1. Introduction

Higher-order link prediction (HLP), which targets the identification of potential multi-node interactions (e.g., simplices) in complex networks, plays a pivotal role in uncovering latent patterns of collective behaviors [1, 2], collaborative relationships [3, 4], and information diffusion mechanisms [5, 6]. Based on pairwise interaction assumptions, traditional link prediction [7] frameworks fail to explain ubiquitous higher-order relational patterns observed in real-world systems [8]. For example, social networks [9, 10] exhibit intricate group interactions that transcend dyadic connections, while chemical reaction networks [11] involve multi-compound processes beyond binary reactions.

Using simple graph models to describe the higher-order relationships in networks leads to structural information loss [12]. Therefore, researchers propose two main approaches: hypergraph-based methods leveraging hyperedges to model multi-interactions [13, 14] and algebraic topology frameworks using simplicial complexes to preserve multi-dimensional interactions [15, 16]. Wang et al. [17] enhance the tuple relationship among nodes by incorporating hyperedges into sequences of random walk nodes using an incidence graph to learn high-quality node representation vectors. Guan et al.[18] deal with semantic fusion in heterogeneous networks utilizing a graph neural network based on hypergraphs and a multilevel reconstruction method to maintain the integrity of different semantic information. Identity-aware Hypergraph Attention Network (ID-HAN) [19] models the critical contribution of each node within a hyperedge by employing a unique attention mechanism. Battiloro et al. [20] design a series of principled self-attention mechanisms to efficiently process data associated with simplex shapes of different orders (e.g., nodes, edges, and triangles). Chavan et al. [21] propose new methods by generalizing Node2vec [22] under different operators, using a 1-hop subgraph in graph2vec [23], and embedding triangles in the graph neural network [24].

Despite these advances, current HLP methods mainly rely on the network's static topology representations, and ignore the temporal evolution inherent in real-world higher-order networks. Real-world networks evolve dynamically [25]. The higher-order interactions among nodes do not occur overnight but are gradually established over time. As a result, it is not easy to accurately reflect the dynamic evolution laws of higher-order structures by relying only on static topological features, which leads to the underperformance of the model when dealing with time-aware tasks. Therefore, this paper proposes a Time-aware Dynamic Embedding (TDE) learning framework that significantly improves the performance of HLP by effectively fusing temporal evolution with higher-order structural features. TDE firstly designs a time-aware simplex temporal structure walk strategy, which captures local and global temporal dependencies of evolving simplices, and combines skip-gram to develop temporal structure embeddings for nodes. Subsequently, TDE constructs a time embedding matrix to encode each node's activity history into time series embedding, thereby modeling the temporal behavior of nodes over time. The temporal structure embedding and time series embedding are integrated through an adaptive fusion mechanism to produce dynamic node representations that jointly reflect structural and temporal characteristics. Furthermore, TDE captures deep spatiotemporal dependencies and uncovers latent patterns in the evolution of higher-order structures by employing a attention-based feature fusion module. This framework allows TDE to effectively capture the dynamics of higher-order interactions, while overcoming the limitations of existing methods that cannot incorporate temporal information in modeling higher-order relationships. The main contributions of this paper are as follows,

- In this paper, we propose a novel time-aware simplex temporal structure walk that captures local and global temporal dependencies across evolving simplices. This mechanism enables the model to reflect better dynamic higher-order interaction patterns often overlooked by static walk strategies.
- TDE combines temporal structure information of nodes with the time dynamic features through the adaptive fusion mechanism to generate a dynamic representation of nodes that can express the relationship between higher-order structure and time. This combination strategy solves the problem of traditional algorithms ignoring the modeling of higher-order structures that evolve with time.
- The performance of TDE is validated on nine real-world datasets, outperforming existing methods in terms of adaptability and prediction accuracy.

2. Related work

In the studies of higher-order network interactions, researchers have proposed various prediction algorithms, which can generally be classified into three categories: similarity-based, matrix optimization-based, and embedding-based.

Similarity-based methods analyze the network structure and similarity metrics, typically offering strong interpretability. Kumar et al. [26] use resource allocation to predict hyperedges without relying on candidate sets, identifying potential node members for new hyperedges. Additionally, [27] introduces a stepwise framework using n-projected graphs to capture interaction patterns involving multiple nodes, reducing information loss in HLP. Benson et al. [28] study the emergence of higher-order structures like triangles, finding that tight node relationships are more likely to lead to future interactions. Jung-Muller et al. [29] propose a Memory Model (MM) for predicting higher-order temporal network events, considering not only the target hyperlink's past activity but also that of its sub- and super-hyperlinks. Ling et al. [30] propose an innovative causal feature selection algorithm based on mutual information, which reduces computational complexity by performing pairwise mutual information comparisons in two stages, thereby enhancing efficiency on high-dimensional data. Additionally, [31] develops an open-source toolbox, Causal Learner, which integrates Bayesian network data generation, global and local causal structure learning algorithms, and various Markov blanket learning methods. Hu et al. [32] integrate network motifs into clustering via tensor

representations and higher-order Markov chains, significantly improving functional module identification.

Matrix optimization-based methods use iterative optimization and matrix factorization techniques to capture higher-order network relationships comprehensively. Matrix Boosting (MATBoost) [33] iteratively optimizes between the occurrence matrix and the adjacency matrix to jointly infer missing higher-order interactions, using constrained optimization to recover missing relationships. Oyetunde et al. [34] combine machine learning with constrained optimization to predict new metabolic reactions through matrix factorization. Maurya et al. [35] use the hypergraph Laplacian tensor's spectral decomposition to predict potential hyperedges by minimizing construction costs. Zhou et al. [36] extend the normalized cut method from simple graphs to hypergraphs, defining the hypergraph Laplacian operator and proposing a new hypergraph embedding technique for inference tasks. He et al. [37] integrate multiple similarity networks using higher-order motif-based Markov chains and matrix optimization to identify functional modules and predict miRNA–disease associations.

Embedding-based methods use network embedding techniques to represent higher-order interactions as dense low-dimensional vectors and capture complex relationships among multiple nodes. Logical Hyperlink Predictor (LHP) [38] uses a logical aggregation layer and a constraint scoring layer to embed logical knowledge into directed hyperedges for predicting missing hyperedges. Li et al. [39] integrate relationship encoding and multi-hop neighborhood features to capture higher-order interaction information and fuse it into embedding representations for higher-order link inference. Additionally, Hypergraph Network Embedding (HNE) [40] converts HLP into a classification task using low-dimensional vector representations. Wang et al. [41] combine binary edges and simplex information to generate a weighted adjacency matrix, aggregating the effects of edges on simplices for HLP. Liu et al. [42] propose Higher-order structure Temporal Graph Neural network (HTGN), that uses hypergraph representation and higher-order structure identification to better model complex group interactions, while reducing training memory via hyperedge feature aggregation. Fusing Higher and Lower-order Representations (FuHLDR) [43] fuses low-order embeddings from graph convolutional networks with higher-order meta-path representations to build drug and disease features, using a Random Vector Functional Link network for association prediction. FuHLDR improves drug repositioning accuracy but relies on domain knowledge for meta-path design, requiring enhanced adaptive mining and generalization. Liang et al. [44] propose a 3D mesh simplification method combining Whale Optimization and Differential Evolution, optimizing operation sequences to effectively preserve geometric features and reduce errors. Despite excellent mesh quality, high computational cost calls for improved efficiency and parallelization.

3. Methodology

3.1. Preliminary

Let $G_{\mathrm{h}} = \{V_{\mathrm{h}}, S_{\mathrm{h}}, T_{\mathrm{h}}\}$ be a higher-order network, where $V_{\mathrm{h}} = \{v_1, v_2, ..., v_n\}$, $S_{\mathrm{h}} = \{s_1, s_2, ..., s_m\}$, and $T_{\mathrm{h}} = \{t_1, t_2, ..., t_m\}$

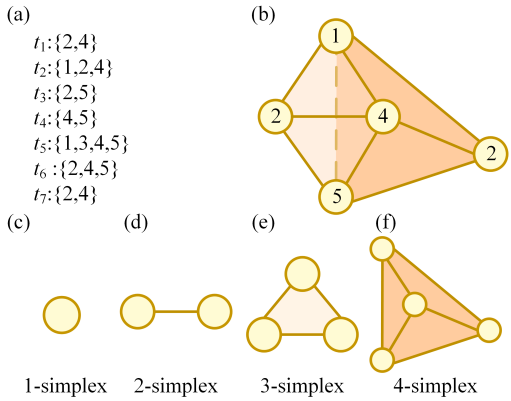

Fig. 1 Higher-order network interpretation diagram. Subfigure (a) represents the structure of a higher-order network dataset, with each timestamp t corresponding to a simplex s. Numbers in braces behind t are nodes forming s. Subfigure (b) shows the structure graph of the higher-order network. Subfigures (c), (d), (e), and (f) illustrate four types of simplices, which are 1-simplex, 2-simplex, 3-simplex, and 4-simplex.

denote the node set, the simplex set, and the timestamp set associated with forming the simplex. Suppose there is a dataset of a higher-order network, as shown in Fig. 1(a). The number of nodes is 5. The dimensions of the simplex set and the timestamp set are both 7. For each simplex s_i in S_{h}, there is a corresponding timestamp t_i in T_{h}. Additionally, for convenience, we define the order of the simplex s_i as $|s_i|$ (some papers use $|s_i|$-1), where the order of the simplex corresponds to the order of interaction among nodes. A simplex can be viewed as a generalization of edges in ordinary graphs: while an edge connects exactly two nodes, a simplex captures interactions among multiple nodes simultaneously, representing higher-order relationships beyond pairwise connections.

In higher-order networks, complex structures such as 5-simplices and beyond are typically formed through the gradual evolution and aggregation of lower-order simplices (e.g., 3-simplices and 4-simplices). In real-world systems, including social networks and biological collectives, the interactions and behavioral patterns represented by the lower-order simplices serve as the foundation for understanding the formation of more complex structures. Therefore, focusing on the prediction of 3-simplices and 4-simplices enables a more fine-grained analysis of higher-order interactions and their influence on the dynamics of the overall system. Based on the above analysis, this paper aims to model and predict the evolution of 3-simplices and 4-simplices, providing a theoretical basis and empirical support for future exploration of more complex higher-order structures.

3.2. TDE

The proposed TDE framework includes the following three key modules: the temporal structure embedding of nodes, the time series embedding of nodes, and the attention-based feature fusion, shown in Fig. 2.

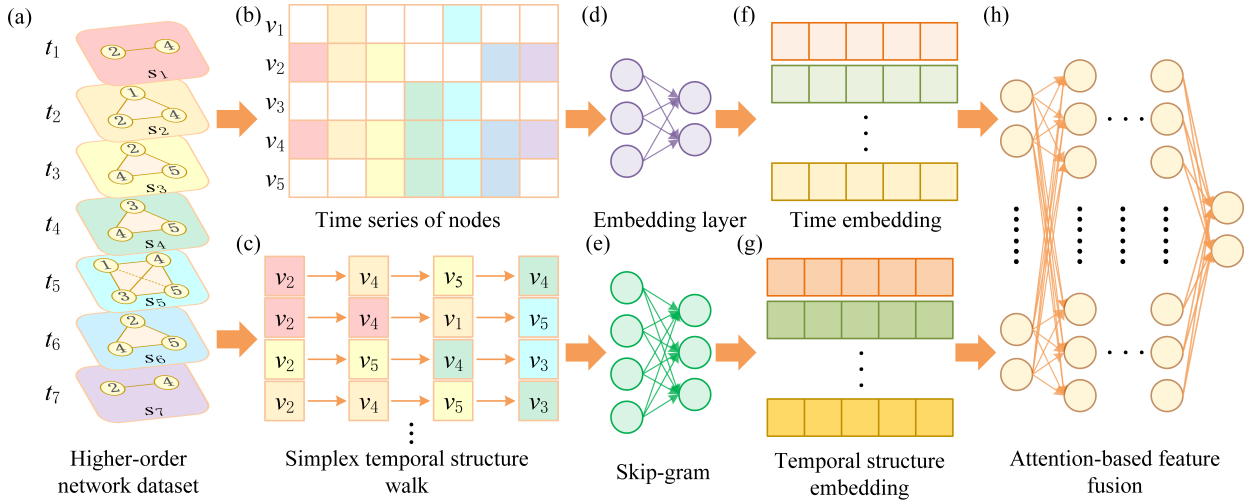

Fig. 2 The framework of TDE. Subfigure (a) shows an example of a simplex temporal structure derived from a higher-order network dataset, presenting different simplices from the first to the seventh. Subfigure (b) represents the time series matrix of nodes. Subfigure (c) illustrates the walk of nodes through the simplex temporal structure, where the different colors indicate changes in the simplices to which the node belongs at various timestamps. Subfigure (d) shows the embedding layer of the neural network. Subfigure (e) illustrates skip-gram, which maps nodes into a low-dimensional space using the temporal walk sequences obtained in subfigure (d) to capture their temporal structural information. Subfigures (f) and (g) represent the time series embedding and higher-order temporal structure embedding of nodes, respectively. Subfigure (h) demonstrates the attention-based feature fusion module, which utilizes the attention mechanism to extract deep features of nodes in the spatial and temporal dimensions. It then outputs the predicted probabilities of simplices by combining the aggregation and softmax layers.

3.2.1. Temporal structure embedding of nodes

The simplex temporal structure in higher-order networks provides natural higher-order relational characteristics, where the evolution of simplices at different timestamps forms complex dynamic relationships among nodes. To model these interactions effectively, capturing the time sequence of simplex formation and the structural dependence of encoding them is necessary. In this paper, we propose a novel simplicial time-structure walk method to effectively capture the dynamic higher-order structural features in higher-order networks. This technique generates higher-order dynamic correlated sequences through time-serialized random walk within simplices. Unlike traditional static random walk, our approach incorporates a time-jump strategy that authentically reflects time interactions in the real world. This method enables the model to flexibly adapt to dynamic changes within the network, effectively describing the temporal evolution patterns of nodes and their higher-order topological features.

Specifically, the method randomly selects a timestamp for each node in which the node appears as the walk's starting point. At each step of the random walk, the method determines whether to move from the simplex at the current time to the simplex at the next timestamp by generating a random number and comparing it with a predefined jump probability p. The random walker then randomly selects sampling nodes through the neighbor set in the simplex. If the current node and timestamp are v and t, the neighbor set of node v at t is $N_t(v) = \{v' | v' \in s_t, v' \neq v\}$, then the probability of selecting the following node v' can be represented as,

$$\mathrm{P}(v'|v,t) = \begin{cases} \frac{1}{|N_t(v)|} & v' \in N_t(v), \\ 0 & v' \notin N_t(v). \end{cases} \tag{1}$$

where $|N_t(v)|$ represents the number of elements in $N_t(v)$.

As shown in Fig. 2, taking the first two walks in Fig. 2(c) as an example, the walk starts at v_2. First, a simplex that contains v_2 is randomly selected from the temporal structure of simplices in Fig. 2(a) (e.g., s_1 at t_1, with a pink background). Next, a random number is generated and compared with jump probability p.

- If the random number is bigger than p, a timestamp jump is made, transitioning to the next simplex containing node v_2 (e.g., s_2 at t_2, with an orange background). At t_2, a node from the neighbor set $N_{t_2}(v_2) = \{v_1, v_4\}$ is randomly selected as the next sampling node.

- If the random number is less than or equal to p, the following sampling node is selected randomly from the neighbor set $N_{t_1}(v_2) = \{v_4\}$ at t_1.

This process is repeated continuously until the subsequent temporal simplex structure no longer contains the current sampled node or the pre-defined walk length l is reached. The node sequence W generated by this random walk method can capture the dynamic associations of nodes across simplices over time. Using W, we further apply word2vec [45] to learn the temporal structure embedding $\mathbf{\Phi}_v \in \mathbb{R}^d$ of nodes, where d represents the embedding dimension,

$$\mathbf{\Phi}_v = \mathrm{word2vec}(W, v, d). \tag{2}$$

The word2vec model, based on the node sequences generated by the random walk, is trained using skip-gram. The model learns low-dimensional representations of nodes by maximizing the conditional probability of predicting the target node given the context nodes. The objective function is,

$$\mathcal{L} = \max \sum_{v \in V_h} \sum_{v_j \in N(v)} \log \mathrm{P}(v_j|v), \tag{3}$$

where $\mathrm{P}(v_j|v)$ represents the probability of generating a context node v_j given the node v. By optimizing the above objective function, each node's temporal structure embedding vector is learned, reflecting the dynamic evolution characteristics of the node's participation in higher-order interactions.

Temporal structure embedding of nodes captures the nodes' dynamic higher-order properties and effectively encodes the features of nodes participating in diverse simplices through the random walk strategy. This embedding approach makes TDE capable of capturing complex, multi-dimensional relationships and evolving network interactions.

3.2.2. Time series embedding of nodes

In higher-order networks, nodes' temporal activity and behavior patterns are often key factors in forming higher-order links. To effectively capture the temporal dynamics of node activity, this paper designs a time series embedding module that models node behavior along the time dimension. This representation method allows the model to distinguish nodes with sustained temporal engagement and those with intermittent or bursty interactions, which is crucial for revealing temporal regularities that precede the formation of higher-order simplices. For instance, in Fig. 1(a), nodes 2 and 4 co-occur across multiple timestamps (at t_1, t_2, t_6, and t_7), exhibiting a stable interaction pattern over time. Such consistent co-participation increases the likelihood that these nodes will form or participate in higher-order simplices during these periods. By learning from these evolving patterns, the time series embedding module enhances the model's ability to capture meaningful temporal signals, thereby improving the accuracy of HLP.

In this paper, we aim to construct a time series embedding for each node to capture its active patterns along the time dimension. Suppose node v's time series is $T_v = \{t_1, t_2, t_3, ..., t_i\}$. We first initialize the node's time series to incorporate temporal information into the node's embedding representation. Then, a Min-Max normalization method is used to normalize the time series, which ensures that the time values are scaled within a standard range and makes the temporal information comparable among nodes in different time distributions. As a result, the learned temporal embeddings retain the relative time positioning of node activations and enhance the model's sensitivity to subtle temporal shifts, which may indicate early signals of higher-order structure emergence. The normalization process is shown in Eq. 4,

$$t'_i = \lfloor \text{Min-Max}(t_i) l_\tau \rfloor = \lfloor \frac{t_i - t_{\min}}{t_{\max} - t_{\min}} l_\tau \rfloor, \tag{4}$$

where Min-Max$(\cdot)$ represents the Min-Max normalization function. $t_{\min}$ and $t_{\max}$ represent the minimum and maximum values in the time series, respectively. l_τ is the length of the time series after initialization where τ represents the annotation of time. The symbol $\lfloor \cdot \rfloor$ represents the operation of rounding down. Then, each t'_i is mapped to a vector space through an embedding layer,

$$\mathbf{h}_{t'_i} = \text{Embedding}(t'_i), \tag{5}$$

where $\mathbf{h}_{t'_i} \in \mathbb{R}^{d_\tau}$ represents the time point embedding of t'_i. d_τ is the dimension of the time point embedding, and Embedding (t'_i) presents the embedding matrix from the embedding layer that transforms t_i into a vector representation. The embedding layer is implemented as a trainable lookup table, where each discrete time point t'_i is mapped to a continuous d_τ-dimensional vector. This operation is equivalent to a one-hot encoding followed by a linear transformation without bias.

The time series embedding of a node is an aggregated expression of multiple time points embedding. We introduce an attention-based adaptive weight aggregation to capture the contributions of different time points to the node's dynamic characteristics. For a node's time series embedding, the weight distribution for each time point is calculated as follows,

$$\alpha_i = \frac{\exp(\mathbf{w}^{\mathrm{T}} \mathbf{h}_{t'_i})}{\sum_j \exp(\mathbf{w}^{\mathrm{T}} \mathbf{h}_{t'_j})}, \tag{6}$$

where α_i represents the time attention weight for t'_i, which adaptively adjusts the importance of each timestamp through time dynamic characteristics. $\mathbf{w}$ is the learned parameter vector. Thus, for the time series embedding $\mathbf{\Phi}_v^\tau$ of node v, it can be expressed as,

$$\mathbf{\Phi}_v^\tau = \sum_i \alpha_i \mathbf{h}_{t'_i}. \tag{7}$$

3.2.3. Attention-based feature fusion module

TDE leverages temporal structure embedding to model higher-order dependencies and time series embedding to capture the evolution of node states. Based on the Transformer [46], this paper introduces an attention-based feature fusion module to enhance the modeling of temporal dependencies in node embeddings. This module employs a multi-head self-attention mechanism to dynamically aggregate historical interactions, effectively capturing both short-term fluctuations and long-term trends. By jointly incorporating temporal structure and time series embeddings into the attention mechanism, the module learns spatiotemporal dependencies that extend beyond simple pairwise interactions, enabling a richer representation of higher-order temporal dynamics. The attention-based feature fusion module further enhances this process by globally attending to temporal contexts and relational patterns across nodes, allowing it to model implicit correlations embedded in complex interaction sequences. This design significantly improves the expressiveness of dynamic higher-order structures and enhances the overall accuracy of HLP.

First, we use a Multi-layer Perceptron (MLP) to fuse the temporal structure embedding $\mathbf{\Phi}_v$ and the time series embedding $\mathbf{\Phi}_v^\tau$, resulting in the dynamic embedding representation $\mathbf{E}_v$ for v,

$$\mathbf{E}_v = \mathrm{MLP}(\mathbf{\Phi}_v + \mathbf{\Phi}_v^{\tau}). \tag{8}$$

Then, for each attention head in the multi-head self-attention mechanism, the calculation is as follows,

$$\mathbf{Q} = \mathbf{E}_v \cdot \mathbf{W}_Q, \mathbf{K} = \mathbf{E}_v \cdot \mathbf{W}_K, \mathbf{V} = \mathbf{E}_v \cdot \mathbf{W}_V, \tag{9}$$

where $\mathbf{W}_Q$, $\mathbf{W}_K$, and $\mathbf{W}_V \in \mathbb{R}^{|\mathbf{E}_v| \times d_{\text{head}}}$ are learnable weight matrices (d_{head} represents each attention head's feature dimension), which map the embedding $\mathbf{E}_v$ into three different feature spaces, *i.e.*, the query space, the key space, and the value space. $\mathbf{W}_Q$ and $\mathbf{W}_K$ help the model compute similarity scores to obtain the attention weights. $\mathbf{W}_V$ applies the calculated attention weights to perform a weighted sum. The output $\mathbf{E}_v^{'}$ of the multi-head attention mechanism is shown in Eqs. 10 and 11,

$$\mathrm{Attention}_i(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}(\frac{\mathbf{Q}\mathbf{K}^{\mathrm{T}}}{\sqrt{d_{\text{head}}}})\mathbf{V}, \tag{10}$$

$$\mathbf{E}_v^{'} = \mathrm{ReLU}(\mathrm{Concat}(\mathrm{Attention}_1, ..., \mathrm{Attention}_H) \cdot W_O), \tag{11}$$

where $\mathrm{Attention}_i(\cdot)$ represents the output of the i-th attention head. $\mathrm{ReLU}(\cdot)$ and $\mathrm{Concat}(\cdot)$ represent the activation function and the concatenation function, respectively. H represents the number of attention heads. W_O is the learnable output projection matrix that maps the concatenated multi-head attention outputs back into the model's embedding space.

Next, $\mathbf{E}_v^{'}$ is aggregated to generate the dynamic embedding $\mathbf{E}_s$ of s,

$$\mathbf{E}_s = \mathrm{ReLU}(\mathrm{AVG}(\mathbf{E}_v^{'})), v \in s, \tag{12}$$

where $\mathrm{AVG}(\cdot)$ denotes the average aggregation function in the aggregation layer, with feature aggregation of simplices.

Finally, the softmax layer is used to output the final prediction results,

$$\hat{y} = \mathrm{softmax}(\mathbf{E}_s), \tag{13}$$

where $\hat{y}$ denotes the prediction result.

3.3. Time complexity analysis

The time complexity of the proposed TDE algorithm can be summarized as follows. The temporal random walk process requires $O(|V_{\text{h}}| \cdot r \cdot l \cdot \bar{d})$, where r is the number of walks per node, l is the walk length, and $\bar{d}$ is the average number of neighbors within a simplex. The skip-gram model used to train the word2vec embeddings has a time complexity of $O(|V_h| \cdot r \cdot l \cdot d)$, where d is the embedding dimension. Finally, the attention-based feature fusion module performs multi-head attention on node embeddings, leading to a complexity of $O(H \cdot |V_{\text{h}}|^2 \cdot d_{\text{head}})$, where H is the number of heads and d_{h} is the per-head dimension. Thus, the total time complexity of the TDE framework is $O(|V_{\text{h}}| \cdot r \cdot l \cdot \bar{d} + |V_h| \cdot r \cdot l \cdot d + H \cdot |V_{\text{h}}|^2 \cdot d_{\text{head}})$.

Algorithm 1 Time-aware Dynamic Embedding (TDE)

Input: Higher-order network $G_{\text{h}} = \{V_{\text{h}}, S_{\text{h}}, T_{\text{h}}\}$, walk length l, jump probability p, embedding dimension d
Output: Predicted probability $\hat{y}$ for k-simplices
for each node $v \in V_{\text{h}}$ **do**
  Randomly select timestamp t where $v \in s_t$
  Initialize walk sequence $W \leftarrow [v]$
  for $i = 1$ to l **do**
    Let $N_t(v) = \{v' \in s_t \setminus \{v\}\}$
    Sample $v' \sim N_t(v)$
    With probability p, jump to next timestamp t' where $v' \in s_{t'}$
    Append v' to W
    Update $v \leftarrow v'$, $t \leftarrow t'$ or t
  end for
end for
Train skip-gram on W to get $\Phi_v = \mathrm{word2vec}(W, d)$
Compute time series embedding Φ_v^{τ} for each node via attention and Embedding layer
Fuse: $\mathbf{E}_v \leftarrow \mathrm{MLP}(\Phi_v + \Phi_v^{\tau})$
Apply self-attention to get $\mathbf{E}_v'$
Aggregate: $\mathbf{E}_s = \mathrm{ReLU}(\mathrm{AVG}(\mathbf{E}_v'))$, $v \in s$
Predict: $\hat{y} = \mathrm{softmax}(\mathbf{E}_s)$
return $\hat{y}$

4. Experiments

4.1. Experimental setup

4.1.1. Datasets

To verify the effectiveness of TDE, we select nine commonly used higher-order network datasets with timestamps. Congress-bills [47] captures US congressional bills and their supporters. Contact networks (Contact-high-school [48], Contact-primary-school [49]) track student interactions in schools. Email networks (Email-Enron [50], Email-Eu [51]) reflect email communication within organizations. NDC-substances [28] records drug ingredients and their release dates. Tagging networks (Tags-ask-ubuntu [28], Tags-math-sx [28]) from Stack Exchange represent tags for questions. Threads-ask-ubuntu [28] contains user records in the forum. For all datasets (the details are shown in Table 1), nodes represent entities and simplices capture higher-order interactions with timestamps. We use the first 70% of simplices as the training set and the rest 30% as the testing set according to time.

4.1.2. Parameters

The thresholds of all algorithms are set to 0.5, meaning that if the predicted probability of s exceeds 0.5, it will occur in the future; otherwise, it will not. In the TDE framework, the temporal structure embedding dimension (d), the time series embedding dimension (d_τ), the per-head dimension (d_{head}), the number of attention heads (H), and the initialized temporal sequence length (l_τ) are set to 128, 16, 16, 4, 100, respectively. The model is trained for 200 epochs. Given the significant impact of jump probability p and walk length l, we will select the optimal values in the experiments in the Subsection 4.2.

Table 1 Statistics of datasets. The number of nodes and simplices, the maximum order of the simplices, the number of 3-simplices, 4-simplices, and 5^+-simplices (simplices of order five or higher), and the temporal duration of the datasets (unit: seconds) are shown here. Numbers in parentheses indicate counts after deduplication.

Datasets	Nodes	Simplices	Max order	3-simplices	4-simplices	5^+-simplices	Time span
Congress-bills	1718	260851(85082)	25	11794(10156)	8766(7764)	53255(51314)	1.06×10^9
Contact-high-school	327	172035(7937)	5	7475(2091)	576(222)	7(7)	3.64×10^8
Contact-primary-school	242	106879(12799)	5	9262(4600)	471(347)	12(9)	1.17×10^5
Email-Enron	143	10883(1542)	18	1231(317)	567(138)	714(193)	1.13×10^8
Email-Eu	998	234760(25791)	25	17969(4938)	6146(2294)	8051(4414)	6.95×10^7
NDC-substances	5311	112405(10025)	25	6589(745)	3257(535)	6413(3854)	2.14×10^{17}
Tags-ask-ubuntu	3029	271233(151441)	5	71949(52282)	44704(39158)	27542(25475)	2.72×10^8
Tags-math-sx	1629	822059(174933)	5	173836(63870)	73308(50892)	33618(29244)	2.34×10^8
Threads-ask-ubuntu	125602	192947(167001)	14	21637(21621)	4560(4560)	1506(1505)	2.42×10^8

4.1.3. Baselines

This paper compares six baseline algorithms to evaluate the proposed method's effectiveness: (1) Deepwalk [52] captures graph structure by learning node embedding from random walk-based "sentences" using word2vec. (2) HTGN [42] employs hypergraph representations and dynamic algorithms to effectively overcome the limitations of traditional temporal graph neural networks in modeling complex higher-order interactions and improving efficiency. (3) MM [29] weights the past activities of the target hyperlink and its sub- and super-hyperlinks using exponential decay based on event recency, capturing the decreasing influence of older interactions. (4) Transformer [46] uses multi-head self-attention to model long-range dependencies, improving node representation with global information. (5) HNE [40] embeds hypergraphs by converting them into bipartite graphs and learning node embedding through random walk and skip-gram. (6) RWTA [25] combines random walk with temporal aggregation to create node sequences with time information, optimizing embedding by weighting temporal features.

4.1.4. Evaluating indicators

We use Average Precision (AP [53]) and Area Under the Curve (AUC [54]) as metrics to assess the algorithms' performance in the evaluation.

AP quantifies the mean value of Precision across all the values of Recall. The calculations are given by Eqs. 14, 15 and 16,

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}}, \tag{14}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}}, \tag{15}$$

$$\text{AP} = \sum_i (\text{Recall}(i) - \text{Recall}(i-1)) \cdot \text{Precision}(i), \tag{16}$$

where TP (True Positive) refers to the number of actual positive samples correctly identified by the model, FN (False Negative) is the number of actual positive samples incorrectly predicted as negative by the model, and FP (False Positive) presents the number of actual negative samples incorrectly predicted as positive by the model. Recall(i) and Precision(i) represent the Recall and Precision corresponding to the i-th threshold, respectively.

AUC evaluates binary classifiers by measuring their ability to identify positive and negative samples across thresholds. We compute it using sampling methods as shown in Eq. 17,

$$\text{AUC} = \frac{N_1 + N_2}{N}. \tag{17}$$

Eq. 17 approximates AUC by repeatedly (N times) comparing prediction scores of randomly selected positive-negative sample pairs. N_1 counts cases where the positive sample's score exceeds the negative's, while N_2 counts cases of ties.

4.2. Sensitivity analysis of parameters p and l

To better analyze and discuss TDE's performance, we investigate the effects of parameters p and l across different datasets. p governs TDE's transition to the next timestamp during random walk, while l determines the step length of the random walk. By tuning p and l, TDE can effectively balance local and global dynamics to achieve optimal performance.

Fig. 3 presents the AP and AUC performances of TDE across nine datasets with two prediction orders (k=3 and k=4) and different p values (from 0.1 to 0.9). In the contact networks (Contact-high-school and Contact-primary-school) and Email-Enron datasets, the AP and AUC performances of TDE generally increase as p rises initially, peaking at p=0.3 in most cases, and then gradually decline with further increases in p. This indicates that recent interactions improve the model's ability to capture meaningful temporal dynamics. However, an overly large p value leads to excessive focus on short-term patterns, neglecting longer-term dependencies and resulting in performance degradation. In the Email-Eu network, as p increases, the AP and AUC performances of TDE remain relatively stable, with a significant decline observed starting from p=0.7. Notably, there is an early performance drop at p=0.5 when k=4, which is attributed to the stochastic nature of TDE's sampling mechanism, leading to inconsistent capture of long-term and short-term information. This inconsistency can reduce the model's ability to effectively represent the temporal structure of interactions. As p continues to increase beyond 0.7, performances of TDE's AP and AUC in Email-Eu network gradually declines due to the growing emphasis on short-term interactions, which weakens the model's capacity to capture broader temporal dependencies. In the Congress-bills, NDC-substances, Tags-ask-ubuntu,

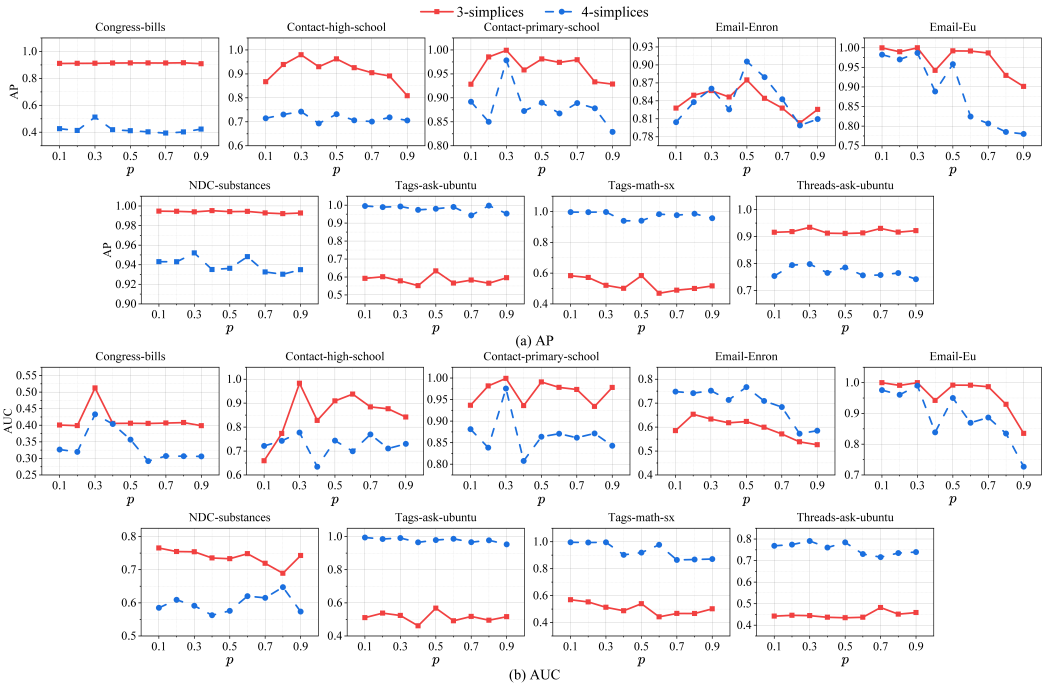

Fig. 3 The AP and AUC performances of TDE at different p. p ranges from 0.1 to 0.9, and the interval is 0.1.

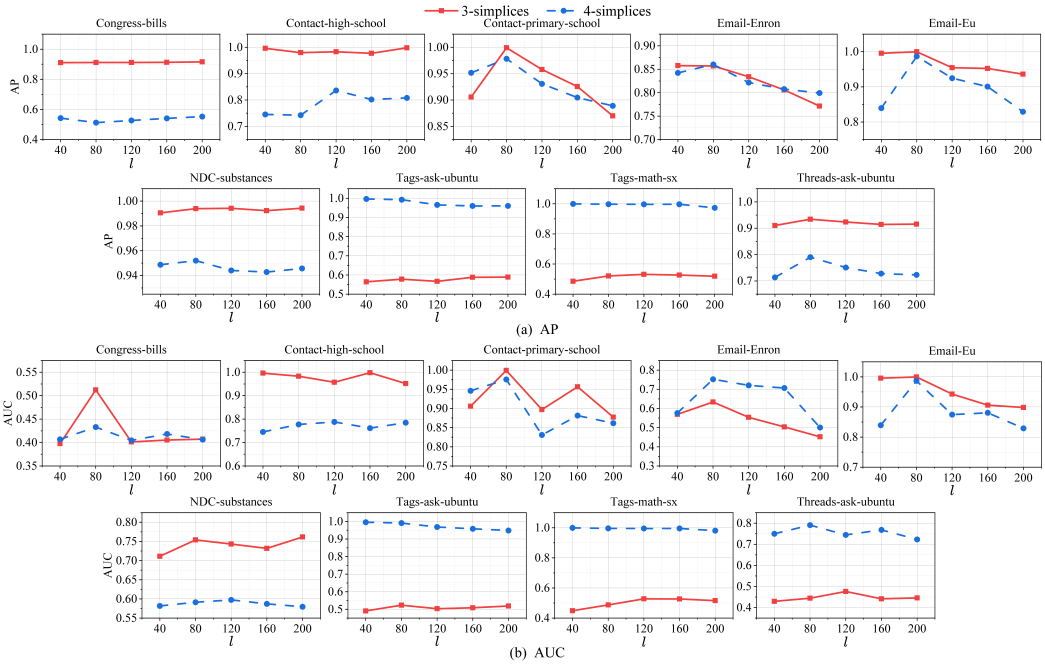

Fig. 4 The AP and AUC performances of TDE at different l. l ranges from 40 to 200, and the interval is 40.

Tags-math-sx, and Threads-ask-ubuntu datasets, the AP and AUC performances of TDE change relatively smoothly as p increases. This phenomenon suggests that in these datasets, TDE's performances are less sensitive to changes in p values. Higher-order interactions depend more on node roles, functions, and the events associated with higher-order interactions. As a result, TDE's performances, when modeling the capture of dynamic higher-order interactions through timestamps, are less affected by time factors, and the performance variation is minimal. Overall, TDE achieves the best performance around p=0.3 across most datasets, suggesting that a balanced consideration of both short-term and long-term interactions yields the most effective temporal modeling.

Fig. 4 presents the AP and AUC performances of TDE across nine datasets with two prediction orders (k=3 and k=4) and different values of l (from 40 to 200). By comparing the

Table 2 The AP performance of seven algorithms in nine datasets. The optimal results are shown in **bold** and the suboptimal values are underlined. k denotes the order of simplices to be predicted.

Orders	Datasets	Deepwalk	MM	HTGN	Transformer	HNE	RWTA	TDE
$k=3$	Congress-bills	0.9094	0.8994	0.7049	0.9300	**0.9324**	0.9312	0.9118
	Contact-high-school	0.5127	0.6685	0.9309	0.5030	0.2476	0.5721	**0.9797**
	Contact-primary-school	0.5473	0.6968	0.8286	0.5215	0.2513	0.4656	**0.9991**
	Email-Enron	0.7501	0.6246	0.7572	0.7047	0.7543	0.7142	**0.8570**
	Email-Eu	0.7109	0.5114	0.9260	0.9979	0.4238	0.6268	**0.9998**
	NDC-substances	0.9930	0.8968	0.9853	0.9712	0.9811	0.9902	**0.9939**
	Tags-ask-ubuntu	0.7752	0.8130	**0.8367**	0.5116	0.5608	0.6805	0.5780
	Tags-math-sx	0.8237	0.7542	**0.9532**	0.5282	0.5145	0.4913	0.5207
	Threads-ask-ubuntu	0.9294	0.6465	0.9293	0.9316	0.9321	0.9009	**0.9342**
$k=4$	Congress-bills	0.4437	**0.8114**	0.6632	0.6460	0.5955	0.5709	0.5123
	Contact-high-school	0.4736	0.6759	0.7235	0.6439	0.0099	0.6766	**0.7424**
	Contact-primary-school	0.5703	0.7519	0.6583	0.5332	0.0099	0.6121	**0.9782**
	Email-Enron	0.5272	0.7946	0.8384	0.4862	0.0267	0.6340	**0.8603**
	Email-Eu	0.7128	0.5995	0.9223	0.9803	0.1042	0.7304	**0.9867**
	NDC-substances	0.9460	0.9390	0.9356	0.8820	0.9299	0.8941	**0.9520**
	Tags-ask-ubuntu	0.6216	0.8524	0.9468	0.9782	0.0138	0.4340	**0.9927**
	Tags-math-sx	0.8065	0.7374	0.9466	0.4667	0.0084	0.5844	**0.9971**
	Threads-ask-ubuntu	0.5135	0.6532	0.7587	0.7083	0.2942	0.6930	**0.7979**

Table 3 The AUC performance of seven algorithms in nine datasets. The optimal results are shown in **bold** and the suboptimal values are underlined. k denotes the order of simplices to be predicted.

Orders	Datasets	Deepwalk	MM	HTGN	Transformer	HNE	RWTA	TDE
$k=3$	Congress-bills	0.4180	**0.8958**	0.6728	0.5142	0.5500	0.5428	0.3975
	Contact-high-school	0.5449	0.6239	0.9397	0.4589	0.4847	0.5965	**0.9830**
	Contact-primary-school	0.5889	0.6687	0.8653	0.5184	0.4904	0.4527	**0.9991**
	Email-Enron	0.5483	0.6221	0.6049	0.4550	0.5338	0.5124	**0.6336**
	Email-Eu	0.7621	0.5108	0.9160	0.9981	0.5715	0.6120	**0.9997**
	NDC-substances	0.7029	0.7125	0.6975	0.3012	0.4418	0.5893	**0.7539**
	Tags-ask-ubuntu	0.7240	0.8265	**0.9294**	0.4329	0.5006	0.6328	0.5235
	Tags-math-sx	0.8113	0.7742	**0.9490**	0.5096	0.5099	0.4673	0.5128
	Threads-ask-ubuntu	0.5170	0.6464	**0.9512**	0.5959	0.5970	0.4020	0.4444
$k=4$	Congress-bills	0.4081	**0.8009**	0.5928	0.6142	0.3591	0.5309	0.4330
	Contact-high-school	0.4586	0.6322	0.7372	0.6983	0.6158	0.7131	**0.7773**
	Contact-primary-school	0.6161	0.6694	0.6517	0.4866	0.5097	0.5679	**0.9756**
	Email-Enron	0.3925	0.7147	0.7419	0.3826	0.5127	0.5305	**0.7521**
	Email-Eu	0.7583	0.5734	0.9138	0.9850	0.6001	0.7261	**0.9902**
	NDC-substances	0.6564	0.9255	**0.9948**	0.2333	0.4871	0.3782	0.5912
	Tags-ask-ubuntu	0.6687	0.8791	0.9413	0.9728	0.4814	0.4074	**0.9909**
	Tags-math-sx	0.8175	0.7134	0.9407	0.4314	0.5012	0.5778	**0.9957**
	Threads-ask-ubuntu	0.4736	0.6530	0.7891	0.7236	0.6586	0.7819	**0.7909**

changes in AP and AUC performances across different l, we observe that the impact of l on TDE varies across datasets. In the Congress-bills network, the AP and AUC performances of TDE remain stable as l increases. This situation indicates that the walk length in the random walk has a negligible impact on TDE's performances, which are primarily influenced by the relationships between the bills and their sponsors. In contact networks (Contact-high-school and Contact-primary-school) and email networks (Email-Enron and Email-Eu), the AP and AUC performances of TDE tend to increase first and then decrease. In most cases, the performances of TDE are optimal when l=80. This phenomenon suggests that higher l makes TDE focus more on the global structural evolution of nodes and weakens the characteristics of local interactions, thus affecting the stability and performance of the model. In the NDC-substances network, the tagging networks (Tags-ask-ubuntu and Tags-math-sx) and

the Threads-ask-ubuntu network, TDE performs more smoothly with increasing l, suggesting there are more stable structural features in these networks. TDE can better capture the evolutionary characteristics of simplices so that the performances will not be affected too much by the change of l. Overall, TDE achieves the best performance in most datasets when the walk length l=80, indicating that this setting provides an effective balance between capturing local interactions and modeling global structural evolution.

Through the sensitivity analysis of parameters p and l, we find that TDE achieves better and more stable performance when the jump probability p and the walk length l are 0.3 and 80, respectively. Thus, we use p=0.3 and l=80 in subsequent experiments.

4.3. Performance comparisons

Table 2 presents the AP performance of seven algorithms across nine datasets under two prediction orders. From Table 2, TDE consistently outperforms baseline methods in most cases, especially in the contact and email networks (Contact-high-school, Contact-primary-school, Email-Enron, and Email-Eu), where higher-order interactions are frequent and exhibit strong temporal dependencies. Meanwhile, TDE achieves the highest AP scores of 0.9797 and 0.7424 (in Contact-high-school), 0.9991 and 0.9782 (in Contact-primary-school), 0.857 and 0.8603 (in Email-Enron), and 0.9998 and 0.9867 (in Email-Eu) for k=3 and k=4, significantly surpassing other algorithms. This means TDE effectively captures the spatiotemporal evolution of complex interactions by leveraging simplex-based temporal structure walk and time series embeddings. In NDC-substances, although the number of simplices is relatively small and the network is structurally stable, TDE still performs strongly, with AP scores of 0.9939 (k=3) and 0.9520 (k=4), demonstrating the ability to extract stable higher-order patterns. In the tagging networks (Tags-ask-ubuntu and Tags-math-sx), when k=3, HTGN achieves better AP scores than TDE, which is mainly due to HTGN's ability to dynamically prune according to hyperedge size and node interaction time and accurately capture active high-order structures. In contrast, TDE fails to effectively handle such complex dynamic high-order patterns. However, as the prediction order increases to k=4, latent higher-order structures become more pronounced over time, and TDE begins to capture the evolving interaction patterns of networks. This leads to a substantial performance boost, with AP scores rising to 0.9927 and 0.9971 in the Tags-ask-ubuntu and Tags-math-sx networks, respectively. In Threads-ask-ubuntu, TDE performs slightly better than other algorithms, with AP scores of 0.9342 at k=3 and 0.7979 at k=4, demonstrating TDE captures the dynamics information of network effectively. Although TDE shows competitive results in the Congress-bills dataset, it slightly lags behind MM, HTGN and HNE algorithms at k=4. This is because the fact that the interactions in the Congress-bills dataset are influenced more by the semantic than by the temporal dynamics. Nonetheless, TDE maintains stable performance and exhibits good generalizability across diverse network types.

Table 3 reports the AUC performance of seven algorithms across diverse datasets and prediction orders. TDE exhibits strong and consistent performance on eight datasets, underscoring its broad applicability and effectiveness in modeling temporal dynamics. In the contact and email networks, such as in the datasets of Contact-high-school, Contact-primary-school, Email-Enron, and Email-Eu, where higher-order interactions are frequent and temporally dependent, TDE significantly outperforms the other six baselines. Specifically, TDE achieves AUC scores of 0.9830 and 0.7773 (in Contact-high-school), 0.9991 and 0.9756 (in Contact-primary-school), and 0.9997 and 0.9902 (in Email-Eu) at k=3 and k=4, illustrating the capability to capture evolving higher-order interaction patterns. In the Congress-bills dataset, the formation of simplices shows weak temporal dependency, leading to poor AUC performance for TDE (0.3975 at k=3 and 0.4330 at k=4). These results are notably lower than MM and Transformer, which exploit global structural features to capture interaction patterns. Notably, HNE and RWTA models, which leverage simplex skeleton structures as their foundation, demonstrate superior performance. This observation implies that the formation of higher-order interactions exhibits stronger dependence on global topological configurations or domain-specific semantic contexts rather than temporal dynamics in Congress-bills, thereby limiting the efficacy of TDE's temporal-aware modeling paradigm. In tagging networks (Tags-ask-ubuntu and Tags-math-sx), TDE's AUC performance follows a trend similar to the AP results. Complex dynamic higher-order patterns make TDE performance poor. However, as the prediction order increases to k=4, higher-order dynamics become evident. TDE effectively captures these patterns, achieving enhanced performance with AUC values of 0.9909 and 0.9957 in Tags-ask-ubuntu and Tags-math-sx, respectively. Similarly, in the Threads-ask-ubuntu, the AUC performance of TDE improves from moderate performance (0.4444) at k=3 to 0.7909 at k=4, proving the ability of TDE in modeling complex temporal interaction patterns even in sparse user activity networks.

4.4. Running time comparison experiments

Table 4 presents a comparison of running times for various algorithms across different datasets and prediction orders. The results show that Deepwalk and Transformer, which rely solely on simple static network features, consistently achieve faster running times across most datasets and prediction tasks. In contrast, methods such as MM and HTGN exhibit significantly longer running times due to their higher model complexity, particularly on large-scale datasets like Tags-math-sx and Threads-ask-ubuntu, where the running time increases substantially. Specifically, HTGN employs complex hypergraph representations and dynamic computation mechanisms that enhance modeling capability but incur considerable computational overhead, leading to longer runtimes. MM utilizes an event recency-based weighting mechanism that offers relatively reasonable efficiency on some datasets; however, its comprehensive consideration of historical events increases computational complexity. Notably, the proposed TDE method maintains high predictive performance while achieving relatively low running times in most cases, especially demonstrating strong time efficiency and computational advantages in higher-order prediction tasks. Consequently, the TDE method, with its excellent trade-off between performance and

Table 4 The running time comparison (seconds) of seven algorithms in nine datasets. The optimal results are shown in **bold** and the suboptimal values are underlined. k denotes the order of simplices to be predicted.

Orders	Datasets	Deepwalk	MM	HTGN	Transformer	HNE	RWTA	TDE
$k = 3$	Congress-bills	373.4540	19063.4870	15796.6442	**42.7790**	2510.5895	2175.9315	1981.9080
	Contact-high-school	105.9615	971.7590	6186.4524	**33.4205**	713.1760	682.4155	497.5685
	Contact-primary-school	89.4195	1317.5735	6970.9279	**33.0875**	944.4070	780.8745	361.2175
	Email-Enron	33.8640	77.0735	406.4439	**3.1855**	73.8200	88.4490	100.1160
	Email-Eu	295.5030	2565.7670	13900.3760	**65.7025**	1467.0995	1470.7330	1504.8535
	NDC-substances	160.4035	1256.5860	2542.8866	**17.1735**	1859.1190	1851.9820	1728.2190
	Tags-ask-ubuntu	625.8780	27657.9350	263971.8397	**408.9470**	9040.3750	11185.7865	13793.4015
	Tags-math-sx	1007.0180	153156.9740	522525.6383	**200.1150**	8205.2955	8120.9190	8018.7830
	Threads-ask-ubuntu	20368.7850	17670.7935	206620.6647	**165.6610**	47712.6525	15782.3915	4972.9205
$k = 4$	Congress-bills	663.7535	46208.4905	4024.1524	**72.1060**	8926.6660	6825.7360	2627.5955
	Contact-high-school	180.1240	2069.0735	470.3140	**41.8665**	3117.8940	1740.8490	717.1890
	Contact-primary-school	156.8070	1406.7075	158.4867	**62.0530**	7668.0405	3655.7985	529.0900
	Email-Enron	64.3630	101.7855	497.4461	**5.4835**	158.7200	162.9395	159.7305
	Email-Eu	508.8750	3417.2845	4539.8393	**87.4995**	8686.2325	6030.6785	2076.7575
	NDC-substances	298.7015	1912.0695	1248.8547	**24.7305**	4173.2505	4512.4155	4673.5690
	Tags-ask-ubuntu	1243.4670	32136.2415	170579.6313	**994.2025**	531769.4965	231608.7935	21488.9665
	Tags-math-sx	1809.9220	99215.7265	207106.9421	**561.2045**	486140.3760	182054.3730	13196.9590
	Threads-ask-ubuntu	35536.2665	2819.7735	48490.0490	**780.2850**	68420.5110	38208.1235	12657.5580

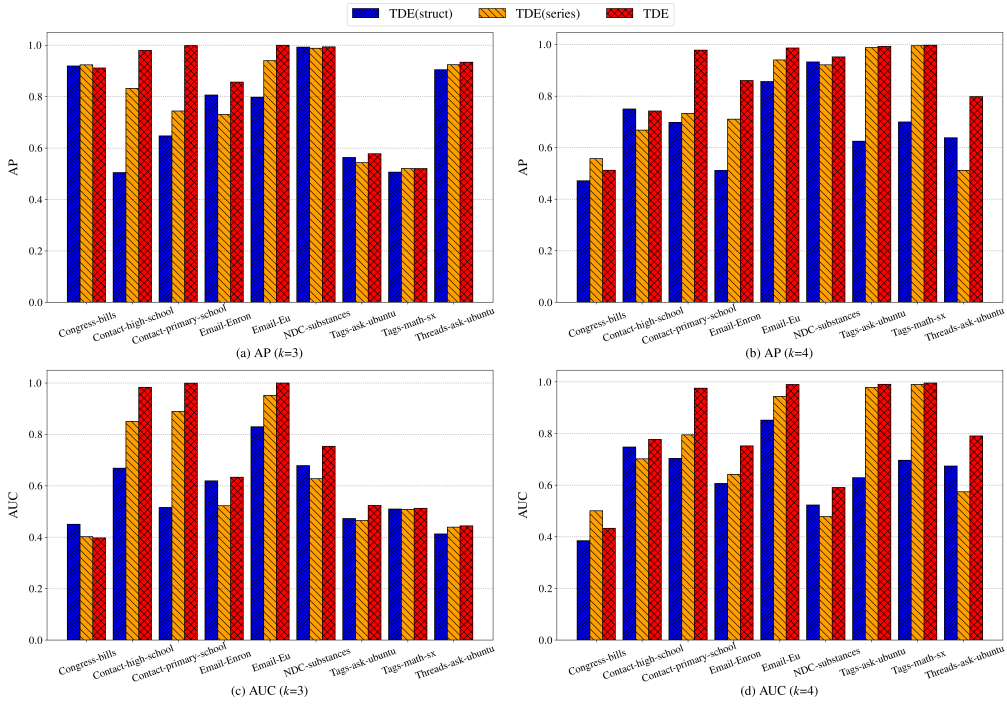

Fig. 5 Performance analysis of AP and AUC for ablation experiments across different TDE patterns on nine datasets at k=3 and k=4.

efficiency, shows great potential for large-scale deployment and real-time inference.

4.5. Ablation experiments

This section presents the ablation experiment of TDE to verify the impact of different modules on HLP. TDE (struct), TDE (series), and TDE represent the application of the temporal structure embedding module, the time series embedding module, and the combination of both features in the framework, respectively. By comparing the performances of all three on different datasets, we analyze the role and complementarity of each feature.

Fig. 5 presents the ablation experiment results of TDE on nine datasets. The results show that TDE (struct) and TDE (series) exhibit distinct performance differences across nine datasets. At the same time, the complete TDE model, integrating both modules, achieves the best AP and AUC scores in most cases. When k=3, as shown in Fig.5(a) and Fig.5(c), TDE generally outperforms both TDE (struct) and TDE (series) in terms of AP and AUC. Notably, in the Contact-primary-school network, the full TDE model achieves significant improvements over both variants, suggesting that the joint modeling of structural patterns and temporal behaviors effectively captures high-order interac-

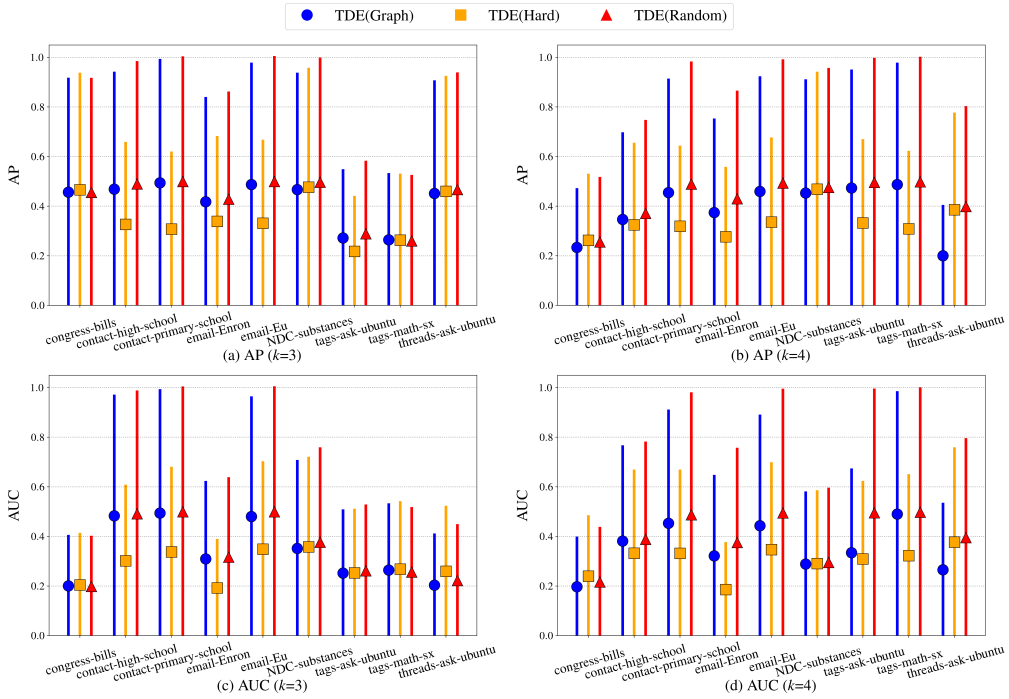

Fig. 6 Performance analysis of AP and AUC for negative sampling experiments across different TDE patterns on nine datasets at k=3 and k=4.

tions. In contrast, in the Congress-bills network, TDE (struct) slightly outperforms the full TDE under k=3, likely due to the dense distribution of 3-order collaborative structures where structural signals dominate. However, when the simplex order increases to k=4, as in Fig.5(b) and Fig.5(d), TDE surpasses both submodules, including in Congress-bills network, indicating that sparser and more complex higher-order interactions benefit more from temporal modeling. Across datasets such as Contact-high-school, Email-Enron, Email-Eu, and NDC-substances networks, TDE consistently outperforms both variants under all metrics and orders, highlighting its stable performance. In the Tags-ask-ubuntu and Tags-math-sx networks, the advantage of TDE becomes more evident at k=4, suggesting that higher-order semantic interactions require a stronger integration of temporal dynamics with topological structures. Furthermore, for Threads-ask-ubuntu, TDE shows the best performance under both AP and AUC, especially at k=4, due to its ability to capture long-range spatiotemporal dependencies in thread-based discussions. These results demonstrate the complementary nature of the two components: TDE (struct) emphasizes time-aware structural walks, while TDE (series) captures periodic behavioral trends of nodes. The attention-based feature fusion in TDE enables effective integration of both, leading to improved adaptability and robustness across network domains. In summary, the ablation study underscores the necessity of temporal-topological fusion in higher-order dynamic systems, with TDE showing superior generalization across interaction types, simplex orders, and application scenarios.

4.6. Negative sampling experiments

To evaluate the impact of different negative sampling strategies on model performance, this section integrates three types of negative sampling methods into the proposed TDE framework: topology-based sampling (Graph), hard negative sampling (Hard), and random negative sampling (Random), denoted as TDE(Graph), TDE(Hard), and TDE(Random), respectively. Specifically, the Graph sampling strategy generates negative samples based on potential closure patterns of higher-order structures in the network, thereby approximating realistic but unobserved higher-order interactions. The Hard sampling strategy selects the top-scoring negative simplices—those misclassified as positive by the model during training—as challenging negative examples, aiming to enhance the model's ability to distinguish ambiguous cases. In contrast, Random sampling selects negative examples uniformly at random from the pool of unobserved candidates, offering the simplest construction process and a balanced sample distribution.

Fig. 6 presents a performance comparison of these three sampling strategies under the TDE framework. TDE(Random) consistently demonstrates superior performance in terms of AP and AUC across most datasets, especially on Email-Enron, Email-Eu, Tags-ask-ubuntu, and Tags-math-sx, where it significantly outperforms the other two strategies for both k=3 and k=4 prediction tasks. These results suggest that although random sampling is relatively simple, its stable and uniform negative sample distribution is beneficial for enhancing the generalization capability of the model. In contrast, while TDE(Graph) is theoretically more aligned with structural characteristics, its performance is less stable on datasets such as Congress-bills and Contact-high-school, potentially due to structural bias introduced during sampling.

TDE(Hard) shows competitive performance on NDC-substances and Tags-math-sx, confirming the effectiveness of hard negative examples. However, its overall volatility may undermine training stability. In summary, although Graph and Hard sampling strategies offer more informative and challenging negative instances, Random sampling, due to its simplicity, stability, and balanced coverage, proves to be a more robust and generalizable choice for higher-order link prediction tasks across diverse network settings.

5. Conclusions and future works

This paper proposes a framework of TDE, providing an innovative and effective solution for HLP in higher-order networks. By leveraging simplex temporal structure walk, TDE captures and embeds dynamic higher-order interactions within the networks. TDE aggregates temporal behavior via a time series embedding module and uses attention-based feature fusion module to integrate temporal and topological features, generating robust embeddings to improve the prediction accuracy. Experimental results across nine real-world datasets demonstrate that TDE consistently outperforms baseline methods in most scenarios, validating its effectiveness and robustness in HLP.

Although TDE performs well in most higher-order networks, it faces limitations in datasets where higher-order interactions are less time-dependent. First, in datasets where the simplex formation depends on semantic information (e.g., bill content or tags), TDE may underperform due to the lack of semantic modeling. Therefore, future work should incorporate text embeddings to capture hidden semantic features. Second, we plan to strengthen the attention mechanism for datasets driven by static topology, improving the model's capacity to represent complex structural patterns. In this direction, the integration of semantic-aware representation learning strategies from attributed graph clustering can be beneficial. For instance, the recently proposed Fuzzy Clustering and Graph Convolution Network (FCGCN) [55] introduces a fuzzy clustering-guided GCN framework that explicitly optimizes clustering quality during representation learning. Inspired by this, we aim to design structure-sensitive regularization objectives that align embedding learning with downstream prediction quality, while mitigating over-smoothing and improving discriminative power for static higher-order structures. Lastly, to improve scalability and broaden applicability, we will extend the model to support the prediction of higher-order simplices beyond 3-simplices and 4-simplices, enabling it to operate in more complex interaction environments.

References

[1] Wan J, Ichinose G, Small M, Sayama H, Moreno Y, Cheng C. Multilayer networks with higher-order interaction reveal the impact of collective behavior on epidemic dynamics. Chaos, Solitons & Fractals, 2022, 164: 112735

[2] Zhang Y, Lucas M, Battiston F. Higher-order interactions shape collective dynamics differently in hypergraphs and simplicial complexes. Nature communications, 2023, 14(1): 1605

[3] Vasilyeva E, Kozlov A, Alfaro-Bittner K, Musatov D, Raigorodskii A, Perc M, Boccaletti S. Multilayer representation of collaboration networks with higher-order interactions. Scientific reports, 2021, 11(1): 5666

[4] Ramos J, Lopes R J, Marques P, Araújo D. Hypernetworks reveal compound variables that capture cooperative and competitive interactions in a soccer match. Frontiers in Psychology, 2017, 8: 1379

[5] Xia Y, Jiang H, Yu S, Yu Z. The dynamic analysis of the rumor spreading and behavior diffusion model with higher-order interactions. Communications in Nonlinear Science and Numerical Simulation, 2024, 138: 108186

[6] Wang D, Zhao Y, Leng H, Small M. A social communication model based on simplicial complexes. Physics Letters A, 2020, 384(35): 126895

[7] Martínez V, Berzal F, Cubero J C. A survey of link prediction in complex networks. ACM computing surveys (CSUR), 2016, 49(4): 1–33

[8] Boccaletti S, De Lellis P, Del Genio C, Alfaro-Bittner K, Criado R, Jalan S, Romance M. The structure and dynamics of networks with higher order interactions. Physics Reports, 2023, 1018: 1–64

[9] He Y J, Xu X K, Xiao J. Predicting higher order links in social interaction networks. IEEE Transactions on Computational Social Systems, 2023, 11(2): 2796–2806

[10] Zhu J, Zhu J, Ghosh S, Wu W, Yuan J. Social influence maximization in hypergraph in social networks. IEEE Transactions on Network Science and Engineering, 2018, 6(4): 801–811

[11] Harada S, Akita H, Tsubaki M, Baba Y, Takigawa I, Yamanishi Y, Kashima H. Dual graph convolutional neural network for predicting chemical networks. BMC bioinformatics, 2020, 21: 1–13

[12] Chen C, Surana A, Bloch A M, Rajapakse I. Controllability of hypergraphs. IEEE Transactions on Network Science and Engineering, 2021, 8(2): 1646–1657

[13] Choo H, Shin K. On the persistence of higher-order interactions in real-world hypergraphs. In: Proceedings of the 2022 SIAM International Conference on Data Mining (SDM). 2022, 163–171

[14] Chen C, Liu Y Y. A survey on hyperlink prediction. IEEE Transactions on Neural Networks and Learning Systems, 2023, 35(11): 15034–15050

[15] Cheng A, Xu Y, Sun P, Tian Y. A simplex path integral and a simplex renormalization group for high-order interactions. Reports on Progress in Physics, 2024, 87(8): 087601

[16] Zhao D, Li R, Peng H, Zhong M, Wang W. Higher-order percolation in simplicial complexes. Chaos, Solitons & Fractals, 2022, 155: 111701

[17] Wang K, Zhu Y, Wang X, Huang J, Cao T. Heterogeneous hypernetwork representation learning with hyperedge fusion. IEEE Transactions on Computational Social Systems, 2024, 11(6): 7646–7657

[18] Guan Y, Sun X, Sun Y. Sparse relation prediction based on hypergraph neural networks in online social networks. World Wide Web, 2023, 26(1): 7–31

[19] Chen Y, Wang X, Chen C. Hyperedge importance estimation via identity-aware hypergraph attention network. In: Proceedings of the 33rd ACM International Conference on Information and Knowledge Management. 2024, 334–343

[20] Battiloro C, Testa L, Giusti L, Sardellitti S, Lorenzo P D, Barbarossa S. Generalized simplicial attention neural networks. IEEE Transactions on Signal and Information Processing over Networks, 2024, 10: 833–850

[21] Chavan N, Potika K. Higher-order link prediction using triangle embeddings. In: 2020 IEEE International Conference on Big Data (Big Data). 2020, 4535–4544

[22] Grover A, Leskovec J. node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. 2016, 855–864

[23] Grohe M. word2vec, node2vec, graph2vec, x2vec: Towards a theory of vector embeddings of structured data. 2020, 1–16

[24] Wu Z, Pan S, Chen F, Long G, Zhang C, Yu P S. A comprehensive survey on graph neural networks. IEEE transactions on neural networks and learning systems, 2020, 32(1): 4–24

[25] Zhang M, Xu B, Wang L. Dynamic network link prediction based on random walking and time aggregation. International Journal of Machine Learning and Cybernetics, 2023, 14(8): 2867–2875

[26] Kumar T, Darwin K, Parthasarathy S, Ravindran B. Hpra: Hyperedge prediction using resource allocation. In: Proceedings of the 12th ACM conference on web science. 2020, 135–143

[27] Yoon S e, Song H, Shin K, Yi Y. How much and when do we need higher-order information in hypergraphs? a case study on hyperedge prediction. In: Proceedings of The Web Conference 2020. 2020, 2627–2633

[28] Benson A R, Abebe R, Schaub M T, Jadbabaie A, Kleinberg J. Simplicial closure and higher-order link prediction. Proceedings of the National

Academy of Sciences, 2018, 115(48): E11221–E11230
[29] Jung-Muller M, Ceria A, Wang H. Higher-order temporal network prediction. In: International Conference on Complex Networks and Their Applications. 2023, 461–472
[30] Ling Z, Li Y, Zhang Y, Yu K, Zhou P, Li B, Wu X. A light causal feature selection approach to high-dimensional data. IEEE Transactions on Knowledge and Data Engineering, 2022, 35(8): 7639–7650
[31] Ling Z, Yu K, Zhang Y, Liu L, Li J. Causal learner: A toolbox for causal structure and markov blanket learning. Pattern Recognition Letters, 2022, 163: 92–95
[32] Hu L, Zhang J, Pan X, Yan H, You Z H. Hiscf: leveraging higher-order structures for clustering analysis in biological networks. Bioinformatics, 2021, 37(4): 542–550
[33] Zhang M, Cui Z, Oyetunde T, Tang Y, Chen Y. Recovering metabolic networks using a novel hyperlink prediction method. arXiv preprint arXiv:1610.06941, 2016
[34] Oyetunde T, Zhang M, Chen Y, Tang Y, Lo C. Boostgapfill: improving the fidelity of metabolic network reconstructions through integrated constraint and pattern-based methods. Bioinformatics, 2017, 33(4): 608–611
[35] Maurya D, Ravindran B. Hyperedge prediction using tensor eigenvalue decomposition. Journal of the Indian Institute of Science, 2021, 101: 443–453
[36] Zhou D, Huang J, Schölkopf B. Learning with hypergraphs: Clustering, classification, and embedding. Advances in neural information processing systems, 2006, 19
[37] He Y, Yang Y, Su X, Zhao B, Xiong S, Hu L. Incorporating higher order network structures to improve mirna–disease association prediction based on functional modularity. Briefings in Bioinformatics, 2023, 24(1)
[38] Yang Y, Li X, Guan Y, Wang H, Kong C, Jiang J. Lhp: Logical hypergraph link prediction. Expert Systems with Applications, 2023, 222: 119842
[39] Li J, Guo X, Jiao P, Wang W. Learning accurate neighborhood-and self-information for higher-order relation prediction in heterogeneous information networks. Neurocomputing, 2025, 613: 128739
[40] Zhao Z, Yang K, Guo J. Link prediction with hypergraphs via network embedding. Applied Sciences, 2022, 13(1): 523
[41] Wang Z, Chen J, Gong M, Shao Z. Higher-order neurodynamical equation for simplex prediction. Neural Networks, 2024, 173: 106185
[42] Liu J, Hua Z, Xie Y, Li B, Shomer H, Song Y, Hassani K, Tang J. Higher-order structure boosts link prediction on temporal graphs. arXiv preprint arXiv:2505.15746, 2025
[43] Zhao B W, Wang L, Hu P W, Wong L, Su X R, Wang B Q, You Z H, Hu L. Fusing higher and lower-order biological information for drug repositioning via graph representation learning. IEEE Transactions on Emerging Topics in Computing, 2023, 12(1): 163–176
[44] Liang Y, He F, Zeng X. 3d mesh simplification with feature preservation based on whale optimization algorithm and differential evolution. Integrated Computer-Aided Engineering, 2020, 27(4): 417–435
[45] Mikolov T, Sutskever I, Chen K, Corrado G S, Dean J. Distributed representations of words and phrases and their compositionality. Advances in neural information processing systems, 2013, 26
[46] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser Ł, Polosukhin I. Attention is all you need. Advances in neural information processing systems, 2017, 30
[47] Fowler J H. Legislative cosponsorship networks in the us house and senate. Social networks, 2006, 28(4): 454–465
[48] Mastrandrea R, Fournet J, Barrat A. Contact patterns in a high school: a comparison between data collected using wearable sensors, contact diaries and friendship surveys. PloS one, 2015, 10(9): e0136497
[49] Stehlé J, Voirin N, Barrat A, Cattuto C, Isella L, Pinton J F, Quaggiotto M, Broeck V. d W, Régis C, Lina B, others . High-resolution measurements of face-to-face contact patterns in a primary school. PloS one, 2011, 6(8): e23176
[50] Klimt B, Yang Y. The enron corpus: A new dataset for email classification research. In: European conference on machine learning. 2004, 217–226
[51] Paranjape A, Benson A R, Leskovec J. Motifs in temporal networks. In: Proceedings of the tenth ACM international conference on web search and data mining. 2017, 601–610
[52] Perozzi B, Al-Rfou R, Skiena S. Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. 2014, 701–710
[53] Everingham M, Van Gool L, Williams C K, Winn J, Zisserman A. The pascal visual object classes (voc) challenge. International journal of computer vision, 2010, 88: 303–338
[54] Bradley A P. The use of the area under the roc curve in the evaluation of machine learning algorithms. Pattern recognition, 1997, 30(7): 1145–1159
[55] Yang Y, Li G, Li D, Zhang J, Hu P, Hu L. Integrating fuzzy clustering and graph convolution network to accurately identify clusters from attributed graph. IEEE Transactions on Network Science and Engineering, 2024