A Appendix

A.1 Pseudocode

Algorithm 1: The T-GNNExplainer algorithm.

```
Input  : target model $f$, temporal graph $\mathcal{G}^k$, target event $e_k$, sparsity $q$, navigator $h_\theta$, rollout $N$.
Output: the best subset $\mathcal{N}^*$ satisfying sparsity threshold $q$.
1  Initialize tree $\mathcal{T} = \{\mathcal{G}^k\}$;
2  for $i = 1, \cdots, N$ do
3      $CurrNode$ = $\mathcal{G}^k$;
4      $PathNodes$ = [$CurrNode$];
       /* move downwardly if the current node is not a leaf.       */
5      while $|CurrNode|/|\mathcal{G}^k| > q$ do
6          if $CurrNode$ is expandable then
               /* expand one child using the node expansion strategy
                  with the help of the navigator.                   */
7              $Child$ = NodeExpansionStrategy($CurrNode$);
8              $CurrNode.children$.append($Child$);
9              $\mathcal{T}$.append($Child$);
           /* select one child using the node selection strategy.   */
10         $CurrNode$ = NodeSelectionStrategy($CurrNode.children$);
11         $PathNodes$.append($CurrNode$);
12     end
13     $LeafNode$=$CurrNode$;
14     $LeafNode.r$ = RewardFunction($LeafNode, e_k$);
       /* update path nodes' statistics using the reward.           */
15     UpdateStatistics($PathNodes$, $LeafNode.r$);
16 end
   /* find the best node having the largest reward and satisfying
      the sparsity criteria $|Node|/|\mathcal{G}^k| \leq q$.                    */
17 $\mathcal{N}^*$ = FindBestTreeNode($\mathcal{T}$, $q$);
18 return $\mathcal{N}^*$;
```

A.2 Synthetic Dataset

Hawkes process: The multivariate Hawkes process (MHP) is the counting process where an arrival of an event can affect the arrival rates of other events. In our synthetic datasets, we use a MHP to capture mutual excitation/inhibition and generate a sequence of events with timestamps.

We assume that there are D event types where the type of an event (node n_u, node n_v, timestamp t) is decided by its nodes n_u and n_v. The intensity function of the i-th type of events at time t is

$$\lambda_i(t) = \mu_i + \sum_{j=1}^{D} \int_0^t \varphi_{ij}(t-s)\mathrm{d}N_j(s) \tag{6}$$

where $\varphi_{ij}(x) = A_{ij}\theta e^{-\theta x}$.

The first term μ_i indicates the endogenous intensity of event type i, while the second term φ_{ij} represents the exogenous influence from other events. In particular, $N_j(s)$ counts the number of occurred event type j within $[0, s]$, and one arrival of event type j at time s will affect the intensity of event type i at time t by the amount $\varphi_{ij}(t-s)$ for $t > s$. A_{ij} is the influence matrix and θ is a time decay factor. Obviously, we could pre-define the parameters μ_i, A_{ij} and θ to decide a MHP.

We show our parameter setting in Fig. 2, where we have 4 types of events $E_0 - E_3$. For two synthetic datasets, we set $\mu_0 = \mu_1 = 0.5$ and $\mu_2 = \mu_3 = 0$. In Synthetic v1, we set $A_{20} = 1$, $A_{21} = -1$, and $A_{32} = 1$. Other A_{ij} are 0. Therefore, E_0 and E_2 work together to let E_3 happen, while E_1 inhibits

E_3. In Synthetic v2, we set $A_{33} = -2$ on the basis of Synthetic v1. The occurrence of E_3 inhibits itself.

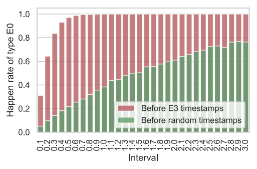
We utilize *tick.hawkes.SimuHawkesExpKernels* in the tick library to generate a sequence of events. We set time decay θ to 10, control the total simulation time as 10000, and then generate ~ 10000 events with timestamps for each synthetic dataset. The node features are created randomly, and the event feature is obtained by adding its ending node features.

Dataset statistics: As we focus on the prediction of E_3, we illustrate the statistics of E_0, E_1, E_2, and E_3 before E_3 happens. Specifically, we compare occurrence rates of all event types before an E_3 timestamp to those before a random timestamp. The x axis in Fig. 7 and Fig. 8 indicates selected intervals from 0.1 to 3.0. While the y axis indicates the happening rate of a specific event type in that interval before an E_3 or a random timestamp.

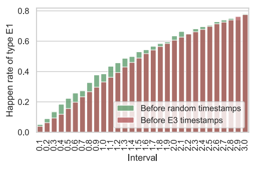
Take the Fig. 7(a) as an example, assuming the interval is 0.3, an E_0 event will happen with probability $\sim 82\%$ in the previous 0.3 time interval before an E_3 event, while the probability before a random timestamp is merely $\sim 16\%$. The other sub-figures in Fig. 7 and Fig. 8 illustrate happening rates of other event types.

We can find that the happening rates of E_0 and E_2 are high before E_3 timestamps, especially when the time interval is small. It is the same as our setting, where E_0 and E_2 act as positive stimulus to E_3. Moreover, the happening rate of E_1 before E_3 is even smaller than that before random timestamps, indicating that E_1 will suppress the happening of E_3. Comparing Fig. 7(d) with Fig. 8(d), we could also conclude that in the Synthetic v2, E_3 itself will suppress E_3 because the happening rate of E_3 before E_3 in Fig. 8(d) is clearly smaller than that in Fig. 7(d).

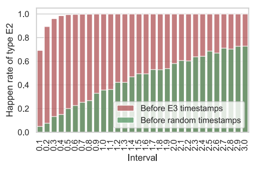
In both Fig. 7 and Fig. 8, the distributions of previous events' happening rates are different for E_3 timestamps and random ones. Hence these signals are captured and utilized by target models to predict whether an E_3 will happen given previous events.



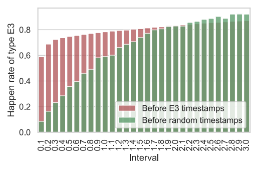
(a) E_0



(b) E_1

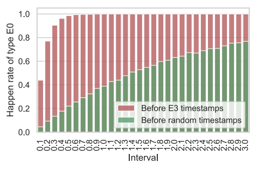


(c) E_2

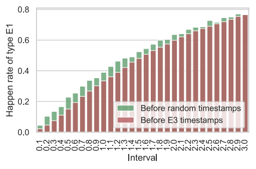


(d) E_3

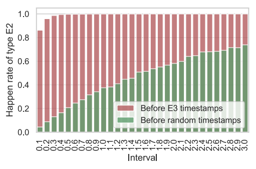
Figure 7: Statistics of the Synthetic v1.



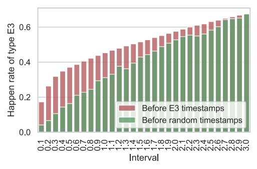
(a) E_0



(b) E_1



(c) E_2



(d) E_3

Figure 8: Statistics of the Synthetic v2.

A.3 Details of Target Models

We list all the hyper-parameters of target models in Table 3. Note that *N degree* indicates the number of neighbour temporal events used for information aggregation in the message passing.

Besides, we also present average precision (AP) of well-trained target models on all datasets in Table 4. The results are averaged over three runs.

Table 3: Hyper-parameters of both models for all datasets

	TGAT		TGN	
	Real-world	Synthetic	Real-world	Synthetic
Hidden dimension	172	4	172	4
Aggregation layers	2	2	2	2
Attention heads	2	2	2	2
N degree	10	10	10	10
Memory dimension	-	-	172	4
Time dimension	172	4	172	4
Node feature dimension	172(zeros)	4	172(zeros)	4
Edge feature dimension	172	4	172	4
Training epoch	10	100	10	100
Learning rate	1e-4	1e-4	1e-4	1e-4

Table 4: Models' average precision (AP) for the inductive event prediction on all datasets.

	Wikipedia	Reddit	Simulate v1	Simulate v2
TGAT	0.9791(0.0000)	0.9750(0.0000)	0.9632(0.0005)	0.9641(0.0000)
TGN	0.9851(0.0000)	0.9664(0.0000)	0.9535(0.0000)	0.9687(0.0000)

A.4 Efficiency comparison with other baselines.

In this section, we compare the running time of all the methods for searching for a solution satisfying a fidelity threshold. The fidelity threshold is identical to that in Sec. 5.4.

The results w.r.t TGAT and TGN are listed in Table 5 and Table 6 respectively. Both tables indicate that T-GNNExplainer with navigator is much faster than T-GNNExplainer without navigator. The running time of T-GNNExplainer with navigator is acceptable in most cases. All the non-search based baselines achieve high efficiency because they conduct a one-pass inference or simply average internal weights. Even though T-GNNExplainer is slower than non-search based baselines, the AUFSC is 86% higher than the leading baseline averaged on all the datasets and models. Moreover, some baselines require retraining on new datasets, e.g., PG, which could be time-consuming once the datasets are large. Hence, considering the explanation quality and intrinsic characteristic of searching, the efficiency of T-GNNExplainer is reasonable and acceptable.

Table 5: Running time comparison of different methods on all the datasets for explaining an instance with TGAT. † indicates withholding the navigator.

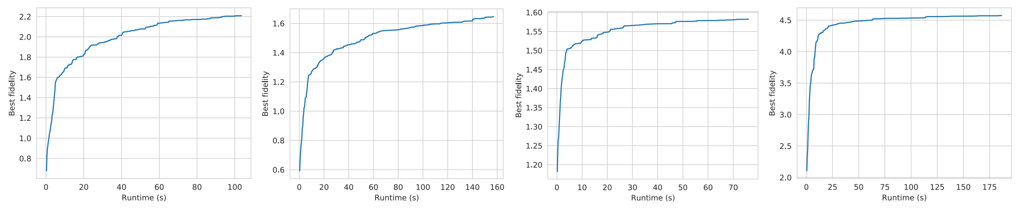
	Methods/Time (s)	Wikipedia	Reddit	Synthetic v1	Synthetic v2
non-search based	ATTN	0.05	0.17	0.05	0.05
	PBONE	0.31	0.39	0.23	0.25
	PG	0.03	0.22	0.03	0.03
search based	T-GNNExplainer†	68.14	158.2	89.74	178.2
	T-GNNExplainer	20.38	28.2	14.49	12.5

Table 6: Running time comparison of different methods on all the datasets for explaining an instance with TGN. † indicates withholding the navigator.

	Methods/Time (s)	Wikipedia	Reddit	Synthetic v1	Synthetic v2
non-search based	ATTN	0.04	0.16	0.03	0.03
	PBONE	0.17	0.31	0.14	0.17
	PG	0.03	0.14	0.10	0.09
search based	T-GNNExplainer†	20.50	40.26	31.95	56.10
	T-GNNExplainer	14.74	8.66	18.00	22.14

Moreover, we can deduce the complexity of our method. Since MCTS is an anytime algorithm, i.e., it can stop at any time based on the rollout limitation, the complexity is $\mathcal{O}(NDC)$. N is the number of rollouts, D is the expansion depth of each rollout determined by the sparsity threshold, and C is a constant including inference time of the navigator, storing time of tree nodes, and other constant-time operations.

We plot runtime-fidelity curves of T-GNNExplainer for TGAT in Fig. 9 to better illustrate the trade-off between efficiency and solutions' quality. Fig. 9 reveals that the best fidelity of found solutions increases steeply at the beginning of searching and the marginal gain decreases with the increase of rollouts in all cases. It means that the method could find a reasonable solution without many rollouts. Hence, in practice, we could use a relatively small number (e.g., [100, 200]) to balance the efficiency and expected solution quality.

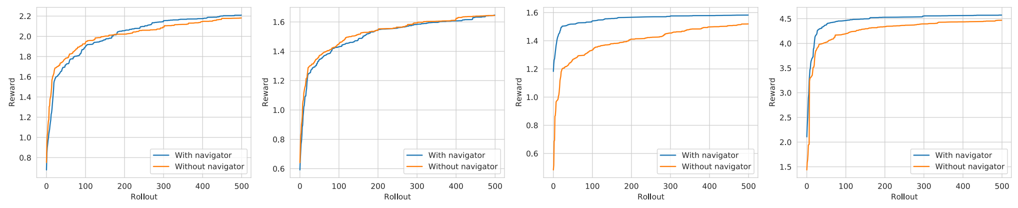


(a) TGAT, Wikipedia (b) TGAT, Reddit (c) TGAT, Simulate v1 (d) TGAT, Simulate v2

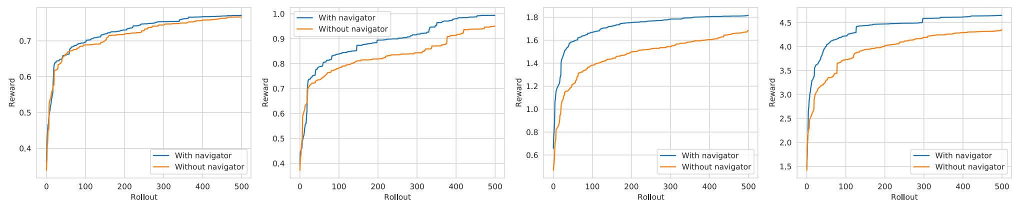
Figure 9: Runtime-fidelity tradeoff of TGAT on all the datasets.

A.5 Rollout-Reward comparison with or without the navigator

In this section, we investigate the effect of the navigator using reward-rollout curves for both target models on all datasets. The reward-rollout curve reflects the best solution's quality difference within a specific rollout limitation. Results in Fig. 10 show that *with navigator* outperforms *without navigator* in most cases, except for the TGAT&Reddit scenario, in which the target model and the navigator may not be trained well because of model capacity and the noisy characteristic of the Reddit dataset. The performance gap between *with navigator* and *without navigator* becomes more significant on synthetic datasets. Since target models achieve better prediction performance, the navigator can be trained more satisfactorily to capture events' importance as well.



(a) TGAT, Wikipedia (b) TGAT, Reddit (c) TGAT, Simulate v1 (d) TGAT, Simulate v2

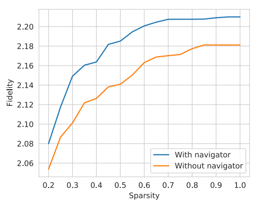


(e) TGN, Wikipedia (f) TGN, Reddit (g) TGN, Simulate v1 (h) TGN, Simulate v2

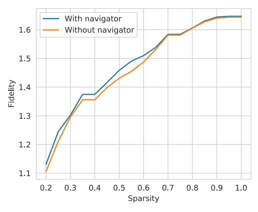
Figure 10: Reward-rollout comparison w/wo the navigator.

A.6 Fidelity-sparsity comparison with or without the navigator

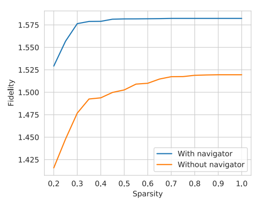
In this section, we plot fidelity-sparsity curves to compare *with navigator* and *without navigator* from a different perspective. The fidelity-sparsity comparison reveals the best solution's quality difference under a specific sparsity threshold after the search terminates. In Fig. 11, we find that the fidelity gaps are generally consistent with those in Fig. 10, i.e., *with navigator* could achieve a higher fidelity than *without navigator* under a given sparsity threshold in most cases. We conclude Fig. 11 and Fig. 10 that the navigator could not only accelerate the search process, but also boost the quality of solutions under the limitation of rollouts and sparsity.



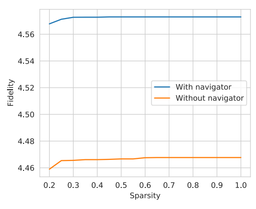
(a) TGAT, Wikipedia



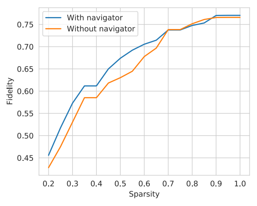
(b) TGAT, Reddit



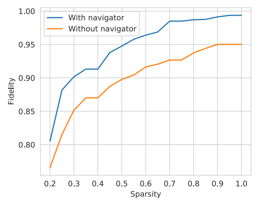
(c) TGAT, Simulate v1



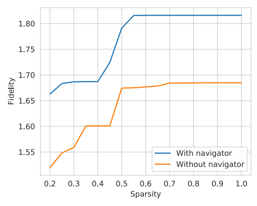
(d) TGAT, Simulate v2



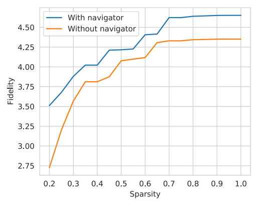
(e) TGN, Wikipedia



(f) TGN, Reddit



(g) TGN, Simulate v1

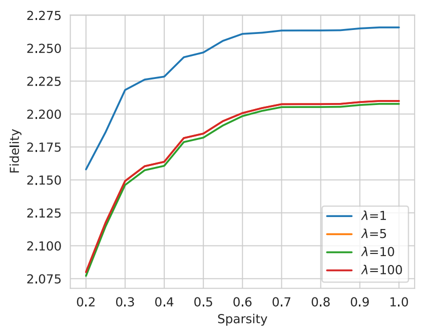


(h) TGN, Simulate v2

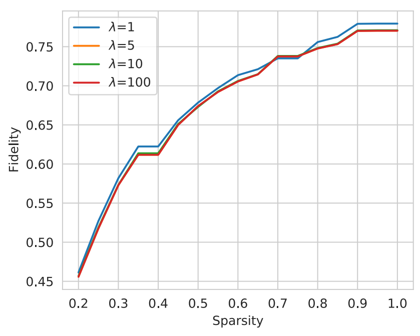
Figure 11: Fidelity-sparsity comparison w/wo the navigator.

A.7 Hyperparameter analysis

In this section, we investigate the effect of the hyperparameter λ in Eq. 4. λ balances the exploitation and exploration in the search process, hence it influences searched solutions' quality as well. We conduct the experiment with both target models on the Wikipedia dataset, and λ is set to 1, 5, 10, and 100, respectively. The results are shown in Fig. 12. We find that a smaller λ is slightly better than a larger one in both scenarios, indicating that more exploitation is preferred in T-GNNExplainer because of the existence of the navigator. However, the absolute difference is insignificant compared with the fidelity scale. In practice, we can set the λ in the range $[1, 10]$ for better performance.



(a) TGAT, Wikipedia



(b) TGN, Wikipedia

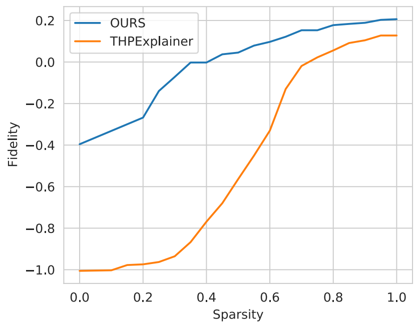
Figure 12: Hyperparameter analysis of λ for TGAT and TGN on Wikipedia.

A.8 Comparison with a self-interpretable baseline

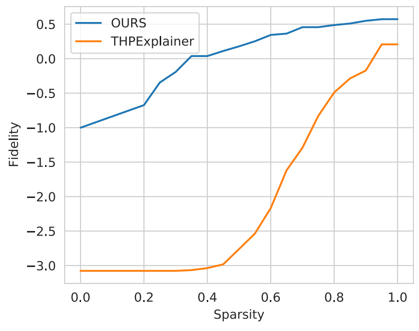
In this section, we compare T-GNNExplainer with a self-interpretable model named Transformer Hawkes Process (THP) Zuo et al. (2020), to investigate the proposed post-hoc explainer's explanation quality with a non-post-hoc one. The THP is based on the point process framework which utilizes a conditional intensity function to model previous events' influence on future ones. The THP adopts a transformer to learn a neural conditional intensity function $\Lambda_\theta(i, j, \delta t)$, where θ is the parameter set, i and j indicate event types, and δt represents a time interval. $\Lambda_\theta(i, j, \delta t)$ models the stimulus intensity of a past event with type i with time interval δt to the happening of event type j at the current timestamp. Since THP considers all event types' correlations, it cannot scale to real-world temporal graphs with tens of thousands of edges (i.e., types). We train and compare with THP on two synthetic datasets. The THP is trained to predict the next event's type and happening time. After training, we can compute each past event's $\Lambda_\theta(\cdot, \cdot, \cdot)$ as a score for interpretation. We denote the intrinsic interpretation method of THP as THPExplainer. Moreover, we regard the THP as a target model to be explained and use the prediction logits for the target event type E_3 to compute reward and run T-GNNExplainer. We compare the fidelity-sparsity curves of T-GNNExplainer and THPExplainer in Fig. 13, and compute Best Fid and AUFSC in Table 7. We find that T-GNNExplainer outperforms THPExplainer on all sparsity thresholds and the improvement is more significant for a smaller sparsity. From both Table 1 and Fig. 13, we could conclude that a search-based method could generally find solutions superior to conditional intensity scores or attentions. These scores are also sensitive to the model's performance. The performance may influence explanation quality as well. For example, THP has about 70% accuracy on simulated datasets because it models both time intervals and event types, which is challenging. TGAT/TGN achieves about 90% accuracy since they only model a binary task. More importantly, these scores may not completely and accurately reflect the decision logic of a complicated neural model. Clarifying the relationship between attention mechanism or conditional intensity scores and model-level interpretation may deserve further investigation.

Table 7: Best fidelity (↑) and AUFSC (↑) achieved by T-GNNExplainer and THPExplainer on synthetic datasets.

	Synthetic v1		Synthetic v2	
	Best Fid	AUFSC	Best Fid	AUFSC
THPExplainer	0.127	-0.485	0.207	-2.046
T-GNNExplainer	0.206	-0.006	0.573	0.021



(a) Simulate v1



(b) Simulate v2

Figure 13: Fidelity-sparsity comparison between T-GNNExplainer and THPExplainer on two synthetic datasets.