2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)

High-Quality Prediction of Tourist Movements using Temporal Trajectories in Graphs

Shima Moghtasedi[1], Cristina Ioana Muntean[2], Franco Maria Nardini[2], Roberto Grossi[1], and Andrea Marino[3]

[1]Dipartimento di Informatica, Università di Pisa, Italy,
Email: {shima.moghtasedi,grossi}@di.unipi.it

[2]ISTI-CNR, Pisa, Italy
Email: {cristina.muntean, francomaria.nardini}@isti.cnr.it

[3]Dipartimento di Statistica, Informatica, Applicazioni "G. Parenti", Università di Firenze, Italy,
Email: andrea.marino@unifi.it

***Abstract*—In this paper, we study the problem of predicting the next position of a tourist given his history. In particular, we propose a model to identify the next point of interest that a tourist will visit in the future, by making use of similarity between trajectories on a graph and taking into account the spatial-temporal aspect of trajectories. We compare our method with a well-known machine learning-based technique, as well as with a popularity baseline, using three public real-world datasets. Our experimental results show that our technique outperforms state-of-the-art machine learning-based methods effectively, by providing at least twice more accurate results.**

***Index Terms*—PoI prediction, temporal trajectory, similarity, graph**

I. Introduction

In the last years, the increasing availability of user-generated content has driven the development of new services for tourism. These services target different tasks ranging from prediction of next places to visit [2], [8] and travel recommendation [15] to intelligent planning [3], booking, etc. These systems are used within location-based social networks and targeted mobile applications to help tourists in achieving a better experience in their activities. In this paper, we study the next-PoI prediction task (next-PoI), which aims at identifying the Point of Interest (in short, PoI) that a tourist will visit with highest probability in the future. Usually, this problem is solved by using machine learning-based techniques. The overall performance of a machine learning-based technique for the next-PoI prediction depends on the effectiveness of the proposed feature set. In particular, given a trajectory dataset, any machine learning model needs to extract a set of useful features for prediction, which is a challenging task. In this work, we introduce a new graph-based method targeting the next-PoI prediction problem. The objective of next-PoI is to reflect the similar behavior of the past tourists, to predict the next position that a tourist will visit. The proposed approach exploits the structural and temporal information available in the past trajectories of tourists, i.e., the sequence of PoIs visited by tourists in a city. We introduce the *trajectory graph*, a graph modeling the behavior of a given set of tourists in a city and we define a similarity function that allows us to identify the similar trajectories over this graph. The new similarity function uses both structural and temporal information to quantify how much two trajectories are close each other. We conduct a comprehensive evaluation of our proposed method and of state-of-the-art competitors on three public datasets, representing the movements of tourists in Pisa, Rome, and Florence. The performance of the methods is evaluated in terms of well-known Information Retrieval metrics, showing that our technique outperforms state-of-the-art machine-learning based competitors by providing at least twice more accurate results.

IEEE/ACM ASONAM 2020, December 7-10, 2020
978-1-7281-1056-1/20/$31.00 © 2020 IEEE

II. Related Work

A. PoI Prediction/Recommendation

An approach to solve the next-PoI prediction problem uses trajectory pattern mining to devise temporally-annotated common patterns (trajectories) of movements from data. Trajectories are a concise representation of the behavior of moving objects as sequences of regions frequently visited with typical travel time. Trajectory-based models are exploited in [7], to predict the most likely locations that are of interest for a user.

In a more generic context, that of movements in a city, a similar problem is predicting the next *check-in*. Noulas et al. [9] study the problem of predicting the next venue a mobile user will visit (in foursquare-like terminology, the next *check-in*), by exploring the predictive power offered by different aspects of the user behavior. In the context

of tourism trajectories, in [8] authors propose a broader set of features, originated from a Flickr dataset, capturing more dimensions of the behavior of tourists. They cast the prediction problem into a "*learning to rank*" task, which allows us to use two effective Machine Learning techniques (Ranking SVM and GBRT) to solve it. Other works predict the next destination of individual tourists using the GPS tracking data, which is not needed by our approach [13], [14].

Similar efforts have been spent in solving the PoI recommendation task. Here, the problem deals with generating a list of potential interesting PoIs for a tourist [3]. It differs from the next-PoI prediction task as it aims at maximizing the satisfaction of the user during the whole tour of the city, while the latter aims at identifying only one PoI as the next candidate to visit.

Newer methods exploit deep learning algorithms based on recurrent and/or sequential models in order to predict the next PoI such as works in [1], [12]. However these methods require a lot of data samples to give satisfactory results. Regarding tourism, such a big amount of data is not currently available. This is not needed by our approach, which does not use data consuming learning models, and requires smaller amounts of data for prediction.

B. Trajectory Similarity

Due to a wide range of application for trajectory similarity, many different similarity functions have been used, e.g., Euclidean distance (ED), Dynamic Time Warping (DTW), distance based on Longest Common Subsequence (LCSS) and edit distance based functions (EDR, ERP). Wang et al. [10] presents a comparative study between widely used trajectory similarity measures. For trajectory analysis in networks, most of the proposed functions are based on the spatial information of trajectories, without considering the proximity between them in graphs, e.g., [5], [11]. These functions are not suitable for the setting of our problem, in which we measure the similarity between trajectories of arbitrary length combining the distance between the traversed nodes on the graph and the time intervals at which these nodes have been traversed [4].

III. Problem Statement

Let $P = \{p_1, p_2, \cdots, p_n\}$ be a set of Points of Interests (PoI) of a city. Let $U = \{u_1, u_2, \cdots, u_m\}$ be a set of users. We consider the movements of user u while visiting the subset of PoIs as a sequence of pairs $(p_i, t_i = [s_i, e_i])$ where s_i and e_i are the times at which user u enters to and exits from p_i. We define a temporal order of the pairs as the *trajectory* of the user u e.g., $T = \langle (p_1, [s_1, e_1]), (p_2, [s_2, e_2]) \cdots, (p_N, [s_N, e_N]) \rangle$, where for each two consecutive pairs we have $s_{i+1} = e_i + 1$. Intuitively, each pair $(p_i, [s_i, e_i]) \in T$ indicates that the user u is in p_i within the time interval $[s_i, e_i]$. We use $\mathcal{T}$ to refer to the set of trajectories T of all the users $u \in U$. For a trajectory T, let s and e be the starting and ending time of T. Given a time instant $i \in [s, e]$, the notation $T(i)$ indicates the unique node $p \in P$ such that there exists a pair $(p, t) \in T$ with $i \in t$.

Problem Given a set of trajectories $\mathcal{T}$ and a user u with its trajectory $Q = \langle (p_1, [s_1, e_1]), (p_2, [s_2, e_2]) \cdots, (p_N, [s_N, e_N]) \rangle$, we aim at predicting PoI in P that the user u will visit after his current visited PoI p_N.

A. Sketch of the Solution

Our basic idea is to use the relation between the trajectories of the tourists to predict their next movement. Given a set of trajectories $\mathcal{T}$ and a query trajectory Q, we process the problem by performing two major steps, which are illustrated in Algorithm 1. We first construct a graph of tourist trajectories on the city, called *trajectory graph* (we use simply graph when clear from the context), for a given dataset of trajectories. We define a similarity function that can be used to compare trajectories over the graph. Then in the second step, we identify the Most Similar Trajectory (in short, MSTRAJ) with respect to a given query trajectory based on our similarity function. In order to predict the next movement of a given tourist u with the trajectory Q, we apply the concept of MSTRAJ to Q. This trajectory (i.e. MSTRAJ) is crucial for our solution, as the current PoI visited by MSTRAJ is more likely to be a PoI that also u will visit in the future. We present the needed concepts to explain the solution in Section IV and then we discuss the method in detail in Section V.

Algorithm 1: Two major steps for next-PoI prediction

Input: $\mathcal{T}, Q$
Result: next-PoI of Q

```
begin
    step1: Build the trajectory graph G w.r.t T
      and Q
    step2: Find MSTRAJ in T w.r.t Q over G
    Report next-PoI by using MSTRAJ
```

IV. Preliminaries

Since the motion of tourists is represented by the PoIs they traversed in the city, we create a graph of their trajectories w.r.t the PoIs they visited, in order to represent the relation between PoIs.

Definition 1 (Trajectory Graph): For a given trajectory dataset $\mathcal{T}$ passing a set of PoIs P, $G = (V, E)$ is an

undirected graph, in which the node set V contains the set of PoIs in P visited by trajectories in $\mathcal{T}$ and E is the set of edges between PoIs, i.e. given two nodes $p_i, p_j \in V$ there is an undirected edge $(p_i, p_j) \in E$ if there exists at least one trajectory in $\mathcal{T}$ that traversed p_i and p_j, consecutively.

A. Similarity Measure

Given the set of trajectories $\mathcal{T}$, and the graph G, in the following, given two trajectories Q and T, we provide a measure of similarity between Q and T. We define the distance between two PoIs $p, v \in V$, respectively belonging to two trajectories, as the shortest path distance between them, $d(p, v)$, in the trajectory graph. The goal is to measure the proximity of two trajectories taking into account the distance between traversed nodes in graph and the time intervals each trajectory spent on the nodes. We restrict the given trajectory T in a specified time interval t as a sequence of nodes are traversed by T in t as $T[t]$. Thus, we can compute the distance between a given node on the graph and a trajectory in a given time interval as $dist(v, T, t) = \frac{\min_{(v_i,t_i)\in T[t]} d(v_i, v)}{D_G}$, where D_G is the diameter of the graph. This distance always is a value between 0 and 1. Having this distance we determine our linear similarity function between two query trajectory Q and target trajectory T in a specified time interval t in an exponential way as $Sim(Q, T, t) = \frac{\sum_{(v_i,t_i)\in Q[t]} |t_i| \times e^{-dist(v_i, T[t_i])}}{|t|}$. Where $|.|$ denotes the length of an interval. For a given query trajectory, we refer to the trajectory with the maximum similarity value, as the Most Similar Trajectory(MSTRAJ). Using the similarity between trajectories we define the next movement of a given trajectory as follows.

Definition 2 (next-PoI): Given a set of trajectories $\mathcal{T}$ over the graph $G(V, E)$, a trajectory $Q = \langle (p_1, t_1), (p_2, t_2) \cdots, (p_N, t_N) \rangle$ of a tourist, and a query time interval $t = [a, b]$, the next-PoI of Q is a node $p_i \in V$, if there exists a trajectory $T_j \in \mathcal{T}$ such that maximizes:

$$\rho_{Ni} \times \tau_{jib} \times Sim(Q, T_j, t = [a, b]), \tag{1}$$

where:

$$\rho_{Ni} = \begin{cases} 1 & \text{if } (p_N, p_i) \in E \\ 0 & \text{otherwise} \end{cases}$$

$$\tau_{jib} = \begin{cases} 1 & \text{if } T_j(b) \text{ is defined and } T_j(b) = p_i \\ 0 & \text{otherwise} \end{cases}$$

The next-PoI problem, based on the above definitions, is then the following one: given a set of trajectories $\mathcal{T}$, the corresponding trajectory graph, and a query trajectory Q, find next-PoI of Q according to the next-PoI definition.

V. Proposed Method

Given the set $\mathcal{T}$ and a query trajectory Q, as demonstrated in Algorithm 1, we first build the trajectory graph G with respect to $\mathcal{T}$. Note that this can be computed in an offline manner as follows: In order to build G, we simply scan all trajectories in $\mathcal{T}$ and update G during the process. Initially, G is an empty graph. For all trajectories $T \in \mathcal{T}$ and for each consecutive pair $(p_i, [s_i, e_i]), (p_j, [s_j, e_j]) \in T$, if the edge (p_i, p_j) does not exist in G, we add it to G. The construction of G is completed once all trajectories are considered. When we process the given query trajectory Q, we update G by scanning Q. Now, for finding next-PoI, we go through the second step of Algorithm 1 over G, finding the most similar trajectory to Q in $\mathcal{T}$.

A. Finding MSTRAJ

Given a set of trajectories $\mathcal{T}$ defined over a graph G, a query trajectory Q and a time interval $t = [a, b]$, we aim at finding MSTRAJ. To this end, we simply compute the similarity score of each trajectory $T \in \mathcal{T}$ with respect to $Q[t]$. Obviously, in the present of $t = [a, b]$, we should only consider those trajectories which are defined for each time instant $i \in t$. We maintain all trajectories in an ordered heap H, according to their similarity scores. Thereafter, we report the trajectory on the top of the heap H (i.e. with maximum similarity score) as MSTRAJ.

B. Finding next-PoI

Given $Q = \langle (p_1, t_1), (p_2, t_2) \cdots, (p_N, t_N) \rangle$ and a time interval $t = [a, b]$. We assume the neighbor nodes of p_N in the trajectory graph G, denoted by $NB(p_N)$, as the candidate next-PoI of Q. All such nodes $p_i \in NB(p_N)$ are indeed such that the value $\rho_{Ni} = 1$ in Equation 1. Therefore, for Q and $t = [a, b]$, if there exists a trajectory $T \in \mathcal{T}$ such that $T(b) \in NB(p_N)$ and satisfies Equation 1, then we consider $T(b)$ as next-PoI of Q, where recall that $T(b)$ indicates the unique node $p \in V$ in which there exists a pair $(p, t) \in T$ with $b \in t$. On the other hand, let T be MSTRAJ. If $T(b) \in NB(p_N)$ then T obviously satisfies the Equation 1, so $T(b)$ is next-PoI of Q. In other words, for a trajectory T, $T(b)$ is next-PoI of Q, if $T(b) \in NB(p_N)$ and T is MSTRAJ w.r.t Q. Based on this idea, we can address the next-PoI problem, for a query $Q = \langle (p_1, [s_1, e_1]), (p_2, [s_2, e_2]) \cdots, (p_N, [s_N, e_N]) \rangle$, looking at the current pair $(p_N, [s_N, e_N]) \in Q$. we identify MSTRAJ w.r.t Q within the query time interval $t = [s_N, e_N]$. Let H be the heap associated to the set of trajectories in $\mathcal{T}$ based on their similarity scores w.r.t Q within t. In searching process, we consider only the trajectory on the top of the heap H and investigate whether the top trajectory satisfies Equation 1. We continue to visit top-ranked trajectory in H finding the trajectory T

maximizing Equation 1. Thereafter, $T(b)$ (i.e. $b = e_N$) will be reported as next-PoI for Q.

VI. Experimental Evaluation

A. Datasets

To evaluate our method we use three different datasets provided in [8] containing tourist movements covering three Italian cities Pisa, Rome, Florence. For making tourist trajectories, the geo-tagged photos from Flickr are collected. The photos are mapped to the set of PoIs aggregated from Wikipedia. Each sequence of PoIs visited by each user is split into the set of daily journeys, which builds the trajectory set.

In our experiments, we consider the time duration between the starting time assigned to two consecutive visited PoIs p_i and p_{i+1} by a user, as the duration the user spent in p_i. Each dataset Pisa, Florence and Rome contains $110, 888, 490$ PoIs and $992, 5984, 12565$ trajectories. Each trajectory has at least two PoIs.

B. Methods

We compare our method with a probability baseline Prob and LearNext [8] described below. Although there are two important state-of-the-art techniques WhereNext [7] and Random Walk [6], we do not include them in our experimental study, since LearNext outperforms them. We evaluate our method using the same datasets and training/test sets in [8].

a) Prob*:* A pure model named "Prob" in [8], uses the trajectories in dataset to build a weighted-directed graph with PoIs as the node set. There is a directed edge from source PoI p_i to the destination PoI p_j, if there exists at least one trajectory in dataset that traverse from p_i to p_j. Each edge is weighted with the number of transaction from source to destination, representing the *transition probability* of each edge. Given p, the method Prob returns the out-neighbor of p with highest probability. Intuitively, this model suggests the most visited PoI from the source p as next-PoI.

b) LearNext*:* is a next-PoI predictor that learns tourists' behavior from their common patterns of movements. The prediction is done accordingly to what the user has already visited in the available attractions. The problem is modeled as an instance of learning to rank, which exploits a feature space composed of 68 features capturing both the behavior of the tourist and the peculiar characteristics of candidate PoIs. The models are trained using GBRT and Ranking SVM as learning methods [8]. They are tested on three collections of trajectories corresponding to popular Italian areas, in particular, data from photos taken in *Pisa*, *Florence*, and *Rome*, which are also used in this paper.

C. Evaluation strategy

The evaluation of our proposed solution is aimed at specifying the effectiveness of the proposed model for predicting next-PoI. To this end, we follow the same evaluation strategy adopted in [8] over the three aforementioned datasets, which is a standard training/test evaluation strategy. For each city, we consider 80% of trajectories as a training set S and 20% of trajectories as the test set S'. The effectiveness of the methods are assessed by means of Success@k (i.e., the percentage of times that the correct answer is in the top-k ranked PoIs) [8]. Specifically, we will use $k = 1$ in our evaluations, which is the topmost PoI. As illustrated in Algorithm 2,

Algorithm 2: Evaluation Block

Input: Training set S
Input: $Q = \langle (p_1, t_1), (p_2, t_2) \cdots, (p_{N-1}, t_{N-1}), (p_N, t_N) \rangle \in$ test set S'
Result: next-PoI

5 Build G by trajectories in S
6 Make $Q.tail$ and $Q.head$
7 Update G by $Q.head$
8 $t = t_{N-1}$
9 Heap H (to maintain trajectories)
10 **for** *each trajectory* T *in* S **do**
11 H.add(T) w.r.t $Sim(Q, T, t)$
12 MsTraj $\leftarrow pop(H)$
13 $p \leftarrow$ last position of MsTraj within t_{N-1}
14 **if** $p \in NB(p_{N-1})$ **then**
15 report p as the next-PoI
16 break

to evaluate the effectiveness of the proposed method, we first build the trajectory graph G, by using the trajectories in the training set S (Line 5). Then, for building the query set, we make use of the following process: We divide each trajectory in S' into two parts: Head and Tail (Line 6). Let $Q = \langle (p_1, t_1), (p_2, t_2) \cdots, (p_N, t_N) \rangle$ be a query trajectory in S'. The tail of Q is the last pair of trajectory (i.e. $Q.tail = \langle (p_N, t_N) \rangle$), and the first $N - 1$ pairs of Q makes the head (i.e. $Q.head = \langle (p_1, t_1), (p_2, t_2) \cdots, (p_{N-1}, t_{N-1}) \rangle$). For each trajectory $Q \in S'$, we aim at using $Q.head$ and predicting the PoI p_N of $Q.tail$ as the next movement of Q. To this end, we update G by making use of $Q.head$ in Line 7. Then, we use G as the underlying graph for trajectories to predict $Q.tail = \langle (p_N, t_N) \rangle$. Based on Definition 2, the next movement of $Q.head$ would be a neighbor PoI of p_{N-1} on G. Let $NB(p_{N-1})$ denotes the neighbor PoIs of p_{N-1} on G. We aim at using $Q.head$ to choose a right PoI $\in NB(p_{N-1})$.

We consider the last pair $\langle(p_{N-1}, t_{N-1})\rangle$ of $Q.head$ as the current location of Q. Through the lines 8-11, by considering t_{N-1} of the current location of Q as the query time interval t, we compute the similarity score between each trajectory in training set and Q within $t = t_{N-1}$. Obviously, we first restrict Q within t, then we compute the similarity scores. Moreover, we make an ordered heap of H of trajectory ids regarding their similarity scores. The trajectory with maximum score (MSTRAJ) is on the top of H. We consider the last PoI p of MSTRAJ within t_{N-1} as next-PoI, if $p \in NB(p_{N-1})$. Otherwise, we continue to search over the trajectories in H (the $while$ loop). In each round, we pick a trajectory with maximum similarity. We stop searching if we find a MSTRAJ with a last position in $NB(p_{N-1})$ within t_{N-1} (lines 12-16). The main task of the evaluation method is to measure how many times our model is able to choose the right PoI by means of Success@1, by following the aforementioned strategy.

D. Effectiveness

In this part, we investigate whether the proposed model is an effective model for predicting next-PoI of a given tourist trajectory. In the first experiments, we measure metric Success@1. The related results are provided for our proposed method MSTRAJ along with the two methods (PROB and LEARNEXT). Table I shows the results of the experiments, where our method outperforms the competitors. The values of Success@1 of our model are highlighted. As we can observe, MSTRAJ provides almost six times more accurate results than PROB for Pisa and Rome, and almost ten times more for Florence, which confirms the effectiveness of our method. This is due to the fact that the proposed model uses the similarity between those parts of trajectories that overlap in time span with the query. This helps to provide more accurate results than the baselines, which use machine learning techniques considering features such as Euclidean distances and PoI frequency. The results confirm that users do not necessarily visit the most frequent or nearest PoI in the future, but tend to visit specific PoIs in similar time intervals.

TABLE I
EFFECTIVENESS IN TERMS OF SUCCESS@1 OF THE PROPOSED METHOD (MSTRAJ) ALONG WITH THE COMPETITORS

Dataset	Predictor	Success@1 %
Pisa	PROB	15.57
	LEARNEXT	40.70
	MSTRAJ	**67.33**
Rome	PROB	12.59
	LEARNEXT	30.95
	MSTRAJ	**77.96**
Florence	PROB	4.96
	LEARNEXT	37.56
	MSTRAJ	**53.57**

VII. CONCLUSION AND FUTURE WORK

In this work, we studied the next-PoI prediction problem for a given tourist trajectory. We introduced a new graph-based method that reflects similar behavior of past tourists to predict the next movements of a new tourist. We evaluated our proposed method with respect to the state-of-the-art competitors on three public datasets of movements in Pisa, Rome, Florence. The performance of the methods shows that our proposal achieves the best performance outperforming well-known competitors based on machine learning by providing at least twice more accurate results, up to 77.96% in Success@1. We aim at applying our method to other datasets and applications such as predicting the next web-page a user will visit, or the next item a user will buy in a specific period of time.

REFERENCES

[1] B. Altaf, L. Yu, and X. Zhang. Spatio-temporal attention based recurrent neural network for next location prediction. In *IEEE(Big Data)*, 2018.

[2] R. Baraglia, C. I. Muntean, F. M. Nardini, and F. Silvestri. Learnext: Learning to predict tourists movements. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, CIKM 2013. ACM.

[3] I. R. Brilhante, J. A. Macedo, F. M. Nardini, R. Perego, and C. Renso. On planning sightseeing tours with tripbuilder. *IPM*, 2015.

[4] R. Grossi, A. Marino, and S. Moghtasedi. Finding structurally and temporally similar trajectories in graphs. In *(SEA 2020)*.

[5] J.-R. Hwang, H.-Y. Kang, and K.-J. Li. Searching for similar trajectories on road networks using spatio-temporal similarity. In *EECADIS*. Springer, 2006.

[6] C. Lucchese, R. Perego, F. Silvestri, H. Vahabi, and R. Venturini. How random walks can help tourism. In *ECIR*. Springer, 2012.

[7] A. Monreale, F. Pinelli, R. Trasarti, and F. Giannotti. Wherenext: a location predictor on trajectory pattern mining. In *SIGKDD*. ACM, 2009.

[8] C. I. Muntean, F. M. Nardini, F. Silvestri, and R. Baraglia. On learning prediction models for tourists paths. *ACM (TIST)*, 2015.

[9] A. Noulas, S. Scellato, N. Lathia, and C. Mascolo. Mining user mobility features for next place prediction in location-based services. In *IEEE 12th international conference on data mining*, ICDM 2012. IEEE Computer Society.

[10] H. Wang, H. Su, K. Zheng, S. Sadiq, and X. Zhou. An effectiveness study on trajectory similarity measures. In *Proceedings of the Twenty-Fourth Australasian Database Conference-Volume 137*, 2013.

[11] Y. Xia, G.-Y. Wang, X. Zhang, G.-B. Kim, and H.-Y. Bae. Spatio-temporal similarity measure for network constrained trajectory data. *CIS*, 4(5), 2011.

[12] H. Xu, P. Wu, J. Wei, Z. Yang, and J. Wang. A meta-path-based recurrent model for next poi prediction with spatial and temporal contexts. In J. Shao, M. L. Yiu, M. Toyoda, D. Zhang, W. Wang, and B. Cui, editors, *APWeb and WAIM Joint Conference on Web and Big Data*. Springer, 2019.

[13] V. W. Zheng, Y. Zheng, X. Xie, and Q. Yang. Collaborative location and activity recommendations with gps history data. In *Proceedings of the 19th international conference on WWW,2010*.

[14] W. Zheng, X. Huang, and Y. Li. Understanding the tourist mobility using gps: Where is the next place? *Tourism Management*, 2017.

[15] Y. Zheng and X. Xie. Learning travel recommendations from user-generated gps traces. *ACM TIST*, 2(1):2, 2011.