymbol{H}_u^{(l-1)}$;
    for $l = k$ **to** 1 **do**
        $\boldsymbol{H}^{(l)} = \boldsymbol{H}^{(l)} + \Delta\boldsymbol{H}^{(l)}$;
Return $\boldsymbol{H}^{(0)}, \boldsymbol{H}^{(1)}, .., \boldsymbol{H}^{(k)}$;

D Propagation Mechanism for Other Matrices

For simplicity, here we only consider the situation where the timestamp of each interaction is different and we can adopt a similar batch updating mechanism in Section C to handle the situation where multiple interactions happen simultaneously. For the updating of other types of temporal walk matrices, let $\boldsymbol{A}(t) = [\boldsymbol{A}^{(0)}(t), \cdots, \boldsymbol{A}^{(k)}(t)] \in \mathbb{R}^{n(k+1) \times d}$ denotes the concatenation of the temporal walk matrices. If the following two situations can be satisfied simultaneously, we can apply the random feature propagation mechanism to implicitly maintain the temporal walk matrices.

Condition 1. After adding a new interaction (u, v, t), the change of each row can be written as the linear combination of other rows, which is for each $1 \leq i \leq n(k+1)$, there exists $k_1, ..., k_m$ and $l_1, ..., l_m$ satisfying.

$$\boldsymbol{A}_i(t^+) = k_1\boldsymbol{A}_{l_1}(t) + \cdots + k_m\boldsymbol{A}_{l_m}(t), \tag{27}$$

where t^+ is the time right after t.

Condition 2. After time moving Δt without adding new interactions, the change of each row can be written as the linear combination of other rows, which is for each $1 \leq i \leq n(k+1)$, there exists $k_1, ..., k_m$ and $l_1, ..., l_m$ satisfying.

$$\boldsymbol{A}_i(t + \Delta t) = k_1\boldsymbol{A}_{l_1}(t) + \cdots + k_m\boldsymbol{A}_{l_m}(t) \tag{28}$$

The motivation behind the conditions is that the projection is a linear operation. If the node representations are the projection of the temporal walk matrix at t (i.e.,$\boldsymbol{H}(t) = \boldsymbol{A}(t)\boldsymbol{P}$) and the updating function of the temporal walk matrix is the linear combination of other rows, then we can apply the same updating function on $\boldsymbol{H}(t)$, which will make $\boldsymbol{H}(t^+)$ still be the projection of the temporal walk matrix. For example, applying (27) to $\boldsymbol{H}(t)$ will get

$$\begin{aligned} \boldsymbol{H}_i(t^+) &= k_1\boldsymbol{H}_{l_1}(t) + \cdots + k_m\boldsymbol{H}_{l_m}(t) \\ &= (k_1\boldsymbol{A}_{l_1}(t) + \cdots + k_m\boldsymbol{A}_{l_m}(t))\boldsymbol{P} = \boldsymbol{A}_i(t^+)\boldsymbol{P} \end{aligned} \tag{29}$$

Then we can ensure that $\boldsymbol{H}(t)$ is always the random projection of $\boldsymbol{A}(t)$ and thus preserve the inner product of the $\boldsymbol{A}(t)$.

D.1 Detailed Updating Mechanism

Based on the above analysis, we give the detailed propagation mechanism of methods in Section 2. For NAT, PINT and DyGFormer, their temporal matrix element $A_{u,v}^{(l)}(t)$ is the number of the l-step temporal walks from u to v and their feature propagation mechanism is shown in Algorithm 4, where the obtained node representations are the projection of the corresponding temporal walk matrix.

For CAWN, its score function for a temporal walk $W = [(w_0, t_0), (w_1, t_1), ..., (w_l, t_l)]$ is defined as $s(W) = \prod_{i=0}^{l-1} \frac{\exp(-\alpha(t_i - t_{i+1}))}{\sum_{(\{w', w\}, t') \in \mathcal{E}_{w_i, t_i}} \exp(-\alpha(t_i - t'))}$. As time goes on, its element of temporal walk

https://doi.org/10.52202/079017-4482

Algorithm 4: Propagation Mechanism for DyGFormer ($\mathcal{G}$,k,n,d_R)

Initialize $\boldsymbol{H}^{(0)}, \boldsymbol{H}^{(1)}, ..., \boldsymbol{H}^{(k)} \in \mathbb{R}^{n \times d_R}$ as zero matrix ;
Fill $\boldsymbol{H}^{(0)}$ with entries independently drawn from $\mathcal{N}(0, \frac{1}{d_R})$;
for $(u, v, t) \in \mathcal{G}$ **do**
    for $l = k$ **to** 1 **do**
        $\boldsymbol{H}_u^{(l)} = \boldsymbol{H}_u^{(l)} + \boldsymbol{H}_v^{(l-1)}$;
        $\boldsymbol{H}_v^{(l)} = \boldsymbol{H}_v^{(l)} + \boldsymbol{H}_u^{(l-1)}$;
Return $\boldsymbol{H}^{(0)}, \boldsymbol{H}^{(1)}, .., \boldsymbol{H}^{(k)}$;

Algorithm 5: Propagation Mechanism for CAWN ($\mathcal{G}$,α,k,n,d_R)

Initialize $\boldsymbol{H}^{(0)}, \boldsymbol{H}^{(1)}, ..., \boldsymbol{H}^{(k)} \in \mathbb{R}^{n \times d_R}$ as zero matrix ;
Fill $\boldsymbol{H}^{(0)}$ with entries independently drawn from $\mathcal{N}(0, \frac{1}{d_R})$;
Initialize $\boldsymbol{d} \in \mathbb{R}^n$ as zero vector ;
Set $t_{\text{prev}} \in \mathbb{R}$ to be zero ;
for $(u, v, t) \in \mathcal{G}$ **do**
    $\boldsymbol{d} = \boldsymbol{d} * \exp(-\alpha(t - t_{\text{prev}}))$;
    for $l = k$ **to** 1 **do**
        $\boldsymbol{H}_u^{(l)} = \frac{d_u}{d_u+1} * \boldsymbol{H}_u^{(l)} + \frac{1}{d_u+1} * \boldsymbol{H}_v^{(l-1)}$;
        $\boldsymbol{H}_v^{(l)} = \frac{d_v}{d_v+1} * \boldsymbol{H}_v^{(l)} + \frac{1}{d_v+1} * \boldsymbol{H}_u^{(l-1)}$;
    $d_u = d_u + 1$;
    $d_v = d_v + 1$;
    $t_{\text{prev}} = t$;
Return $\boldsymbol{H}^{(0)}, \boldsymbol{H}^{(1)}, .., \boldsymbol{H}^{(k)}$;

matrices will not change. When a new interaction (u, v, t) happens, for a temporal walk matrix $\boldsymbol{A}^{(l)}(t)$, its u-th row and v-th row will change. Formally, let $\boldsymbol{d}(t) \in \mathbb{R}^n$ denotes a time decay degree vector, where for each node u, $d_u(t)$ is defined as $d_u(t) = \sum_{(\{u,v'\},t') \in \mathcal{E}_{u,t}} \exp(-\alpha(t' - t))$ with $\mathcal{E}_{u,t}$ denoting the set of interactions attached to u before t, then the updating function of the temporal walk matrix $\boldsymbol{A}^{(k)}(t)$ can be represented as

$$\begin{aligned} \boldsymbol{A}_u^{(l)}(t^+) &= \frac{d_u(t)}{d_u(t)+1} * \boldsymbol{A}_u^{(l)}(t) + \frac{1}{d_u(t)+1} * \boldsymbol{A}_v^{(l-1)}(t), \\ \boldsymbol{A}_v^{(l)}(t^+) &= \frac{d_v(t)}{d_v(t)+1} * \boldsymbol{A}_v^{(l)}(t) + \frac{1}{d_v(t)+1} * \boldsymbol{A}_u^{(l-1)}(t), \end{aligned} \tag{30}$$

where $\frac{d_u(t)}{d_u(t)+1} * \boldsymbol{A}_u^{(l)}(t)$ corresponds to the change in score of the old l-step temporal walks begging from u and $\frac{1}{d_u(t)+1} * \boldsymbol{A}_v^{(l-1)}(t)$ correspond to change caused by the newly generated l-step temporal walk from u (the same for v and similar analysis about the updating of temporal walk matrix can be found in Appendix B.1). Then we can give the propagation mechanism for CAWN in Algorithm 5 based on the above analysis, where $\boldsymbol{d}$ corresponds to the time decay degree vector.

E Broader Impact

We proposed an effective and efficient temporal link prediction method, which may advance real-world scenarios that rely on link prediction as a cornerstone, such as recommendation systems. For potential negative impacts, overly accurate link prediction in some contexts may lead to imbalanced outcomes such as reduced diversity in recommendation systems.

F Experimental Settings

F.1 Datasets

Experiments are conducted on the following 13 benchmark datasets [3] collected by [16].

- **Wikipedia** is a bipartite interaction graph that records the edits on Wikipedia pages over a month. Nodes represent users and pages, and links denote the editing behaviors with timestamps. Each link is associated with a 172-dimensional Linguistic Inquiry and Word Count (LIWC) feature [27].
- **Reddit** is a bipartite graph capturing user posts under subreddits over a month. Nodes represent users and subreddits, while links denote timestamped posts. Each link carries a 172-dimensional LIWC feature.
- **MOOC** is a bipartite interaction network of online courses, where nodes represent students and course content units (e.g., videos, problem sets). Links indicate students' access to specific content units and have a 4-dimensional feature.
- **LastFM** is a bipartite network detailing song-listening behaviors of users over one month. Nodes are users and songs, and links represent listening activities.
- **Enron** records email communications between employees of the Enron Energy Corporation over three years.
- **Social Evo.** is a mobile phone proximity network tracking daily activities within an undergraduate dormitory for eight months, with each link having a 2-dimensional feature.
- **UCI** is an online communication network with nodes representing university students and links representing posted messages.
- **Flights** is a dynamic flight network showing air traffic development during the COVID-19 pandemic. Nodes represent airports, and links denote tracked flights. Each link has a weight indicating the number of flights between two airports per day.
- **Can. Parl.** is a dynamic political network recording interactions between Canadian Members of Parliament (MPs) from 2006 to 2019. Nodes represent MPs from electoral districts, and links are formed when two MPs vote "yes" on a bill. The weight of each link represents the number of times one MP voted "yes" in support of another MP in a year.
- **US Legis.** is a senate co-sponsorship network tracking social interactions between US Senators. The weight of each link indicates the number of times two senators co-sponsored a bill in a given congress.
- **UN Trade** captures food and agriculture trade between 181 nations over more than 30 years. The weight of each link represents the total normalized agriculture import or export values between two countries.
- **UN Vote** records roll-call votes in the United Nations General Assembly. A link between two nations increases in weight each time both vote "yes" on an item.
- **Contact** tracks the evolution of physical proximity among approximately 700 university students over a month. Each student has a unique identifier, and links indicate close proximity, with weights revealing the extent of physical closeness between students.

The statistics of the datasets are shown in Table 3, where #N&L Feat stands for the dimensions of node and link features.

F.2 Baselines

We select the following eleven popular baselines:

- **JODIE** [3] designs a recurrent architecture to maintain a memory vector for each node and a projection layer to map the node memories into future representation trajectories.
- **DyRep** [28] considers each link as a temporal point process [29] and designs a deep temporal point process model to capture the dynamics of the observed process.

[3] `https://zenodo.org/record/7213796#.Y1c06y8r30o`

https://doi.org/10.52202/079017-4482

Table 3: Statistics of the datasets.

Datasets	Domains	#Nodes	#Links	#N&L Feat	Bipartite	Duration	Unique Steps	Time Granularity
Wikipedia	Social	9,227	157,474	– & 172	True	1 month	152,757	Unix timestamps
Reddit	Social	10,984	672,447	– & 172	True	1 month	669,065	Unix timestamps
MOOC	Interaction	7,144	411,749	– & 4	True	17 months	345,600	Unix timestamps
LastFM	Interaction	1,980	1,293,103	– & –	True	1 month	1,283,614	Unix timestamps
Enron	Social	184	125,235	– & –	False	3 years	22,632	Unix timestamps
Social Evo.	Proximity	74	2,099,519	– & 2	False	8 months	565,932	Unix timestamps
UCI	Social	1,899	59,835	– & –	False	196 days	58,911	Unix timestamps
Flights	Transport	13,169	1,927,145	– & 1	False	4 months	122	days
Can. Parl.	Politics	734	74,478	– & 1	False	14 years	14	years
US Legis.	Politics	225	60,396	– & 1	False	12 congresses	12	congresses
UN Trade	Economics	255	507,497	– & 1	False	32 years	32	years
UN Vote	Politics	201	1,035,742	– & 1	False	72 years	72	years
Contact	Proximity	692	2,426,279	– & 1	False	1 month	8,064	5 minutes

- **TGAT** [14] proposes a time encoding function based on Bochner's Theorem [30] and combines it with the graph attention mechanism [31] to learn dynamic node representations.
- **TGN** [21] proposes a general framework for temporal graph learning, which includes a memory module to maintain node memories and an embedding module to aggregate node memories.
- **CAWN** [6] captures the evolution pattern of the temporal graph by causal anonymous walks. For a given link, CAWN first sampled a set of temporal walks beginning from the two end nodes respectively and constructs relative encodings. The sampled walks together with relative encodings are then mapped into node representations via a sequential model.
- **EdgeBank** [16] is a statistical method, that gives the likelihood of a link based on historical interactions between the two end nodes.
- **TCL** [32] propose a graph transformer architecture [33] to learning node representations, which samples neighbor nodes based on BFS and maps them into the node representation.
- **NAT** [7] propose a dictionary-type neighborhood representation to efficiently capture the correlation information between nodes, which maintains a series of N-caches to store the neighborhood information and use them to decode the pairwise information between nodes.
- **PINT** [8] proposes an injective temporal message passing mechanism to learn node representations and relative positional features constructed based on temporal walk counting to inject pairwise information between nodes.
- **GraphMixer** [13] proposes a simplified temporal graph learning architecture, which employs MLP-Mixer to learn the representation of a node from its historical interaction sequences.
- **DyGFormer** [9] proposes a transformer architecture [34] to learning node representations, which include a patching technique to capture the long-term histories of a node to and a neighbor co-occurrence encoding scheme to capture the correlation between nodes.

G Additional Experimental Results

G.1 Performance Comparison

The results under other settings are shown in Table 6, Table 7, Table 8, Table 9 and Table 10.

G.2 Efficicency Analysis

Efficiency analysis results on Reddit, UCI, and Wikipedia are shown in Figure 7.

G.3 Scalability Analysis

The scalability analysis result of PINT is shown in Table 4, where OOM indicates the out-of-memory error.

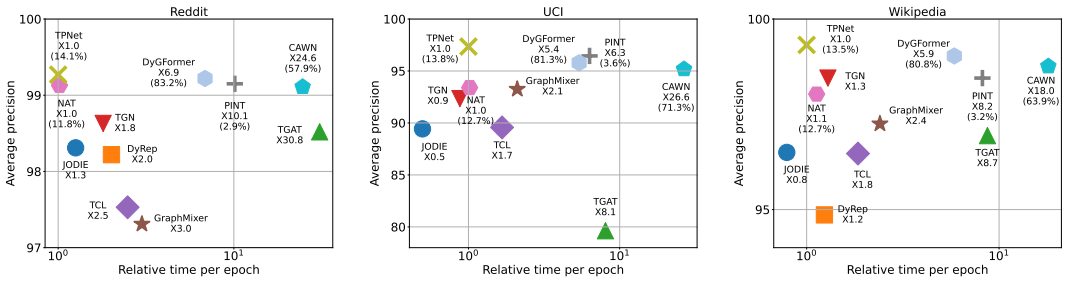


Figure 7: Efficiency analysis on more datasets.

Table 4: Scalability analysis of PINT.

Edge Number	PINT	
	Time	Memory
100000	51.18	92.32
1000000	594.77	2175.80
10000000	N/A	OOM
100000000	N/A	OOM

Table 5: Average norm of node representations from different layers.

Dataset	Layer1	Layer2	Layer3
Wikipedia	1.16E+01	1.23E+03	2.06E+05
Reddit	4.28E+01	2.52E+05	2.54E+09
MOOC	5.70E+00	2.50E+03	1.52E+06
LastFM	9.64E+01	1.81E+04	4.78E+06
Enron	2.73E-01	1.50E+00	4.60E-02
Social Evo.	2.81E+03	7.30E+06	1.56E+10
UCI	1.88E+01	8.67E+02	6.27E+04
Flights	3.38E+01	8.12E+03	3.29E+06
Can. Parl.	8.36E+00	7.49E+02	1.40E+04
US Legis.	4.39E+01	2.75E+03	1.22E+05
UN Trade	1.67E+02	1.58E+04	1.04E+06
UN Vote	2.41E+02	3.36E+04	3.22E+06
Contact	9.88E+00	8.83E+02	9.36E+04

G.4 Influence of Node Representation Dimension

Results on Wikipedia, Enron, and UCI are shown in Figure 8.

G.5 Statistic Analysis of Node Representations

Table 5 shows the mean of node representation norm at different layers, where aEb indicates $a \times 10^b$.

https://doi.org/10.52202/079017-4482

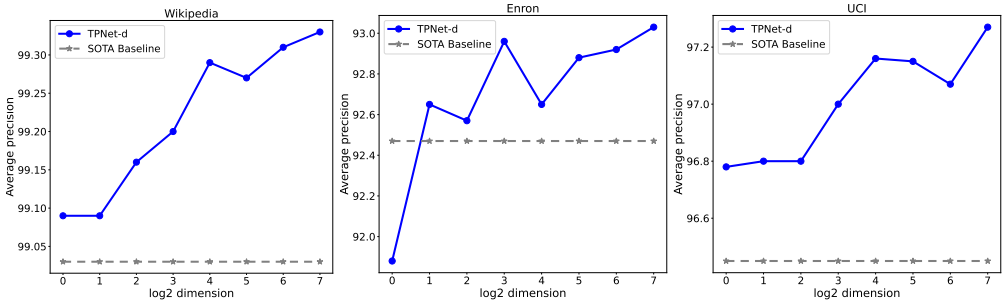


Figure 8: Influence of Node Representation Dimension

Table 6: Inductive results for random negative sampling.

Metrics	Datasets	JODIE	DyRep	TGAT	TGN	CAWN	TCL	GraphMixer	NAT	PINT	DyGFormer	RPNet
AP	Wikipedia	94.82±0.20	92.43±0.37	96.22±0.07	97.83±0.04	98.24±0.03	96.22±0.17	96.65±0.02	96.30±0.08	98.03±0.04	98.59±0.03	**98.91±0.01**
	Reddit	96.50±0.13	96.09±0.11	97.09±0.04	97.50±0.07	98.62±0.01	94.09±0.07	95.26±0.02	98.24±0.04	98.56±0.05	98.84±0.02	**98.86±0.01**
	MOOC	79.63±1.92	81.07±0.44	85.50±0.19	89.04±1.17	81.42±0.24	80.60±0.22	81.41±0.21	83.62±1.19	87.90±0.98	86.96±0.43	**95.07±0.26**
	LastFM	81.61±3.82	83.02±1.48	78.63±0.31	81.45±4.29	89.42±0.07	73.53±1.66	82.11±0.42	92.24±0.93	92.42±0.64	94.23±0.09	**95.36±0.11**
	Enron	80.72±1.39	74.55±3.95	67.05±1.51	77.94±1.02	86.35±0.51	76.14±0.79	75.88±0.48	87.18±1.24	88.12±0.30	89.76±0.34	**90.34±0.28**
	Social Evo.	91.96±0.48	90.04±0.47	91.41±0.16	90.77±0.86	79.94±0.18	91.55±0.09	91.86±0.06	87.44±5.48	92.40±0.60	93.14±0.04	**93.24±0.07**
	UCI	79.86±1.48	57.48±1.87	79.54±0.48	88.12±2.05	92.73±0.06	87.36±2.03	91.19±0.42	87.31±0.28	94.72±0.15	94.54±0.12	**95.74±0.05**
	Flights	94.74±0.37	92.88±0.73	88.73±0.33	95.03±0.60	97.06±0.02	83.41±0.07	83.03±0.05	96.74±0.22	97.54±0.06	97.79±0.02	**97.97±0.04**
	Can. Parl.	53.92±0.94	54.02±0.76	55.18±0.79	54.10±0.93	55.80±0.69	54.30±0.66	55.91±0.82	61.90±2.52	50.32±0.86	**87.74±0.71**	68.09±1.55
	US Legis.	54.93±2.29	57.28±0.71	51.00±3.11	58.63±0.37	53.17±1.20	52.59±0.97	50.71±0.76	60.41±0.74	59.71±1.36	54.28±2.87	**61.71±0.84**
	UN Trade	59.65±0.77	57.02±0.69	61.03±0.18	58.31±3.15	65.24±0.21	62.21±0.12	62.17±0.31	69.57±1.45	60.37±0.78	64.55±0.62	**86.53±0.29**
	UN Vote	56.64±0.96	54.62±2.22	52.24±1.46	58.85±2.51	49.94±0.45	51.60±0.97	50.68±0.44	**66.60±0.98**	57.43±1.24	55.93±0.39	58.00±3.21
	Contact	94.34±1.45	92.18±0.41	95.87±0.11	93.82±0.99	89.55±0.30	91.11±0.12	90.59±0.05	96.12±0.08	97.41±0.14	98.03±0.02	**98.39±0.02**
	Avg. Rank	7.62	8.69	7.92	6.15	6.15	8.46	7.85	4.69	4.15	3.00	**1.23**
AUC	Wikipedia	94.33±0.27	91.49±0.45	95.90±0.09	97.72±0.03	98.03±0.04	95.57±0.20	96.30±0.04	95.82±0.18	97.76±0.05	98.48±0.03	**98.90±0.01**
	Reddit	96.52±0.13	96.05±0.12	96.98±0.04	97.39±0.07	98.42±0.02	93.80±0.07	94.97±0.05	98.00±0.04	98.38±0.07	98.71±0.01	**98.73±0.02**
	MOOC	83.16±1.30	84.03±0.49	86.84±0.17	91.24±0.99	81.86±0.25	81.43±0.19	82.77±0.24	84.72±1.31	90.27±0.96	87.62±0.51	**95.55±0.25**
	LastFM	81.13±3.39	82.24±1.51	76.99±0.29	82.61±3.15	87.82±0.12	70.84±0.85	80.37±0.18	91.60±1.31	92.15±0.68	94.08±0.08	**95.36±0.06**
	Enron	81.96±1.34	76.34±4.20	64.63±1.74	78.83±1.11	87.02±0.50	72.33±0.99	76.51±0.71	87.95±0.58	87.97±0.61	**90.69±0.26**	90.21±0.49
	Social Evo.	93.70±0.29	91.18±0.49	93.41±0.19	93.43±0.59	84.73±0.27	93.71±0.18	94.09±0.07	88.15±6.36	94.78±0.37	95.29±0.03	**95.47±0.04**
	UCI	78.80±0.94	58.08±1.81	77.64±0.38	86.68±2.29	90.40±0.11	84.49±1.82	89.30±0.57	83.78±0.37	93.18±0.17	92.63±0.13	**94.40±0.03**
	Flights	95.21±0.32	93.56±0.70	88.64±0.35	95.92±0.43	96.86±0.02	82.48±0.01	82.27±0.06	96.97±0.20	97.69±0.05	97.80±0.02	**98.05±0.04**
	Can. Parl.	53.81±1.14	55.27±0.49	56.51±0.75	55.86±0.75	58.83±1.13	55.83±1.07	58.32±1.08	62.70±2.91	49.64±0.69	**89.33±0.48**	69.21±1.31
	US Legis.	58.12±2.35	61.07±0.56	48.27±3.50	62.38±0.48	51.49±1.13	50.43±1.48	47.20±0.89	64.22±0.65	61.89±1.52	53.21±3.04	**65.29±0.61**
	UN Trade	62.28±0.50	58.82±0.98	62.72±0.12	59.99±3.50	67.05±0.21	63.76±0.07	63.48±0.37	69.15±2.33	64.05±0.72	67.25±1.05	**86.88±0.23**
	UN Vote	58.13±1.43	55.13±3.46	51.83±1.35	61.23±2.71	48.34±0.76	50.51±1.05	50.04±0.86	**68.55±0.90**	59.01±1.88	56.73±0.69	54.82±4.04
	Contact	95.37±0.92	91.89±0.38	96.53±0.10	94.84±0.75	89.07±0.34	93.05±0.09	92.83±0.05	95.70±0.06	97.66±0.12	98.30±0.02	**98.51±0.01**
	Avg. Rank	7.46	8.62	7.92	5.69	6.46	8.77	8.00	4.77	3.92	2.77	**1.62**

Table 7: Transductive results for historical negative sampling.

Metric	Dataset	JODIE	DyRep	TGAT	TGN	CAWN	EdgeBank	TCL	GraphMixer	NAT	PINT	DyGFormer	TPNet
AP	Wikipedia	83.01±0.66	79.93±0.56	87.38±0.22	86.86±0.33	71.21±1.67	73.35±0.00	89.05±0.39	90.90±0.10	68.36±4.47	**91.33±0.76**	82.23±2.54	81.55±4.10
	Reddit	80.03±0.36	79.83±0.31	79.55±0.20	81.22±0.61	80.82±0.45	73.59±0.00	77.14±0.16	78.44±0.18	80.58±0.91	**83.93±1.11**	81.57±0.67	81.02±1.31
	MOOC	78.94±1.25	75.60±1.12	82.19±0.62	87.06±1.93	74.05±0.95	60.71±0.00	77.06±0.41	77.77±0.92	85.42±3.26	84.80±1.57	85.85±0.66	**92.69±0.95**
	LastFM	74.35±3.81	74.92±2.46	71.59±0.24	76.87±4.64	69.86±0.43	73.03±0.00	59.30±2.31	72.47±0.49	78.75±1.63	84.95±0.43	81.57±0.48	**87.74±0.50**
	Enron	69.85±2.70	71.19±2.76	64.07±1.05	73.91±1.76	64.73±0.36	76.53±0.00	70.66±0.39	77.98±0.92	72.90±1.38	**85.41±0.84**	75.63±0.73	80.79±1.68
	Social Evo.	87.44±6.78	93.29±0.43	95.01±0.44	94.45±0.56	85.53±0.38	80.57±0.00	94.74±0.31	94.93±0.31	91.29±4.89	96.73±0.29	**97.38±0.14**	96.80±0.41
	UCI	75.24±5.80	55.10±3.14	68.27±1.37	80.43±2.12	65.30±0.43	65.50±0.00	80.25±2.74	84.11±1.35	75.38±1.28	**93.96±0.20**	82.17±0.82	86.34±0.80
	Flights	66.48±2.59	67.61±0.99	**72.38±0.18**	66.70±1.64	64.72±0.97	70.53±0.00	70.68±0.24	71.47±0.26	64.74±1.83	66.82±1.44	66.59±0.49	69.10±1.27
	Can. Parl.	51.79±0.63	63.31±1.23	67.13±0.84	68.42±3.07	66.53±2.77	63.84±0.00	65.93±3.00	74.34±0.87	77.72±1.78	63.84±5.36	**97.00±0.31**	86.61±3.57
	US Legis.	51.71±5.76	86.88±2.25	62.14±6.60	74.00±7.57	68.82±8.23	63.22±0.00	80.53±3.95	81.65±1.02	91.12±1.97	73.58±4.16	85.30±3.88	**94.55±0.62**
	UN Trade	61.39±1.83	59.19±1.07	55.74±0.91	58.44±5.51	55.71±0.38	81.32±0.00	55.90±1.17	57.05±1.22	78.65±1.16	70.07±2.28	64.41±1.40	**85.22±1.22**
	UN Vote	70.02±0.81	69.30±1.12	52.96±2.14	69.37±3.93	51.26±0.04	**84.89±0.00**	52.30±2.35	51.20±1.60	71.39±2.68	71.79±2.53	60.84±1.58	74.68±1.38
	Contact	95.31±2.13	96.39±0.20	96.05±0.52	93.05±2.35	84.16±0.49	88.81±0.00	93.86±0.21	93.36±0.41	96.84±0.57	97.61±0.18	97.57±0.06	**98.02±0.15**
	Avg. Rank	8.23	7.69	7.54	5.92	10.31	8.08	7.77	6.23	5.85	3.62	4.23	**2.46**
AUC	Wikipedia	80.77±0.73	77.74±0.33	82.87±0.22	82.74±0.32	67.84±0.64	77.27±0.00	85.76±0.46	87.68±0.17	69.32±2.78	**89.25±0.49**	78.80±1.95	79.89±2.47
	Reddit	80.52±0.32	80.15±0.18	79.33±0.16	81.11±0.19	80.27±0.30	78.58±0.00	76.49±0.16	77.80±0.12	79.36±0.31	**82.98±0.63**	80.54±0.29	81.87±0.49
	MOOC	82.75±0.83	81.06±0.94	80.81±0.67	88.00±1.80	71.57±1.07	61.90±0.00	72.09±0.56	76.68±1.40	84.93±2.89	87.44±1.74	87.04±0.35	**93.45±0.67**
	LastFM	75.22±2.36	74.65±1.98	64.27±0.26	77.97±3.04	67.88±0.24	78.09±0.00	47.24±3.13	64.21±0.73	75.89±2.21	81.89±0.96	78.78±0.35	**84.64±0.45**
	Enron	75.39±2.37	74.69±3.55	61.85±1.43	77.09±2.22	65.10±0.34	79.59±0.00	67.95±0.88	75.27±1.14	73.22±2.18	**83.80±0.68**	76.55±0.52	81.16±1.28
	Social Evo.	90.06±3.15	93.12±0.34	93.08±0.59	94.71±0.53	87.43±0.15	85.81±0.00	93.44±0.68	94.39±0.31	91.87±4.52	96.82±0.24	**97.28±0.07**	97.22±0.30
	UCI	78.64±3.50	57.91±3.12	58.89±1.57	77.25±2.68	57.86±0.15	69.56±0.00	72.25±3.46	77.54±2.02	71.52±1.61	**92.05±0.36**	76.97±0.24	80.42±0.64
	Flights	68.97±1.87	69.43±0.90	72.20±0.16	68.39±0.95	66.11±0.71	**74.64±0.00**	70.57±0.18	70.37±0.23	67.11±2.16	69.52±0.90	68.09±0.43	71.82±0.82
	Can. Parl.	62.44±1.11	70.16±1.70	70.86±0.94	73.23±3.08	72.06±3.94	63.04±0.00	69.95±3.70	79.03±1.01	81.47±2.58	73.89±4.50	**97.61±0.40**	86.39±3.73
	US Legis.	67.47±6.40	91.44±1.18	73.47±5.25	83.53±4.53	78.62±7.46	67.41±0.00	83.97±3.71	85.17±0.70	94.63±1.16	83.45±2.85	90.77±1.96	**96.28±0.44**
	UN Trade	68.92±1.40	64.36±1.40	60.37±0.68	63.93±5.41	63.09±0.74	86.61±0.00	61.43±1.04	63.20±1.54	80.68±0.98	76.95±1.92	73.86±1.13	**88.90±1.00**
	UN Vote	76.84±1.01	74.72±1.43	53.95±3.15	73.40±5.20	51.27±0.33	**89.62±0.00**	52.29±2.39	52.61±1.44	76.47±2.15	76.61±3.05	64.27±1.78	78.43±1.09
	Contact	96.35±0.92	96.00±0.23	95.39±0.43	93.76±1.29	83.06±0.32	92.17±0.00	93.34±0.19	93.14±0.34	96.79±0.50	97.34±0.16	97.17±0.05	**97.73±0.11**
	Avg. Rank	6.77	7.31	8.38	5.62	10.31	7.54	8.54	7.08	6.46	3.15	4.77	**2.08**

Table 8: Inductive results for historical negative sampling.

Metrics	Datasets	JODIE	DyRep	TGAT	TGN	CAWN	TCL	GraphMixer	NAT	PINT	DyGFormer	RPNet
AP	Wikipedia	68.69±0.39	62.18±1.27	84.17±0.22	81.76±0.32	67.27±1.63	82.20±2.18	**87.60±0.30**	47.37±4.39	78.22±2.90	71.42±4.43	71.28±4.33
	Reddit	62.34±0.54	61.60±0.72	63.47±0.36	64.85±0.85	63.67±0.41	60.83±0.25	64.50±0.26	61.51±0.73	**67.56±0.83**	65.37±0.60	62.15±1.72
	MOOC	63.22±1.55	62.93±1.24	76.73±0.29	77.07±3.41	74.68±0.68	74.27±0.53	74.00±0.97	69.30±0.95	73.74±1.66	80.82±0.30	**81.85±1.60**
	LastFM	70.39±4.31	71.45±1.76	76.27±0.25	66.65±6.11	71.33±0.47	65.78±0.65	76.42±0.22	70.21±0.78	77.96±1.55	76.35±0.52	**82.27±1.22**
	Enron	65.86±3.71	62.08±2.27	61.40±1.31	62.91±1.16	60.70±0.36	67.11±0.62	72.37±1.37	62.69±1.02	**80.47±1.52**	67.07±0.62	74.60±1.35
	Social Evo.	88.51±0.87	88.72±1.10	93.97±0.54	90.66±1.62	79.83±0.38	94.10±0.31	94.01±0.47	84.89±4.62	95.75±1.06	**96.82±0.16**	96.38±0.18
	UCI	63.11±2.27	52.47±2.06	70.52±0.93	70.78±0.78	64.54±0.47	76.71±1.00	81.66±0.49	51.56±0.75	**85.49±0.22**	72.13±1.87	78.48±1.18
	Flights	61.01±1.65	62.83±1.31	64.72±0.36	59.31±1.43	56.82±0.57	64.50±0.25	**65.28±0.24**	51.63±1.10	53.46±0.61	57.11±0.21	54.67±0.76
	Can. Parl.	52.60±0.88	52.28±0.31	56.72±0.47	54.42±0.77	57.14±0.07	55.71±0.74	55.84±0.73	61.56±2.68	50.61±1.76	**87.40±0.85**	68.97±1.60
	US Legis.	52.94±2.11	62.10±1.41	51.83±3.95	61.18±1.10	55.56±1.71	53.87±1.41	52.03±1.02	64.61±3.02	59.37±1.84	56.31±3.46	**66.95±1.81**
	UN Trade	55.46±1.19	55.49±0.84	55.28±0.71	52.80±3.19	55.00±0.38	55.76±1.03	54.94±0.97	70.04±2.07	57.35±1.65	53.20±1.07	**78.83±0.53**
	UN Vote	61.04±1.30	60.22±1.78	53.05±3.10	63.74±3.00	47.98±0.84	54.19±2.17	48.09±0.43	64.91±1.58	**65.75±2.86**	52.63±1.26	65.24±1.62
	Contact	90.42±2.34	89.22±0.66	**94.15±0.45**	88.13±1.50	74.20±0.80	90.44±0.17	89.91±0.36	84.13±1.78	90.68±0.46	93.56±0.52	93.56±0.57
	Avg. Rank	7.46	7.62	5.69	6.31	8.08	5.92	5.23	7.62	4.23	4.62	**3.15**
AUC	Wikipedia	61.86±0.53	57.54±1.09	78.38±0.20	75.75±0.29	62.04±0.65	79.79±0.96	**82.87±0.21**	40.95±4.99	73.29±2.33	68.33±2.82	67.95±2.77
	Reddit	61.69±0.39	60.45±0.37	64.43±0.27	64.55±0.50	**64.94±0.21**	61.43±0.26	64.27±0.13	58.32±0.63	64.02±0.46	64.81±0.25	62.37±0.83
	MOOC	64.48±1.64	64.23±1.29	74.08±0.27	77.69±3.55	71.68±0.94	69.82±0.32	72.53±0.84	67.99±1.68	75.92±2.48	80.77±0.63	**84.46±0.88**
	LastFM	68.44±3.26	68.79±1.08	69.89±0.28	66.99±5.62	67.69±0.24	55.88±1.85	70.07±0.20	64.18±0.65	75.02±0.66	70.73±0.37	**77.10±0.78**
	Enron	65.32±3.57	61.50±2.50	57.84±2.18	62.68±1.09	62.25±0.40	64.06±1.02	68.20±1.62	61.69±0.68	**78.90±1.29**	65.78±0.42	74.50±1.02
	Social Evo.	88.53±0.55	87.93±1.05	91.87±0.72	92.10±1.22	83.54±0.24	93.28±0.60	93.62±0.35	84.89±4.72	96.29±0.56	**96.91±0.09**	96.76±0.14
	UCI	60.24±1.94	51.25±2.37	62.32±1.18	62.69±0.90	56.39±0.10	70.46±1.94	75.98±0.84	42.59±0.96	**81.34±0.29**	65.55±1.01	71.35±0.84
	Flights	60.72±1.29	61.99±1.39	63.38±0.26	59.66±1.04	56.58±0.44	**63.48±0.23**	63.30±0.19	48.39±1.37	49.76±0.89	56.05±0.21	53.08±0.87
	Can. Parl.	51.62±1.00	52.38±0.46	58.30±0.61	55.64±0.54	60.11±0.48	57.30±1.03	56.68±1.20	61.72±2.76	48.93±2.35	**88.68±0.74**	69.11±1.18
	US Legis.	58.12±2.94	67.94±0.98	49.99±4.88	64.87±1.65	54.41±1.31	52.12±2.13	49.28±0.86	66.95±4.17	62.82±2.18	56.57±3.22	**68.37±1.62**
	UN Trade	58.73±1.19	57.90±1.33	59.74±0.59	55.61±3.54	60.95±0.80	61.12±0.97	59.88±1.17	70.43±2.07	62.61±1.70	58.46±1.65	**80.70±1.03**
	UN Vote	65.16±1.28	63.98±2.12	51.73±4.12	68.59±3.11	48.01±1.77	54.66±2.11	45.49±0.42	67.69±1.92	**69.47±3.01**	53.85±2.02	65.75±1.85
	Contact	90.80±1.18	88.88±0.68	93.76±0.41	88.84±1.39	74.79±0.37	90.37±0.16	90.04±0.29	84.74±1.44	91.99±0.28	**94.14±0.26**	93.47±0.43
	Avg. Rank	7.23	8.08	5.92	6.00	7.46	6.00	5.38	7.92	4.31	4.38	**3.31**

https://doi.org/10.52202/079017-4482

Table 9: Transductive results for inductive negative sampling.

Metric	Dataset	JODIE	DyRep	TGAT	TGN	CAWN	EdgeBank	TCL	GraphMixer	NAT	PINT	DyGFormer	TPNet
AP	Wikipedia	75.65±0.79	70.21±1.58	87.00±0.16	85.62±0.44	74.06±2.62	80.63±0.00	86.76±0.72	**88.59±0.17**	53.85±6.25	82.39±2.70	78.29±5.38	79.35±5.52
	Reddit	86.98±0.16	86.30±0.26	89.59±0.24	88.10±0.24	**91.67±0.24**	85.48±0.00	87.45±0.29	85.26±0.11	81.71±1.03	87.59±0.65	91.11±0.40	88.19±0.33
	MOOC	65.23±2.19	61.66±0.95	75.95±0.64	77.50±2.91	73.51±0.94	49.43±0.00	74.65±0.54	74.27±0.92	77.28±1.94	75.70±1.43	81.24±0.69	**88.18±0.97**
	LastFM	62.67±4.49	64.41±2.70	71.13±0.17	65.95±5.98	67.48±0.77	75.49±0.00	58.21±0.89	68.12±0.33	64.02±1.48	72.24±0.92	73.97±0.50	**77.99±1.30**
	Enron	68.96±0.98	67.79±1.53	63.94±1.36	70.89±2.72	75.15±0.58	73.89±0.00	71.29±0.32	75.01±0.79	70.34±1.73	**80.23±1.37**	77.41±0.89	75.36±1.81
	Social Evo.	89.82±4.11	93.28±0.48	94.84±0.44	95.13±0.56	88.32±0.27	83.69±0.00	94.90±0.36	94.72±0.33	92.12±4.53	97.15±0.25	**97.68±0.10**	97.35±0.32
	UCI	65.99±1.40	54.79±1.76	68.67±0.84	70.94±0.71	64.61±0.48	57.43±0.00	76.01±1.11	80.10±0.51	57.55±0.81	**83.61±0.23**	72.25±1.71	77.26±1.57
	Flights	69.07±4.02	70.57±1.82	75.48±0.26	71.09±2.72	69.18±1.52	**81.08±0.00**	74.62±0.18	74.87±0.21	59.16±1.48	61.99±1.70	70.92±1.78	64.78±1.50
	Can. Parl.	48.42±0.66	58.61±0.86	68.82±1.21	65.34±2.87	67.75±1.00	62.16±0.00	65.85±1.75	69.48±0.63	78.03±1.27	59.15±6.20	**95.44±0.57**	85.59±3.08
	US Legis.	50.27±5.13	83.44±1.16	61.91±5.82	67.57±6.47	65.81±8.52	64.74±0.00	78.15±3.34	79.63±0.84	88.40±2.34	67.78±4.29	81.25±3.62	**91.05±1.21**
	UN Trade	60.42±1.48	60.19±1.24	60.61±1.24	61.04±6.01	62.54±0.67	72.97±0.00	61.06±1.74	60.15±1.29	78.38±2.24	69.72±1.84	55.79±1.02	**86.61±0.99**
	UN Vote	67.79±1.46	67.53±1.98	52.89±1.61	67.63±2.67	52.19±0.34	66.30±0.00	50.62±0.82	51.60±0.73	72.53±1.94	68.37±1.92	51.91±0.84	**75.05±1.41**
	Contact	93.43±1.78	94.18±0.10	94.35±0.48	90.18±3.28	89.31±0.27	85.20±0.00	91.35±0.21	90.87±0.35	91.13±1.48	94.05±0.52	94.75±0.28	**95.84±0.32**
	Avg. Rank	9.08	8.62	5.92	6.23	7.62	7.77	6.69	6.38	7.31	5.08	4.46	**2.85**
AUC	Wikipedia	70.96±0.78	67.36±0.96	81.93±0.22	80.97±0.31	70.95±0.95	81.73±0.00	82.19±0.48	**84.28±0.30**	52.11±5.83	77.03±1.93	75.09±3.70	75.36±3.41
	Reddit	83.51±0.15	82.90±0.31	87.13±0.20	84.56±0.24	**88.04±0.29**	85.93±0.00	84.67±0.29	82.21±0.13	76.01±1.04	81.02±0.68	86.23±0.51	81.64±0.42
	MOOC	66.63±2.30	63.26±1.01	73.18±0.33	77.44±2.86	70.32±1.43	48.18±0.00	70.36±0.37	72.45±0.72	75.56±2.39	77.00±2.24	80.76±0.76	**89.07±0.63**
	LastFM	61.32±3.49	62.15±2.12	63.99±0.21	65.46±4.27	67.92±0.44	**77.37±0.00**	46.93±2.59	60.22±0.32	58.54±2.37	69.51±0.61	69.25±0.36	72.48±0.90
	Enron	70.92±1.05	68.73±1.34	60.45±2.12	71.34±2.46	75.17±0.50	75.00±0.00	67.64±0.86	71.53±0.85	69.62±1.12	**78.91±0.96**	74.07±0.64	75.44±1.38
	Social Evo.	90.01±3.19	93.07±0.38	92.94±0.61	95.24±0.56	89.93±0.15	87.88±0.00	93.44±0.72	94.22±0.32	92.53±4.20	97.10±0.22	97.51±0.06	**97.59±0.25**
	UCI	64.14±1.26	54.25±2.01	60.80±1.01	64.11±1.04	58.06±0.26	58.03±0.00	70.05±1.86	74.59±0.74	49.73±1.20	**79.42±0.31**	65.96±1.18	70.85±0.96
	Flights	69.99±3.10	71.13±1.55	73.47±0.18	71.63±1.72	69.70±0.75	**81.10±0.00**	72.54±0.19	72.21±0.21	59.73±1.61	59.61±1.65	69.53±1.17	64.21±1.30
	Can. Parl.	52.88±0.80	63.53±0.65	72.47±1.18	69.57±2.81	72.93±1.78	61.41±0.00	69.47±2.12	70.52±0.94	80.03±1.75	65.30±6.98	**96.70±0.59**	85.05±2.71
	US Legis.	59.05±5.52	89.44±0.71	71.62±5.42	78.12±4.46	76.45±7.02	68.66±0.00	82.54±3.91	84.22±0.91	93.04±1.35	79.09±3.45	87.96±1.80	**94.48±0.50**
	UN Trade	66.82±1.27	65.60±1.28	66.13±0.78	66.37±5.39	71.73±0.74	74.20±0.00	67.80±1.21	66.53±1.22	80.77±1.31	75.76±1.54	62.56±1.51	**89.56±0.87**
	UN Vote	73.73±1.61	72.80±2.16	53.04±2.58	72.69±3.72	52.75±0.90	72.85±0.00	52.02±1.64	51.89±0.74	77.36±1.52	74.11±2.51	53.37±1.26	**79.13±1.16**
	Contact	94.47±1.08	94.23±0.18	94.10±0.41	91.64±1.72	87.68±0.24	85.87±0.00	91.23±0.19	90.96±0.27	92.51±1.16	94.71±0.33	95.01±0.15	**95.93±0.23**
	Avg. Rank	8.08	8.23	6.77	6.31	7.31	7.00	7.00	6.54	7.46	5.08	5.00	**3.23**

Table 10: Inductive results for inductive negative sampling.

Metrics	Datasets	JODIE	DyRep	TGAT	TGN	CAWN	TCL	GraphMixer	NAT	PINT	DyGFormer	RPNet
AP	Wikipedia	68.70±0.39	62.19±1.28	84.17±0.22	81.77±0.32	67.24±1.63	82.20±2.18	**87.60±0.29**	47.37±4.39	78.23±2.89	71.42±4.43	71.29±4.33
	Reddit	62.32±0.54	61.58±0.72	63.40±0.36	64.84±0.84	63.65±0.41	60.81±0.26	64.49±0.25	61.51±0.73	**67.56±0.83**	65.35±0.60	62.14±1.72
	MOOC	63.22±1.55	62.92±1.24	76.72±0.30	77.07±3.40	74.69±0.68	74.28±0.53	73.99±0.97	69.30±0.95	73.74±1.66	80.82±0.30	**81.85±1.60**
	LastFM	70.39±4.31	71.45±1.75	76.28±0.25	69.46±4.65	71.33±0.47	65.78±0.65	76.42±0.22	70.21±0.78	77.96±1.55	76.35±0.52	**82.27±1.22**
	Enron	65.86±3.71	62.08±2.27	61.40±1.30	62.90±1.16	60.72±0.36	67.11±0.62	72.37±1.38	62.69±1.02	**80.47±1.52**	67.07±0.62	74.60±1.35
	Social Evo.	88.51±0.87	88.72±1.10	93.97±0.54	90.65±1.62	79.83±0.39	94.10±0.32	94.01±0.47	84.89±4.62	95.75±1.06	**96.82±0.17**	96.38±0.18
	UCI	63.16±2.27	52.47±2.09	70.49±0.93	70.73±0.79	64.54±0.47	76.65±0.99	81.64±0.49	51.58±0.76	**85.50±0.22**	72.13±1.86	78.50±1.18
	Flights	61.01±1.66	62.83±1.31	64.72±0.37	59.32±1.45	56.82±0.56	64.50±0.25	**65.29±0.24**	51.60±1.11	53.43±0.61	57.11±0.20	54.63±0.75
	Can. Parl.	52.58±0.86	52.24±0.28	56.46±0.50	54.18±0.73	57.06±0.08	55.46±0.69	55.76±0.65	61.71±2.77	50.66±1.67	**87.22±0.82**	68.87±1.68
	US Legis.	52.94±2.11	62.10±1.41	51.83±3.95	61.18±1.10	55.56±1.71	53.87±1.41	52.03±1.02	64.61±3.02	59.37±1.84	56.31±3.46	**66.95±1.81**
	UN Trade	55.43±1.20	55.42±0.87	55.58±0.68	52.80±3.24	54.97±0.38	55.66±0.98	54.88±1.01	69.96±2.08	57.26±1.68	52.56±1.70	**78.95±0.52**
	UN Vote	61.17±1.33	60.29±1.79	53.08±3.10	63.71±2.97	48.01±0.82	54.13±2.16	48.10±0.40	64.81±1.54	**65.70±2.82**	52.61±1.25	65.33±1.59
	Contact	90.43±2.33	89.22±0.65	**94.14±0.45**	88.12±1.50	74.19±0.81	90.43±0.17	89.91±0.36	84.13±1.77	90.68±0.46	93.55±0.52	93.57±0.57
	Avg. Rank	7.38	7.77	5.54	6.23	8.08	5.92	5.23	7.62	4.23	4.77	**3.15**
AUC	Wikipedia	61.87±0.53	57.54±1.09	78.38±0.20	75.76±0.29	62.02±0.65	79.79±0.96	**82.88±0.21**	40.95±4.99	73.30±2.33	68.33±2.82	67.96±2.77
	Reddit	61.69±0.39	60.44±0.37	64.39±0.27	64.55±0.50	**64.91±0.21**	61.36±0.26	64.27±0.13	58.31±0.63	64.02±0.46	64.80±0.25	62.37±0.83
	MOOC	64.48±1.64	64.22±1.29	74.07±0.27	77.68±3.55	71.69±0.94	69.83±0.32	72.52±0.84	68.00±1.68	75.93±2.48	80.77±0.63	**84.46±0.88**
	LastFM	68.44±3.26	68.79±1.08	69.89±0.28	66.99±5.61	67.68±0.24	55.88±1.85	70.07±0.20	64.18±0.65	75.02±0.66	70.73±0.37	**77.10±0.78**
	Enron	65.32±3.57	61.50±2.50	57.83±2.18	62.68±1.09	62.27±0.40	64.05±1.02	68.19±1.63	61.69±0.68	**78.90±1.29**	65.79±0.42	74.50±1.02
	Social Evo.	88.53±0.55	87.93±1.05	91.88±0.72	92.10±1.22	83.54±0.24	93.28±0.60	93.62±0.35	84.89±4.72	96.29±0.56	**96.91±0.09**	96.76±0.14
	UCI	60.27±1.94	51.26±2.40	62.29±1.17	62.66±0.91	56.39±0.11	70.42±1.93	75.97±0.85	42.63±0.97	**81.35±0.29**	65.58±1.00	71.37±0.84
	Flights	60.72±1.29	61.99±1.39	63.40±0.26	59.66±1.05	56.58±0.44	**63.49±0.23**	63.32±0.19	48.36±1.37	49.74±0.90	56.05±0.22	53.05±0.87
	Can. Parl.	51.61±0.98	52.35±0.52	58.15±0.62	55.43±0.42	60.01±0.47	56.88±0.93	56.63±1.09	61.83±2.92	48.93±2.24	**88.51±0.73**	68.98±1.21
	US Legis.	58.12±2.94	67.94±0.98	49.99±4.88	64.87±1.65	54.41±1.31	52.12±2.13	49.28±0.86	66.95±4.17	62.82±2.18	56.57±3.22	**68.37±1.62**
	UN Trade	58.71±1.20	57.87±1.36	59.98±0.59	55.62±3.59	60.88±0.79	61.01±0.93	59.71±1.17	70.34±2.04	62.54±1.73	57.28±3.06	**80.78±1.03**
	UN Vote	65.29±1.30	64.10±2.10	51.78±4.14	68.58±3.08	48.04±1.76	54.65±2.20	45.57±0.41	67.61±1.90	**69.44±2.98**	53.87±2.01	65.85±1.84
	Contact	90.80±1.18	88.87±0.67	93.76±0.40	88.85±1.39	74.79±0.38	90.37±0.16	90.04±0.29	84.74±1.44	92.00±0.28	**94.14±0.26**	93.47±0.43
	Avg. Rank	7.23	8.00	5.85	6.00	7.46	6.00	5.46	7.92	4.31	4.46	**3.31**

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction list the limitations of existing relative encodings, and the goal of this paper is to tackle such limitations.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitation in the appendix.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

https://doi.org/10.52202/079017-4482

Justification: All theorems include the full sets of assumptions and proof.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide the implementation details of our method in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We include our codes in the supplemental material.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Implementation details are shown in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in the appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report the mean and standard deviation in tables that record the experimental results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

https://doi.org/10.52202/079017-4482

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We list the compute resources of our experiments in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our paper follows the NeurIPS Code of Ethics in every respect.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss the potential societal impacts in the appendix.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: The proposed method does not have a high risk for misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We follow the license of each asset and cite them properly.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

https://doi.org/10.52202/079017-4482

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [No]

    Justification: We do not provide new assets in this paper.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper does not include crowdsourcing experiments and research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not include crowdsourcing experiments and research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.