Personalized Recommendation of Location-based Services using Spatio-temporal-aware Long and Short Term Neural Network

XIAOSHUANG SHENG¹, FAN WANG¹, YIFAN ZHU¹, TENGFEI LIU¹, and HUAN CHEN¹

¹Hello Inc, Hangzhou, China, 310013

Corresponding author: XIAOSHUANG SHENG (e-mail: shengxiaoshuang10699@hellobike.com).

This work was supported in part by Hello Inc.

ABSTRACT User behavioral data are critical for predicting the next item in a recommendation system, which can be acquired by location-based services. However, existing approaches directly use latitude and longitude information, rather than fully exploitation of the location information. In order to better utilize location information and user behavior data, considering the shortcomings of existing models, a spatial-temporal long and short-term neural network (SLSTNN) is proposed in this paper for location-based personalized service recommendation. SLSTNN is the first attempt to comprehensively characterize the long and short term sequences of users, which is achieved by a double-layer attention mechanism and integrated with a deep neural network to improve the representation of spatio-temporal data. In addition, the explicit feature cross-network is employed to characterize user profiles and context features. Experimental results demonstrate that the proposed SLSTNN framework outperforms the state-of-the-art methods with an improvement of online conversion rate by 2.14%. SLSTNN addresses the insufficient feature crossing problem in the simple sequence model and can be potentially used in many recommendation systems.

INDEX TERMS neural network, recommendation system, deep learning, feature interactions

I. INTRODUCTION

WITH the prevalence of platform economy, users can easily access mobile platforms anywhere in daily life. As a result, large amounts of behavior logs on location-based services have been generated. Based on such user behavior data and location data, recommending the next item that may be of interest to users has become a key task to improve the user’s experience while bringing new value to the platform. Ranking models are an important part of recommendation systems.

Unlike traditional recommendation systems, location-based recommendation systems suffer from new challenges. First, location information is usually fed into the model directly as features, while the potential information cannot be exploited extensively. Second, as the travel industry is different from e-commerce when users decide whether to travel, users’ short-term interest is more important than long-term interest. In e-commerce, many ranking models have shifted from traditional methods to deep models. Current deep models capture the diversity of user interests [1], the evolution of interests over time [2], as well as long sequential models [3], [4]. The probability of a driver completing a passenger order should be obtained to recommend a more suitable passenger to the driver in the travel industry. Existing ranking models cannot fully express the relationship between user choice and current location, long-term user interest, and short-term user interest.

Existing studies usually integrate users’ long-term preferences and short-term preferences directly into deep learning models indiscriminately. For example, [5], [6] directly adopt the Embedding&MLP paradigm for long-term and short-term preference sequences, i.e., the input features are first mapped as low-dimensional embedding vectors, then transformed into fixed-length vectors in a grouping manner, and finally fed into a multilayer perceptron (MLP) in series to learn the nonlinear relationships between features. In [1], [2], they use attention mechanisms to tap into the diversity of user interests and the evolution of interests over time. These
methods do not show the different effects of long-term and short-term interests on user decisions.

In this paper, we propose the Spatial-temporal-aware Long and Short Term Neural Network (SLSTNN) for predicting the next recommended item in location-based services. We adopted Uber’s H3 address coding for location information to solve the problems of non-cooccurrence and sparsity of items. The advantage is that the spatial information of the item can be incorporated into the model. Hierarchical attention mechanisms represent users’ long-term and short-term preferences. Specifically, this includes three steps: 1) to embed users and projects into low-dimensional dense space; 2) to use the attention network of the first layer to calculate the different weights of the items in the user’s long-term sequence, and then use the weights to compress the item vector to generate the user’s long-term representation, and 3) to use the attention network of the second layer to combine the user’s short-term behavior sequence with the user’s long-term representation obtained in the previous step.

In addition to the long-term and short-term sequence data, the user’s portrait features and context features are also very important for the ranking model. Many models follow the Embedding&MLP paradigm, leading to feature crossover being implicit. Implicit feature crossing is not necessarily sufficient. We use the explicit feature crossover network to crossover user portrait and context features to improve the model performance.

In summary, the primary contributions of this paper are summarized as follows:

- We employ Uber’s H3 address coding to encode the location information, which addresses the sparse spatio-temporal data problem and improves the generalization ability of the model.
- Based on the hierarchical attention mechanism, we integrate users’ long-term and short-term interest sequence data into the model, which solves the problem that short-term interest is more important than long-term interest in location-based services.
- Through an explicit feature intersection structure, user portraits, and contextual features are fully intersected to make up for the lack of feature representation in simple sequence models.
- Experimental results on two public datasets demonstrate that our model consistently outperforms state-of-the-art methods in terms of both Cross-Entropy and Area Under Curve.

In order to better integrate location information and user behavior data, SLSTNN model is proposed. Firstly, SLSTNN uses Uber’s H3 address coding to encode the location information, which solves the problem of sparse data. Then SLSTNN uses the hierarchical attention mechanism to represent the user’s long-term and short-term sequence vectors, which change with different target items. Finally, SLSTNN uses an explicit feature crossover network to realize the intersection of base features, which reduces the workload of artificial feature intersection. The rest of the paper is organized as follows. We discuss related work in section 2 and introduce the background about the hitch riding services platform in section 3. Section 4 describes in detail the design of SLSTNN model. We present experiments in section 5 and conclude in section 6.

II. RELATED WORK

Traditional recommendation systems have explored embedding&MLP. User features are compressed into a fixed-length representation vector regardless of the candidate items. Some work has proposed to map the original sparse features into multiple fixed-length representation vectors [5]–[7]. All the vectors are then concatenated to obtain the overall representation vector of the instance. However, these methods are limited in capturing high-level user-item interactions, as the fixed-length vectors can not effectively represent the sequential behavior of users.

Compared with the traditional recommendation algorithm, the recommendation algorithm for location-based services depends on the current user’s location. Baidu Map [8] was the first to try to use users’ positioning patterns to recommend the best travel routes. Efficient and reliable recommendation results are provided in real-time by constantly collecting user feedback and updating the number of lines in the recommendation system. Reinforcement learning aims to find the best policy model for decision making and has been shown to be powerful for sequential recommendations. Didi [9] proposed a deconfounded multi-agent environment reconstruction (DEMER) approach in order to learn the environment together with the hidden confounder. DigiT.DSS.Lab [10] proposed a predictive vehicle ride sharing system for commuting, which has impact on the smart cities’ green ecosystem. Yang et al. [11] proposed that feature engineering focuses on traffic recommendations for application scenarios of multi-modal models and designs from multiple perspectives of users, travel, and services. However, these methods do not better integrate location information and user behavior information into the model.

More recently, sequence recommendation systems have attracted increasing research attention. Standard sequence recommendation algorithms are straightforward, and some common sequence modeling approaches, such as Pooling [12], Recurrent Neural Network [13], Convolutional Neural Networks [14], Attention [1], Memory Network [15], Transformer [16], etc. The attention mechanism has been used to activate relevant user behaviors [1]. In [2], not only the continuous interest is effectively captured, but also it establishes the evolution of interest corresponding to the target item. Pi et al. [3], [4] used user behavior sequence data with a maximum length of 54000, which is designed for very long sequence data. The above-mentioned methods usually encode user behaviors into a single representation. However, there are usually multiple aspects of a user’s interest preferences. Therefore, the user’s sequence behaviors are encoded as multiple interest representation vectors [17]. There are
other methods for sequential recommendation, such as using comparison learning to do sequential recommendation tasks [18]. Long-term and short-term interactions may have different effects on the user’s current interests. Therefore, it is necessary to distinguish between long-term and short-term user behavior. Ying et al. [19] used hierarchical attention networks to model long-term and short-term user behavior. ZHI et al. [20] proposed denoising user-aware memory network (DUMN), which constructs four feedback sequences of users to model users’ preferences in a fine-grained way. Kun et al. [21] proposed an all-MLP model with learnable filters for sequential recommendation task. The all-MLP architecture endows our model with lower time complexity, and the learnable filters can adaptively attenuate the noise information in the frequency domain. We follow the similar pipeline, but contributes in that three aspects: 1) we use Uber’s H3 address encoding to encode location information about users’ long- and short-term preferences; 2) the proposed framework models the impact of long-term and short-term serial data on user decisions through a hierarchical attention mechanism, and 3) our model describes user portraits and contextual features through an explicit feature crossover network.

III. BACKGROUND
When the driver comes to the hitch riding services platform to browse the order, the matching recommendation system consists of two main stages: recall and ranking. The matching recommendation system is described in detail in Fig.1.

- Recall stage: The system obtains the longitude and latitude of the driver’s current position, and obtains the list of candidate passenger orders according to the starting distance and departure time difference between the driver and the passenger.

- Ranking stage: The system predicts the probability of order completion for each passenger by a ranking model of recall candidate passenger orders, and shows the order list with a high probability of order completion in priority. On each day, many drivers come to the system to browse orders, leaving a lot of user behavior, which is very important for the ranking model. For example, if a driver often goes to Xiaoshan Airport in Hangzhou, China, the order whose destination is near Xiaoshan Airport in the passenger order list should have a higher ranking. If another driver likes to take a higher price order, the higher price order in the passenger order list should rank higher.

IV. THE PROPOSED SLSTNN FRAMEWORK
A. FEATURE REPRESENTATION
Five kinds of features are used in the online hitch riding service system: driver and passenger portrait features, context features, order matching features, driver’s long-term sequence data (i.e. history completion order sequence data), and driver’s short-term real-time click behavior sequence data. We will give the explanation as follows.

1) Driver and passenger portrait features: The driver’s portrait mainly shows the number of orders received and completed in the past seven days, fourteen days, and twenty-eight days, as well as the driver’s safety and performance ability.

2) Context features: The context features include the driver’s city, the weather of the day, etc.

3) Order matching features: The matching relationship between driver and passenger orders in the current time and space, such as the distance between the driver’s starting point and passenger’s starting point, the difference between driver’s browsing time and passenger’s departure time, etc.

4) Driver’s long-term sequence data: In our system, each passenger order has a unique order ID. Once the driver accepts the passenger order, the order will not be presented to the system, resulting in very sparse data. To reduce the sparsity of order data and increase the generalization ability of the model, Uber’s H3 address coding is adopted to recode the destination of the order. The resolution of H3 is available at https://h3geo.org/docs/core-library/restable. Due to the business concern, we chose the resolution of 6 and 7, and 8. H3 resolution corresponds to the size of the regular hexagon block. The greater the resolution, the smaller the area of the corresponding regular hexagon. Assuming that the driver has completed three orders in history, the longitude and latitude of the destination of these orders are [(30.278456, 120.12427), (30.304176, 120.20589), (30.214071, 120.17415)]. Here, the H3 resolution of 6 is used to calculate the regular hexagonal block code corresponding to the location information of the three orders. Through calculation, the driver history sequence is: [86309a427ffffff, 86309a547ffffff, 86309a42ffffff]. The calculation logic of other H3 resolutions is the same as above, and the description will not be repeated here. We show the information of regular hexagonal blocks on the map, as shown in Fig.2. It can be seen that the locations in the same regular hexagon block share the same block code, which reduces the sparsity of...
spatio-temporal data and is very helpful to improve the effectiveness of the deep learning model. One of the innovations of this paper is to re-encode the location information through Uber’s H3 coding to reduce the sparsity of spatio-temporal data.

5) Driver short-term real-time click behavior sequence data: A driver browsing order on the platform without creating an order is not clear about the destination, departure time, and price range of the order s/he is interested in. Real-time statistics on the clicking behavior of the driver on the platform is required. As described above, we can obtain the driver’s short-term clicking destination order sequence by Uber’s H3 coding. We can also access the short-term click order price sequence, starting distance sequence, and time difference sequence. In this way, we can predict the driver’s destination range, time range, and price range, which is of great help for our system to provide better service to the driver.

The entire set of features used in our system consists of five categories, as shown in Table 1. Among which, a driver long term and short term behavior sequence features are typically multi-hot encoding vectors and contain rich information about the driver. Data in our system is mostly in a multi-group categorical form. For example, \{gender = Female, driver_click_city_sequence = {beijing, shanghai}\}, which is normally transformed into high-dimensional sparse binary features via encoding [5]. Mathematically, encoding vector of i-th feature group can be formalized as \( t_i \in R^{K_i} \). \( K_i \) denotes the dimensionality of feature group \( i \), which means feature group \( i \) contains \( K_i \) unique ids. \( t_i[j] \) is the \( j \)-th element of \( t_i \) and \( t_i[j] \in 0, 1, \sum_{j=1}^{K_i} t_i[j] = k \). Vector \( t_i \) with \( k = 1 \) refers to one-hot encoding and \( k > 1 \) refers to multi-hot encoding. Then one instance can be represented as \( x = [t_1^T, t_2^T, \ldots, t_M^T]^T \) in a group-wise manner, where \( M \) is the number of feature groups. And \( \sum_{i=1}^{M} K_i = K \), where \( K \) is dimensionality of the entire feature space. In this way, the aforementioned instance with two groups of features are illustrated as:

\[
\begin{align*}
0, 1 & , \ldots, 1, 1, 0 \\
\text{gender}=\text{Female, driver_click\_city\_sequence}=\text{beijing, shanghai}
\end{align*}
\]

**B. SYSTEM ARCHITECTURE OF THE PROPOSED SLSTNN FRAMEWORK**

Aiming at the particularity of location-based service recommendation system, we developed a ranking model, consisting of five layers, as shown in Fig.3.

1) Embedding Layer: Since the inputs are high dimensional binary vectors, embedding layer is used to transform them into low dimensional dense representations. For the i-th feature group of \( t_i \), let \( W^i = [w^i_1, \ldots, w^i_j, \ldots, w^i_{K_i}] \in R^{D \times K_i} \) represent the i-th embedding dictionary, where \( w^i_j \in R^D \) is an embedding vector with dimensionality of \( D \). If \( t_i \) is one-hot vector with j-th element \( t_i[j] = 1 \), the embedded representation of \( t_i \) is a single embedding vector \( e_i = w^i_j \). If \( t_i \) is multi-hot vector with \( t_i[j] = 1 \) for \( j \in \{i_1, i_2, \ldots, i_k\} \), the embedded representation of \( t_i \) is a list of embedding vectors: \( \{e_{i_1}, e_{i_2}, \ldots, e_{i_k}\} = \{w^i_{i_1}, w^i_{i_2}, \ldots, w^i_{i_k}\} \).

2) Long-term attention-based pooling layer: In the model, the fixed length of the sequence was set as \( H \), and the long-term sequence of user only retained the latest \( H \) data. Each passenger order can be regarded as a target order, and the driver’s preference for each target passenger order is different. How to characterize the relationship between the driver’s long-term behavior sequence and passenger order? Here we refer to deep interest network(DIN)’s [1] activation unit. Instead of expressing all user’s diverse interests with the same vector, DIN adaptively calculates the representation vector of user interests by considering the relevance of historical behaviors w.r.t. candidate passenger order. This representation vector varies over different passenger orders. Specifically, activation units are applied to the user behavior features,
which performs as a weighted sum pooling to adaptively calculate user representation $\alpha_{t-1}^{long-term}$ given a candidate passenger order $A$, as shown in Eq.(1)

$$
\alpha_{t-1}^{long-term} = f(v_A, e_1, e_2, \ldots, e_H) = \sum_{j=1}^{H} a(e_j, v_A) e_j,
$$

where $\{e_1, e_2, \ldots, e_H\}$ is the list of embedding vectors of behaviors of driver $U$ with length of $H$. $v_A$ is the embedding vector of candidate passenger order $A$. In this way, $\alpha_{t-1}^{long-term}$ varies over different passenger order. $a(\cdot)$ is a feed-forward network with output as the activation weight, as illustrated in Fig.4. Apart from the two input embedding vectors, $a(\cdot)$ adds the out product of them to feed into the subsequent network, which is an explicit knowledge to help relevance modeling.

3) **Long-short-term attention-based pooling layer**: The long-term and short-term sequences correspond to the driver’s day-level interest and second-level interest, respectively. In our system, the driver’s selection of passenger orders is mainly a single decision, which is more affected by the current real-time click behavior than the long-term day-level sequence. Therefore, our second activation unit takes the driver long-term sequence representation vector obtained from the previous layer and the embedding vector of the driver’s real-time click sequence as the input is given a candidate passenger order $A$, as shown in Eq.(2)

$$
\beta_{t}^{long-short-term} = f(v_A, \alpha_{t-1}^{long-term}, c_1, c_2, \ldots, c_H) = b_1(\alpha_{t-1}^{long-term}, v_A) \alpha_{t-1}^{long-term} + \sum_{j=1}^{H} h(c_j, v_A) c_j,
$$

where $\{c_1, c_2, \ldots, c_H\}$ is the list of embedding vectors of behaviors of driver $U$ with a length of $H$. $v_A$ denotes the embedding vector of candidate passenger order $A$. In this way, $\beta_{t}^{long-short-term}$ varies over different passenger order. $b_1(\cdot)$ and $h(\cdot)$ is a feed-forward network with output as the activation weight, as illustrated in Fig.4.

4) **Explicit cross layer**: Both driver and passenger’s profile and contextual features are essential for a single decision. We follow the approach in [23] to implement feature crossing. The cross-network applies explicit feature crossing efficiently. Each cross layer has the following:

$$
x_{t+1} = x_0 x_0^T w_l + b_l + x_l = f(x_l, w_l, b_l) + x_l,
$$

FIGURE 3. System architecture of the proposed SLSTNN framework.
where \( x_l, x_{l+1} \in \mathbb{R}^d \) are column vectors denoting the outputs from the \( l \)-th and \((l+1)\)-th cross layers, respectively; \( w_l, b_l \in \mathbb{R}^d \) are the weight and bias parameters of the \( l \)-th layer. Each cross-layer adds back its input after a feature feature crossing \( f \), and the mapping function \( f : \mathbb{R}^d \mapsto \mathbb{R}^d \) fits the residual of \( x_{l+1} - x_l \). The cross vector product of \( X \) and \( X^T \) is used to obtain the cross combination of all elements. After layer stacking, any combined feature of bounded order can be obtained. When the cross layer is stacking with the \( l \) layer, the highest cross order can reach the order of \( l+1 \). Here we show some examples. For convenience, we first set \( b \) to be the zero vector. Let \( X_0 = \begin{bmatrix} x_{0,1} \\ x_{0,2} \end{bmatrix} \), then we have,

\[
X_1 = X_0X_0^TW_0 + X_0 = \begin{bmatrix} x_{0,1} \\ x_{0,2} \end{bmatrix} \begin{bmatrix} w_{0,1} \\ w_{0,2} \end{bmatrix} \begin{bmatrix} x_{0,1} \\ x_{0,2} \end{bmatrix} + \begin{bmatrix} x_{0,1} \\ x_{0,2} \end{bmatrix} \quad (4)
\]

We further calculate the \( X_2 \) and have the following,

\[
X_2 = X_0X_1^TW_1 + X_1. \quad (5)
\]

It can be seen from Eq.(4) and Eq.(5) that when the cross-layer is stacked with \( l \) layers, the highest cross order can reach \( l+1 \), and all the cross combinations are included, which is the subtlety of DCN [22].

5) MLP layer: The concatenation layer combines the outputs from long and short-term sequence representation and explicit feature cross representation and feeds the concatenated vector into an MLP layer.

The loss function is the log loss along with a regularization term

\[
\text{Loss} = -\frac{1}{N} \sum_{i=1}^{N} y_i \log (p_i) + (1 - y_i) \log (1 - p_i) + \lambda \sum_{l} \|w_l\|^2, \quad (6)
\]

where \( p_i \) are the probabilities computed from the model output, \( y_i \) are the true labels, \( N \) is the total number of inputs, and \( \lambda \) is the \( L_2 \) regularization parameter.

Once the training of the proposed model is finished, we predict the probability of a driver completing a passenger order based on the model.

The pseudocode of algorithm SLSTNN is as follows.

**Algorithm 1 SLSTNN algorithm**

- **Input:** training set \( D \), number of iterations \( M \), number of batches \( Q \), model parameter \( W \)
- **1:** Use Uber’s H3 address coding to encode the location information
- **2:** Random initialization model parameter \( W \)
- **for** \( n = 1, M \) **do**
- **4:** Randomly reorder the samples in the training set \( D \); \( \) **for** batch = 1, len( training_set )/Q **do**
- **6:** The embedding vector of the long-term sequence is substituted into the attention network of the first layer to obtain the embedding representation \( \alpha_{l-1} \) of the long-term sequence
- **7:** The embedding vector of short-term sequence and the vector \( \alpha_{l-1} \) of the previous step are substituted into the attention network of the second layer to obtain the embedding representation \( \beta_{l} \) of long-term and short-term sequence
- **8:** \( x_l \) is obtained by substituting the embedding vector of profile and context features into explicit cross layer
- **9:** Splice \( \alpha_{l-1}, \beta_{l} \) into MLP layer
- **10:** Calculate loss
- **11:** Updating model parameters through deep learning optimizer
- **12:** end for
- **end for**

V. EXPERIMENTAL RESULTS

A. DATASET

We evaluate the effectiveness and efficiency of our proposed SLSTNN on the following two datasets.

- **Gowalla Dataset:** Gowalla dataset records the time and point-of-interest information of check-ins from users in the location-based social networking site, Gowalla. For long-term sequence, we only use the past half year data. Items which have been observed by less than 30 users...
Logloss measures the distance between the predicted score is insensitive to the class imbalance problem. In contrast, one, it only considers the order of predicted instances and aspects. AUC measures the probability that a positive in-

The statistics of all the above datasets is shown in Table 2. Volume of Hitch Riding Services Dataset is much larger than Gowalla Dataset, which brings more challenges.

B. EVALUATION METRICS

We use Area Under the ROC curve (AUC), cross entropy (Logloss), and accuracy(Accuracy) to evaluate the performance of our model. Three metrics are widely used in many fields and they evaluate system performance from different aspects. AUC measures the probability that a positive instance can be ranked higher than a randomly chosen negative one, it only considers the order of predicted instances and is insensitive to the class imbalance problem. In contrast, Logloss measures the distance between the predicted score and the true label for each instance. Accuracy refers to the ratio of correctly predicted sample numbers to total predicted sample numbers. The formulas are shown below:

\[ \text{AUC} = \frac{\sum \text{pred}_{\text{pos}} > \text{pred}_{\text{neg}}}{\text{positiveNum} \times \text{negativeNum}}, \]

\[ \text{Logloss} = -\frac{1}{n} \sum_i \left[ y_i \log(y_i') + (1 - y_i) \log(1 - y_i') \right], \]

where \( \text{pred}_{\text{pos}} \) is the number of positive samples predicted, \( \text{pred}_{\text{neg}} \) is the number of negative samples predicted, \( \text{positiveNum} \) is the number of positive samples in the data set, \( \text{negativeNum} \) is the number of negative samples in the data set, \( y_i' \) is the probability computed from the model output, \( y_i \) is the true label, \( n \) is the total number of inputs, \( \text{pred}_{\text{right}} \) is number of correct samples predicted in the test set, \( \text{totalNum} \) is total number of samples in the test set.

C. PERFORMANCE COMPARISON

Baselines: We compare the performance of our model with the following baselines:

1) Logistic regression (LR) [23]. LR is one of the widely used shallow model before the emergence of deep networks for CTR prediction tasks. We consider it as a weak baseline.

2) Wide&Deep [5]. The Wide&Deep model has been widely adopted in real industrial applications. It consists of two parts: 1) the wide model, which handles the manually designed cross product features, and 2) the deep model, which automatically extracts nshort-termar relations among features.

3) DeepFM [6]. It imposes a factorization machine as a “wide” module in Wide&Deep saving feature engineering jobs.

4) DCN [22]. It keeps the benefits of a DNN model, and it introduces a novel cross network that is more efficient in learning feature interactions.

5) xDeepFM [7]. The integrated CIN and DNN modules help the model learn higher-order feature interactions both explicitly and implicitly.

6) SHAN [19]. It models long-term and short-term user behaviors using hierarchical attention networks.

7) DIN [1]. It uses the mechanism of attention to activate related user behaviors.

8) DIEN [2]. It not only captures sequential interests more effectively but also models the interest evolving process that is relative to the target item.

The long-sequence model [3, 4] is not suitable for our scene because the hitched ride itself is a low-frequency behavior and has a strong relationship with geographical location. The hitched ride is a single decision, and the behavior too long ago is not informative enough, so we do not compare it with the long-sequence model.

Settings: We implement all models with Python 3.7 and Tensorflow1.15. We run all models on 2 V100 GPUs. For SLSTNN, we set the embedding size 128. We also experimented with other settings and found that small changes did not change the results much. We tuned other parameters and set the learning rate as 0.001, dropout rate as 0.7, L2 loss weight as 0. We randomly selected 10% of training set as validation set. The optimal setting of other hyper-parameters can refer to the hyper-parameter investment part of the paper.

| Dataset                  | Hitch Riding Services Dataset | Gowalla Dataset |
|--------------------------|-------------------------------|----------------|
| Samples                  | 30 million                    | 642000         |
| Users                    | 1.38 million                  | 15171          |
| Item                     | 6.24 million                  | 13193          |
| Average length of long-term sequence | 15.2                        | 5.4            |
| Average length of short-term sequence | 7.6                          | 2.3            |

Accuracy = \( \frac{\text{pred}_{\text{right}}}{\text{totalNum}} \)
For baseline models, we used default parameter settings as in their original papers or implementations.

**Test Performance:** The comparison results of different methods on both Gowalla dataset and Hitch Riding Services dataset are shown in Table 3. All numbers are average results of ten repeated experiments. It is clear that all the deep networks significantly outperform the LR model, demonstrating the power of deep learning. DeepFM and xDeepFM with specifically designed structures perform better than Wide&Deep. DIN and DIEN sequence models with attention mechanisms perform better than DeepFM and xDeepFM, which simply sum or average the embedded vectors for sequence processing. On two datasets, the effect of DIEN is basically the same as that of DIN since there is no evolutionary process of users’ interest when making decisions, which does not reflect the advantages of DIEN. SLSTNN model has the longest training time among all models, which is related to the complexity of network structure. The proposed SLSTNN performs best among all the approaches.

**D. ABLATION STUDY**

To evaluate the contribution of each component for the performance, we conduct the ablation study. SLSTNN-L means only the user’s long-term taste is modeled, while SLSTNN-S only considers user’s short-term preference. SLSTNN-WITHOUT-CROSS means that the cross-network is removed. As shown in Fig.5 and Fig.6, SLSTNN-S outperforms SLSTNN-L, which demonstrates that short-term sequence information is more important on predicting the next item task. SLSTNN-WITHOUT-CROSS plays a greater role in the Hitch Riding Services Dataset than in the Gowalla dataset. This is mainly because the user portrait features of the Hitch Riding Services Dataset are richer, and feature crossover is more necessary.

**E. INVESTIGATION OF HYPER-PARAMETERS**

We also investigate the impact of hyper-parameters on SLSTNN from three aspects: 1) the number of hidden layers and the number of neurons per layer and activation functions in long-term-sequence attention-based pooling layer; 2) the number of hidden layers and the number of neurons per layer and activation functions in long-term and short-term attention-based pooling layer; and 3) The number of hidden layers and the number of neurons per layer in the explicit cross-layer. We conduct experiments by holding the best settings for the MLP layer while varying the settings of the other layers. The SLSTNN network adjusts the hyperparameters from bottom to top. After the optimal hyperparameters are determined for each layer, they are kept constant and the next hyperparameters are tuned.

1) **Long-term attention-based pooling layer:** The performance of the long-term sequence attention-based pooling layer increases with the depth of the network at the beginning, as shown in Fig.7(a). However, model performance degrades when the network depth is greater than 3. It might be caused by overfitting evidenced that we notice that the loss of training data still keeps decreasing when more hidden layers are added. The optimal pyramid structure is 64 neurons in the first layer, 32 neurons in the second layer, and 16 neurons in the third layer, as shown in Fig.7(b). The best activation function is leakyRelu, as shown in Fig.7(c).

2) **Long-short-term attention-based pooling layer:** The best performance is achieved when the number of network layers is 3, as shown in Fig.8(a). The optimal pyramid structure is 128 neurons in the first layer, 64 neurons in the second layer, and 32 neurons in the third layer, as shown in Fig.8(b). The best activation function is leakyRelu, as shown in Fig.8(c).

3) **Explicit cross layer:** The best performance is achieved when the number of layer is 3, as shown in Fig.9. If there are too many feature cross layers, the model is easily overfitted. When the number of layer exceeds 3, the model prediction time increases, but the model effect also decreases.

Based on the above analysis, we conclude that the more the number of neurons or the number of neurons, it is more easily to overfit. To avoid such problem, we employ dropout [24] technique in the training process.

**F. RESULT FROM ONLINE A/B TEST**

A/B test is a random trial, usually with two variables A and B. On the premise of maintaining a single variable, we use the control variable method to compare the data of a and B, and draw the experimental conclusion. We also conducted careful online A/B testing in the hitch riding services during the period from 2021-10 to 2021-11. As shown in Table 4, compared with the online model Xdeepfm, the proposed SLSTNN improves the conversion of orders completed by drivers by 2.14%. This is a significant improvement and demonstrates the effectiveness of the proposed approaches.

![Figure 5](image-url) **FIGURE 5. Influence of components at AUC.**

![Figure 6](image-url) **FIGURE 6. Influence of components at Logloss.**

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/
### TABLE 3. Model comparison in the hitch riding services dataset.

| Model        | Gowalla Dataset |           |           | Hitch Riding Services Dataset |           |
|--------------|-----------------|-----------|-----------|-------------------------------|-----------|
|              | AUC             | Logloss   | Accuracy  | Running time | AUC       | Logloss   | Accuracy  | Running time |
| LR [23]      | 0.8524          | 0.2598    | 0.751     | 6min          | 0.8024    | 0.2969    | 0.723     | 68min        |
| Wide&Deep [5]| 0.9322          | 0.2146    | 0.844     | 40min         | 0.8838    | 0.2339    | 0.825     | 395min       |
| DeepFM [6]   | 0.9351          | 0.2138    | 0.876     | 42min         | 0.8859    | 0.2309    | 0.852     | 415min       |
| DCN [22]     | 0.9354          | 0.2131    | 0.882     | 48min         | 0.886     | 0.2297    | 0.857     | 420min       |
| xDeepFM [7]  | 0.9419          | 0.2117    | 0.891     | 50min         | 0.8935    | 0.2265    | 0.861     | 437min       |
| SHAN [9]     | 0.9425          | 0.2178    | 0.897     | 47min         | 0.8986    | 0.2235    | 0.875     | 418min       |
| DIN [1]      | 0.9457          | 0.2019    | 0.898     | 52min         | 0.8978    | 0.2209    | 0.884     | 460min       |
| DIEN [2]     | 0.9437          | 0.2156    | 0.905     | 55min         | 0.8881    | 0.2307    | 0.889     | 540min       |
| SLSTNN (ours)| 0.9589          | 0.1983    | 0.913     | 56min         | 0.9144    | 0.2119    | 0.897     | 550min       |

### FIGURE 7. Effect of network hyper-parameters of long-term sequence attention-based pooling layer.

### FIGURE 8. Effect of network hyper-parameters of long-term and short-term sequence attention-based pooling layer.

### TABLE 4. Result from online a/b test on hitch riding services.

| Group       | Model  | Click through rate | Conversion rate |
|-------------|--------|--------------------|-----------------|
| Control group | xdeepfm | 49.10%             | 29.50%          |
| Experience group | SLSTNN | 51.06%             | 30.13%          |

SLSTNN. Now SLSTNN has been deployed online and serves the main traffic.

### VI. CONCLUSION

In this paper, we proposed a spatial-temporal-aware long short term neural network for personalized recommendation of location-based services. We employed Uber’s H3 address encoding to encode location information to address the problem of data sparsity. Meanwhile, we used a hierarchical attention mechanism to represent a user’s long- and short-term sequence vector, which varies with different target items. In addition, we used an explicit feature intersection network to implement the basic features, which reduces the workload of manual feature intersection. Our experiments show that our model outperforms state-of-the-art methods in terms of AUC, Logloss, and Accuracy on two real-world datasets.
The proposed SLSTNN improves drivers’ order completion conversion rate on the hitch riding services by 2.14%, which brings better revenue to the platform.

There are two directions for future work. First, there are more and more scenarios on hitch riding services. One model for each scenario has brought great development costs and a waste of resources. In the future, we will explore how to model multiple scenes. Second, our current modeling goal is too single. In the future, we need to integrate more goals into the model.

ACKNOWLEDGMENT

The authors would like to thank for Sheng Peng, Chongxiang Zheng who did the challenging work for online system implementation. Colleagues in charge of the project obtain features through MMAP (memory-based mapping file) and realize the call of TF deep model through Java language.

REFERENCES

[1] Guorui Zhou, et al, “Deep interest network for click-through rate prediction,” in Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2018, pp. 1059–1068.

[2] Guorui Zhou, Na Mou, et al, “Deep interest evolution network for click-through rate prediction,” in Proceedings of the 33nd AAAI Conference on Artificial Intelligence, 2019, Honolulu, USA.

[3] Qi Pi, Weijie Bian, et al, “Practice on Long sequential User Behavior Modeling for Click-through Rate Prediction,” in Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM, 2019, 1059–1068.

[4] Pi, Q., Zhou, G., Zhang, et al, “Search-based User Interest Modeling with Lifelong sequential Behavior Data for Click-Through Rate Prediction,” in Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 2685–2692.

[5] Cheng H. et al, “Wide & deep learning for recommender systems,” in Proceedings of the 1st Workshop on Deep Learning for Recommender Systems. ACM, 2016.

[6] Huifeng Guo, Ruiming Tang, et al, “DeepFM: A Factorization-Machine based Neural Network for CTR Prediction,” in Proc. 26th International Joint Conference on Artificial Intelligence, 2017, pp. 1725–1731.

[7] Huifeng Guo, Ruiming Tang, et al, “xDeepFM: Combining Explicit and Implicit Feature Interactions for Recommender Systems,” in Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2018, pp. 1754–1763.

[8] Hao Liu, Yongxin Tong, et al, “Hydra: A Personalized and Context-Aware Multi-Modal Transportation Recommendation System,” in Proc. 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 2314–2324.

[9] Wenjie Shang, Yang Yu, et al, “Environment Reconstruction with Hidden Confounders for Reinforcement Learning based Recommendation,” in Proc. 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 566–576.

[10] Anagnostopoulos, “Predictive Vehicle Ride Sharing Recommendation System for Smart Cities Commuting,” Journal of SMART CITIES, no. 1, pp. 177-191, January, 2021.

[11] Yang Liu, Cheng Lyu, et al, “Exploring a large-scale multi-modal transportation recommendation system,” Journal of Transportation Research Part C: Emerging Technologies, vol. 126, 103070, May, 2021.

[12] Paul Covington, Jay Adams, and Emre Sargin, “Deep neural networks for youtube recommendations,” in Proceedings of the 10th ACM Conference on Recommender Systems. ACM, 2016, pp. 191–198.

[13] Balázs Hidasi, Alexandros Karatzoglou, et al, “Session-based Recommendations with Recurrent Neural Networks,” in ICLR, 2016.

[14] Jiaxi Tang, Ke Wang, “Personalized Top-N sequential Recommendation via Convolutional sequence Embedding,” in Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, 2018, pp. 565–573.

[15] Xu Chen, Hongteng Xu, Yongfeng Zhang, et al, “sequential recommendation with User Memory Networks,” in Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, 2018, pp. 108–116.

[16] Fei Sun, Jun Liu, Jian Wu, et al, “BERT4Rec: sequential recommendation with Bidirectional Encoder Representations from Transformer,” in Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 2019, pp. 1441–1450.

[17] Yukuo Cen, Jianwei Zhang, Xu Zou, et al, “Controllable Multi-Interest Framework for Recommendation,” in Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 2942–2951.

[18] Jianxin Ma, Chang Zhou, Hongxia Yang, et al, “Disentangled Self-Supervision in sequential Recommendations,” in Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 483–491.

[19] Haochao Ying, Fuzhen Zhuang, Fuzheng Zhang, et al, “sequential Recommender System based on Hierarchical Attention Networks,” in Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, 2018, pp. 3926–3932.

[20] Zhi Bian, Shaojun Zhou, et al, “Denoising User-aware Memory Network for Recommendation,” in Proc. RecSys ’21: Fifteenth ACM Conference on Recommender Systems, 2021, pp. 400–410.

[21] Kun Zhou, Hui Yu, et al, “Filter-enhanced MLP is All You Need for Sequential Recommendation,” in Companion Proceedings of the Web Conference 2022. Accepted. DOI: https://doi.org/10.1145/3485447.3512111.

[22] Ruoxi Wang, et al, “Deep & Cross Network for Ad Click Predictions,” in Proc. 23th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2017, pp. 1–7.

[23] H. Brendan Mcmahan, H. Brendan Holt, et al, “Ad Click Prediction: a View from the Trenches,” in Proc. 23th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2014, pp. 1222–1230.

[24] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting,” Journal of Machine Learning Research, vol. 15, no. 1, pp. 1929–1958, January, 2014.

XIAOSHUANG SHENG received the Master degree from Zhejiang University, China, in 2015. She is currently working in Hello Inc. Her research interests include recommendations and trading ecology.
FAN WANG received his master’s degree from Wuhan University. He is currently the leader of the four-wheel algorithm of Hello Inc. His research interests include machine learning and deep learning.

YIFAN ZHU was born in 1995. He received the master’s degree from Hohai University. He is currently an engineer in Hello Inc. His research interests include recommendation system, named entity recognition, deep learning.

TENGFEI LIU was born in 1993. He received the master’s degree from East China Normal University. He is currently an engineer in Hello Inc responsible for the recommendation system.

HUAN CHEN is an algorithm researcher and leader of Hello Inc’s AI & Map Platform team. His research interests include recommendation and natural language processing.

***