Transformer-based Multi-task Learning for Queuing Time Aware Next POI Recommendation

Sajal Halder[1], Kwan Hui Lim[2], Jeffrey Chan[1], and Xiuzhen Zhang[1]

[1] School of Computing Technologies, RMIT University, Australia
[2] Singapore University of Technology and Design, Singapore
sajal.halder@rmit.edu.au, kwanhui_lim@sutd.edu.sg,
jeffrey.chan@rmit.edu.au, xiuzhen.zhang@rmit.edu.au

Abstract. Next point-of-interest (POI) recommendation is an important and challenging problem due to different contextual information and wide variety in human mobility patterns. Most of the prior studies incorporated user travel spatiotemporal and sequential patterns to recommend next POIs. However, few of these previous approaches considered the queuing time at POIs and its influence on user's mobility. The queuing time plays a significant role in affecting user mobility behaviour, e.g., having to queue a long time to enter a POI might reduce visitor's enjoyment. Recently, attention based recurrent neural networks-based approaches show promising performance in next POI recommendation but they are limited to single head attention which can have difficulty finding the appropriate complex connections between users, previous travel history and POI information. In this research, we present a problem of queuing time aware next POI recommendation and demonstrate how it is non-trivial to both recommend a next POI and simultaneously predict its queuing time. To solve this problem, we propose a multi-task, multi head attention transformer model called TLR-M. The model recommends next POIs to the target users and predicts queuing time to access the POIs simultaneously. By utilizing multi-head attention, the TLR-M model can integrate long range dependencies between any two POI visit efficiently and evaluate their contribution to select next POIs and to predict queuing time. Extensive experiments on eight real datasets show that the proposed model outperforms than the state-of-the-art baseline approaches in terms of precision, recall and F1 score evaluation metrics. The model also predicts and minimizes the queuing time effectively.

Keywords: Points of Interest (POI) · POI Recommendation · Transformer · Multi-tasking · Multi-head attention · Queuing time

1 Introduction

Travel and tourism are popular leisure activities and a trillion-dollar industry across the world. To improve the travel and tourism experience, appropriate next point-of-interest (POI) recommendation based on tourist personalized interest has attracted much attention from the researchers in recent years [2, 7, 21]. These personalized next POI recommendations can be challenging because visitors can have multiple criteria and different preferences when choosing a POI to visit next. Some visitors may prefer the nearest available POI that they are mildly interested in, while other visitors might

Fig. 1: Influence of queuing time along with spatiotemporal features in POI recommendation.

prefer one that they are very interested in despite travelling a longer distance. Others may have dynamic preferences and their previous visit history is not that important to consider. Most of the deep learning technique cannot handle multiple conflicting of near and long-distance preferences as well as recent and past visit influence simultaneously. LSTM or RNN based approaches focus on recent visits and nearest preferences based on spatiotemporal dependencies. Thus, learning spatiotemporal dependencies can be challenging. In addition, another factor that affects visitor's satisfaction is the length of queuing time. Fig. 1 depicts an example showing the significance of queuing time. Assume at lunch time (1:00 PM), a visitor wants to go to a restaurant for lunch. If the next POI recommendation model does not consider queuing time of these POIs, it may recommend nearby restaurant A or B according to the distance and other users' sequential patterns. However, these two are crowded places and users have to wait a long time having their lunch which is generally undesirable. Thus, a queuing time aware next POI recommendation model, which takes POIs queuing information along with spatiotemporal dependencies and personalized interest is more likely to recommend restaurant C to the user as next move. These kinds of queuing related activities are significant in many other real-life applications, e.g., theme park and popular tourist attracts, restaurants, concerts and festivals. In addition, with the COVID-19 pandemic, there is a need to keep physical distance and queuing takes on a health dimension, making queuing influence even more significant. To the best of our knowledge, this is the first work to consider queuing time and its prediction for the next POI recommendation.

These challenges inspired us to build a model that can capture complex spatiotemporal dependencies along with queuing time influence in next POIs recommendation. The problems of POI recommendation and queue time prediction are inter-dependent. Thus, one single model that jointly recommends top-k POIs and predict queuing time simultaneously is necessary.

Existing studies on next POI recommendation have considered spatiotemporal preferences [11] but did not consider user preferences. In another group of prior research, user identification is considered and attention-based spatiotemporal influence based *ATST-LSTM* [7] and self-attentive network *SANST* [6] have been proposed. All these works are appropriate for next POI recommendation, but they are not capable of multi-tasking (recommend POIs and predict queuing time) simultaneously. Recently, attention-based transformer shows significant improvement to capture all dependen-

cies at once using non-recurrent encoder-decoder model in volatility prediction [20, 17] and natural language processing [4]. Transformer allows multi-tasking which uses an attention mechanism to compute the dependencies of its input and output. Therefore, in this work, we propose multi-attention layers-based transformer network leverages to complex spatiotemporal dependencies. After that, we use a multi-tasking approach to recommend POIs and predict queuing time simultaneously. The main contributions of the paper can be summarized as follows:

- This work discusses the significance of queuing time aware next POI recommendation model in which queuing time affects POI selection. More specifically, the model captures user behaviour along with spatiotemporal and queuing time influences.
- We develop a multi attention transformer-based multi-task learning model for next top-k POI recommendation and queue time prediction, simultaneously. The model can recommend appropriate next POIs because of the advantages of two parallel joint learning processes.
- Experiment results using eight real-life datasets show our proposed transformer model outperforms the state-of-the-art next personalized POI recommendation based on precision, recall, F1-score and is able to predict queuing time effectively.

2 Related Works

This research focuses on next top-k POI recommendation and queuing time prediction. In this section, we briefly describe state-of-the-art research related to these areas.

POI recommendation has attracted significant attention because of its importance in both academy and industry. POI recommendation accuracy depends on multiple factors. The previous study *LORE* [22] incorporates geographical influence and social influence into a unified recommendation framework for check-in data. To solve temporal and spatial dependencies simultaneously convolutional LSTM [18] network has been proposed. Moreover, some recent works [19, 16] have employed convolutional neural network and multi-layer preceptors to POI recommendation. Huang et al. [7] proposed an attention-based spatiotemporal long and short-term memory (ATST-LSTM) network for the next POI recommendation. Zhou et al. [23] proposed generative and discriminator based POI recommendation model that maximize the learned probabilities distributions and optimize the differences between recommend POIs and true check-ins. Lim et al. [9] introduced queuing time as an important factor in itinerary recommendation. Therefore, in this work, we introduce the queuing time aware of top-k POI recommendation.

Transformer network-based model improves accuracy across a variety of NLP tasks. The model can capture all words dependencies in a sentence to predict next word. Recently, some research works in transformer-based model [20, 17] show the significant improvements in volatility prediction and event forecasting using multi-head attention technique. It has been shown that the transformer model is faster than the recurrent and convolutional layers-based models and improved performance using the multi-headed self-attention technique [14]. Multi-task learning approach has

been used for a variety of research areas i.e., sentence classification and tagging [15], entity recognition and semantic labelling [1], and two different financial forecasting [20]. Inspired by transformer multi-task learning, we use multi-head attention-based transformer model for next top-k POI recommendation and predict queue time simultaneously. The multi-head attention model can capture POIs relationships among other POIs in multiple ways and it is effective to handle users' dynamic behaviours. Our proposed TLR-M model differs from the state-of-the-art POI recommenders in various aspects. First, we introduce complex spatiotemporal dependencies along with POI sequence in transformer model. Second, we present multi-task learning in POI recommendation that can recommend top-k POI and predict queuing time simultaneously. Most importantly, the approach can set up the relationship among heterogeneous features (i.e. geographical, time and user identity features etc.) automatically using multi-head attention mechanism.

3 Preliminary and Problem Statement

In this section, we first describe key preliminary definitions and then describe the problem statement.

Definition 1 *Point of Interest (POI): A POI p is defined as a uniquely identified location (e.g., roller coaster, museum, hotel and etc.) that has longitude and latitude values. A sequence represents a set of POIs, $P = \{p_1, p_2, \cdots, p_n\}$ that user visits sequentially.*

Definition 2 *Visit Activity: User visit activity is a quadri-tuple $v^u_{t_k} = (p^u_{t_k}, l^u_{t_k}, t_k, u)$ which represents the user u visits POI $p^u_{t_k}$ with location $l^u_{t_k}$ at timestamp t_k.*

Definition 3 *Visit Sequence: A user visit sequence is a set of visit activities of the user, represented by $V_u = \left\{v^u_{t_1}, v^u_{t_2}, \cdots, v^u_{t_i}\right\}$. All users historical visit sequences in a dataset are defined by $V^U = \left\{V_{u_1}, V_{u_2}, \cdots, V_{u_{|U|}}\right\}$, here $|U|$ is the number of all users.*

Definition 4 *Visit Trajectory: A user's visit trajectory is a subset of user's visit sequence i.e. $V_u = \cup_i S^u_i$, represented by $S^u_i = \left\{v^u_{t_k}, v^u_{t_{k+1}}, \cdots, v^u_{t_{k+n-1}}\right\}$, where sequence length is n. In the sequence if the time difference between two consecutive POI visits is more than six hours, we divided it into different trajectories, all the isolated POI visits are ignored.*

Definition 5 *Queuing Time Trajectory: The queuing time is a triplet $q^p_{T_k} = (p^u_{T_k}, T_k, q_i)$ represents the user u need to wait q_i time to access the POI $p^u_{T_k}$ at timestamps T_k. The queuing time sequence is a set of queuing time triplet $S^{u_i}_q = \left\{q^p_{T_k}, q^p_{T_{k+1}}, \cdots, q^p_{T_{k+n-1}}\right\}$. All database queuing time trajectory indicates by $Q^U = \cup_i S^{u_i}_q$, where $u_i \in U$. The length of visit sequence and queuing time sequence will be same. The timestamps T_k may be hour based or half hour based time interval.*

Problem Statement: Given the input of all users' visit trajectories V^U and queuing time trajectories Q^U during past T timestamp, the output of our proposed multi-task learning model is to recommend next top-k POIs to the users and predict the prospective queuing time of recommended POIs, simultaneously. The model can recommend a fixed set of POIs (top-5 or top-10) and can optimize queuing time between original time and predicted queuing time.

4 Proposed TLR-M Model

In this section, we describe our proposed **T**ransformer based **L**earning **R**ecommendation using **M**ulti-tasking **TLR-M** model. We capture the global dependencies between users visit trajectories and POIs queuing influences using multi-head self-attention mechanism. The self-attention mechanism overcomes two limitations of RNN based top-k prediction tasks. Firstly, the RNN model is hard to support parallel work because of its recursive nature. Secondly, RNN can not capture the whole sequence information directly. The purpose of using self-attention is two-folds. It captures the whole sequence information flow directly and it permits parallel operations that join multiple learning objectives effectively.

Our proposed model uses multi-head self-attention based on two pairs of encoder and decoder. Fig. 2(a) illustrates the architecture of TLR-M model. Some transformer based classifier used encoder only rather than encoder and decoder. Encoder based model can generate global attention based transition matrix but can not generate personalized attention based different recommendation appropriately. However, we use encoder and decoder together to capture personalize correlation between POIs visits whole sequence directly without sequential propagation. Here, the spatial, the temporal,user inter-dependencies among the time and geographical locations and queuing time influence are jointly considered which performed by the attentive learning. The model takes POIs visit trajectories as quadruplet $(p^u_{t_i}, l^u_{t_i}, t_i, u_{t_i})$ input in transformer *Encoder-1* and queuing time trajectories as triplet $(p^u_{t_i}, T_i, q_i)$ input in transformer *Encoder-2*. Thus, the inputs (x^1_t and x^2_t) of the two transformer encoders are as follows:

$$x^1_t = W_p p^u_{t_i} + W_l l^u_{t_i} + W_t t^u_i + W_u u_{t_i} \text{ and } x^2_t = W_p p^u_{t_i} + W_T T_i + W_q q_{t_i} \quad (1)$$

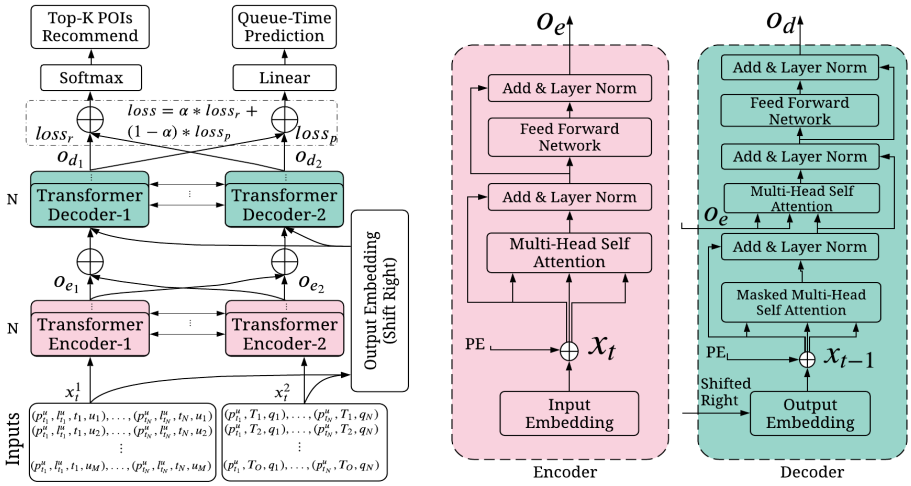
where $p^u_{t_i}$, $l^u_{t_i}$, t^u_i and u_{t_i} represent POI IDs, spatial, temporal context and user vector respectively. W_p, W_l, W_t and W_u are transition matrices. Besides this, q_{t_i} is the queuing time and T_i is timestamps. W_T and W_q are transition matrices.

Unlike recurrent networks (LSTMs and GRUs), the transformer network can process the input sequentially, POI after POI (as token after token). The transformer uses positional encoding to keep a separate embedding table with input vectors. The model use POI position in the trajectory instead of POI index in the table. Thus, the positional embedding table is much smaller than the one-hot encoding table. Positional embeddings may train with the rest of the deep network or pre-computed by the following sinusoidal formula. Here, we use pre-computed positional embedding sinusoidal signal manner [14].

$$PE_{pos,2i} = \sin\left(\frac{pos}{10000^{2i/E_{size}}}\right) \text{ and } PE_{pos,2i+1} = \cos\left(\frac{pos}{10000^{2i/E_{size}}}\right) \quad (2)$$

where E_{size} and pos denotes the embedding size and relative position of POIs in trajectories, respectively. We define $2i$ and $2i+1$ to indicate the embedding element index with the even and odd position, respectively.

Encoder consists of N layers and each layer are composed of multi head self-attention, fully connected feed forward followed by layers normalization [14] depicts



(a) Multi-tasking model (b) Single Encoder and Decoder Unit

Fig. 2: The architecture of TLR-M model.

in Fig 2(b). In our model Fig 2(a), the inputs of first encoder layers in *Encoder-1* and *Encoder-2* come from the element wise addition between inputs embedding latent vector and positional encoding represent by $x_e^1 = x_t^1 + PE$ and $x_e^2 = x_t^2 + PE$. The output of first encoder layer feeds as input embedding in the next layer. Thus, the Nth layer outputs of two encoders (Encoder-1 and Encoder-2) are o_{e_1} and o_{e_2}.

$$\begin{aligned} o_{e_1} &= lNo(x_e^1 + FFN(lNo(x_e^1 + MulH(Q,K,V)))) \\ o_{e_2} &= lNo(x_e^2 + FFN(lNo(x_e^2 + MulH(Q,K,V)))) \end{aligned} \tag{3}$$

where *lNo(.)* is layer normalization, *FFN(.)* is fully connected feed-forward network and *MulH(.)* is multi-head attention mechanism.

These two encoder outputs are concatenated ($o_e = o_{e_1} + o_{e_2}$) and fed into the decoders that share the impact of top-k and queuing time together. The decoder unit in Fig 2(b) consists of six layers, among them masked multi head attention uses to avoid the significance of padding token. We use padding token to construct the same length of visit trajectories and queuing time trajectories. Decoder takes same input as encoder input but in this case, information is shifted one position right, ensure that the prediction output of position t_{i+1} only depends on known outputs up to time t_i. Then, these embeddings added with positional embeddings and construct $x_d^1 = x_{t-1}^1 + PE$ and $x_d^2 = x_{t-1}^1 + PE$. This output embedding transforms through the mask layer, multi head attention and feed forward sub-layers using add and normalization functions. Each output of decoder repeatedly uses as input in the decoder and transformed until N repetition. Then, the output of N^{th} decoder feeds into soft-max layer or fully connected layer based on the target output. *Decoder-1* and *Decoder-2* performed based on following equation.

$$
\begin{aligned}
o_{d_1} &= lNo(x_d^1 + FFN(lNo(x_d^1 + MulH(o_e, o_e, MulHM(Q, K, V))))) \\
o_{d_2} &= lNo(x_d^2 + FFN(lNo(x_d^2 + MulH(o_e, o_e, MulHM(Q, K, V)))))
\end{aligned} \tag{4}
$$

These two decoder outputs are updated during the training phase using two different loss functions. We have used soft parameter sharing architecture in which each task has its own parameter setting based model. The parameters of these two models are then regularized to reduce the difference among them and encourage the parameters to be similar. In the training phase, we integrate multi-layer perceptron dropout technique to prevent over-fitting. To recommend top-K POIs with higher probabilities, the outputs of *Decoder-1* feeds into softmax layer and the queuing prediction component, we use *ReLU* as the activation function for the fully connected layer defined by $\hat{y} = softmax(o_{d_1})$ and $\hat{y}_i^q = ReLU(o_{d_2})$.

The proposed model uses two objective functions for best top-K POI recommendation and appropriate queuing time prediction. Thus, the first objective function uses sparse-cross-entropy as a loss function for accurate top-k recommendation as follows.

$$
loss_r = -\frac{1}{N}\sum_{i=1}^{N}[y_i log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i)] \tag{5}
$$

where y_i is the original output and $\hat{y}_i$ is the predicted output.

In the *Decoder-2* output, we find the queuing probability corresponding POI distributions. Then, to reduce the difference between predicted probability and likelihood probability we use mean square error loss function as follows.

$$
loss_p = -\sum_{i=1}^{N}[(y_i^q - \hat{y}_i^q)^2] \tag{6}
$$

where y_i^q and $\hat{y}_i^q$ represent original queuing time and predicted queuing time respectively. Therefore, our objective function is a weighted average of these two loss functions using weight parameter $\alpha \in [0, 1]$.

$$
loss = \alpha \times loss_r + (1 - \alpha) \times loss_p \tag{7}
$$

We use Adam-optimizer and applied the trick of decay learning rate with the steps until it reaches convergence. Finally, the *TLR-M* extract our two desire goals simultaneously that are to recommend top-k POIs and predict proceeding queuing time.

4.1 Algorithm

Algorithm 1 depicts an overview of our proposed TLR-M model. It takes two sets of inputs including POI sequence, spatiotemporal, users, and queue time features. First, we initialize all parameters in line 1. Then based on two minibatch inputs x_b^1 and x_b^2 we train our proposed model in lines 2-10. These inputs feed into the encoders and generate outputs using multi attention-based feed-forward network using equation 3 in line 4. The two encoder outputs fed as input into the decoders with right-shifted encoder inputs in line 5. After that, the output of the multi-head attention layer is normalized

Algorithm 1: TLR-M Model

Data: (x^1, x^2)= Model inputs, PE = positional embedding, b_{size}: batch size
Result: TLR-M model $\{M\}_u$: [top-k POIs], [Queue time]
1 **TRAIN MODEL:** Initialize all parameters
2 **for** $(x_b^1, x_b^2) \leftarrow sample(x^1, x^2, b_{size})$ **do**
3 $x_e^1, x_e^2 = x_b^1 + PE, x_b^2 + PE$
4 Using equation 3 find o_e^1 and o_e^2
5 $x_d^1, x_d^2 = Input(RightShift(x_e^1, x_e^2), o_e^1, o_e^2)$
6 Using equation 4 fund decoder output o_d^1 and o_d^2
7 $\hat{y} = softmax(o_d^1)$ and $\hat{y}^q = Relu(o_d^2)$
8 Calculate loss J using equations 5, 6, 7
9 Build the learned TLR-M Model $\{M\}_u$
10 Update the parameters.
11 **end**
12 **TEST MODEL** : $\hat{y}_{test}$, $\hat{y}^q_{test}$ = Predict output based on Model $\{M\}_u$ and test data.
13 $Return\ \hat{y}_{test}, \hat{y}^q_{test}$;

and passed fully connected feed-forward network. It generates two probabilities distributions as outputs in line 6. The outputs are passed with softmax and rectified linear unit to find top-k POIs recommendation and queuing time prediction probabilities in line 7. Using this probability, we applied two loss functions and achieved our goal in line 8. Furthermore, using the loss functions of equations 5, 6 and 7 we train our proposed model $\{M\}_u$ and update all parameters in lines 10. After constructing the model, we predict the next top-k potential POIs $\hat{y}_{test}$ using our test data and predict queuing time $\hat{y}^q_{test}$ in line 12. Finally, we measure our evaluation matrices based on output $\hat{y}_{test}$ and $\hat{y}^q_{test}$ compare to original POI and queuing time.

5 Experiments

In this section, we present experimental setup, datasets, baseline algorithms and evaluation metrics. For these comparisons, our proposed *TLR-M* and the existing baseline methods are implemented in the Python language. Training and testing sets selection are important factors in the deep learning model. At first, we construct itinerary based on visiting POI where the first t steps used as a model design and $t + 1$ step is used as a next target POI. Thus, we construct all the prefixes of the input trajectories and make sub-trajectories as per standard practice [13]. Then, among these itineraries, we select training set using 70% random itineraries and the testing set using remaining 30% itineraries. For baseline models, we used authors' publicly shared codes.

To analyse the significance of the proposed models, we have used *t-test* statistical method. Experimental results show that TLR-M significantly out-performs all baseline with at least 96.5% confidence interval ($p \leq 0.035$), based on t-test.

Dataset	# Photos	POI Visits	# Users	# POIs	Dataset	# Chick-ins/	POI Visits	# Users	# POIs
Epcot	90,435	38,950	2,725	17	Edinburgh	82,060	33,944	1,454	29
Magic Kingdom	133,221	73,994	3,342	27	Melbourne	17,087	5,871	911	242
California Adv	193,069	57,177	2,593	25	Foursquare	315,084	315,084	7,642	6,202
Budapest	36,000	18,513	935	39	Gowalla	407,894	407,894	5,628	7,283

Table 1: Parameters description of various datasets.

5.1 Datasets and Baseline Algorithms

For our experiments, we use various datasets comprising three theme parks, three cities and two social networks [9, 10, 12]. The visit sequences of POIs are constructed based on photos taken or check-in times to these POIs. If the time gap between two consecutive photos taken time or check-in time is greater than 6 hours, it is considered as a new visit sequence. Among these datasets Foursquare and Gowalla do not hold queuing time information, thus we describe the results differently. Foursquare and Gowalla datasets are sparse. Hence, we consider the users who have at least 20 records and the POIs that has been visited at least 20 times. All other datasets, we filter out those users and POIs with fewer than 3 visits and 3 visitors, respectively. The variation of eight datasets is shown in Table 1.

Several baselines are described to compare the performance of our proposed *TLR-M* model that plays a significant role in the next POI recommendation. Among a large number of existing works, we have considered several recent works as baselines that outperform than the other baselines [3, 8, 5]. To best evaluate the performance of our proposed *TLR-M* model, we compare our proposed models with four recent baselines which are ST-RNN [11], STACP [12], APOIR [23] and ATST-LSTM [7]. Queuing time information are not always present in dataset i.e. Gowalla and Foursquare datasets contain only check-in time. If there are check-out times information, we can define queuing time as the time difference between one check-in and previous check-out time. However, to show the transformer-based POI recommendation efficiency, we develop a single-tasking TLR model. It takes only visit trajectories information as input and uses single encoder and decoder instead of pair encoders and decoders. All parameters are same, and the objective function is $loss_r$ in Equation 5. To predict queuing time prediction, we develop a single-tasking TLR_q model using queuing trajectories and $loss_p$ loss function.

5.2 Performance Evaluation

To compare the performance of our model against the various baselines, we use Precision@k, Recall@k, F1-Score@k and root mean square error (RMSE) metrics [11]. Theme parks and cites datasets are dense, thus we evaluate the sub-trajectories based recommendation accuracies. On the other hand, Gowalla and Foursquare datasets are very sparse datasets that case we consider users based accuracies. We combine test sub-trajectories target and predicted top-K POIs and find accuracy metrics as per research paper [7].

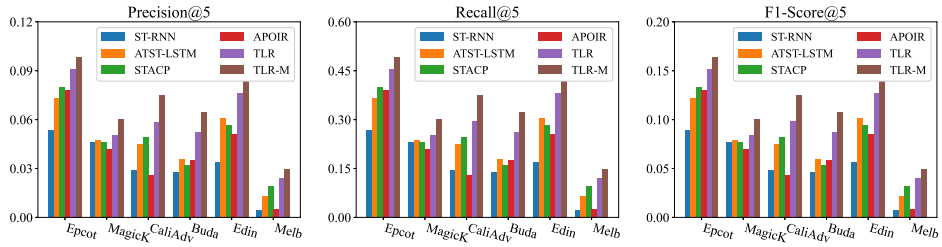


Fig. 3: Results comparison among proposed TLR, TLR-M and various baselines, in terms of Precision@5, Recall@5 and F1-Score@5 for six datasets.

5.3 Results and Discussion

The performance of *TLR* and *TLR-M* models with state-of-the-art POI recommendations are evaluated based on several experiments. The main evaluation process of POI recommendation is how accurately the recommended POIs reflect visitors real visit POIs. Fig. 3 shows the results of proposed models against the various baselines for the datasets against the various evaluation metrics. The proposed *TLR* model performance against the existing POIs recommendation is the best in terms of all evaluation metrics. Three sub-figures in Fig. 3 show the precision@5, recall@5 and F1-score@5 results based on the six datasets.

It shows that our proposed multi-tasking model *TLR-M* outperforms all the baselines as well as our proposed single task *TLR* model. Because we have used multi-head attention-based transformer that can capture POI visit full trajectories relationship more efficiently than the RNN or CNN based approach. The transformer architecture can capture all inputs and outputs dependent relations efficiently. On the other hand, queuing time impact has been added at training time that also increases next POI recommendation efficiency. The results of evaluation metrics differ dataset to dataset because we consider top 5 POIs among all POIs. Thus, Melbourne dataset results show a lower score than the other datasets because of a comparatively larger number of POIs. Our results show the same output pattern for k values 3 and 10. In this experiment, we run each model 10 times and reported average values as a metric value.

The proposed *TLR-M* model not only outperforms in top-k POI recommendation but also predicts queuing time very well. To compare *TLR-M* model with single queuing time prediction model, we use single TLR_q model in which predicts target POIs queuing time as output. In this case, root mean square error loss function has been used. We have developed $ATST_q$ model applying the same input (queuing time and POI sequence) in $ATST\_LSTM$ model. The $ATST_q$ model is unable to predict the queuing time effectively, as shown by RMSE value that is at least 10 times higher than the *TLR-M* model. The table 2 shows that our multi-task model *TLR-M* outperforms single task LTR_q and $ATST_q$ models.

Performance Analysis for Larger Datasets The results in three theme park and three city datasets show that our models outperform the state-of-the-art baselines,

and we proceed to evaluate the performance of our proposed model on large-scale datasets. To show the performance, we use two large check-in datasets Foursquare and Gowalla datasets with the increasing number of POIs. Fig. 4 shows the proposed TLR model outperforms than the baselines in terms of Recall@10 value. These data sets are very sparse, thus we consider a various number of POIs to evaluate performance comparison. Here, we did not compare our multi-tasking model TLR-M because the queuing time information is missing in these datasets. The results show that all algorithms values decrease with the increase of POIs number. Our proposed TLR model shows the best results because it can capture POI trajectories inter dependencies efficiently. Based on these results, it is obvious that our model outperforms on small and large datasets than the baselines.

Dataset	$ATST_q$	TLR_q	TLR-M
Epcot	1319.9	173.1	**102.5**
MagicK	925.6	90.5	**84.7**
CaliAdv	1834.2	108.7	**101.5**
Buda	2157.5	147.3	**129.2**
Edin	1755.3	136.7	**113.2**
Melb	2602.5	132.4	**88.8**

Table 2: RMSE results between single and multi-task. Small value are better.

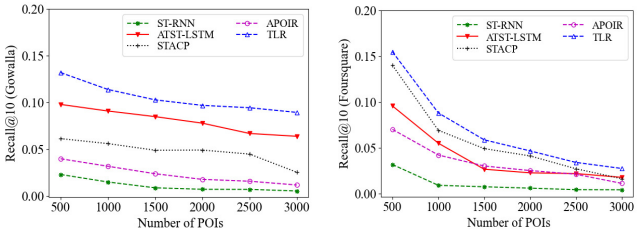


Fig. 4: Performance comparison based on two sparse datasets.

Effects of Parameters: We explore the effect of batch size and multi loss functions balancing factor (α). We consider learning rate = 0.001, dropout rate = 0.5, number of head = 4, training step = 200 and hidden size = 128. Our results show that the effect of batch size increases first then decreases with the value increases (due to page limit we could not show the figures). We find that batch size 32 is best for these algorithms. Besides this, we explore the effect of balancing factor α. We observe that the best value for the balance factor fluctuates between 0.50 to 0.75 in different datasets. Therefore, we set default value 0.5 to provide the equal significance of two-loss functions.

6 Conclusion

In this paper, we study the new research topic, queuing time aware next POI recommendation. By incorporating sequential, spatial, temporal and queuing time influences, we have proposed multi-head transformer-based multi-task learning recommendation model *TLR-M* that recommends top-k POIs and predicts queuing time simultaneously. By leveraging the attention technique instead of RNN architecture, the model can capture whole trajectory dependencies directly and efficiently. Experiment results based on eight datasets show that our proposed *TLR-M* model significantly outperforms the various state-of-the-art models.

In this work, we have studied the queuing time aware top-k POI recommendation problem. Our future research direction is to construct a full itinerary considering the budget time and social relationship influence that users get maximum entertainment.

References

1. Alonso, H.M., Plank, B.: When is multitask learning effective? semantic sequence prediction under varying data conditions. In: EACL. pp. 1–10 (2017)
2. Chang, B., Park, Y., Park, D., Kim, S., Kang, J.: Content-aware hierarchical point-of-interest embedding model for successive poi recommendation. In: IJCAI. pp. 3301–3307 (2018)
3. Chen, X., Xu, H., Zhang, Y., Tang, J., Cao, Y., Qin, Z., Zha, H.: Sequential recommendation with user memory networks. In: WSDM. pp. 108–116 (2018)
4. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
5. Feng, S., Li, X., Zeng, Y., Cong, G., Chee, Y.M., Yuan, Q.: Personalized ranking metric embedding for next new poi recommendation. In: IJCAI (2015)
6. Guo, Q., Qi, J.: Sanst: A self-attentive network for next point-of-interest recommendation. arXiv preprint arXiv:2001.10379 (2020)
7. Huang, L., Ma, Y., Wang, S., Liu, Y.: An attention-based spatiotemporal lstm network for next poi recommendation. IEEE Transactions on Services Computing (2019)
8. Li, X., Cong, G., Li, X.L., Pham, T.A.N., Krishnaswamy, S.: Rank-geofm: A ranking based geographical factorization method for point of interest recommendation. In: SIGIR. pp. 433–442. ACM (2015)
9. Lim, K.H., Chan, J., Karunasekera, S., Leckie, C.: Personalized itinerary recommendation with queuing time awareness. In: SIGIR. pp. 325–334. ACM (2017)
10. Lim, K.H., Chan, J., Leckie, C., Karunasekera, S.: Personalized trip recommendation for tourists based on user interests, points of interest visit durations and visit recency. Knowledge and Information Systems **54**(2), 375–406 (2018)
11. Liu, Q., Wu, S., Wang, L., Tan, T.: Predicting the next location: A recurrent model with spatial and temporal contexts. In: AAAI (2016)
12. Rahmani, H.A., Aliannejadi, M., Baratchi, M., Crestani, F.: Joint geographical and temporal modeling based on matrix factorization for point-of-interest recommendation. In: ECIR. pp. 205–219. Springer (2020)
13. Tan, Y.K., Xu, X., Liu, Y.: Improved recurrent neural networks for session-based recommendations. In: Proceedings of the 1st Workshop on Deep Learning for Recommender Systems. pp. 17–22 (2016)
14. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: NIPS. pp. 5998–6008 (2017)
15. Wang, S., Che, W., Liu, Q., Qin, P., Liu, T., Wang, W.Y.: Multi-task self-supervised learning for disfluency detection. arXiv preprint arXiv:1908.05378 (2019)
16. Wang, S., Wang, Y., Tang, J., Shu, K., Ranganath, S., Liu, H.: What your images reveal: Exploiting visual contents for point-of-interest recommendation. In: WWW. pp. 391–400 (2017)
17. Wu, X., Huang, C., Zhang, C., Chawla, N.V.: Hierarchically structured transformer networks for fine-grained spatial event forecasting. In: The Web Conference 2020. pp. 2320–2330. ACM (2020)
18. Xingjian, S., Chen, Z., Wang, H., Yeung, D.Y., Wong, W.K., Woo, W.c.: Convolutional lstm network: A machine learning approach for precipitation nowcasting. In: NIPS. pp. 802–810 (2015)
19. Yang, C., Bai, L., Zhang, C., Yuan, Q., Han, J.: Bridging collaborative filtering and semi-supervised learning: a neural approach for poi recommendation. In: SIGKDD. pp. 1245–1254 (2017)
20. Yang, L., Ng, T.L.J., Smyth, B., Dong, R.: Html: Hierarchical transformer-based multi-task learning for volatility prediction. In: The Web Conference 2020. pp. 441–451 (2020)
21. Yin, H., Wang, W., Wang, H., Chen, L., Zhou, X.: Spatial-aware hierarchical collaborative deep learning for poi recommendation. TKDE **29**(11), 2537–2551 (2017)
22. Zhang, J.D., Chow, C.Y., Li, Y.: Lore: Exploiting sequential influence for location recommendations. In: SIGSPATIAL. pp. 103–112. ACM (2014)
23. Zhou, F., Yin, R., Zhang, K., Trajcevski, G., Zhong, T., Wu, J.: Adversarial point-of-interest recommendation. In: WWW. pp. 3462–34618 (2019)