Predicting Human Mobility with Self-attention and Feature Interaction

Jingang Jiang, Shuo Tao, Defu Lian(✉), Zhenya Huang, and Enhong Chen

Institute of Smart City Research (Wuhu) and School of Data Science, University of Science and Technology of China, Hefei, China
{liandefu,cheneh}@ustc.edu.cn, {jjg1210,taoshuo,huangzhy}@mail.ustc.edu.cn

Abstract. Mobility prediction plays an important role in a wide range of location-based applications and services. However, two important challenges are not well addressed in existing literature: 1) explicit high-order interactions of spatio-temporal features are not systemically modeled; 2) most existing algorithms place attention mechanisms on top of recurrent network, so they can not allow for full parallelism and are inferior to self-attention for capturing long-range dependence. To this end, we propose MoveNet, a self-attention based sequential model, to predict each user's next destination based on her most recent visits and historical trajectory. MoveNet first introduces a cross based learning framework for modeling feature interactions. With self-attention on both the most recent visits and historical trajectory, MoveNet can use an attention mechanism to capture user's long-term regularity in a more efficient and effective way. We evaluate MoveNet with three real-world mobility datasets, and show that MoveNet outperforms the state-of-the-art mobility predictor by around 10% in terms of accuracy, and simultaneously achieves faster convergence and over 4x training speedup.

Keywords: Human mobility · Self-attention · Feature interaction

1 Introduction

With the development of location-acquisition techniques and the prevalence of smart devices, human daily routines are much easier to digitize and share with friends in social network websites. Mobility understanding and prediction are of vital importance in a wide range of applications and services, ranging from urban planning [10], traffic forecasting [12] and epidemic control [11] to location-based advertisements and recommendation [30].

The key in mobility prediction is how to capture useful mobility patterns from historical traces. Previous work about mobility prediction is mainly based on either Markov models or recurrent models. Markov models predict next locations conditioning on a few recent visits, and was successfully applied for location

X. Wang et al. (Eds.): APWeb-WAIM 2020, LNCS 12318, pp. 117–131, 2020.
https://doi.org/10.1007/978-3-030-60290-1_9

prediction in GPS trajectories [1], cell tower traces [20] and check-in traces [7,13]. The finding in these work is that pattern frequency determines the order of Markov models, so they are strongly correlated with pattern-based methods [18]. In practice, whatever type of dataset is available, Markov models with Kneser-Ney smoothing [3] or hierarchical Pitman-Yor Process [22] models are used, because they integrate different orders of models. Regarding cell tower traces, 93% human movements can be predicted based on sequential patterns [20], but for social check-ins, mobility predictability is much lower [15]. This is because users often check in unseen locations, while Markov models fail to predict in this case. Leveraging sequential recommendation techniques, Factorizing Personalized Markov Chain (FPMC) [4] and Personalized Ranking Metric Embedding (PRME) [6] are adapted to learn location transitions with distributional representation. Although they can deal with the prediction of unseen locations, these methods sometimes perform worse than Markov models, due to not well capturing sequential patterns. One of underlying reasons may lie in the BPR loss, which does not perform as well as expected in recommendation tasks [14].

The success of recurrent neural networks (RNN) in language modeling motivates researchers to apply RNN-like models for mobility prediction. The pioneer work in [24] separately models short-term sequential contexts and long-term sequential contexts, by replacing (factorizing) Markov models with RNN models and performing optimization with respect to the sampled softmax functions. Spatio-temporal statistics between consecutive visits are also put into the gate functions to control information flow [28]. To further model long-range dependence in long-term sequential contexts, attention mechanisms are applied on top of RNNs [5].

Among these existing works, two important challenges are not well addressed. First, spatio-temporal features generally include location id and time id, and lack consideration for the influence of explicit high-order interaction between features. This may help to distinguish mobility modeling from sequential recommendation and may lead to the improvements of mobility prediction. Second, recurrent networks are time-consuming to train particularly for long sequences and can not be comparable to self-attention mechanisms for capturing long-range dependence according to [23].

To this end, we propose MoveNet, a self-attention based sequential model, to predict movements based on both the most recent visits and the whole historical trajectory. MoveNet first embeds spatial-temporal information of check-ins, including user, time and location, and then models high-order interactions of spatial-temporal feature in the most recent visits based on a cross-based learning framework [16]. Following that, we apply self-attention to the embedding representations of the most recent visits to capture short-term preference and to the embedding of the whole historical trajectory to capture long-term regularity. Long-term regularity in the current context is then extracted by an attention mechanism, by considering representation of the most recent visit from self attention as query, and representations of the historical trajectory as memory. In order to promote efficiency, only the last k representations in the historical trajectory are used for attention, but this does not lead to large performance degradation.

The contributions can be summarized as follows:

- We propose a self-attention based sequential model for predicting user movements, which promotes efficiency and effectiveness of processing lengthy historical trajectories by allowing for full parallelism and capability of modeling long-range dependence.
- We model high-order interactions of spatio-temporal features based on a cross-based learning framework, so that user-location, location-time, user-time interactions can be naturally incorporated. This framework is very general, so it is possible to integrate more useful features of locations and contexts.
- We conduct extensive experiments by evaluating MoveNet on three real-world check-in datasets. The results show that MoveNet not only can outperform the SOTA predictor by about 10%, but is also faster than the SOTA predictor in terms of empirical convergence and running time cost. Moreover, the effect of self-attention and high-order interaction modeling has been verified.

2 Related Works

In computer science, mobility can be predicted based on Markov models, machine learning models, sequence pattern mining, and recurrent networks.

Ashbrook and Starner applied second-order Markov model to predict future movement after automatically clustering the GPS trajectories into meaningful location sequences[1]. Song et al. reported empirical evaluation results of location predictors on WiFi mobility data, and observed that second-order Markov model performed best [21]. Chen et al. investigated variable-order Markov model for next location prediction [2]. Lian et al. utilized Markov models with Kneser-Ney smoothing to predict next check-in location [13] and Gao et al. applied Pitman-Yor process to seamlessly integrate different orders of Markov models [8]. To model long-term dependence between locations, Mathew et al. trained a Hidden Markov Model for each user [17]. Markov models are closely correlated with frequent sequential pattern mining, since transitions with large probability may correspond to frequent sequential patterns. Therefore, several sequential pattern mining based methods [18,27] are proposed, which first extract frequent patterns and predict next location based pattern matching. Treating next location as classes, mobility prediction can be cast into multi-class classification problem [19]. Input features are usually sparse, so such models also suffer from low prediction accuracy of unseen locations.

To better deal with the prediction of unseen locations, personalized transition probability are factorized based on pairwise interaction tensor factorization [4,29] or metric embedding [6] so that locations are represented by distributional representation. To capture long-range dependence between visits, recurrent networks such as LSTM or GRU are used to model location sequences [24]. The problem of gradient vanish in recurrent networks restricts the capability of capture long-term dynamics. Therefore, this work splits historical trajectories into long-term contexts and short-term contexts, and use RNN for short-term contexts and GRU for long-term contexts. To further address the gradient vanish problem, attention mechanisms are usually applied on top of LSTM when modeling long-term context [5]. In order to capture the periodicity in the long-term trajectory pattern, Gao et al. proposed a variational attention mechanism [9]. achieving higher accuracy of mobility prediction.

3 Preliminary

In this paper, we study the mobility prediction problem in the sparse check-in datasets. In the check-in dataset, a user's check-in sequence is $T^u = q_1 \rightarrow q_2 \rightarrow \cdots \rightarrow q_n$, where $q_i = (u, t_i, l_i)$ denotes a check-in record, indicating a user u checked in a location l_i at time t_i. Note that the check-in sequence is subject to chronological order. That is, for any two check-in two records q_i and q_j with $i < j$, we have $t_i < t_j$. Since users selectively issue check-ins when he visited/stayed at some locations due to privacy concerns, the time intervals between consecutive check-ins are usually not even. Therefore, we split each user's check-in sequence into multiple sessions, such that the time interval between consecutive sessions are larger than a given threshold Δt. In a formal way, $\tilde{T}^u = S_1^u \rightarrow S_2^u \rightarrow \cdots \rightarrow S_m^u$, where m is the number of sessions and $S_i^u = q_{i_1} \rightarrow \cdots \rightarrow q_{i_{k_i}}$ of length k_i denotes the i-th session, being a sub-sequence of T^u. The problem of mobility prediction is defined as follows:

Definition 1 (Mobility Prediction). *Given the most recent incomplete session $S_m^u = q_{m_1} \rightarrow q_{m_2} \rightarrow \cdots \rightarrow q_{m_j}$ and the historical sessions $S_1^u \rightarrow S_2^u \rightarrow \cdots \rightarrow S_{m-1}^u$ of a user u, predict the next location $l_{m_{j+1}}$ at which the user will check in.*

Note that such a definition can be applied for mobility prediction with continuously-recorded trajectories, like GPS trajectories. This is because mobility prediction is usually conducted on sequence of stay points[1] while time intervals between consecutive stay of points are also not even.

4 MoveNet

The whole framework of MoveNet is shown in Fig. 1, where we model historical sessions and the most recent session separately.

The goal of modeling historical sessions is to capture long-term regularity. Consequently, the historical sessions are concatenated as the whole trajectory and then fed into a spatial-temporal embedding module. After being concatenated with user embedding, they are further fed into a self-attention module, yielding a sequence of check-in representations with long-range dependence encoded. To improve efficiency, we only use the representations for the last k check-ins as memory slots for subsequent attention use. This is also motivated

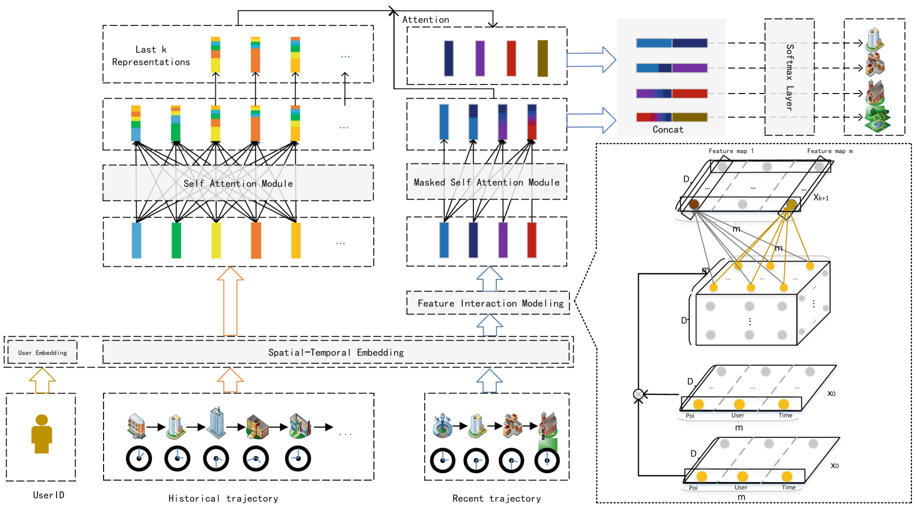

Fig. 1. Framework of MoveNet

by the observation that using the representations of the whole historical trajectory only leads to a very small improvement.

The objective of modeling the trajectory in the most recent session is to capture short-term user preference. As such, the trajectory in the most recent session is first embedded, and then fed together with user embedding into the feature interaction modeling module, in order to capture high-order interaction of spatial-temporal features. Note that the feature interaction modeling module is not applied in the former part, because it is more time-consuming than simple operators and attention mechanism may be a better practice for capturing long-term regularity. The most recent sequence of check-in representation is then fed into the masked self-attention module to capture sequential dependence. Here, the mask is used out of causality concerns, i.e., considering the first j items to predict the $(j+1)$-th check-in location. Following that, each check-in representation in the most recent trajectory attends over k memory slots obtained from modeling historical sessions. Being concatenated with the attended representations, each check-in representation is then used in the softmax layer for multi-class classification.

4.1 Spatial-Temporal Embedding

For check-in locations, we denote by $\boldsymbol{L} \in \mathbb{R}^{N\times d}$ the location embedding matrix. For check-in time, we first convert it to a tuple (hour of day, weekend or not) and denote by $\mathbf{T} \in \mathbb{R}^{48\times d}$ the time embedding matrix. For each user, we do not have her any side information, so each user is embedded with the embedding matrix $\boldsymbol{U} \in \mathbb{R}^{M\times d}$. Note that the same dimension of embedding matrices is required in the feature interaction modeling module. Since each check-in location is attached with a GPS position, it is also possible to represent each GPS position with a vector, such that a metric value between any two vectors can approximate their sphere distance. However, we do not observe any significant improvements and do not take them into account. Moreover, we will use self-attention for capturing long-range dependence, and usually incorporate positional embedding for encoding relative order information. However, we do also not observe any significant improvements and thus not take them into account.

4.2 Self-attention Module

In the SOTA mobility predictor [5], RNNs like LSTM and GRU are used to process input sequences. However, the capacity of learning long-range dependencies is limited, and its sequential processing style makes it less efficient and less parallelizable. With the great success of Transformer in machine translation, self-attention has been applied to various sequential processing tasks, due to the advantages of capturing long-range dependence and fully parallelizable.

The basis of self-attention is the scaled dot-product attention, which is defined by [23] as follows:

$$\text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d}}\right)\boldsymbol{V} \tag{1}$$

where $\boldsymbol{Q}$, $\boldsymbol{K}$, $\boldsymbol{V}$ represent queries, keys and values, respectively. The attention layer computes a weighted sum of values in $\boldsymbol{V}$, where the weight reflects the similarity of each query to keys. $\sqrt{d}$ is a scale factor to avoid overly large values of inner product.

Assume the self-attention module takes $\boldsymbol{X} \in \mathbb{R}^{n\times d}$, a sequence of n representations as input, which is obtained from either direct concatenation or the feature interaction modeling module, and convert it into the query, key, value matrices via linear projections. In particular, the output of the self-attention module is calculated by

$$\boldsymbol{Y} = \text{SA}(\boldsymbol{X}) = \text{Attention}\left(\boldsymbol{X}\boldsymbol{W}^{\boldsymbol{Q}}, \boldsymbol{X}\boldsymbol{W}^{\boldsymbol{K}}, \boldsymbol{X}\boldsymbol{W}^{\boldsymbol{V}}\right) \tag{2}$$

where $\boldsymbol{W}^Q, \boldsymbol{W}^K, \boldsymbol{W}^V \in \mathbb{R}^{d\times d}$ are projection matrices. Unlike self-attention in the Transformer, we don't use multi-head attention, since we do not observe benefit of using more than one head.

Following the Transformer, we feed the output of self-attention into a feed-forward network (FFN) to encode a non-linearity transformation following weighted summation. When applied on $\boldsymbol{Y}_j$, the j-th row of $\boldsymbol{Y}$, FFN produces the following output:

$$\boldsymbol{Z}_j = \text{FFN}(\boldsymbol{Y}_j) = \text{ReLU}(\boldsymbol{Y}_i\boldsymbol{W}^{(1)} + \boldsymbol{b}^{(1)})\boldsymbol{W}^{(2)} + \boldsymbol{b}^{(2)}, \tag{3}$$

where $\boldsymbol{W}^{(1)} \in \mathbb{R}^{d\times 4d}$, $\boldsymbol{W}^{(2)} \in \mathbb{R}^{4d\times d}$, $\boldsymbol{b}^{(1)} \in \mathbb{R}^{4d}$, $\boldsymbol{b}^{(2)} \in \mathbb{R}^{d}$. Here each representation is first transformed into a 4-times larger space and is then transformed back after applying the ReLU activation. Note that the linear transformations are the same at different positions.

Causality. When the self-attention module is employed in modeling the most recent session, causality should be imposed, such that future check-ins can not be used for computing representation of the current one. However, this is easy to implement by incorporating mask into self-attention of sequence.

Stacking. Through the transformation of self-attention and feed-forward network, $\boldsymbol{Z}_i$ aggregates representation of all sequentially-dependent check-ins. It might be useful to learn more complex dependence by applying multiple self-attention blocks, each of which consists of self-attention and feed-forward network. Note that parameters in feed-forward network are varied from block to block. Moreover, in order to stabilize and speed up the model training, we perform the following operations

$$f(x) = \text{LayerNorm}(x + \text{Dropout}(\text{Sublayer}(x))) \tag{4}$$

where Sublayer(x) denotes self-attention or feed-forward network and LayerNorm(x) denotes layer normalization.

4.3 Feature Interaction Modeling

In the STOA mobility predictor [5], embedding vectors are directly concatenated, without considering high-order interaction of these vectors. This may greatly affect the accuracy of mobility prediction. According to our empirical observations, as shown in Fig. 2, feeding user embedding together with spatial-temporal embedding into the self-attention module performs better than concatenating it with the outputs of the self-attention module. Therefore, the feature interaction modeling module takes user embedding, location embedding and time embedding as input, and should take both second-order and third-order interactions into account. Motivated by the cross-based learning framework [16], we transform the feature matrix $\boldsymbol{X}^0$, which stacks three embedding vectors by row, into $\boldsymbol{X}^1$ and $\boldsymbol{X}^2$ of the same shape through the following equations:

$$\begin{aligned}
\boldsymbol{X}^1_{h,*} &= \sum_{i=1}^{3}\sum_{j=1}^{3} W^{h,1}_{ij}\left(\boldsymbol{X}^0_{i,*} \circ \boldsymbol{X}^0_{j,*}\right), \\
\boldsymbol{X}^2_{h,*} &= \sum_{i=1}^{3}\sum_{j=1}^{3} W^{h,2}_{ij}\left(\boldsymbol{X}^0_{i,*} \circ \boldsymbol{X}^1_{j,*}\right),
\end{aligned} \tag{5}$$

where $\boldsymbol{X}^1_{h,*}$ denotes the h-th row of $\boldsymbol{X}^1$, $\boldsymbol{W}^{h,1}, \boldsymbol{W}^{h,2} \in \mathbb{R}^{3\times 3}$ denote the parameter matrices of the second-order and third-order interactions respectively. $\circ$ represents the Hadamard product, i.e., element-wise multiplication between two vectors. Here, $\boldsymbol{X}^1$ captures second-order interaction between any two of three embedding vectors, and $\boldsymbol{X}^2$ captures third-order interactions among three embedding vectors.

As shown in Fig. 1, the calculation of $\boldsymbol{X}^a$, $a = \{1, 2\}$ can be achieved by two steps. First, introduce a 3-order tensor $\boldsymbol{O}^a \in \mathbb{R}^{3\times 3\times d}$, which consists of the outer products of $\boldsymbol{X}^{a-1}_{*,f}$ and $\boldsymbol{X}^{0}_{*,f}$ for each dimension f, s.t. $1 \leq f \leq d$. By regarding $\boldsymbol{O}^a$ as an image of size 3×3, $\boldsymbol{X}^a$ is then obtained by applying convolution on the image with $\boldsymbol{W}^{h,a}$ as a filter.

4.4 Attention and Predict

In this section, we discuss how to aggregate short-term preference with long-term regularity and how to train the parameters. Motivated by [5], we will apply use the attention mechanism for this task, by considering the check-in representation of the recent trajectory as query, and the check-in representations of the historical trajectory as memory slots. It is worth mentioning that we use only the last k historical representations. Denoting by $\boldsymbol{Z}^{(q)}_j$ the representation of the last check-in q_j of the most recent trajectory, and by $\boldsymbol{Z}^{(v)}_i$ the representation of the i-th check-in in the historical trajectory.

$$\boldsymbol{V}_j = \sum_i \frac{\exp\left(\langle \boldsymbol{Z}^{(q)}_j, \boldsymbol{Z}^{(v)}_i \rangle\right)}{\sum_{i'} \exp\left(\langle \boldsymbol{Z}^{(q)}_j, \boldsymbol{Z}^{(v)}_{i'} \rangle\right)} \boldsymbol{Z}^{(v)}_i \tag{6}$$

where $\langle x, y \rangle$ denotes dot product between vector x and y. $\boldsymbol{V}_j$ is then concatenated with $\boldsymbol{Z}^{(q)}_j$, and passed it into a fully connected network for prediction. We then use the cross-entropy loss for parameter optimization.

5 Experiments

We will evaluate the proposed algorithm with three check-in datasets, reporting results of the comparison with competing baselines, ablation study and sensitivity analysis.

5.1 Datasets

The three datasets are Foursquare check-in datasets in different cities or at different periods. Table 1 summarizes dataset statistics. Note that the NYC-1 dataset [5] spans from Feb. 2010 to Jan. 2011, while both the NYC-2 dataset and the TKY dataset [25] span from Apr. 2012 to Feb. 2013. These data are the check-in information actively shared by users on the website, including user ID, timestamp, GPS location and poi ID.

Following [5], we split the trajectory into multiple sessions by setting time interval $\Delta t = 72$ hours and then filter out sessions with less than 5 records and users with less than 5 sessions. We then use the first 80% sessions as the training set and the left 20% sessions as the testing set.

Table 1. Dataset statistics

	City	Users	Check-ins	POIs
NYC-1	New York	886	82,575	10,497
NYC-2	New York	935	118,600	13,962
TKY	Tokyo	2,108	323,987	21,395

5.2 Settings

Table 2 gives the default setting of hyperparameters, some of them, such as learning rate, may be fine-tuned to achieve a better accuracy of mobility prediction. Moreover, we use last 10 check-in representations in the historical trajectory as memory slots for attention use. The reason for $k = 10$ is that the experiment shows that when k is around 10, the result is better.

Table 2. The default settings of hyperparameters

Parameter	Value	Parameter	Value
Learning rate	1e−4	location emb. dim	200
Gradient clip	5.0	time emb. dim	200
Decay of lr	0.1	user emb. dim	200
L2 penalty	1e−5	hidden size	600
#block	2	#head	1

5.3 Baselines

We compared the proposed algorithm with the following baselines:

- **Markov Model**, the first-order Markov model, the first algorithm was used in mobility prediction [1].
- **RNN**, a GRU model, is applied on the most recent check-in trajectories, with user, time and location as input at each time step.
- **DeepMove** [5], is the state-of-the-art mobility predictor. It only embeds location and time, and applies GRU for modeling the most recent trajectory, and uses the attention mechanism to capture long-term regularity. The representations from attention and RNN are then concatenated with user embedding to predict the next location.
- **RNN+SA**$_{\text{tl}}$ [26], first applies GRU on the most recent sequence of location and time, and then uses the self-attention module to capture long-range dependence. The subscript (tl) indicates only time and location included. User embedding is concatenated with each output of self-attention.

– **SA**$_{\mathrm{utl}}$, a variant of MoveNet, which only model the most recent trajectory without feature interaction. The subscript indicates user, time and location included.
– **FI+SA**$_{\mathrm{utl}}$, a variant of MoveNet, which only model the most recent trajectory but feature interaction included.

5.4 Comparison with Baselines

Table 3. The comparison results with baselines

	NYC-1			NYC-2			TKY		
	Acc@1	Acc@5	Acc@10	Acc@1	Acc@5	Acc@10	Acc@1	Acc@5	Acc@10
Markov	0.0820	0.1190	0.1212	0.1304	0.1976	0.2019	0.1255	0.1860	0.1900
RNN	0.1453	0.2995	0.3466	0.1883	0.3728	0.4260	0.1259	0.2572	0.3067
DeepMove	0.1322	0.2911	0.3419	0.1763	0.3702	0.4302	0.1451	0.2965	0.3547
RNN+SA$_{\mathrm{tl}}$	0.1446	0.3041	0.3495	0.1887	0.3902	0.4456	0.1158	0.2433	0.2915
SA$_{\mathrm{utl}}$ (ours)	0.1507	0.3213	0.3701	0.1966	0.4008	0.4574	0.1335	0.2683	0.3205
FI+SA$_{\mathrm{utl}}$ (ours)	0.1491	0.3303	**0.3865**	**0.1976**	0.4196	0.4856	0.1326	0.2711	0.3237
MoveNet (ours)	**0.1534**	**0.3318**	0.3843	0.1972	**0.4227**	**0.4888**	**0.1474**	**0.3100**	**0.3683**

We report the accuracy of the mobility predictor in terms of Acc@1, Acc@5 and Acc@10, and show the results in Table 3. Acc@k means whether the top k items in the predicted result have the correct item. From this table, we have the following observations.

First, the proposed MoveNet outperforms the state-of-the-art predictor, i.e., DeepMove, by 9.82%, 10.91% and 9.95% on average in terms of Acc@1, Acc@5 and Acc@10. The Markov model is the worst of all, indicating the power of neural network sequential models.

Second, self-attention (SA$_{\mathrm{utl}}$) is better than RNN, the relative improvements are 4.72%, 6.37% and 6.22% on average in terms of Acc@1, Acc@5 and Acc@10. Placing self-attention on top of RNN can improve the accuracy of mobility predictor, but does not perform as well as the self-attention model.

Third, incorporating feature interaction modeling can lead to 2.85% and 3.87% improvements on average in terms of Acc@5 and Acc@10 by comparing FI+SA$_{\mathrm{utl}}$ with SA$_{\mathrm{utl}}$. The self-attention model with feature interaction modeling even performs comparatively to MoveNet.

Finally, modeling long-term regularity can be beneficial, by comparing MoveNet with FI+SA$_{\mathrm{utl}}$ in the TKY dataset. However, improvements in the other two datasets are marginal.

5.5 How to Use User Embedding

In order to understand how to better incorporate user embedding, we evaluate two GRU models with the most recent trajectory in the NYC-2 dataset. The first GRU model (U+GRU$_{tl}$) takes location and time as input at each step, and concatenate the output of GRU at each step with user embedding for prediction. The second GRU model (GRU$_{utl}$) takes user, time and location as input, and directly uses the output of GRU for prediction. We then report the Acc@1 and Acc@5 of these two GRUs with user embedding size varied. We can observe that GRU$_{utl}$ is much better than U+GRU$_{tl}$. However, with the growing size of user embedding, the margin between them first increases and then decrease. Overall, we always suggest the second way to incorporate user embedding.

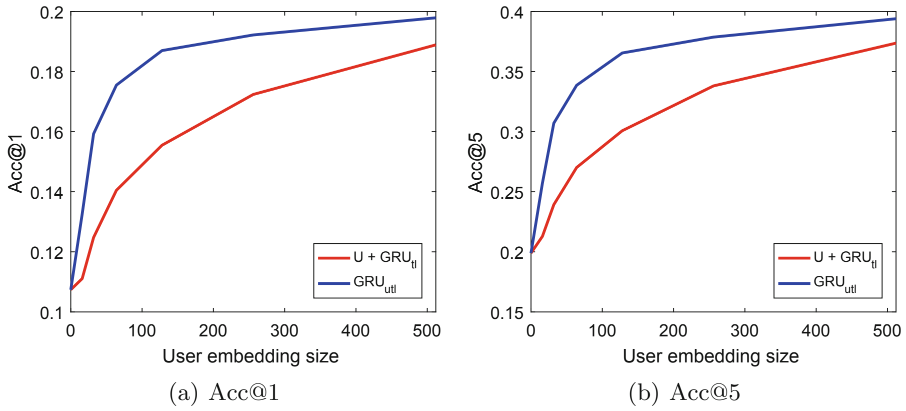

Fig. 2. Accuracy comparison with check-in features

5.6 When Long-Term Regularity Takes Effect

We observe the long-term regularity is only useful in the TKY dataset. To understand when long-term regularity can take effect, we compare the predictability of mobility behavior [20] in these datasets and plot the distribution of predictability over popularity in Fig. 3. We can observe that in the TKY dataset much more users are highly-predictable (>0.71). Predicting mobility for these users is more dependent on the long-term regularity. Also, the statistics show trajectory length in the TKY dataset is around 20% longer than that in the NYC-2 dataset. Therefore, the long-term regularity plays more important role in the TKY dataset.

5.7 Sensitivity Analysis

In natural language processing, the number of blocks and heads in the self-attention module is set to 6 and 8, respectively [23]. These settings may not the best choice for mobility predictor. To this end, we vary the number of blocks from 1 to 3 and the number of heads from 1 to 6 and report the results of evaluation in Table 4. We can make the following observations. First, 2 blocks are better

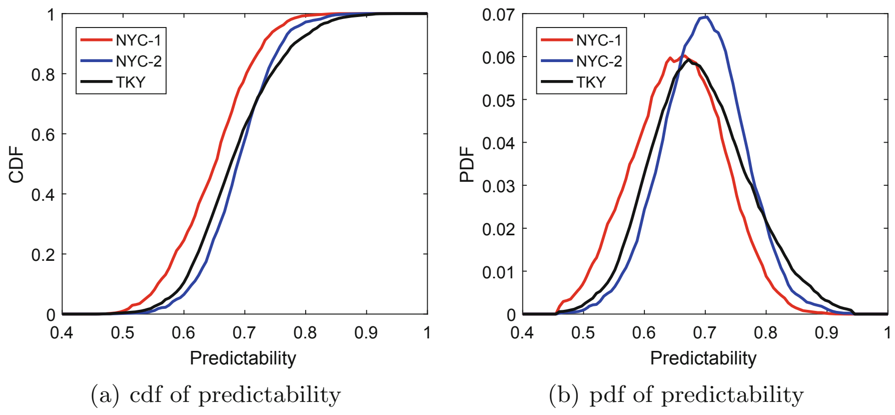

Fig. 3. Predictability of mobility behavior in the three datasets

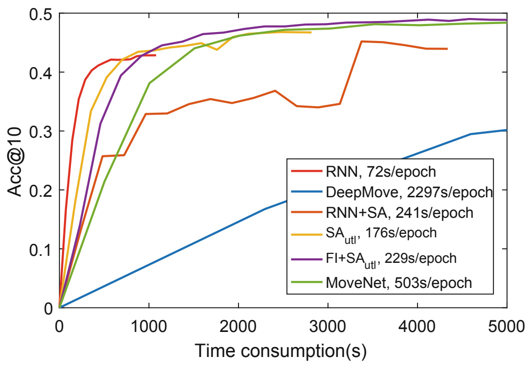

Fig. 4. Training efficiency comparison with baselines

in the NYC datasets, while in the TKY dataset the accuracy is dramatically degraded with the increasing number of blocks. Second, 1 head is better in the NYC datasets, and an increasing number of heads leads to accuracy degradation. However, in the TKY dataset, the better number of heads is 3. In other words, these two parameters should be fine-tuned from task to task, from dataset to dataset.

Table 4. Acc@5 w.r.t the number of heads (#H) and blocks (#B).

#B	NYC-1	NYC-2	TKY	#H	NYC-1	NYC-2	TKY
1	0.3166	0.3889	0.2785	1	0.3213	0.4008	0.2683
2	0.3213	0.4008	0.2683	3	0.3060	0.3880	0.2787
3	0.3215	0.4001	0.0877	6	0.3019	0.3882	0.2779

5.8 Training Efficiency Comparison

Comparing MoveNet with DeepMove, we replace RNN with self-attention, so training efficiency can be significantly promoted due to the capacity of parallel computing. Therefore, we record time cost of training in each epoch and show the results in Fig. 4. We can observe that MoveNet is not only 4x faster in each epoch of training, but also converges faster than DeepMove. And we use the same training strategy as DeepMove, when the accuracy of two consecutive epochs is no longer improved, we will use a smaller learning rate to train from the previous best model and stop training when the learning rate is small enough.

6 Conclusion

In this paper, we proposed MoveNet, a new mobility predictor, based on self-attention and feature interaction modeling. MoveNet is not only faster than the SOTA predictor in terms of empirical convergence and running time cost, but also outperform the competing baselines for mobility prediction according to evaluation with three real-world datasets.

Acknowledgments. This paper is partially supported by Municipal Program on Science and Technology Research Project of Wuhu City (No. 2019yf05) and the Fundamental Research Funds for the Central Universities.

References

1. Ashbrook, D., Starner, T.: Learning significant locations and predicting user movement with GPS. In: Proceedings of ISWC 2002, pp. 101–108. IEEE (2002)
2. Chen, M., Liu, Y., Yu, X.: NLPMM: a next location predictor with Markov modeling. In: Tseng, V.S., Ho, T.B., Zhou, Z.-H., Chen, A.L.P., Kao, H.-Y. (eds.) PAKDD 2014. LNCS (LNAI), vol. 8444, pp. 186–197. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-06605-9_16
3. Chen, S.F., Goodman, J.: An empirical study of smoothing techniques for language modeling. In: Proceedings of ACL 1996, pp. 310–318. ACL (1996)
4. Cheng, C., Yang, H., Lyu, M.R., King, I.: Where you like to go next: successive point-of-interest recommendation. In: Proceedings of IJCAI 2013, pp. 2605–2611. AAAI Press (2013)
5. Feng, J., et al.: Deepmove: predicting human mobility with attentional recurrent networks. In: Proceedings of the 2018 World Wide Web Conference, pp. 1459–1468. International World Wide Web Conferences Steering Committee (2018)
6. Feng, S., Li, X., Zeng, Y., Cong, G., Chee, Y.M., Yuan, Q.: Personalized ranking metric embedding for next new poi recommendation. In: Proceedings of IJCAI 2015, pp. 2069–2075. AAAI Press (2015)
7. Gao, H., Tang, J., Liu, H.: Exploring social-historical ties on location-based social networks. In: Proceedings of ICWSM 2012 (2012)
8. Gao, H., Tang, J., Liu, H.: Mobile location prediction in spatio-temporal context. In: Proceedings of the Mobile Data Challenge at the 10th International Conference on Pervasive Computing (2012)

9. Gao, Q., Zhou, F., Trajcevski, G., Zhang, K., Zhong, T., Zhang, F.: Predicting human mobility via variational attention. In: The World Wide Web Conference, pp. 2750–2756. ACM (2019)
10. Horner, M., O'Kelly, M.: Embedding economies of scale concepts for hub network design. J. Transp. Geogr. **9**(4), 255–265 (2001)
11. Hufnagel, L., Brockmann, D., Geisel, T.: Forecast and control of epidemics in a globalized world. Proc. Natl. Acad. Sci. U. S. A. **101**(42), 15124–15129 (2004)
12. Kitamura, R., Chen, C., Pendyala, R., Narayanan, R.: Micro-simulation of daily activity-travel patterns for travel demand forecasting. Transportation **27**(1), 25–51 (2000)
13. Lian, D., Xie, X., Zheng, V.W., Yuan, N.J., Zhang, F., Chen, E.: CEPR: a collaborative exploration and periodically returning model for location prediction. ACM Trans. Intell. Syst. Technol. **6**(1), 1–27 (2015)
14. Lian, D., Zhao, C., Xie, X., Sun, G., Chen, E., Rui, Y.: GEOMF: joint geographical modeling and matrix factorization for point-of-interest recommendation. In: Proceedings of KDD 2014, pp. 831–840. ACM (2014)
15. Lian, D., Zhu, Y., Xie, X., Chen, E.: Analyzing location predictability on location-based social networks. In: Tseng, V.S., Ho, T.B., Zhou, Z.-H., Chen, A.L.P., Kao, H.-Y. (eds.) PAKDD 2014. LNCS (LNAI), vol. 8443, pp. 102–113. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-06608-0_9
16. Lian, J., Zhou, X., Zhang, F., Chen, Z., Xie, X., Sun, G.: xDeepFM: combining explicit and implicit feature interactions for recommender systems. In: Proceedings of KDD 2018, pp. 1754–1763. ACM (2018)
17. Mathew, W., Raposo, R., Martins, B.: Predicting future locations with hidden Markov models. In: Proceedings of the 2012 ACM Conference on Ubiquitous Computing, pp. 911–918. ACM (2012)
18. Monreale, A., Pinelli, F., Trasarti, R., Giannotti, F.: Wherenext: a location predictor on trajectory pattern mining. In: Proceedings of KDD 2009, pp. 637–646. ACM (2009)
19. Noulas, A., Scellato, S., Lathia, N., Mascolo, C.: Mining user mobility features for next place prediction in location-based services. In: Proceedings of ICDM 2012, pp. 1038–1043. IEEE (2012)
20. Song, C., Qu, Z., Blumm, N., Barabási, A.: Limits of predictability in human mobility. Science **327**(5968), 1018–1021 (2010)
21. Song, L., Kotz, D., Jain, R., He, X.: Evaluating location predictors with extensive Wi-Fi mobility data. In: Proceedings of INFOCOM 2004, vol. 2, pp. 1414–1424. IEEE (2004)
22. Teh, Y.W.: A hierarchical Bayesian language model based on Pitman-Yor processes. In: Proceedings of ACL 2006, pp. 985–992. ACL (2006)
23. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
24. Yang, C., Sun, M., Zhao, W.X., Liu, Z., Chang, E.Y.: A neural network approach to jointly modeling social networks and mobile trajectories. ACM Trans. Inf. Syst. (TOIS) **35**(4), 36 (2017)
25. Yang, D., Zhang, D., Zheng, V.W., Yu, Z.: Modeling user activity preference by leveraging user spatial temporal characteristics in LBSNs. IEEE Trans. Syst. Man Cybern.: Syst. **45**(1), 129–142 (2014)
26. Zeng, J., He, X., Tang, H., Wen, J.: A next location predicting approach based on a recurrent neural network and self-attention. In: Wang, X., Gao, H., Iqbal, M., Min, G. (eds.) CollaborateCom 2019. LNICST, vol. 292, pp. 309–322. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-30146-0_21

27. Zhang, C., Han, J., Shou, L., Lu, J., La Porta, T.: Splitter: mining fine-grained sequential patterns in semantic trajectories. Proc. VLDB Endow. **7**(9), 769–780 (2014)
28. Zhao, P., et al.: Where to go next: a spatio-temporal gated network for next poi recommendation. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 5877–5884 (2019)
29. Zhao, S., Zhao, T., Yang, H., Lyu, M.R., King, I.: Stellar: spatial-temporal latent ranking for successive point-of-interest recommendation. In: Proceedings of AAAI 2016 (2016)
30. Zheng, V., Zheng, Y., Xie, X., Yang, Q.: Towards mobile intelligence: learning from GPS history data for collaborative recommendation. Artif. Intell. **184**, 17–37 (2012)