DEEPSTE: Deep Spectral Temporal Embeddings for Dynamic Graph Representation Learning

Qiang Huang[1] , **Ke Liu**[2] , **Renjie Gong**[1] , **Sijing Zhang**[2] , **Hao Wang**[1*] , **Shanshan Feng**[1]
Xiao Yan[3*] , **Jiawei Jiang**[1*]
[1]School of Computer Science, Wuhan University
[2]Wuhan Dameng Database Co., Ltd
[3]Institute for Math and AI, Wuhan University
{huangq23cs, gongrenjie, wanghao.cs, victor_fengss, yanxiaosunny, jiawei.jiang}@whu.edu.cn
{liuke, zsj}@dameng.com

Abstract

Temporal embeddings play a crucial role in dynamic graph neural networks (DGNNs) by capturing the temporal dynamics of interactions. However, existing Random Fourier Feature (RFF)-based methods in DGNNs directly sample Fourier frequencies from a fixed, data-independent distribution $p(\omega)$, neglecting the temporal characteristics of dynamic graphs and thereby limiting representational capacity. We propose DEEPSTE, a deep spectral temporal embedding framework for dynamic graphs. DEEPSTE learns RFF representations via Monte Carlo importance sampling with a tractable proposal distribution $q(\omega)$ (e.g., `Gaussian`) to approximate the feature map of a shift-invariant or positive-definite kernel whose latent spectral density is analytically intractable. DEEPSTE adopts a data-dependent scale parameter η, estimated from interaction intervals, to construct the frequency proposal distribution $q(\omega)$ reflecting time–frequency uncertainty. The frequency DNN $f_{\boldsymbol{\omega}}$ and the importance-weighting DNN $g_{\mathbf{w}}$, initialized from $q(\omega)$, are jointly optimized to model the importance-sampled spectral representation and learn adaptive temporal features. Experiments demonstrate the effectiveness of DEEPSTE on dynamic link prediction and node classification tasks, while also revealing insights such as temporal embedding decay and accelerated convergence.

1 Introduction

Dynamic graphs are prevalent in real-life systems such as social networks, recommender systems, biological interactomes, and financial transaction networks, where the topological structure evolves over time [Kazemi *et al.*, 2020; Pareja *et al.*, 2020; Rossi *et al.*, 2020; Wang *et al.*, 2021a; Lu *et al.*, 2022; Zhang *et al.*, 2022; Huang *et al.*, 2023; Qian *et al.*, 2024; Huang *et al.*, 2025; Yuan *et al.*, 2025]. However, representation learning on dynamic graphs poses significant challenges due to continuous temporal dynamics and evolving graph topology, in contrast to the static and time-invariant nature of conventional graphs [Kipf and Welling, 2016; Veličković *et al.*, 2017; Ying *et al.*, 2018; Xu *et al.*, 2020]. Dynamic graph neural networks (DGNNs), such as TGAT [Xu *et al.*, 2020], TGN [Rossi *et al.*, 2020], APAN [Wang *et al.*, 2021a], CAWN [Wang *et al.*, 2021b], and SimpleDyG [Wu *et al.*, 2024], model dynamic graphs by aggregating information from temporal neighbors. A core component of these models is the temporal embedding kernel, which encodes continuous time to capture latent temporal patterns. For example, on e-commerce platforms, purchases typically reflect long-term interests, while views capture short-term preferences. Since older interactions may be less relevant to the current topology, modeling temporal relevance is crucial for accurate recommendation.

Kernel methods, such as Support Vector Machines, offer strong theoretical guarantees and effectively approximate complex nonlinear decision boundaries. However, their scalability to large dynamic graphs is hindered by the high computational and memory overhead of nonlinear kernel evaluations [Joachims, 2006; Cortes *et al.*, 2010; Si *et al.*, 2017; Li *et al.*, 2019]. Random Fourier Features (RFFs) approximate shift-invariant kernels by mapping input data into a randomized feature space via Monte Carlo direct sampling from the prior spectral distribution, e.g., `Gaussian` and `Laplacian` kernels. This enables scalable learning by replacing costly kernel matrix decomposition with explicit mappings, allowing efficient linear methods to approximate nonlinear models [Rahimi and Recht, 2007]. To address the limitation of positional encoding in self-attention, which captures discrete orderings but fails to model continuous time, [Xu *et al.*, 2019] propose a translation-invariant time kernel based on RFFs that embeds time spans into high-dimensional representations and enables self-attention to capture temporal patterns. TGAT adopts a learnable linear projection to approximate the frequency components of the Fourier basis in RFFs, enabling the model to dynamically learn the spectral distribution [Xu *et al.*, 2020]. These RFF methods have been widely incorporated into DGNNs to model temporal dynamics in continuous-time edges [Wu *et al.*, 2024].

Despite their success, existing RFF-based methods in

*Corresponding authors.

DGNNs struggle with two fundamental challenges.

• *Data-Independent Direct Sampling*. Existing RFF-based methods sample Fourier frequencies from a fixed, data-independent distribution, neglecting the latent, analytically intractable spectral distribution. This limits the expressiveness of the RFF representations and prevents adaptation to real-world dynamic graphs with complex temporal patterns [Li *et al.*, 2019; Xu *et al.*, 2019; Chung *et al.*, 2025].

• *Shallow Temporal Embeddings*. Merely using a single linear layer to approximate the frequency component in vanilla RFFs is insufficient to capture complex temporal patterns in dynamic graphs. This limited model depth constrains the ability to learn adaptive temporal embeddings that accurately represent continuous-time edge dynamics.

To address the aforementioned challenges, we propose **DEEPSTE**, a novel **DEEP S**pectral **T**emporal **E**mbedding framework for dynamic graph representation learning, which introduces the following two key advances:

• *Data-dependent Importance Sampling*. To approximate the analytically intractable spectral density distribution in real-world dynamic graphs, DEEPSTE constructs RFFs via Monte Carlo importance sampling with a tractable, data-dependent proposal distribution (e.g., `Gaussian`), which is estimated from the empirical distribution of interaction time intervals and reflects the time–frequency uncertainty principle.

• *Adaptive Temporal Features*. DEEPSTE employs multilayer perceptrons (MLPs) to learn adaptive temporal embeddings. Guaranteed by the generalization error bounds we establish, the frequency DNN and importance-weighting DNN are jointly optimized to approximate non-analytic frequency and importance weighting patterns in importance-sampled RFFs, enabling DEEPSTE to capture temporal dynamics through the learnable DNN components. Contributions can be summarized as follows:

- We propose DEEPSTE, a novel deep spectral temporal embedding for dynamic graphs, which reformulates RFFs via data-dependent Monte Carlo importance sampling estimated from interaction time intervals and consistent with the time–frequency uncertainty principle.
- We establish generalization error bounds that guarantee the DEEPSTE design, which employs a frequency initialization DNN and an importance-weighting DNN to approximate shift-invariant positive-definite kernels, enabling the learning of adaptive temporal embeddings.
- Extensive experiments on dynamic graphs show that DEEPSTE improves the dynamic link prediction and node classification tasks by 2.32% and 1.32%, respectively, while uncovering insights on DNN depth, temporal embedding decay effect, and faster convergence.

2 Preliminaries

Dynamic Graphs. A dynamic graph over continuous time $t \in (0, T]$, denoted $\mathcal{G}(t) = (\mathcal{V}(t), \mathcal{E}(t))$, is defined by a time-ordered sequence of edge events: $X(t) = \big(x(t_1), \ldots, x(t_n)\big)$. Each event $x(t_i) \in X(t)$ represents an interaction between nodes u_i and v_i at time t_i with edge feature $\mathbf{e}_i \in \mathbb{R}^e$, $x(t_i) = (u_i, v_i, \mathbf{e}_i, t_i)$. The set of neighbors of a node $u \in \mathcal{V}(t)$ at time t is defined as $\mathcal{N}_u(t) = \{v \mid (u, v) \in \mathcal{E}(t)\}$.

RFFs-based Temporal Embedding. The temporal embedding function $\phi(t)$ in DGNNs maps the continuous-time $t \in \mathbb{R}$ into a d-dimensional vector space, $\phi : \mathbb{R} \to \mathbb{R}^d$ [Xu *et al.*, 2019; Xu *et al.*, 2021]. Random Fourier Features (RFFs) explicitly map inputs into a low-dimensional feature space via Monte Carlo direct sampling from the kernel's prior spectral distribution, resulting in the approximation $k(t_1, t_2) \approx \mathbf{z}(t_1)'\mathbf{z}(t_2)$ [Rahimi and Recht, 2007; Xu *et al.*, 2019]. Based on Bochner's theorem, any continuous, shift-invariant and positive-definite kernel $k(t_1, t_2) = k(t_1 - t_2)$ on $\mathbb{R}^d$ can be represented as the Fourier transform of a non-negative finite measure. Specifically, if k is integrable, we have $k(t_1 - t_2) = \int_{\mathbb{R}} e^{i\omega(t_1 - t_2)} p(\omega)\, \mathrm{d}\omega = \mathbb{E}_\omega[\xi_\omega(t_1)\xi_\omega(t_2)^*]$ with spectral density $p(\omega)$, feature map $\xi_\omega(t) = e^{i\omega t}$, and * for complex conjugation. For the real part $\mathrm{Re}(k(t_1 - t_2))$, we have $\mathbb{E}_\omega\left[\cos\big(\omega(t_1 - t_2)\big)\right] = \mathbb{E}_\omega\left[\cos(\omega t_1)\cos(\omega t_2) + \sin(\omega t_1)\sin(\omega t_2)\right]$. This motivates mapping time into a finite-dimensional random feature space:

$$\phi^{\mathcal{B}}(t) := \sqrt{\frac{1}{d}}\left[\cos(\omega_1 t), \sin(\omega_1 t), \ldots, \cos(\omega_d t), \sin(\omega_d t)\right],$$

where $\omega_i \overset{\text{i.i.d.}}{\sim} p(\omega)$. By Mercer's theorem, periodic, translation-invariant kernels admit a Fourier decomposition $k(t_1, t_2) = \sum_{j=1}^{\infty} c_j\, \mathbf{z}_j(t_1)\, \mathbf{z}_j(t_2)$, where c_j are Fourier coefficients and $\{\mathbf{z}_j\}$ are the corresponding eigenfunctions. This leads to the Mercer random feature map:

$$\phi^{\mathcal{M}}_{\omega}(t) := [\ldots, \sqrt{c_{2j}}\cos(\frac{j\pi t}{\omega}), \sqrt{c_{2j+1}}\sin(\frac{j\pi t}{\omega}), \ldots],$$

where c_j are the kernel's Fourier coefficients. Truncating the expansion and combining multiple frequencies yields the final Mercer temporal embedding function as follows:

$$\phi^{\mathcal{M}}(t) := \left[\phi^{\mathcal{M}}_{\omega_1}(t), \ldots, \phi^{\mathcal{M}}_{\omega_k}(t)\right].$$

3 Deep Spectral Temporal Embeddings

In this section, we present DEEPSTE, a framework that integrates data-dependent Monte Carlo importance sampling with adaptive temporal embeddings learned via DNNs.

3.1 Data-dependent Importance Sampling

Temporal Embedding. By Bochner's theorem, a continuous, shift-invariant kernel $k(t_1, t_2) = k(t_1 - t_2)$ admits the Fourier representation, $k(t_1 - t_2) = \langle \phi(t_1), \phi(t_2) \rangle = \int_{\mathbb{R}} e^{i\omega(t_1 - t_2)} \frac{p(\omega)}{q(\omega)}\, q(\omega)\, \mathrm{d}\omega$, where $q(\omega)$ is a proposal distribution and $w(\omega) = \frac{p(\omega)}{q(\omega)}$ is the importance weight with spectral density $p(\omega)$. By estimating the real part of the above integral, we obtain the Bochner Fourier features:

$$\begin{aligned}\phi^{\mathcal{B}}_{\text{imp}}(t) := \sqrt{\frac{1}{d}}\big[&\sqrt{w_1}\cos(\omega_1 t), \ldots, \sqrt{w_d}\cos(\omega_d t), \\ &\sqrt{w_1}\sin(\omega_1 t), \ldots, \sqrt{w_d}\sin(\omega_d t)\big],\end{aligned} \tag{1}$$

where $\omega_i \overset{\text{i.i.d.}}{\sim} q(\omega)$ and $w_i = \frac{p(\omega_i)}{q(\omega_i)}$.

For the mercer embedding, we reformulate the kernel as an importance-weighted expectation with Fourier coefficients encoded in the spectral density implicitly, $k(t_1, t_2) =$

Proposal Distribution	*PDF*	*Variance*
`Gaussian` $\mathcal{N}(0,\sigma^2)$	$\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{\omega^2}{2\sigma^2}\right)$	σ^2
`Laplace` $(0,b)$	$\frac{1}{2b}\exp\left(-\frac{\lvert\omega\rvert}{b}\right)$	$2b^2$
`Cauchy` $(0,\gamma)$	$\frac{1}{\pi\gamma\left(1+\frac{\omega^2}{\gamma^2}\right)}$	–

Table 1: Common proposal distributions $q(\omega)$ and their probability density functions (PDFs) and variances.

$\mathbb{E}_{\omega\sim q(\omega)}[\frac{p(\omega)}{q(\omega)}\,\mathbf{z}_\omega(t_1)\,\mathbf{z}_\omega(t_2)]$. Thus, we obtain the Mercer Fourier features with frequency ω and importance weights:

$$\phi'^{\mathcal{M}}_{\omega}(t) := \Big[\ldots, \sqrt{w'_{2j}}\cos\left(\tfrac{j\pi t}{\omega}\right),\ \sqrt{w'_{2j+1}}\sin\left(\tfrac{j\pi t}{\omega}\right), \ldots\Big],$$

where $w'_j \propto c_j \frac{p(\omega_j)}{q(\omega_j)}$ denote the importance weight $\frac{p(\omega_j)}{q(\omega_j)}$ combined with the Fourier coefficient c_j. The full Mercer feature map is constructed by concatenating the sampled blocks corresponding to different frequency components:

$$\phi^{\mathcal{M}}_{\text{imp}}(t) := \left[\phi'^{\mathcal{M}}_{\omega_1}(t),\ \phi'^{\mathcal{M}}_{\omega_2}(t),\ \ldots,\ \phi'^{\mathcal{M}}_{\omega_k}(t)\right], \tag{2}$$

where $\omega_j \overset{\text{i.i.d.}}{\sim} q(\omega)$, each $\phi^{\mathcal{M}}_{\omega_j}(t)$ is a spectral embedding associated with the periodic kernel component k_{ω_j}.

Data-dependent Proposal Distributions $q(\omega)$**.** Specifically, DEEPSTE uses `Gaussian`, `Laplacian`, and `Cauchy` distributions as proposal distributions $q(\omega)$, chosen for their analytical tractability and complementary spectral properties, as summarized in Table 1. The scale parameter of the proposal distribution $q(\omega)$ controls the spectral bandwidth and influences the range of frequencies sampled. Instead of using a fixed parameter, we adopt an adaptive data-dependent approach and learn it jointly with the model. To approximate the latent spectral properties of dynamic graphs, we estimate the scale parameter from the empirical distribution of interaction time intervals. According to Claim 1 in [Li *et al.*, 2019], the optimal frequency sampling distribution satisfies

$$q(\omega) \propto f_e\left(-2\sigma_T^2\|\omega\|^2\right), \tag{3}$$

where $f_e(\cdot)$ is a positive monotonically increasing function that also depends on the kernel bandwidth, σ_T^2 denotes the variance of the time-interval distribution. This form shows that σ_T directly controls the spectral concentration of $q(\omega)$, with larger σ_T leading to a sharper decay in ω and thus a more low-frequency–concentrated distribution, consistent with the time–frequency uncertainty principle.

Thus, given a set of i.i.d. normalized time intervals $\Delta\tilde{T} = \{\Delta\tilde{t}_1, \ldots, \Delta\tilde{t}_n\}$ between nodes and their temporal neighbors in dynamic graphs, we rescale the empirical temporal dispersion as $\tilde{\sigma}_T = \sqrt{\mathrm{Var}(\Delta\tilde{T}) + h^2}/h$, where $\mathrm{Var}(\Delta\tilde{T})$ denotes the empirical variance and $h = 1$ is a small constant introduced to prevent degenerate scaling and ensure numerical stability. Since prior results suggest that the spectral concentration of $q(\omega)$ increases (i.e., corresponding to a smaller scale parameter) with the temporal variance σ_T^2, the scale of the frequency-domain proposal should decrease accordingly. We therefore adapt the scale parameter $\eta_T \in \{\sigma, b, \gamma\}$ according to the inverse relation $\mathrm{Std}(\omega) \sim \frac{1}{\sigma_T}$. In practice, we adopt

$$\eta_T = \eta_0/\tilde{\sigma}_T, \tag{4}$$

where η_0 is a base scale controlling the spectral width. Consequently, the resulting proposal distribution $q(\omega)$ adapts to empirical temporal statistics, yielding a data-dependent distribution motivated by Fourier uncertainty intuition governing the spectral behavior with respect to temporal dispersion.

3.2 Learning Adaptive Features via DNNs

To handle complex frequency distributions, and high-dimensional features in importance-sampled RFFs, we use MLPs to jointly learn frequencies and their importance weights in an adaptive manner. By the Universal Approximation Theorem [Hornik *et al.*, 1989; Lu *et al.*, 2017], there exists an MLP f can approximate the function with an error bounded by $\sup_{x\in\mathcal{X}}|f(x)-\phi(x)| < \epsilon'$. We state the following claim, which provides guidelines for constructing DNN-based RFFs with a guaranteed error bound (see Appendix).

Claim 1. *Let $\mathcal{X} \subset \mathbb{R}$ be compact, and let $k : \mathbb{R}\times\mathbb{R}\to\mathbb{R}$ be a continuous, shift-invariant, positive-definite kernel, weights $w_i = p(\omega_i)/q(\omega_i) \le W$, $\sigma^2_{p,q} := \mathbb{E}_q[w(\omega)^2\omega^2]$. Let $f(t)$ be the DNN feature map. If $\epsilon' \le \frac{\epsilon}{2\,C(\sqrt{W}+\sqrt{d})(\sqrt{W}+1)}$, then*

$$\begin{aligned}&\Pr\Big(\sup_{t_1,t_2\in\mathcal{X}}\left|f(t_1)'f(t_2) - k(t_1,t_2)\right| \ge \epsilon\Big)\\ &\qquad\le 4\sqrt{2}\sigma_{p,q}\sqrt{\frac{t_{\max}}{\epsilon}}\exp\Big(-\frac{d\epsilon^2}{16W^2}\Big).\end{aligned} \tag{5}$$

Proof Sketch. The key is to decompose the DNN kernel approximation error via an intermediate RFF map:

$$\begin{aligned}&\left|f(t_1)'f(t_2) - k(t_1,t_2)\right|\\ &= \left|f(t_1)'f(t_2) - \phi(t_1)'\phi(t_2) + \phi(t_1)'\phi(t_2) - k(t_1,t_2)\right|\\ &\le \underbrace{\left|\phi(t_1)'\phi(t_2) - k(t_1,t_2)\right|}_{\text{❶ RFF sampling error}} + \underbrace{\left|f(t_1)'f(t_2) - \phi(t_1)'\phi(t_2)\right|}_{\text{❷ DNN approximation error}},\end{aligned}$$

corresponding to the RFF sampling error and the DNN approximation error, respectively, by the triangle inequality.

❶ Based on prior work [Rahimi and Recht, 2007; Xu *et al.*, 2019], and by Hoeffding's inequality, we derive the former, which attains its minimum at $N^\star = \sigma_{p,q}\sqrt{\frac{2t_{\max}}{\epsilon}}\exp\Big(\frac{d\epsilon^2}{16W^2}\Big)$.

❷ For the DNN approximation error, exploiting the 1-Lipschitz property of trigonometric functions, we obtain $\left|\sqrt{\frac{w_i}{d}}\cos(\omega_i t) - g_{\mathbf{w},i}\cos(f_{\boldsymbol{\omega},i}(t))\right| \le \sqrt{\frac{w_i}{d}}\left|f_{\boldsymbol{\omega},i}(t) - \omega_i t\right| + \left|g_{\mathbf{w},i} - \sqrt{\frac{w_i}{d}}\right| \le \sqrt{\frac{W}{d}}\epsilon'_\omega + \epsilon'_w \le \left(\sqrt{\frac{W}{d}}+1\right)\epsilon',\ \epsilon' := \max\{\epsilon'_\omega, \epsilon'_w\}$. Applying the Euclidean norm $\|\cdot\|_2$ and the inequality $\sqrt{a+b+c} \le \sqrt{a}+\sqrt{b}+\sqrt{c}$, gives $\|f(t)-\phi(t)\| \le C_1(\sqrt{W}+\sqrt{d})\epsilon'$. As $\|\phi(t)\| \le \sqrt{W}, \|f(t)\| \le C_2(\sqrt{W}+1)$, utilizing the Cauchy–Schwarz inequality, we obtain $\sup_{t_1,t_2\in\mathcal{X}}|f(t_1)'f(t_2) - \phi(t_1)'\phi(t_2)| \le \sup_{t_1,t_2\in\mathcal{X}}\|f(t_1) - \phi(t_1)\|\|f(t_2)\| + \|\phi(t_1)\|\|f(t_2) - \phi(t_2)\| \le C(\sqrt{W}+\sqrt{d})(\sqrt{W}+1)\epsilon' \le \epsilon/2$, gives $\epsilon' \le \frac{\epsilon}{2C(\sqrt{W}+\sqrt{d})(\sqrt{W}+1)}$, where $C = C_1(C_2+1)$ absorbs all normalization and norm-aggregation constants. □

Algorithm 1: Training process of DEEPSTE

Input: Dynamic graph $\mathcal{G}(\mathcal{V}(t), \mathcal{E}(t))$; proposal distribution $q(\omega)$; frequency DNN $f_{\boldsymbol{\omega}}$ and importance-weighting $g_{\mathbf{w}}$.

Scale parameter η_T with Eq. (4).
for *each layer* $l = 1, \ldots, L$ **do**
  Sample weight parameters $\mathbf{W}^{(l)} \sim q(\omega; \eta_T^2 \mathbf{I})$.
  Initialize layer l in $f_{\boldsymbol{\omega}}$ and $g_{\mathbf{w}}$ with weights $\mathbf{W}^{(l)}$.
end
for *iterations from 1 to I* **do**
  for *number of steps* **do**
    Batch time intervals $T_b = (\Delta t_1, \ldots, \Delta t_b)$.
    Spectral temporal embedding $\phi_{\mathrm{DNN}}(T_b)$ using Bochner Eq. (6) or Mercer Eq. (7).
    Compute DGNN model loss function $\mathcal{L}_{\mathcal{G}}$.
    Update DNNs $f_{\boldsymbol{\omega}}$, $g_{\mathbf{w}}$ via $\nabla_{f_{\boldsymbol{\omega}}} \mathcal{L}_{\mathcal{G}}$, $\nabla_{g_{\mathbf{w}}} \mathcal{L}_{\mathcal{G}}$.
  end
end

Guided by the existing analysis, the importance-sampled Bochner RFF $\phi^{\mathcal{B}}_{\mathrm{imp}}(t)$ in Equation (1) consists of two components: frequencies sampled from the proposal distribution and their corresponding importance weights. Thus, we employ two DNNs: a frequency DNN $f_{\boldsymbol{\omega}} : \mathbb{R} \to \mathbb{R}^d$ to simulate frequency components ω, and an importance-weighting $g_{\mathbf{w}} : \mathbb{R}^{2d} \to \mathbb{R}^d$ to approximate the square root of corresponding importance weights, $\sqrt{w(\omega)/d}$. To incorporate both the proposal distribution and the data-dependent scale parameter, we initialize the DNN weights using samples from $q(\omega)$, aiming to better approximate the underlying sample features. The DEEPSTE Bochner temporal embedding is defined as:

$$\phi^{\mathcal{B}}_{\mathrm{DNN}}(t) := g_{\mathbf{w}}\left([\cos(f_{\boldsymbol{\omega}}(t)) \oplus \sin(f_{\boldsymbol{\omega}}(t))]\right) \in \mathbb{R}^d, \quad (6)$$

where $\oplus$ denotes vector concatenation.

Correspondingly, for the importance-sampled Mercer RFF $\phi^{\mathcal{M}}_{\mathrm{imp}}(t)$ with m Fourier components as introduced in Equation (2), the frequency DNN $f_{\boldsymbol{\omega}}: \mathbb{R} \to \mathbb{R}^{d/m}$, where $d \bmod m = 0$, is responsible for learning the reciprocal of spectral frequencies, $\frac{1}{\omega}$. The importance-weighting $g_{\mathbf{w}} : \mathbb{R}^{2d} \to \mathbb{R}^d$ aims to approximate the square root of the importance weights combined with the Fourier coefficients, $\sqrt{cw(\omega)}$. The DEEPSTE Mercer temporal embedding is defined as:

$$\begin{aligned}\phi^{\mathcal{M}}_{\mathrm{DNN}}(t) :=& g_{\mathbf{w}}\left([\cos\left(\pi f_{\boldsymbol{\omega}}(t) \oplus \cdots \oplus m\pi f_{\boldsymbol{\omega}}(t)\right) \oplus \right. \\ & \left. \sin\left(\pi f_{\boldsymbol{\omega}}(t) \oplus \cdots \oplus m\pi f_{\boldsymbol{\omega}}(t)\right)]\right) \in \mathbb{R}^d.\end{aligned} \quad (7)$$

The harmonic components $\{k\pi f_{\boldsymbol{\omega}}(t)\}_{k=1}^{m}$ correspond to multiples of the learned base frequency, facilitating approximation of the kernel's spectral density. The cosine and sine terms form a $2d$-dimensional intermediate representation, which is transformed by $g_{\mathbf{w}}$ to yield the final embedding. Algorithm 1 presents the training procedure of DEEPSTE.

4 Experimental Evaluation

4.1 Experimental Settings

Datasets. The five benchmark datasets, including four small-scale and one large-scale, with time spans of up to 10^7 are summarized in Table 2 and introduced in Appendix. For transductive tasks, graphs are chronologically split 70%/15%/15% for training/validation/testing. For inductive tasks, 10% of nodes are masked as unseen, consistent with previous work.

Dataset	*# Nodes*	*# Edges*	d_V	d_E	t_{max}	$\tilde{\sigma}_T$
UCI	2K	60K	172	100	1.67×10^7	3.26
Wikipedia	9K	157K	172	172	2.68×10^6	1.45
MOOC	7K	412K	172	4	2.57×10^6	1.75
USLegis	1K	60K	172	1	1.10×10^1	1.01
StackOverflow	201K	5M	172	172	6.73×10^7	1.19

Table 2: Statistics of the datasets used in the experiments.

Baselines. We compare DEEPSTE against 5 state-of-the-art temporal embedding methods commonly used in DGNNs, including: the TGAT-Timer [Xu *et al.*, 2020], RNN [Cho *et al.*, 2014], Attention [Vaswani *et al.*, 2017], and two advanced spectral temporal embedding: Sampling-Bochner and Sampling-Mercer [Xu *et al.*, 2019; Xu *et al.*, 2020]. All baselines are built upon six representative backbone DGNNs—including attention-based (e.g., TGAT, APAN), memory-augmented (e.g., TGN), and motif-based models (e.g., CAWN)—as well as two efficient DGNN frameworks, DyG [Wu *et al.*, 2024] and TGL [Zhou *et al.*, 2022], with ROC AUC used as the evaluation metric.

Hyparparameters. The base scale parameter η_0 in Equation (4) is set to 0.01. The m in the DEEPSTE-Mercer (7) is set to 4 to evenly partition the embedding dimension. The hidden dimension of the DGNNs is set to 128. The learning rate is set to 0.001, and early stopping is applied with a patience of 3 epochs over 3 runs. Detailed configurations are in the appendix. Experiments were conducted on NVIDIA A800 GPUs. The code for DEEPSTE is publicly available at https://github.com/johnnyhuangcs/DeepSTE.

4.2 Main Results

This section summarizes the experimental results for the dynamic link prediction and node classification tasks.

Dynamic Link Prediction. We compare DEEPSTE against baseline temporal embeddings under both transductive and inductive settings on 4 backbone DGNNs. As shown in Table 3, DEEPSTE consistently outperforms all baseline temporal embeddings across all settings and datasets on all 4 backbone DGNNs, achieving the highest ROC AUC scores. For TGAT, DEEPSTE-Mercer achieves the most significant improvement on the USLegis dataset, yielding an 11.24% increase in ROC AUC over the best-performing baseline (TGAT-Timer). The average improvement on TGAT across all datasets is 5.66%. DEEPSTE consistently outperforms the best baselines across three DGNNs (TGN, CAWN, and APAN), with an average increase of 2.32% across all models and datasets, demonstrating its effectiveness in dynamic link prediction. Notably, on TGAT, DEEPSTE-Mercer achieves the highest ROC AUC scores on 7 out of 8 datasets across both transductive and inductive settings, whereas DEEPSTE-Bochner attains the top score on only one time. How-

DGNN	Time Encoder	Transductive				Inductive			
		UCI	Wikipedia	MOOC	USLegis	UCI	Wikipedia	MOOC	USLegis
TGAT	TGAT-Timer	76.71 ± 0.32	75.54 ± 0.48	60.23 ± 0.46	60.16 ± 1.37	67.54 ± 0.08	75.39 ± 0.65	57.40 ± 0.55	*56.07 ± 1.36*
	RNN	75.51 ± 1.51	75.04 ± 0.40	*60.70 ± 0.23*	48.47 ± 1.80	71.15 ± 0.45	74.09 ± 0.27	60.15 ± 0.80	54.84 ± 1.98
	Attention	54.97 ± 1.97	73.30 ± 0.13	55.60 ± 0.34	43.35 ± 0.49	51.93 ± 1.19	72.11 ± 0.15	54.20 ± 0.75	43.70 ± 0.21
	Sampling-Bochner	61.81 ± 1.36	72.17 ± 0.27	59.66 ± 1.84	50.83 ± 1.23	58.46 ± 1.39	71.01 ± 0.41	57.40 ± 0.33	49.20 ± 1.62
	Sampling-Mercer	73.16 ± 1.96	73.54 ± 0.06	59.93 ± 1.44	55.94 ± 1.34	69.03 ± 1.61	75.82 ± 0.14	58.05 ± 3.00	55.77 ± 0.32
	DEEPSTE-Bochner	*81.16 ± 1.76*	*80.44 ± 0.53*	60.61 ± 0.17	*61.48 ± 0.45*	*77.76 ± 2.77*	**78.82 ± 0.36**	*60.37 ± 0.09*	55.09 ± 1.28
	DEEPSTE-Mercer	**83.02 ± 1.90**	**80.47 ± 0.08**	**61.10 ± 0.04**	**71.40 ± 1.43**	**80.09 ± 0.62**	*78.62 ± 0.23*	**61.20 ± 0.00**	**65.50 ± 0.31**
TGN	TGAT-Timer	87.52 ± 1.41	92.50 ± 2.75	87.33 ± 0.77	80.86 ± 0.08	78.81 ± 1.68	90.86 ± 1.68	84.46 ± 0.76	59.71 ± 0.69
	RNN	87.23 ± 0.29	95.81 ± 0.03	88.67 ± 0.13	77.55 ± 0.17	83.86 ± 0.08	95.31 ± 0.02	*87.69 ± 0.83*	55.58 ± 1.40
	Attention	85.74 ± 0.08	95.70 ± 0.03	87.74 ± 0.30	74.58 ± 0.19	82.20 ± 0.83	95.17 ± 0.01	87.03 ± 0.37	59.89 ± 0.70
	Sampling-Bochner	86.41 ± 0.16	95.56 ± 0.07	78.15 ± 1.68	78.44 ± 0.33	82.72 ± 0.12	94.83 ± 0.04	76.80 ± 1.83	59.98 ± 0.06
	Sampling-Mercer	86.74 ± 0.10	95.69 ± 0.03	85.08 ± 0.42	76.31 ± 0.60	83.07 ± 0.27	95.04 ± 0.03	85.88 ± 0.29	*60.23 ± 0.73*
	DEEPSTE-Bochner	**90.89 ± 0.14**	**98.27 ± 0.00**	**91.63 ± 0.53**	*81.23 ± 0.06*	**85.95 ± 2.06**	*95.45 ± 0.26*	**88.43 ± 1.62**	**60.68 ± 1.44**
	DEEPSTE-Mercer	*89.23 ± 0.12*	*97.84 ± 0.24*	*89.15 ± 1.75*	**81.27 ± 0.36**	*85.18 ± 0.83*	**95.58 ± 0.64**	87.35 ± 1.49	58.82 ± 1.34
CAWN	TGAT-Timer	*86.69 ± 0.00*	97.12 ± 0.03	67.02 ± 0.02	69.40 ± 0.12	85.56 ± 0.05	97.14 ± 0.06	66.46 ± 0.15	72.36 ± 0.12
	RNN	85.11 ± 0.02	97.17 ± 0.01	66.37 ± 0.11	84.59 ± 0.11	85.02 ± 0.04	97.18 ± 0.01	65.86 ± 0.08	87.45 ± 0.07
	Attention	82.75 ± 0.02	96.52 ± 0.03	60.97 ± 0.08	69.07 ± 0.37	81.39 ± 0.04	96.31 ± 0.00	59.07 ± 0.14	71.94 ± 0.24
	Sampling-Bochner	85.32 ± 0.29	96.94 ± 0.01	63.40 ± 0.22	74.37 ± 0.60	85.42 ± 0.30	96.96 ± 0.00	63.49 ± 0.24	78.15 ± 0.13
	Sampling-Mercer	83.99 ± 0.05	96.87 ± 0.05	63.43 ± 0.16	75.56 ± 0.82	83.94 ± 0.12	96.90 ± 0.01	63.49 ± 0.25	78.79 ± 0.34
	DEEPSTE-Bochner	86.45 ± 0.10	**97.63 ± 0.02**	*67.22 ± 0.03*	*86.02 ± 0.14*	*86.20 ± 0.13*	**97.64 ± 0.01**	*67.87 ± 0.16*	*87.96 ± 0.17*
	DEEPSTE-Mercer	**87.08 ± 0.04**	*97.57 ± 0.02*	**67.63 ± 0.03**	**86.55 ± 0.23**	**86.38 ± 0.16**	*97.59 ± 0.00*	**68.46 ± 0.02**	**88.22 ± 0.04**
APAN	TGAT-Timer	75.75 ± 0.61	84.45 ± 1.49	98.98 ± 0.08	75.95 ± 0.92	84.01 ± 0.29	84.09 ± 1.85	99.07 ± 0.23	74.50 ± 1.69
	RNN	73.12 ± 0.62	86.93 ± 2.36	98.10 ± 1.14	74.75 ± 1.70	82.26 ± 0.75	87.27 ± 1.48	98.10 ± 1.22	82.31 ± 0.96
	Attention	71.59 ± 0.47	90.69 ± 0.91	*99.29 ± 0.04*	72.47 ± 1.07	79.25 ± 0.22	92.01 ± 2.13	99.34 ± 0.03	80.10 ± 2.76
	Sampling-Bochner	74.37 ± 1.80	85.54 ± 0.31	85.58 ± 1.00	71.03 ± 1.25	82.56 ± 1.35	86.50 ± 2.01	80.58 ± 1.10	71.24 ± 1.76
	Sampling-Mercer	*76.35 ± 0.51*	87.95 ± 1.78	98.72 ± 0.21	68.52 ± 1.74	83.75 ± 0.11	90.69 ± 1.88	*99.40 ± 0.04*	64.72 ± 1.06
	DEEPSTE-Bochner	**76.38 ± 0.98**	*90.69 ± 0.18*	99.27 ± 0.23	**79.94 ± 1.14**	*84.30 ± 1.21*	**93.77 ± 0.79**	99.32 ± 0.03	**83.82 ± 1.82**
	DEEPSTE-Mercer	76.19 ± 0.26	**91.04 ± 2.65**	**99.35 ± 0.06**	*79.65 ± 1.35*	**85.03 ± 0.15**	*92.09 ± 1.27*	**99.41 ± 0.13**	*82.66 ± 2.85*

Table 3: ROC AUC results of DGNNs on the dynamic link prediction task across benchmark datasets under transductive and inductive settings. The best, second-best results are highlighted in **bold**, *underlined*, respectively.

Time Encoder	Transductive		Inductive	
	DyG	TGL-TGN	DyG	TGL-TGN
TGAT-Timer	84.33 ± 0.21	88.52 ± 0.12	67.28 ± 0.24	67.31 ± 0.11
RNN	82.46 ± 0.02	87.02 ± 0.00	64.91 ± 0.17	61.99 ± 0.66
Attention	77.67 ± 0.60	88.26 ± 0.01	59.14 ±1.20	66.57 ± 0.10
Sampling-B.	82.23 ± 0.61	88.84 ± 0.06	60.97 ±1.08	66.02 ± 0.03
Sampling-M.	82.80 ± 0.31	89.06 ± 0.08	63.33 ± 2.04	67.27 ± 0.10
DeepSTE-B.	**85.64 ± 0.00**	*89.46 ± 0.93*	*68.41 ± 0.25*	*67.75 ± 0.19*
DeepSTE-M.	*85.34 ± 0.16*	**89.65 ± 0.06**	**68.61 ± 0.35**	**68.24 ± 0.11**

Table 4: Link prediction performance of time encoders on the large StackOverflow dataset with the efficient DyG and TGL-TGN.

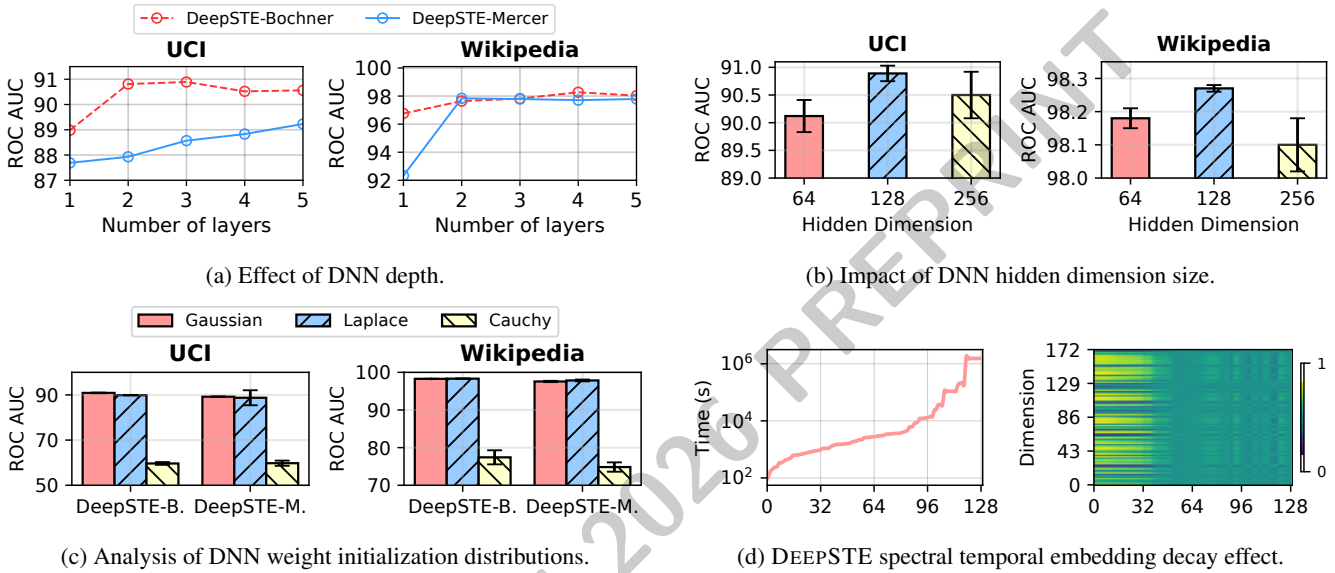
ever, on TGN, DEEPSTE-Bochner achieves more top scores than DEEPSTE-Mercer. DEEPSTE-Bochner and DEEPSTE-Mercer achieve the highest ROC AUC scores 13 and 19 times, respectively, across all settings. This suggests that DEEPSTE-Mercer appears to be more effective at capturing temporal dynamics than DEEPSTE-Bochner. For the baseline temporal embedding methods, we observe that the spectral-temporal method Sampling-Mercer achieves the second-best ROC AUC score three times, while TGAT-Timer and RNN each achieve it twice. Spectral temporal methods appear to be more effective than classical temporal embeddings.

We further evaluate our method on large-scale dynamic graphs using efficient DGNN models (e.g., DyG and TGL-TGN) whose training time does not exceed 10,000 seconds (see appendix for details). As shown in Table 4, our proposed DEEPSTE achieves an average accuracy improvement of 1.04% over the strongest baselines, with a maximum improvement of 1.33%, demonstrating its effectiveness on large dynamic graphs and recent efficient DGNNs.

Dynamic Node Classification. Table 5 shows dynamic node classification results on Wikipedia and MOOC, which have node labels. DEEPSTE consistently achieves the best results across various settings and datasets, generally outperforming baseline temporal embedding methods. DEEPSTE-Mercer yields its largest performance gain on the MOOC dataset when integrated with the TGAT DGNN backbone, showing a 5.34% increase in ROC AUC compared to the best-performing baseline, Sampling-Mercer. On average, it achieves a 1.32% improvement across all datasets. Similar to

Time Encoder	TGAT		TGN		CAWN		APAN	
	Wikipedia	MOOC	Wikipedia	MOOC	Wikipedia	MOOC	Wikipedia	MOOC
TGAT-Timer	77.19 ± 1.68	54.24 ± 0.24	87.74 ± 0.64	*61.58 ± 0.20*	77.33 ± 0.91	63.56 ± 0.07	*85.02 ± 0.06*	66.21 ± 0.10
RNN	75.57 ± 1.86	56.30 ± 0.25	84.91 ± 2.09	59.28 ± 1.47	76.05 ± 0.17	64.58 ± 0.05	82.89 ± 0.29	66.52 ± 1.27
Attention	76.54 ± 1.25	54.94 ± 0.26	83.25 ± 1.37	61.41 ± 1.91	76.20 ± 0.19	64.76 ± 0.68	84.66 ± 0.86	67.83 ± 0.49
Sampling-Bochner	76.63 ± 1.64	55.68 ± 0.72	87.07 ± 1.20	58.95 ± 0.49	75.88 ± 0.12	63.89 ± 0.02	82.60 ± 0.64	65.87 ± 0.33
Sampling-Mercer	75.86 ± 0.59	56.43 ± 0.09	87.05 ± 0.65	57.54 ± 0.80	74.92 ± 0.74	63.76 ± 0.51	82.39 ± 0.33	*68.28 ± 0.72*
DEEPSTE-Bochner	**77.93 ± 1.92**	*60.66 ± 0.52*	*88.07 ± 0.18*	61.08 ± 0.22	*77.78 ± 0.88*	*65.22 ± 1.12*	84.67 ± 0.36	**68.36 ± 0.38**
DEEPSTE-Mercer	*77.51 ± 1.00*	**61.77 ± 0.62**	**88.42 ± 0.45**	**62.09 ± 0.15**	**77.80 ± 0.74**	**66.87 ± 0.53**	**85.63 ± 0.37**	67.64 ± 0.43

Table 5: ROC AUC results on the dynamic node classification task using various time encoders evaluated on the Wikipedia and MOOC datasets. The best, second-best results are highlighted in **bold**, *underlined*, respectively.



(a) Effect of DNN depth.

(b) Impact of DNN hidden dimension size.

(c) Analysis of DNN weight initialization distributions.

(d) DEEPSTE spectral temporal embedding decay effect.

Figure 1: Analysis of key factors in DEEPSTE-Bochner influencing TGN performance on the transductive link prediction task, along with a visualization of DEEPSTE temporal embedding decay as interaction time intervals increase in the UCI dataset.

the results observed in dynamic link prediction, DEEPSTE-Mercer consistently outperforms DEEPSTE-Bochner across most experimental settings, achieving the highest ROC AUC scores in 6 out of 8 cases, compared to only 2 for DEEPSTE-Bochner, suggesting potentially better temporal modeling.

4.3 Micro Experiments

We conduct an in-depth analysis of the impact of various factors on the performance of DGNNs, as shown in Figure 1.

Effect of DNN Depth. We investigate the impact of the number of linear layers in the DNNs on the performance of TGN. As shown in Figure 1a, increasing the number of layers generally leads to significantly improved performance compared to using a single linear layer. On the UCI dataset, DEEPSTE-Bochner achieves the best performance with 3 linear layers, while DEEPSTE-Mercer performs best with 5 layers. On the Wikipedia dataset, the corresponding optimal numbers of layers are 4 and 2, respectively. These results show that increasing DNN depth effectively enhances model performance, highlighting the effectiveness of deep models.

Impact of Hidden Dimension Size. Figure 1b illustrates the impact of hidden dimension size in the MLPs on TGN performance. The results show that increasing the hidden dimension generally improves performance, with the optimal result achieved at size 128. This suggests that larger hidden dimensions enhance the DNNs' ability.

Influence of Sampling Distribution. As illustrated in Algorithm 1, the DNN weights are initialized using a proposal distribution $q(\omega)$. We analyze the effect of different sampling distributions (`Gaussian`, `Laplace`, and `Cauchy`), with the results presented in Figure 1c. The proposal distribution $q(\omega)$ is initialized with a data-driven scale parameter for each dataset, as shown in Table 2. The results indicate that the choice of sampling distribution significantly affects model performance. The `Gaussian` and `Laplace` distributions generally yield the best results, while the `Cauchy` distribution performs the worst. Therefore, the `Cauchy` distribution may be less suitable as a sampling distribution for Random Fourier Features, as its heavy-tailed characteristics can introduce high variance in the sampled features.

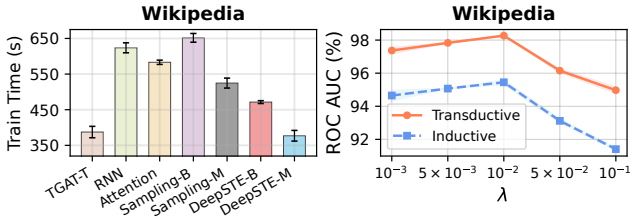


Figure 2: (a) Total training time for temporal embedding methods. (b) Sensitivity analysis of the hyperparameter η_0 under different initialization values. Experiments are conducted using TGN with the DeepSTE-Bochner temporal embedding on the Wikipedia dataset.

	UCI	Wikipedia
DeepSTE-Bochner	90.89 ± 0.14	98.27 ± 0.00
– (w/o) Data-dependent η	87.53 ± 0.76	97.72 ± 0.11
– (w/o) Frequency DNN f_{ω}	88.87 ± 0.26	95.01 ± 0.21
– (w/o) Weight DNN $g_{\mathbf{w}}$	86.15 ± 0.10	96.17 ± 0.07
– (w) Positional embedding	91.15 ± 0.22	98.35 ± 0.15

Table 6: Results of the ablation study on the DeepSTE-Bochner temporal embedding module within the TGN framework for the transductive link prediction task.

Spectral Temporal Embedding Decay Effect. Figure 1d visualizes the temporal embeddings generated by DeepSTE-Bochner on the UCI dataset. The left panel shows 128 interaction intervals, while the right displays the corresponding 172-dimensional temporal embeddings. The model assigns higher weights to recent interactions, with embedding magnitudes decaying and becoming more uniform over time. This indicates a stronger emphasis on temporally close events, potentially enhancing representation learning.

Training Efficiency. Figure 2(a) and the appendix compare DeepSTE's training efficiency with temporal embedding baselines. DeepSTE converges in fewer epochs and achieves shorter overall training times. On Wikipedia, DeepSTE-Mercer achieves the fastest training time and the fewest required epochs. Specifically, DeepSTE-M and DeepSTE-B consistently rank among the top-3 methods in training time across all model–dataset combinations, with mean ranks of 2.83 and 3.25, respectively (see appendix for details).

Scale Parameter Initialization. Figure 2(b) shows the effect of the base scale parameter η_0, varied from 10^{-3} to 10^{-1}. DeepSTE performs best at 10^{-2}, suggesting a favorable balance between high- and low-frequency temporal patterns.

4.4 Ablation Studies

We conduct ablation studies to evaluate the impact of each component in DeepSTE, as shown in Table 6. Removing the data-driven scale parameter η reduces performance by 3.36% on UCI and 0.55% on Wikipedia, underscoring its importance. Excluding the frequency DNNs f_{ω} leads to drops of 2.02% and 3.26%, respectively, indicating the importance of phase information. Eliminating the importance-weighting $g_{\mathbf{w}}$ results in declines of 4.74% on UCI and 2.10% on Wikipedia, underscoring the significance of frequency-specific weighting. Combining DeepSTE with positional embeddings yields minor gains (0.26% on UCI and 0.08% on Wikipedia), suggesting that positional information slightly enhances DGNNs' recognition of ordering information.

5 Related Work

DGNNs. DGNNs effectively model dynamic graphs by representing them as event sequences that capture evolving topological structures. Specifically, attention-based DGNNs employ a self-attention mechanism with functional time encoding to capture temporal dependencies and dynamic topological structures [Xu *et al.*, 2020]. TGN represents a continuous-time dynamic graph as a sequence of time-stamped events and leverages a novel combination of memory modules and graph-based operators, using memory-based aggregation and update mechanisms to model temporal dynamics [Rossi *et al.*, 2020]. Motifs-based DGNNs represent temporal networks using causal anonymous walks derived from temporal random walks, capturing temporal motifs in network dynamics [Wang *et al.*, 2021b; Jin *et al.*, 2022]. Recent advanced DGNNs employ Transformer-based architectures with temporal encoding to model temporal dynamics [Wu *et al.*, 2024; Yu *et al.*, 2023; Feng *et al.*, 2025].

Temporal Embedding. The temporal embedding kernel is crucial in DGNNs, enabling the capture of temporal dynamics in contrast to static graphs. Pioneering works employ a single linear layer to project time intervals, thereby enabling the learning of dynamic embedding trajectories or exogenous features in temporal interactions [Kumar *et al.*, 2019; Chung *et al.*, 2025]. Similarly, DyRep also utilizes a linear layer to project time intervals as exogenous features that influence the evolution of embeddings [Trivedi *et al.*, 2019]. RFF-based temporal embeddings use translation-invariant kernels to map continuous time spans into vectors, extending the self-attention mechanism from sequence order to the temporal domain [Xu *et al.*, 2019]. The TGAT timer employs a linear layer to learn the phase patterns of standard RFFs introduced in [Xu *et al.*, 2019], which are subsequently integrated into the attention mechanism [Xu *et al.*, 2020]. This approach has demonstrated strong performance and been widely adopted in later DGNNs for dynamic graph representation learning.

6 Conclusion

We propose DeepSTE, a deep spectral temporal embedding framework for dynamic graphs that encodes continuous time via Random Fourier Features. The framework employs Monte Carlo importance sampling of the temporal kernel to estimate the target spectral density, together with a data-dependent scale parameter that adapts the frequency distribution of the kernel. Frequency and importance-weighting networks, initialized from proposal distributions, are jointly optimized to learn adaptive temporal embeddings through importance-sampled RFF components. Extensive experiments demonstrated consistent improvements across dynamic link prediction and node classification tasks, while also revealing phenomena such as temporal embedding decay and accelerated convergence for temporal modeling.

Acknowledgments

This work was sponsored by National Natural Science Foundation of China (NO. 62472327, No. 62441229), Key R&D Program of Hubei Province (No. 2023BAB077), and Sichuan Clinical Research Center for Imaging Medicine (YXYX2402).

References

[Cho *et al.*, 2014] Kyunghyun Cho, Bart Van Merriënboer, Çağlar Gulçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1724–1734, 2014.

[Chung *et al.*, 2025] Hsing-Huan Chung, Shravan Chaudhari, Xing Han, Yoav Wald, Suchi Saria, and Joydeep Ghosh. Between linear and sinusoidal: Rethinking the time encoder in dynamic graph learning. *arXiv preprint arXiv:2504.08129*, 2025.

[Cortes *et al.*, 2010] Corinna Cortes, Mehryar Mohri, and Ameet Talwalkar. On the impact of kernel approximation on learning accuracy. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 113–120. JMLR Workshop and Conference Proceedings, 2010.

[Feng *et al.*, 2025] ZhengZhao Feng, Rui Wang, TianXing Wang, Mingli Song, Sai Wu, and Shuibing He. A comprehensive survey of dynamic graph neural networks: Models, frameworks, benchmarks, experiments and challenges. *IEEE Transactions on Knowledge and Data Engineering*, 2025.

[Hornik *et al.*, 1989] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[Huang *et al.*, 2023] Shenyang Huang, Farimah Poursafaei, Jacob Danovitch, Matthias Fey, Weihua Hu, Emanuele Rossi, Jure Leskovec, Michael Bronstein, Guillaume Rabusseau, and Reihaneh Rabbany. Temporal graph benchmark for machine learning on temporal graphs. *Advances in Neural Information Processing Systems*, 36:2056–2073, 2023.

[Huang *et al.*, 2025] Xuanwen Huang, Wei Chow, Yize Zhu, Yang Wang, Ziwei Chai, Chunping Wang, Lei Chen, and Yang Yang. Enhancing cross-domain link prediction via evolution process modeling. In *Proceedings of the ACM on Web Conference 2025*, pages 2158–2171, 2025.

[Jin *et al.*, 2022] Ming Jin, Yuan-Fang Li, and Shirui Pan. Neural temporal walks: Motif-aware representation learning on continuous-time dynamic graphs. *Advances in Neural Information Processing Systems*, 35:19874–19886, 2022.

[Joachims, 2006] Thorsten Joachims. Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 217–226, 2006.

[Kazemi *et al.*, 2020] Seyed Mehran Kazemi, Rishab Goel, Kshitij Jain, Ivan Kobyzev, Akshay Sethi, Peter Forsyth, and Pascal Poupart. Representation learning for dynamic graphs: A survey. *Journal of Machine Learning Research*, 21(70):1–73, 2020.

[Kipf and Welling, 2016] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[Kumar *et al.*, 2019] Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1269–1278, 2019.

[Li *et al.*, 2019] Yanjun Li, Kai Zhang, Jun Wang, and Sanjiv Kumar. Learning adaptive random features. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 4229–4236, 2019.

[Lu *et al.*, 2017] Zhou Lu, Hongming Pu, Feicheng Wang, Zhiqiang Hu, and Liwei Wang. The expressive power of neural networks: A view from the width. *Advances in neural information processing systems*, 30, 2017.

[Lu *et al.*, 2022] Mingxuan Lu, Zhichao Han, Susie Xi Rao, Zitao Zhang, Yang Zhao, Yinan Shan, Ramesh Raghunathan, Ce Zhang, and Jiawei Jiang. Bright-graph neural networks in real-time fraud detection. In *Proceedings of the 31st ACM international conference on information & knowledge management*, pages 3342–3351, 2022.

[Pareja *et al.*, 2020] Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao Schardl, and Charles Leiserson. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 5363–5370, 2020.

[Qian *et al.*, 2024] Hao Qian, Hongting Zhou, Qian Zhao, Hao Chen, Hongxiang Yao, Jingwei Wang, Ziqi Liu, Fei Yu, Zhiqiang Zhang, and Jun Zhou. Mdgnn: Multi-relational dynamic graph neural network for comprehensive and dynamic stock investment prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 14642–14650, 2024.

[Rahimi and Recht, 2007] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20, 2007.

[Rossi *et al.*, 2020] Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637*, 2020.

[Si *et al.*, 2017] Si Si, Cho-Jui Hsieh, and Inderjit S Dhillon. Memory efficient kernel approximation. *Journal of Machine Learning Research*, 18(20):1–32, 2017.

[Trivedi *et al.*, 2019] Rakshit Trivedi, Mehrdad Farajtabar, Prasenjeet Biswal, and Hongyuan Zha. Dyrep: Learning representations over dynamic graphs. In *International conference on learning representations*, 2019.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[Veličković *et al.*, 2017] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[Wang *et al.*, 2021a] Xuhong Wang, Ding Lyu, Mengjian Li, Yang Xia, Qi Yang, Xinwen Wang, Xinguang Wang, Ping Cui, Yupu Yang, Bowen Sun, et al. Apan: Asynchronous propagation attention network for real-time temporal graph embedding. In *Proceedings of the 2021 international conference on management of data*, pages 2628–2638, 2021.

[Wang *et al.*, 2021b] Yanbang Wang, Yen-Yu Chang, Yunyu Liu, Jure Leskovec, and Pan Li. Inductive representation learning in temporal networks via causal anonymous walks. In *International Conference on Learning Representations (ICLR)*, 2021.

[Wu *et al.*, 2024] Yuxia Wu, Yuan Fang, and Lizi Liao. On the feasibility of simple transformer for dynamic graph modeling. In *Proceedings of the ACM web conference 2024*, pages 870–880, 2024.

[Xu *et al.*, 2019] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Self-attention with functional time representation learning. *Advances in neural information processing systems*, 32, 2019.

[Xu *et al.*, 2020] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs. *arXiv preprint arXiv:2002.07962*, 2020.

[Xu *et al.*, 2021] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. A temporal kernel approach for deep learning with continuous-time information. *arXiv preprint arXiv:2103.15213*, 2021.

[Ying *et al.*, 2018] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 974–983, 2018.

[Yu *et al.*, 2023] Le Yu, Leilei Sun, Bowen Du, and Weifeng Lv. Towards better dynamic graph learning: New architecture and unified library. *Advances in Neural Information Processing Systems*, 36:67686–67700, 2023.

[Yuan *et al.*, 2025] Qi Yuan, Yang Liu, Yateng Tang, Xinhuan Chen, Xuehao Zheng, Qing He, and Xiang Ao. Dynamic graph learning with static relations for credit risk assessment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 39, pages 13133–13141, 2025.

[Zhang *et al.*, 2022] Mengqi Zhang, Shu Wu, Xueli Yu, Qiang Liu, and Liang Wang. Dynamic graph neural networks for sequential recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 35(5):4741–4753, 2022.

[Zhou *et al.*, 2022] Hongkuan Zhou, Da Zheng, Israt Nisa, Vasileios Ioannidis, Xiang Song, and George Karypis. Tgl: a general framework for temporal gnn training on billion-scale graphs. *Proceedings of the VLDB Endowment*, 15(8):1572–1580, 2022.