2021 International Conference on Data Mining Workshops (ICDMW) | 978-1-6654-2427-1/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICDMW53433.2021.00012

DynaPosGNN: Dynamic-Positional GNN for Next POI Recommendation

Junbeom Kim*, Sihyun Jeong*, Goeon Park*, Kihoon Cha, Ilhyun Suh, Byungkook Oh
Samsung Research, Samsung Electronics. Co., Ltd, Seoul, Korea
{junbeom.kim, sihyun.jeong, goeon.park, kihoon.cha, ilhyun.suh, byungkook.oh}@samsung.com

***Abstract*—Location-based offline user activity data plays a key role to understand user interest and context in various user-friendly services. Especially, the next Point-Of-Interest (POI) recommendation task is focusing on the user's next location based on previous user trajectory. Previous studies for the next POI recommendations generally utilized the order of each visit as temporal characteristic and positions as spatial characteristic based on the Recurrent Neural Networks (RNN) mechanism. However, they are just interested in the next location right after a user's current trajectory not considering specific arrival time or the prediction time. In this paper, we propose Dynamic-Positional Graph Neural Network (DynaPosGNN), a novel next POI recommendation model considering specific arrival time in offline user activities. DynaPosGNN can predict users' next location by analyzing the correlation between arrival time and two spatial dynamic graphs called 'User-POI graph' and 'POI-POI graph'. Our experiments conducted on two real-world datasets show that DynaPosGNN outperforms existing next POI recommendation models.**

***Index Terms*—Next POI recommendation, Successive POI recommendation, Offline user activity, Dynamic Graph, Graph Neural Network**

I. Introduction

With a plethora of social networking service usages, service providers became able to collect a variety of private user information like a daily log. Most of the existing social networking services such as Foursquare, Facebook, Instagram, and Yelp have aspect of Location-based social networking (LBSN) services, which are based on user visit history and offline experience. This could be fundamental for an effective advertisement targeting strategy because we can identify user preference, social role, and basic profile from offline activities of users and utilize them. One effective targeting strategy is based on Point-Of-Interest (POI) recommendation. Research on POI recommendation [1] is mainly interested in the future location of users by analyzing previous visit history regardless of time. Similarly, the next POI recommendation called 'Successive POI Recommendation [2]' task finds the next location of users, but this recommendation is different from the traditional problem in terms of successive prediction. This means that the next POI recommendation problem is restricted to the next visit right after the current trajectory. Recently, most researchers are interested in the next POI recommendation task [3], [4] because this could be applied to a wide variety of services like nearby shop recommendations, attraction to a competitor, and route recommendation based on mobility flow analysis.

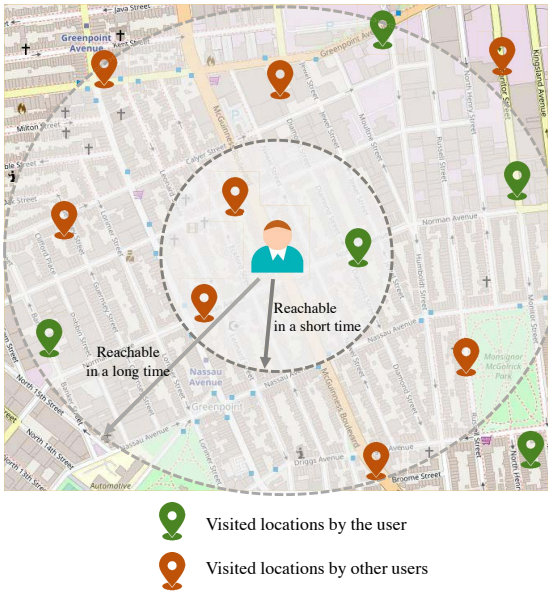

Fig. 1. Mobility prediction of a user based on the prediction time and visit history. The location where a user is expected to go depends on the prediction time, and in the case of a short time, the public's local visit history could be more helpful than the user's previous history.

Most of the next POI recommendation models rely on sequence pattern analysis [5]–[7]. Recurrent Neural Networks (RNN) enable the prediction model to capture the temporal patterns of user visit history. Recent works are utilizing spatial features like POI coordinates at RNN model for spatio-temporal pattern analysis [8]–[10]. Most of the existing issues are related to how to represent spatio-temporal features and how to capture the regular or irregular visit patterns. Recently, graph representation of POIs [11], [12] in the real world enables comprehensive analysis on the temporal correlation among the POIs, and regional proximity.

However, we should consider several existing challenges in the next POI recommendation task. First, previous methods predict where to go next right after the user's last visited location, which does not take into account the prediction time. In other words, the immediate next visit can be in 10

*Contributed Equally.

978-1-6654-2427-1/21/$31.00 ©2021 IEEE
DOI 10.1109/ICDMW53433.2021.00012

minutes or in 3 days, but the models are often using the same mechanism in both cases. 10 minutes is predictable at a distance where users can walk from their current location, but not three days. Secondly, the dynamicity of visit history is usually omitted by existing methodologies for representing spatial POI graphs. When existing methodologies represent relationships between POI in graph form, they usually include the number of consecutive visits and transfers between POI on a single edge, which does not include information about when each visit occurred. For example, there are several visits to a POI, which can be extremely old or very recent. The up-to-date information about a visit to a location is missing, even though it may be important information for the prediction of a user's next visit.

In this paper, we propose Dynamic-Positional Graph Neural Network (DynaPosGNN) to solve the existing problems mentioned above. Our main contributions are as follows:

- Firstly, DynaPosGNN is a more realistic next POI recommendation model where users are expected to visit at a particular point in time, rather than simply predicting where to go after a current visit. For example, a POI predicted to be after 10 minutes and 3 days from the user's current location should be considered differently depending on the user's geological accessibility in the given time (Figure 1). The model we propose carefully considers related factors in accordance with the prediction time.
- Secondly, we use multi-edge graph representation to consider the dynamics of user visit history. Unlike the existing methods, we utilize multiple edges between POI nodes and represent each visit history over time as distinct edges. This is because the history of consecutive visits between POI may occur a long time ago, or recently, or periodically. We applied these mechanisms to both the 'User-POI graph' containing the visit history of each user and the 'POI-POI graph' containing the visit history for the entire user set.

In the following section, we discuss the existing studies on the next POI recommendation problem and GNN-based approaches for general recommendation tasks. Section III explains our new problem formulation with given prediction time. Section IV mainly illustrates our dynamic graph-based POI recommendation model, DynaPosGNN. In the section V and VI show our evaluation and insights based on experimental results with real-world datasets.

II. Related Work

A. Next POI Recommendation

In the next POI recommendation task, the way to consider spatio-temporal characteristics of user visits is critical for model performances. Therefore, most of the recent research relies on both characteristics. To be specific, spatial analysis of existing studies is mostly focusing on the exact location of each POI, and related features are normally represented as coordinate or encoded ones. GeoSAN [13] used hierarchical gridding on original coordinates and applied self-attention mechanism to the model. STAN [14] replaced such GPS gridding with a simple linear interpolation for spatial discretization. Some models care about the distance between distinct POIs because accessibility is highly related to distances, practically. Therefore, with spatial features, the model could make predictions considering the geographical accessibility of users in local view. FPMC-LR [15] was based on a personalized Markov chain considering users' movement constraints like localized region. The spatial graph used in STP-UDGAT [12] represented connectivity between POIs based on distances. LSTPM [16] included geo-dilated LSTM to capture the probable geographical accessibility in user sequences.

Temporal analysis plays a key role in successive POI recommendation. Existing models considering temporal features have contribution on discovering long/short-term patterns and periodicity of users. Recently, RNN or LSTM based models performed nicely in this kind of successive POI recommendation with temporal characteristics. Previous studies on session-based recommender systems [17] such as GRU4Rec [18], ATEM [19], and SASRec [20] became fundamental research for the next-POI recommendation task. DeepMove [21] aimed to capture the periodicity of user behavior with recurrent module and historical attention. Similarly, LSTPM [16] used an LSTM-based model to find long/short-term analysis by focusing each user's daily sequences and whole sequence. Moreover, time consumption between POIs can figure out more realistic accessibility than spatial features sometimes. Time-LSTM [22] considered time intervals between actions when it applied LSTM to sequential recommendation. STP-UDGAT [12] utilized both time consumption graph (Temporal graph) and POI connectivity graph (Spatial graph) to analyze the various aspects of accessibility. ST-RNN [23] and STGN [24] were RNN and LSTM-based next POI recommendation models using both time and distance interval between POIs, respectively.

Besides considering spatio-temporal characteristics, user preference analysis is also a critical factor to achieve high performance on the next location prediction task. Prediction model could take account of visited POIs in the view of users. To capture the user preferences in POI category, CatDM [25] encoded the time series of POI categories for "Category-aware deep model". STP-UDGAT [12] used two kinds of preferences: a preference graph for global user view, and a personalized preference graph for individual user view.

B. GNN for Recommendation

Graph Neural Network (GNN) [26] is a recursive neural network for representing graph-structured data. Recently, GNN is widely used in recommendation systems [27]–[30]. SR-GNN [31] proposed a session-based recommendation model using GNN to represent complicated transitions of user actions. Similarly, GC-SAN [32] presented a novel graph contextual model adapting self-attention network. GC-MC [33] introduced graph convolutional matrix completion in recommendation system employing a bipartite User-Item graph as a representation of

movie ratings. However, these studies are not applied to the next POI recommendation task. STP-UDGAT [12] is the first research incorporating various factors to learn POI-POI and User-User relationships using GAT [34] to predict users' next visits.

III. Problem Definition

The existing next POI recommendation problem is mainly focusing on the prediction of the next location based on previous visit history. However, most of the real-world LBSN data consist of user visit trajectories of various lengths from a minute to numerous days. Therefore, a task for prediction of a user's next location with visit history is highly related to the prediction of the user's offline behavioral pattern and future check-in time, i.e. the update time of user state in the LBSN. To consider the overall user behavioral pattern, we formulated our problem as follows:

Let $U = \{u_1, u_2, ...u_M\}$ be a set of M users and $V = \{v_1, v_2, ..., v_N\}$ be a set of N POIs. Each user u_m has POI visit sequence $s_{u_m} = \{v_{m_1}, v_{m_2}, ...v_{m_i}\}$ with visit time $\{t_1, t_2, ...t_i\}$.

Suppose a target user u_T with visit sequence $s_u = \{v_1, v_2, ...v_c\}$ is currently in a place v_c at time t_c. The goal is to recommend ordered set of POIs from V for given prediction time $t_f > t_c$ where next POI visit v_f should be highly ranked.

IV. Proposed Model

As we redefine the problem including the prediction time, our recommendation model should take into accounts other potentially important other information such as time difference and geographical distance. DynaPosGNN computes both temporal and spatial features from user visit history and timing to be predicted to solve the problem. The key point of our approach is learning the influence of related features by prediction time. To achieve an efficient learning process considering every record, we should aggregate every user's visit history together while not missing each visit record. Some previous studies using a graph representation of sequential visit histories like SR-GNN [31] and STP-UDGAT [12] ignored the timestamp of each visit. We define a concept of dynamic graph maintaining each timestamp, and propose DynaPosGNN, a novel analytical approach for the graph.

A. Dynamic Graphs

In order to utilize users' POI visit history efficiently, we introduce two dynamic graphs (Figure 2).

1) User-POI Graph: A User-Item bipartite graph has been commonly used to express the interactions between user and item such as watching movies or clicking on the Internet shopping malls. This graph has a finite value on each edge. For example, an edge could have a binary value (0 or 1) when it comes to clicking item scenarios. Similarly, an edge could have a value from 1 to 5 for movie rating service. If we apply finite value on a single edge between user-item pairs (e.g. the number of visits) in a bipartite graph, we might also neglect the timestamp of each visit. To preserve each timing of visit occurrence, we define the multi-edge concept between nodes as follows.

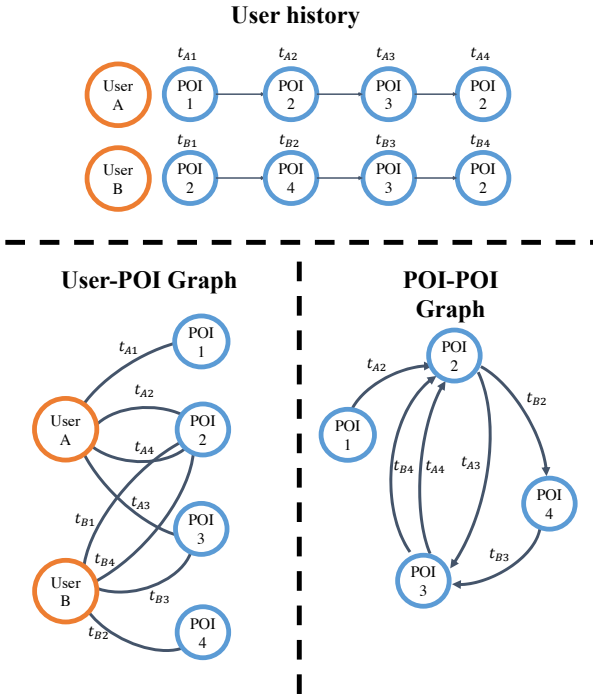

Fig. 2. An example of User-POI graph and POI-POI graph generation.

Definition 1 (User-POI Graph): User-POI Graph (UPG) is an undirected, dynamic bipartite graph that has all users U and POIs V as nodes. If user u visits POI v at time t, new edge e_t is created with visit time t as an edge feature, so that UPG can have multiple edges between two nodes. For a given time T, UPG_T is the subgraph of UPG which consists of edges created before T.

2) POI-POI Graph: The way to represent Item-Item or POI-POI graph is diverse by different criteria in the previous studies. SR-GNN [31], one of the sequential recommendation studies used an Item-Item directed graph in accordance with click sequences. Also, STP-UDGAT [12] defined different types of graphs by using order of visit, distance, and travel time as edge features. In this paper, we define a way to maintain more information on the POI-POI relationship than existing approaches.

Definition 2 (POI-POI Graph): POI-POI Graph (PPG) is a directed, dynamic graph which has all POIs V as node. If any user u visits POIs v_a, v_b in a row, new edge e_t is created from v_a to v_b with visit time t_b. PPG can also have multiple edges between two nodes. For given time T, PPG_T is the subgraph of PPG which consists of edges created before T.

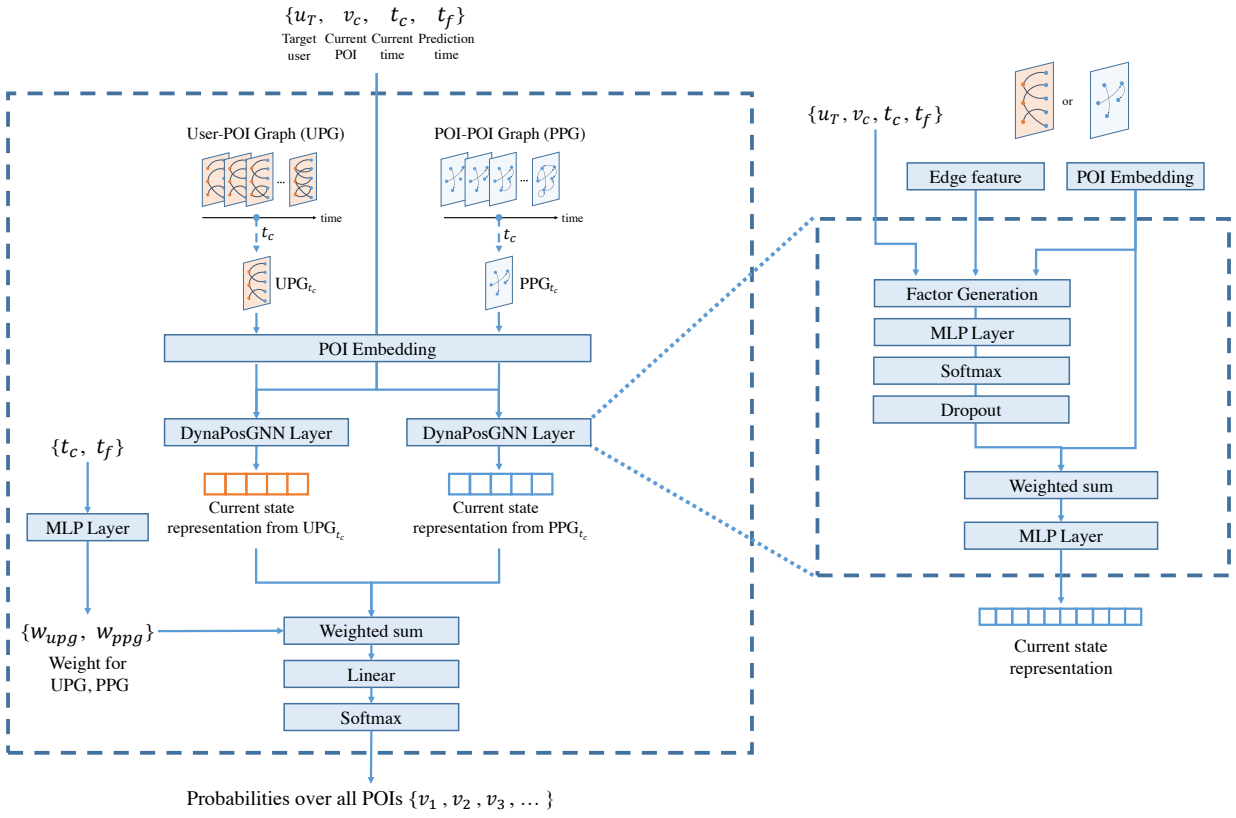

Fig. 3. The architecture of DynaPosGNN model.

B. Factors related to the influence of past records

As we preserve all past records, we have time information of past, current, and prediction time and geolocation of current and past POIs as well. Combining the given information, we can consider the following four factors for each past record. We denote past records as $\{p_1, p_2, ...p_n\}$ where each record $p_k = (v_k, t_k)$ has visited POI v_k and time t_k.

1) **Time interval between current and prediction time** (Δt_{cf}): Time interval between current time t_c and prediction time t_f

$$\Delta t_{cf} = t_f - t_c \tag{1}$$

is the key among four factors. Since human movement speed is limited, the boundary of possible movement is determined according to the prediction time. The degree of influence of sequential and historical effect also relevant to future time interval. Since Δt_{cf} can be arbitrarily large, we use $1/exp(\Delta t_{cf})$ in model to convert values between 0 and 1.

2) **Time interval between current and past record** p_k (Δt_{cp_k}): Without direct information about users, a record of user history is the only information that can help user understanding. However, the older the past records, the less influential they would be, and the more recent visits would have a significant impact on choosing the next destination. Hence for each past record t_p, the time interval

$$\Delta t_{cp_k} = t_c - t_{p_k} \tag{2}$$

between the current time t_c could be an important factor to determine the influence of past visit records at the present time. Since Δt_{cp_k} can also be large, we convert value to $1/exp(\Delta t_{cp_k})$ in model.

3) **Hour time difference between prediction time and past record** p_k (d_h): The behavioral pattern of people is often repeated over 24-hour cycle based on their lifestyle, and they are likely to visit the same places at similar times. For example, they usually have lunch at noon, and go to a pub in the evening. Thus we take into account the difference between time that visit occurred. We define "Hour time" as time excluding date and expressing in hour unit, which value is between 0 and 24. Then we define the difference between two "Hour time" as follows: For two "Hour time" $h_f, h_{p_k} \in [0, 24)$, the difference between two "Hour time" is

$$d_h(h_f, h_{p_k}) = \min(|h_f - h_{p_k} - 24|, |h_f - h_{p_k}|, |h_f - h_{p_k} + 24|) \tag{3}$$

which value is between 0 and 12.

4) **Geographical distance** (d_{hav}): If the next prediction time is close to the current time like 10 minutes, the places where the user can go in time are given priority for prediction. On the other hand, if there is a large gap between the current time and the prediction time, the impact of geographical distances would be reduced naturally. We use the general formula for geographical distance, haversine distance, defined as

$$d_{hav}(v_c, v_{p_k}) = Haversine(v_c, v_{p_k}). \quad (4)$$

Similar to time intervals, the range of haversine distance is large. Hence, we convert distance to $1/exp(d_{hav})$.

C. DynaPosGNN

With the four factors we mentioned above, DynaPosGNN computes aggregation weights that indicate the correlation between the current user state and related POI visit records.

Formally, GNNs consist of two steps; Aggregate and Combine [35]. In the Aggregate step, GNN gathers features from connected neighbor nodes. After aggregation, the process of combining gathered information is performed. In the view of GNN, DynaPosGNN also gathers features from connected neighbors in the Aggregate step. DynaPosGNN layer uses all edges in User-POI Graph but uses outward edge only in POI-POI Graph since aggregated information from outward edges contains next movements of the user who visit a current location.

In Combine step, DynaPosGNN calculates the importance of each visit record based on the current state. For each connection $\{e_1, ...e_k\}$ between target user u_t or current location v_c and v_k with edge feature t_k, the weight of visit record use all given information in four factor formulas defined as equation (1-4).

$$w_k = f\big(\Delta t_{cf}, \Delta t_{ck}, d_h(t_f - t_k), d_{hav}(v_c, v_k)\big)$$

In DynaPosGNN layer, we use Multi-Layered Perceptron (MLP) to model and learn weight function f. With the weight calculated above, the output embedding of DynaPosGNN layer is

$$\text{Embedding}(v_c, t_c, t_f) = \text{MLP}\Big(\sum_{i \in \mathcal{N}_c} \alpha_i \ \text{Emb}_i\Big)$$

where α_i is normalized weight using softmax function.

$$\alpha_k = \text{Softmax}(w_k) = \frac{\exp(w_k)}{\sum_{i \in \mathcal{N}_c} \exp(w_i)}$$

From DynaPosGNN layers, we obtain two current state representations, from UPG_{t_c} and PPG_{t_c}. Both outputs have the information of the current state, one from the view of the user and the other from the view of the current location. The relative importance between two representations also varies depending on prediction time. To enable the model to learn this relation, we add a representation weight generator that receives current time t_c and prediction time t_f. With weights, we compute the probability distribution over all POIs as follows:

$$\mathbf{p} = \text{Softmax}\big(\mathbf{W_s}(w_{upg}\mathbf{h_{upg}} + w_{ppg}\mathbf{h_{ppg}})\big)$$

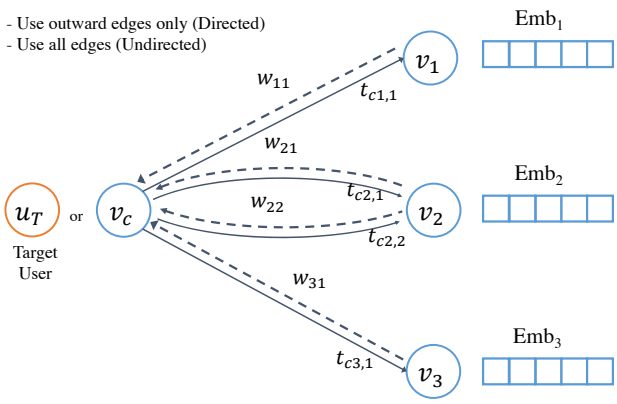

Fig. 4. Structure of DynaPosGNN Layer.

where $\mathbf{W_s} \in \mathbb{R}^{d \times N}$ is learnable weight with hidden dimension d and number of POIs N, $\mathbf{h_{upg}}$, $\mathbf{h_{ppg}}$ are current state representation of User-POI graph and POI-POI graph.

V. Experiment Setup

To experiment with our proposed model with other state-of-the-art methods, we conduct experiments on two widely used real-world LBSN datasets, namely Gowalla [36] and Foursqaure [37].

A. Datasets

Foursquare is collected in New York City and Tokyo from April 2012 to February 2013 while Gowalla is collected from all over the world from February 2009 to October 2010. To experiment in a more realistic environment, we divide Gowalla into city-level datasets. Then, we use four cities as experimental data; New York City (FS-NYC), Tokyo (FS-TKY) from Foursquare and San Francisco (GW-SF), Austin (GW-AUS) from Gowalla. For all city-level datasets, we remove users with less than 10 check-ins and POIs with less than 10 visits by users. The statistics of the preprocessed datasets are shown in Table I. The density is calculated based on Bert4Rec [38]. Lastly, we take the first 65% of each user's trajectory as a training set, the next 5% as a validation set, and the remaining 30% as a testing set.

TABLE I
Statistics of LBSN Datasets.

	#User	#POI	#Check-in	Density
FS-NYC	950	1,167	45,632	4.12%
FS-TKY	2,274	2,865	333,184	5.11%
GW-SF	1,049	1,650	59,679	3.45%
GW-AUS	2,751	2,994	186,863	2.27%

B. Baselines

- TOP: This method ranks the POIs based on the frequency of visits by all users.

TABLE II
PERFORMANCE COMPARISON ON FOUR LBSN DATASETS IN TERMS OF *HR@K* AND *NDCG@K*.

					FS-NYC			
	HR@1	**HR@5**	**HR@10**	**HR@20**	**NDCG@1**	**NDCG@5**	**NDCG@10**	**NDCG@20**
TOP	0.0234	0.0860	0.1313	0.1748	0.0234	0.0546	0.0692	0.0801
UTOP	0.1879	0.3685	0.4292	0.4790	0.1879	0.2843	0.3039	0.3166
RNN	0.1801	0.3128	0.3600	0.4090	0.1801	0.2519	0.2671	0.2795
LSTM	0.1993	0.3579	0.4092	0.4558	0.1993	0.2842	0.3008	0.3126
GRU	0.1941	0.3448	0.3953	0.4455	0.1941	0.2751	0.2915	0.3042
DeepMove	0.1631	0.2904	0.3347	0.3791	0.1631	0.2302	0.2447	0.2559
LSTPM	0.1906	0.3420	0.3907	0.4381	0.1906	0.2705	0.2863	0.2982
STGN	0.1153	0.1537	0.1763	0.2072	0.1153	0.1353	0.1426	0.1504
STP-UDGAT	0.1127	0.2366	0.2902	0.3480	0.1127	0.1782	0.1955	0.2100
DynaPosGNN	**0.2198**	**0.3824**	**0.4363**	**0.4956**	**0.2198**	**0.3073**	**0.3246**	**0.3396**
					FS-TKY			
	HR@1	**HR@5**	**HR@10**	**HR@20**	**NDCG@1**	**NDCG@5**	**NDCG@10**	**NDCG@20**
TOP	0.0381	0.1326	0.1746	0.2183	0.0381	0.0871	0.1005	0.1115
UTOP	0.2044	0.4689	0.5620	0.6295	0.2044	0.3437	0.3741	0.3913
RNN	0.2216	0.4398	0.5152	0.5828	0.2216	0.3379	0.3623	0.3795
LSTM	0.2359	0.4873	0.5682	0.6354	0.2359	0.3700	0.3962	0.4133
GRU	0.2326	0.4840	0.5667	0.6335	0.2326	0.3665	0.3934	0.4104
DeepMove	0.2498	0.4520	0.5320	0.6006	0.2498	0.3568	0.3828	0.4002
LSTPM	0.2512	0.4852	0.5705	0.6483	0.2512	0.3756	0.4033	0.4230
STGN	0.1594	0.2669	0.3229	0.3838	0.1594	0.2156	0.2337	0.2491
STP-UDGAT	0.0779	0.2143	0.2875	0.3683	0.0779	0.1480	0.1717	0.1921
DynaPosGNN	**0.2686**	**0.5181**	**0.6058**	**0.6799**	**0.2686**	**0.4012**	**0.4297**	**0.4485**
					GW-SF			
	HR@1	**HR@5**	**HR@10**	**HR@20**	**NDCG@1**	**NDCG@5**	**NDCG@10**	**NDCG@20**
TOP	0.0091	0.0407	0.0722	0.1050	0.0091	0.0245	0.0349	0.0432
UTOP	0.1069	0.2334	0.2888	0.3441	0.1069	0.1733	0.1911	0.2051
RNN	0.0935	0.1871	0.2298	0.2793	0.0935	0.1432	0.1569	0.1694
LSTM	0.1171	0.2290	0.2801	0.3352	0.1171	0.1762	0.1928	0.2067
GRU	0.1047	0.2072	0.2546	0.3108	0.1047	0.1588	0.1741	0.1883
DeepMove	0.0898	0.1802	0.2245	0.2730	0.0898	0.1369	0.1512	0.1634
LSTPM	0.1206	0.2367	0.2974	0.3622	0.1206	0.1813	0.2008	0.2172
STGN	0.0734	0.1231	0.1433	0.1707	0.0734	0.0997	0.1062	0.1131
STP-UDGAT	0.0516	0.1215	0.1608	0.2071	0.0516	0.0875	0.1002	0.1119
DynaPosGNN	**0.1363**	**0.2760**	**0.3401**	**0.4104**	**0.1363**	**0.2093**	**0.2300**	**0.2478**
					GW-AUS			
	HR@1	**HR@5**	**HR@10**	**HR@20**	**NDCG@1**	**NDCG@5**	**NDCG@10**	**NDCG@20**
TOP	0.0156	0.0543	0.0829	0.1168	0.0156	0.0360	0.0453	0.0538
UTOP	0.1000	0.2134	0.2653	0.3205	0.1000	0.1593	0.1761	0.1901
RNN	0.0844	0.1652	0.2094	0.2609	0.0844	0.1267	0.1409	0.1539
LSTM	0.1053	0.2079	0.2582	0.3175	0.1053	0.1592	0.1755	0.1904
GRU	0.0981	0.1961	0.2452	0.3037	0.0981	0.1495	0.1653	0.1800
DeepMove	0.0763	0.1569	0.1993	0.2502	0.0763	0.1179	0.1316	0.1444
LSTPM	0.0992	0.2055	0.2647	0.3311	0.0992	0.1542	0.1733	0.1901
STGN	0.0617	0.1227	0.1562	0.2026	0.0617	0.0931	0.1039	0.1157
STP-UDGAT	0.0338	0.0940	0.1350	0.1851	0.0338	0.0645	0.0778	0.0904
DynaPosGNN	**0.1223**	**0.2556**	**0.3258**	**0.3978**	**0.1223**	**0.1916**	**0.2143**	**0.2326**

- UTOP: This method ranks the POIs based on the frequency of visits by each user.
- RNN [39]: This method is a typical model for learning time-series data. It learns temporal dependencies between visited POIs.
- LSTM [40]: This method is one of the variants of RNN model. It is composed of three gates and cell state module to improve learning short-term and long-term preferences.
- GRU [41]: This method is one of the variants of RNN model. It contains two gates to model short-term and long-term preferences.
- DeepMove [21]: This method uses RNN module for learning current trajectory and attention mechanism for learning historical trajectory.
- LSTPM [16]: This method is a LSTM-based model. It models long-term preferences by using a nonlocal networks and short-term preferences by using a geo-dilated RNN module.
- STGN [24]: This method adds two pairs of spatio-temporal gates into the LSTM module to learn short-term preferences and long-term preferences.
- STP-UDGAT [12]: This method is a GNN-based model. It learns POI-POI and User-User relationships from each user as well as from all users in terms of spatial, temporal, and preference factors.

C. Evaluation Metrics

To evaluate performance how these models predict the next POI accurately, we exploit two well-known metrics that are mainly used to evaluate recommendation systems; *Hit Rate*

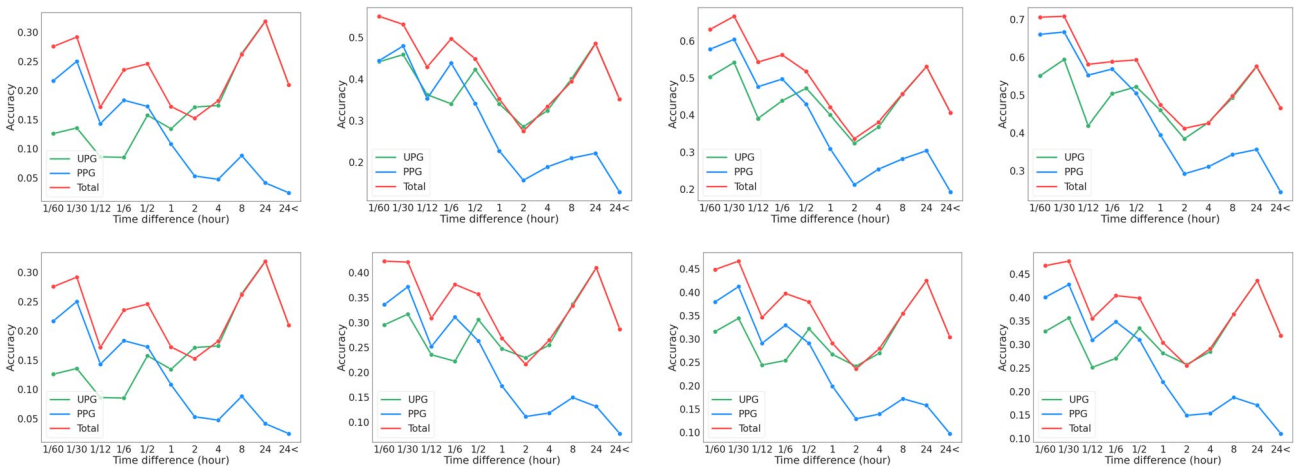

Fig. 5. Results of evaluating the impact of User-POI graph and POI-POI graph on performance (FS-NYC). The first row is HR@k of result from User-POI graph, POI-POI graph, and full module and the second row is NDCG@k for k=1,5,10,20.

(HR) and *Normalized Discounted Cumulative Gain (NDCG)*. *HR@K* measures whether a target location is within the top-*K* recommendation list, and *NDCG@K* measures the quality of recommendation according to the ranking of the target location among top-*K*. In this paper, we use cutoff $K \in \{1, 5, 10, 20\}$ for evaluation.

D. Experimental Settings

For our proposed model, we select hyperparameters (embedding dimension of location and dropout rate) with optimization on FS-NYC, and apply the same setting to other datasets. As a result, we set the embedding dimension of location to 512 and the dropout rate as 0.5. we adopt an early stopping strategy with patience of 10 on the validation loss. We use Adam optimizer with batch size of 64 and learning rate of 0.001. For all baselines, the first 70% is used for a training set, and the remaining 30% is used for a testing set. For RNN, LSTM, and GRU, we set both cell state size and POI embedding dimension to 500. For other baselines, we set the same hyper-parameters based on each model's framework.

VI. Results

A. Method Comparison

Experimental results are summarized in Table II. The best score is marked in bold, and the second-best score is underlined. We observe that our model outperforms all baselines on four datasets in both metrics. Compared to the second-best results, our proposed model shows an average improvement of 11.23%, and 12.26% in terms of *HR* and *NDCG* respectively on all datasets. In the results, all methods perform worse in low-density datasets such as GW-AUS and GW-SF. However, the highest performance gain is of 19.71% and 20.07% respectively in *HR* and *NDCG* on the GW-AUS which is the lowest density among all datasets. Therefore, we can figure out that our model works robustly. LSTPM is the most competitive method among the baselines, and LSTM has always higher scores than RNN and GRU. Interestingly, UTOP shows the second-highest result in a few datasets. This further verifies that a lot of users go to the places where they have already visited.

B. Analysis on Parameters

In this section, we vary major parameters of DynaPosGNN: embedding dimension and dropout. Firstly, Figure 6 shows the impact of embedding dimension in every dataset. To summarize, we figure out 512 is the best embedding dimension of location for DynaPosGNN. In terms of stability, the gap between the best/worst recommendation performance is 2.87%, and that percentage is allowable. Second, Figure 6 shows that the best dropout rate is 0.5 which is generally used for other GNN algorithms.

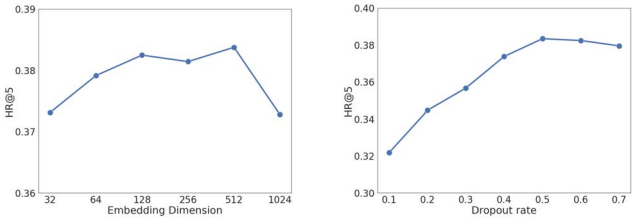

Fig. 6. Impact of embedding dimension and dropout rate.

C. Effectiveness of User-POI graph and POI-POI graph

The major factor in recommendation depends on the interval between the future prediction time and the current time, and two dynamic graphs include different aspects of user history, respectively. As we mentioned in the previous section, the User-POI graph is a user-centric POI graph and the POI-POI graph is a global view of overall mobility flow.

According to our investigation, each graph helps recommendation performance in different importance if the time difference between the current time and prediction time is short or long. Figure 5 is a result of our ablation study on FS-NYC dataset to observe relative performance change by varying time differences. When a prediction time of a user is close to the current time, locally close POIs have more priority based on the POI-POI graph. The result indicates DynaPosGNN adopts the impact of the POI-POI graph more than the User-POI graph in this scenario because the range of movement of the user is limited. These two graphs are complementary, and we found cross points of two lines about a 30-minute difference. The red lines in Figure 5 show complemented final results with two dynamic graphs.

D. Impact of four factors

We conducted a supplementary evaluation to discover the effectiveness of four factors in DynaPosGNN. We introduced the four key factors in Section IV: Time interval between current and prediction time (Δt_{cf}), Time interval between current and past record (Δt_{cp}), Hour time difference between prediction time and past record (d_h), and Geographical distance (d_{hav}). This experiment aims to evaluate the performance of the next POI recommendation, excluding each factor in the model at each trial. The comparison of the results of all factors is shown in Figure 7.

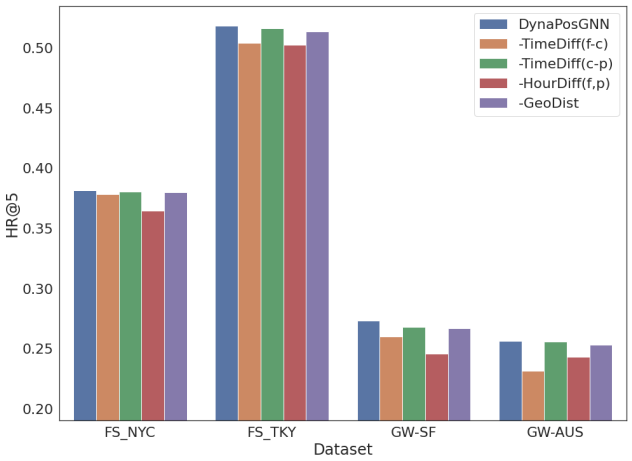

Fig. 7. Performance evaluation by factors.

From these results, we discovered that 'Hour time difference' had the greatest impact on the performance. The next was 'Time interval between current and prediction time (hereafter referred to as the 'Time interval')' which reflects the difference between the prediction time and the current time. However, 'Hour time difference' means the difference in time excluding date, and 'Time interval' means the absolute time difference including the date. In the case of 'Hour time difference', for example, the behavioral characteristics would not be much different at 6 PM and 7 PM in general when we think of dinner time. In addition, 'Time interval' plays a key role in the comprehensive utilization of two dynamic graphs as the time it takes for a user to arrive at the next POI. Although methods such as LSTPM [16] and DeepMove [21] utilize users' time-specific activity information similar to the 'Hour time difference', there is a problem of performance degradation in the environment with a lack of data because each time slot is computed independently without association. DynaPosGNN has great strength in that it provides a flexible understanding of the users' activity time by 'Hour time difference'.

VII. Conclusion

Offline user activity data from LBSN services is important for advertising, traffic management, and user understanding-based services. Accordingly, the next POI recommendation task became a major challenge with LBSN data. In this paper, we pointed out that most of the previous next POI recommendation research is interested in the next POI just right after the current user location regardless of the prediction time. What problem we deal with is the next POI recommendation considering specific future timing that a user will arrive at a certain location. To achieve high accuracy on recommendation, we applied two dynamic graphs with multi-edge representation to our approach, DynaPosGNN. It adjusts the impact of user-centric trajectory and public movement records by given prediction time. Evaluation on two real-world datasets showed that DynaPosGNN has higher performance in terms of HR and NDCG than existing methods. Moreover, the ablation study on two graphs illustrated an efficient recommendation mechanism based on remarkable complementary of our approach.

For one of the probable future research directions, we plan to redefine the human mobility prediction problem as a trajectory recommendation. In this problem, what we aim to recommend after the current user location would be several sets of sequential next POIs. We expect that detailed context and behavioral semantic information of users could be derived by accurately predicted future trajectory, not a single next POI.

References

[1] Y. Yu and X. Chen, "A survey of point-of-interest recommendation in location-based social networks," in *Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[2] M. Ye, P. Yin, and W.-C. Lee, "Location recommendation for location-based social networks," in *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*, 2010, pp. 458–461.

[3] Z. Zhang, C. Li, Z. Wu, A. Sun, D. Ye, and X. Luo, "Next: a neural network framework for next poi recommendation," *Frontiers of Computer Science*, vol. 14, no. 2, pp. 314–333, 2020.

[4] S. Feng, L. V. Tran, G. Cong, L. Chen, J. Li, and F. Li, "Hme: A hyperbolic metric embedding approach for next-poi recommendation," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020, pp. 1429–1438.

[5] K. Zhao, Y. Zhang, H. Yin, J. Wang, K. Zheng, X. Zhou, and C. Xing, "Discovering subsequence patterns for next poi recommendation." in *IJCAI*, 2020, pp. 3216–3222.

[6] Y. Wu, K. Li, G. Zhao, and X. Qian, "Long-and short-term preference learning for next poi recommendation," in *Proceedings of the 28th ACM international conference on information and knowledge management*, 2019, pp. 2301–2304.

[7] Y. Wu, K. Li, G. Zhao, and Q. Xueming, "Personalized long-and short-term preference learning for next poi recommendation," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[8] L. Huang, Y. Ma, S. Wang, and Y. Liu, "An attention-based spatiotemporal lstm network for next poi recommendation," *IEEE Transactions on Services Computing*, 2019.

[9] Y.-C. Chen, T. Thaipisutikul, and T. K. Shih, "A learning-based poi recommendation with spatiotemporal context awareness," *IEEE Transactions on Cybernetics*, 2020.

[10] T. Qian, B. Liu, Q. V. H. Nguyen, and H. Yin, "Spatiotemporal representation learning for translation-based poi recommendation," *ACM Transactions on Information Systems (TOIS)*, vol. 37, no. 2, pp. 1–24, 2019.

[11] Y.-S. Lu and J.-L. Huang, "Glr: A graph-based latent representation model for successive poi recommendation," *Future Generation Computer Systems*, vol. 102, pp. 230–244, 2020.

[12] N. Lim, B. Hooi, S.-K. Ng, X. Wang, Y. L. Goh, R. Weng, and J. Varadarajan, "Stp-udgat: Spatial-temporal-preference user dimensional graph attention network for next poi recommendation," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 845–854.

[13] D. Lian, Y. Wu, Y. Ge, X. Xie, and E. Chen, "Geography-aware sequential location recommendation," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 2009–2019.

[14] Y. Luo, Q. Liu, and Z. Liu, "Stan: Spatio-temporal attention network for next location recommendation," *arXiv preprint arXiv:2102.04095*, 2021.

[15] C. Cheng, H. Yang, M. R. Lyu, and I. King, "Where you like to go next: Successive point-of-interest recommendation," in *Twenty-Third international joint conference on Artificial Intelligence*, 2013.

[16] K. Sun, T. Qian, T. Chen, Y. Liang, Q. V. H. Nguyen, and H. Yin, "Where to go next: Modeling long-and short-term user preferences for point-of-interest recommendation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020, pp. 214–221.

[17] S. Wang, L. Cao, Y. Wang, Q. Z. Sheng, M. A. Orgun, and D. Lian, "A survey on session-based recommender systems," *ACM Computing Surveys (CSUR)*, vol. 54, no. 7, pp. 1–38, 2021.

[18] D. Jannach and M. Ludewig, "When recurrent neural networks meet the neighborhood for session-based recommendation," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, 2017, pp. 306–310.

[19] S. Wang, L. Hu, L. Cao, X. Huang, D. Lian, and W. Liu, "Attention-based transactional context embedding for next-item recommendation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[20] W.-C. Kang and J. McAuley, "Self-attentive sequential recommendation," in *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 197–206.

[21] J. Feng, Y. Li, C. Zhang, F. Sun, F. Meng, A. Guo, and D. Jin, "Deepmove: Predicting human mobility with attentional recurrent networks," in *Proceedings of the 2018 world wide web conference*, 2018, pp. 1459–1468.

[22] Y. Zhu, H. Li, Y. Liao, B. Wang, Z. Guan, H. Liu, and D. Cai, "What to do next: Modeling user behaviors by time-lstm." in *IJCAI*, vol. 17, 2017, pp. 3602–3608.

[23] Q. Liu, S. Wu, L. Wang, and T. Tan, "Predicting the next location: A recurrent model with spatial and temporal contexts," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, 2016.

[24] P. Zhao, A. Luo, Y. Liu, F. Zhuang, J. Xu, Z. Li, V. S. Sheng, and X. Zhou, "Where to go next: A spatio-temporal gated network for next poi recommendation," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[25] F. Yu, L. Cui, W. Guo, X. Lu, Q. Li, and H. Lu, "A category-aware deep model for successive poi recommendation on sparse check-in data," in *Proceedings of The Web Conference 2020*, 2020, pp. 1264–1274.

[26] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE transactions on neural networks*, vol. 20, no. 1, pp. 61–80, 2008.

[27] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, "Graph neural networks for social recommendation," in *The World Wide Web Conference*, 2019, pp. 417–426.

[28] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 974–983.

[29] Q. Tan, N. Liu, X. Zhao, H. Yang, J. Zhou, and X. Hu, "Learning to hash with graph neural networks for recommender systems," in *Proceedings of The Web Conference 2020*, 2020, pp. 1988–1998.

[30] S. Wang, L. Hu, Y. Wang, X. He, Q. Z. Sheng, M. A. Orgun, L. Cao, F. Ricci, and P. S. Yu, "Graph learning based recommender systems: A review," *arXiv preprint arXiv:2105.06339*, 2021.

[31] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan, "Session-based recommendation with graph neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 346–353.

[32] C. Xu, P. Zhao, Y. Liu, V. S. Sheng, J. Xu, F. Zhuang, J. Fang, and X. Zhou, "Graph contextualized self-attention network for session-based recommendation." in *IJCAI*, vol. 19, 2019, pp. 3940–3946.

[33] R. v. d. Berg, T. N. Kipf, and M. Welling, "Graph convolutional matrix completion," *arXiv preprint arXiv:1706.02263*, 2017.

[34] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[35] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *International Conference on Learning Representations*, 2019.

[36] E. Cho, S. A. Myers, and J. Leskovec, "Friendship and mobility: user movement in location-based social networks," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2011, pp. 1082–1090.

[37] D. Yang, D. Zhang, V. W. Zheng, and Z. Yu, "Modeling user activity preference by leveraging user spatial temporal characteristics in lbsns," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 1, pp. 129–142, 2014.

[38] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang, "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.

[39] J. L. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.

[40] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[41] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.