R4-Reviewer-N4rj

W1:

W1: Some descriptions are not clear enough in this paper, see D1.

Response-to-W1:

Thanks for your comment. Please find the corresponding response to D1.

W2:

W2: There are some concerns about the experiments in this paper, see D2, D3.

Response-to-W2:

Thanks for your comment. Please find the corresponding responses to D2 and D3.

W3:

W3: The experiments of this paper can be improved, see D4, D5.

Response-to-W3:

Thanks for your comment. Please find the corresponding responses to D4 and D5.

Q1:

D1: The authors introduce their graph learning module in Section 4.2. However, they do not provide clear instructions on how to integrate the outputs with the trajectory learning in Section 4.3.

Response-to-Q1:

The collaborative transition graph learning module, as mentioned in Section 4.2, learns POI embeddings and timeslot rotations. Our ROTAN model, illustrated in Figure 4, comprises three modules: Firstly, we pre-train a collaborative transition graph mentioned in Section 4.2 to obtain the POI embeddings and timeslot rotations. Secondly, we utilize the pre-trained POI embeddings and timeslot rotations in the trajectory learning module mentioned in Section 4.3. Finally, we recommend time-specific next POIs by using the trajectory representations and input target time in Section 4.4. In this framework, the learned POI embeddings and timeslot rotations, extracted from the transition graph, are used in the following trajectory learning module. In the ROTAN framework, we utilize two independent transformers to capture the different temporal patterns for users and POIs (i.e., Section 4.3.1 and Section 4.3.2). Hence, the embeddings are used in Eq. (4) and Eq. (9), respectively. Note that their parameters will be updated during the training phases.

Q2:

D2: In Section 5.1.1, the authors propose to use the preprocessing method of [46], but the statistical information presented in this paper is inconsistent with that of [46].

Response-to-Q2:

Yes, we use the dataset and preprocessing method described in [46]. The statistical analysis presented in [46] was conducted on the raw data. However, our statistical analysis is based on the preprocessed data, which may result in some differences. We follow the preprocessing methods outlined in [46] to obtain our preprocessed data. Hence, the statistical information obtained from our preprocessed data is accurate and aligns with the paper [46].

Table1. The raw data and preprocessed data statistics of datasets in [46].

		NYC	TKY	CA
Raw data	#Users	1,048	2,282	3,957
	#POIs	4,981	7,833	9,690
	#Check-ins	103,941	405,000	238,369
Preprocessed data	#Users	1,047	2,281	3,951
	#POIs	4,937	7,821	9,670
	#Check-ins	80,166	306,345	168,922

Q3:

D3: In Section 5.3, the results of the ablation experiments indicate a significant decline in performance when ROTAN does not consider the target time. These results are lower than those reported in [46]. This may suggest that the model's performance heavily relies on the target time, but it may not be effective enough when only learning from historical information.

Response-to-Q3:

Different from conventional methods (e.g., GetNext and AGRAN), STHGCN uses the hypergraph to capture trajectory similarities and correlations. Note that our work and all the compared baselines do not utilize trajectory-level information. To make a fair comparison, we do not include paper [2] in our manuscript.

To address the reviewer's concern, we provide the performance of ROTAN without using target time in the following tables. We observe that ROTAN (w/o Target Time) still significantly outperforms all evaluated baselines (in Section 5.1.2), indicating the effectiveness of our model when only learning from historical information. Moreover, while the performance of ROTAN (w/o Target Time) is worse than STHGCN on NYC and TKY datasets, its performance on CA is comparable to STHGCN. This is because STHGCN additionally exploits trajectory similarity information.

While paper [2] serves as a strong baseline, it employs spatiotemporal hypergraph learning for the next POI recommendation. It heavily relies on higher-order information in user trajectories and collaborative relations among trajectories. In fact, the performance of paper [2] is significantly better than the RNN/transformer-based baselines (Section 5.1.2), since it additionally utilizes trajectory-level information. A trajectory-level hypergraph is first constructed, and then a hypergraph transformer is utilized to make the next POI recommendation. Although the performance is excellent, the process is complex and computationally expensive.

Table2. The results of baselines and ROTAN (with/without target time) on NYC.

Models	**Acc@1**	**Acc@5**	**Acc@10**	**MRR**
STAN	0.2231	0.4582	0.5734	0.3253
GETNext	0.2406	0.4815	0.5811	0.3528
CLSPRec	0.1784	0.3830	0.4591	0.2691
AGRAN	0.2121	0.4519	0.5529	0.3179
STHGCN[46]	0.2702	0.5337	0.6119	0.3889
ROTAN(w/o target time)	0.2499	0.5007	0.5943	0.3634
TPG	0.2555	0.5005	0.5932	0.3669

ROTAN	0.3106	0.5281	0.6131	0.4104

Table3. The results of baselines and ROTAN (with/without target time) on TKY.

Models	**Acc@1**	**Acc@5**	**Acc@10**	**MRR**
STAN	0.1963	0.3798	0.4464	0.2852
GETNext	0.1829	0.4045	0.4961	0.2853
CLSPRec	0.1453	0.3394	0.4106	0.2340
AGRAN	0.1428	0.3737	0.4605	0.2471
STHGCN[46]	0.2947	0.5161	0.5922	0.3964
ROTAN(w/o target tIme)	0.2291	0.4531	0.5469	0.3345
TPG	0.1420	0.3631	0.4492	0.2436
ROTAN	0.2458	0.4626	0.5392	0.3475

Table4. The results of baselines and ROTAN (with/without target time) on CA.

Models	**Acc@1**	**Acc@5**	**Acc@10**	**MRR**
STAN	0.1104	0.2348	0.3018	0.1869
GETNext	0.1526	0.3278	0.3946	0.2364
CLSPRec	0.0891	0.1815	0.2013	0.1302
AGRAN	0.1199	0.3148	0.4017	0.2140
STHGCN[46]	0.1683	0.3406	0.4179	0.2512
ROTAN(w/o target tIme)	0.1685	0.3523	0.4381	0.2535
TPG	0.1749	0.3285	0.3860	0.2479
ROTAN	0.2199	0.3718	0.4334	0.2931

Q4:

D4: As mentioned in D2, this paper employs the preprocessing method from [46], so [46] should also be used as a baseline method for experiments. Additionally, this paper lacks sufficient experiments. To address this, the authors could refer to the experiments conducted in [46, 49], e.g., they can conduct experiments on users with varying levels of activity and trajectory lengths.

Response-to-Q4:

Different from conventional methods (e.g., GetNext and AGRAN), STHGCN [46] uses hypergraphs to capture trajectory similarities and correlations. Note that our work and all compared baselines do not utilize trajectory-level information. To ensure a fair comparison, we do not include paper [2] in our manuscript. Additionally, STHGCN [46] is complex and computationally expensive, with a running time much larger than previous RNN/Transformer methods.

We provide additional experimental results for users with varying levels of activity and trajectory lengths in the following tables. Following [46, 49], we divided users/trajectories into different groups and evaluated their performances. Furthermore, we reproduce TPG [25] under the same experimental setup, and the results are presented in the tables. We can see that the performance of ROTAN is significantly better than GetNext [49], and STHGCN [46]. and TPG in most cases. These experimental results indicate that ROTAN performs well for both active and normal user groups. It outperforms other baselines for different kinds of trajectories.

Table5. The results of users with varying levels of activity on NYC.

User Groups	**Models**	**Acc@1**	**Acc@5**	**Acc@10**	**MRR**
Very active	GETNext	0.2692	0.5639	0.6995	0.3933
Normal	GETNext	0.2421	0.4739	0.5422	0.4254
Inactive	GETNext	0.1224	0.3471	0.4394	0.2319
Very active	STHGCN	0.3085	-	-	0.4402
Normal	STHGCN	0.3050	-	-	0.4265
Inactive	STHGCN	0.1460	-	-	0.3933
Very active	TPG	0.3129	0.5798	0.6791	0.4315

Q5:

Normal	TPG	0.2722	0.5264	0.5825	0.3767
Inactive	TPG	0.0388	0.0780	0.1024	0.0626
Very active	ROTAN	0.3822	0.6244	0.7264	0.4961
Normal	ROTAN	0.3078	0.5276	0.5949	0.4056
Inactive	ROTAN	0.0102	0.0286	0.0494	0.0293

Table6. The results of trajectory lengths on NYC.

Trajectory	**Models**	**Acc@1**	**Acc@5**	**Acc@10**	**MRR**
Long trajs	GETNext	0.2452	0.5378	0.6698	0.3695
Middle trajs	GETNext	0.2441	0.4927	0.5881	0.3732
Short trajs	GETNext	0.2186	0.4561	0.5269	0.3570
Long trajs	STHGCN	0.3184	-	-	0.4401
Middle trajs	STHGCN	0.2545	-	-	0.3795
Short trajs	STHGCN	0.2703	-	-	0.3783
Long trajs	TPG	0.3048	0.5482	0.6270	0.4042
Middle trajs	TPG	0.2681	0.5127	0.5824	0.3749
Short trajs	TPG	0.2396	0.4082	0.4881	0.3191
Long trajs	ROTAN	0.3432	0.5664	0.6307	0.4449
Middle trajs	ROTAN	0.3012	0.5282	0.6011	0.4057
Short trajs	ROTAN	0.2781	0.4403	0.5059	0.3555

Q5:

D5: Unlike most previous POI recommendation works, this paper utilizes the next temporal information as a known condition for predicting POIs. In the existing baseline methods, only a portion of experiments in TPG aligns with this paper. However, TPG also conducts experiments on conventional POI recommendation. To better demonstrate the effectiveness of ROTAN, the authors should consider conducting similar experiments with TPG.

Response-to-Q5:

Thanks for the suggestion. We have carefully checked TPG[25], and it doesn't conduct extensive experiments on conventional POI recommendations. Similarly, it conducts an ablation study as our work by removing the temporal-based prompt (i.e., remove TP).

To make thorough comparisons, we provide extensive results in the following tables. We can learn that our ROTAN outperforms TPG in various experimental settings, including without target time, without both time embeddings and target time, and the default setting. The results demonstrate the superiority of the proposed ROTAN over the latest TPG method.

Table7. The results of different experiment settings of TPG and ROTAN on NYC.

	Acc@1	**Acc@5**	**Acc@10**	**MRR**
TPG(w/o Target time)	0.2592	0.4928	0.5854	0.3635
TPG(w/o sequence Time and Target time)	0.2533	0.4991	0.5737	0.3613
TPG	0.2555	0.5005	0.5932	0.3669
ROTAN(w/o Target time)	0.2499	0.5007	0.5943	0.3634
ROTAN(w/o sequence Time and Target time)	0.2628	0.5024	0.6096	0.3754
ROTAN	0.3106	0.5281	0.6131	0.4104

Table8. The results of different experiment settings of TPG and ROTAN on TKY.

	Acc@1	**Acc@5**	**Acc@10**	**MRR**
TPG(w/o Target time)	0.1770	0.3915	0.4748	0.2767
TPG(w/o sequence Time and Target time)	0.1588	0.3689	0.4583	0.2577
TPG	0.1420	0.3631	0.4492	0.2436
ROTAN(w/o Target time)	0.2291	0.4531	0.5469	0.3345
ROTAN(w/o sequence Time and Target time)	0.2139	0.4636	0.5539	0.3276
ROTAN	0.2458	0.4626	0.5392	0.3475

Table9. The results of different experiment settings of TPG and ROTAN on CA

	Acc@1	**Acc@5**	**Acc@10**	**MRR**
TPG(w/o Target time)	0.1661	0.3366	0.4019	0.2456
TPG(w/o sequence Time and Target time)	0.1612	0.3394	0.4105	0.2441

TPG	0.1749	0.3285	0.3860	0.2479
ROTAN(w/o Target time)	0.1685	0.3523	0.4381	0.2535
ROTAN(w/o sequence Time and Target time)	0.1832	0.3671	0.4524	0.2697
ROTAN	0.2199	0.3718	0.4334	0.2931