Trajectory Test-Train Overlap in Next-Location Prediction Datasets

Massimiliano Luca[1,2*], Luca Pappalardo[3], Bruno Lepri[2] and Gianni Barlacchi[+4]

[1]Free University of Bolzano, Piazza Domenicani, 3, Bolzano, 39100, Italy.
[2]Bruno Kessler Foundation, Via Sommarive, 19, Trento, 38123, Italy.
[3]ISTI-CNR, Via Moruzzi, 1, Pisa, 56127, Italy.
[4]Amazon Alexa AI, Berlin, Germany.

*Corresponding author(s). E-mail(s): mluca@fbk.eu;
Contributing authors: luca.pappalardo@isti.cnr.it; lepri@fbk.eu;

Abstract

Next-location prediction, consisting of forecasting a user's location given their historical trajectories, has important implications in several fields, such as urban planning, geo-marketing, and disease spreading. Several predictors have been proposed in the last few years to address it, including last-generation ones based on deep learning. This paper tests the generalization capability of these predictors on public mobility datasets, stratifying the datasets by whether the trajectories in the test set also appear fully or partially in the training set. We consistently discover a severe problem of trajectory overlapping in all analyzed datasets, highlighting that predictors memorize trajectories while having limited generalization capacities. We thus propose a methodology to rerank the outputs of the next-location predictors based on spatial mobility patterns. With these techniques, we significantly improve the predictors' generalization capability, with a relative improvement on the accuracy up to 96.15% on the trajectories that cannot be memorized (i.e., low overlap with the training set).

Keywords: Human Mobility; Next-Location Prediction; Deep Learning; Generalization; Test-Train Overlap

[+] work done prior joining Amazon

1 Introduction

Next-location prediction is the task of forecasting which location an individual will visit, given their historical trajectories. It is crucial in many applications such as travel recommendation, and optimization [1, 2], early warning of potential public emergencies [3–6], location-aware advertisements and geomarketing, and recommendation of friends in social network platforms [7–11]. Predicting an individual's location is challenging as it requires capturing human mobility patterns [12, 13] and combining heterogeneous data sources to model multiple factors influencing human displacements (e.g., weather, transportation mode, presence of points of interest and city events).

The striking development of Deep Learning (DL) and the availability of large-scale mobility data has offered an unprecedented opportunity to design powerful next-location predictors (NLs) and has driven test-set performance on mobility data to new heights [13]. However, little work has been done on how challenging these benchmarks are, what NLs learn, and their actual generalization capabilities. Although some studies investigate the predictability of human whereabouts and its relationship with the trajectories' features [14, 15], we know comparatively little about how the individuals' trajectories are distributed in mobility benchmarks, making it hard to understand and contextualize our observed results. Recent studies in natural language processing [16, 17] and computer vision [18] show that DL models excel on specific test sets but are not solving the underlying task. In this paper, we investigate whether it is the case for NLs too.

We perform an extensive study of the test sets of several public next-location benchmark datasets [13] and evaluate a set of state-of-the-art DL-based NLs on their generalization capability. We identify three levels of generalization that an NL should exhibit: (i) *known mobility*, requiring no generalization beyond recognizing trajectories seen during the training phase; (ii) *fragmentary mobility*, requiring generalization to novel compositions of previously observed trajectories; and (iii) *novel mobility*, requiring generalization to a sequence of movements not present in the training set. It is unclear how well state-of-the-art NLs perform on each of these three scenarios.

To address this compelling issue, we stratify mobility data by whether the trajectories in the test set also appear fully or partially in the training set. We quantify the overlap between trajectories with three measures accounting for different ways of computing the percentage of locations in the test trajectories that are also in the training trajectories.

We find that, in five next-location benchmark datasets, there is a severe problem of trajectory overlapping between the test and training sets when composing them randomly: $\sim$ 43% to 72% of test trajectories overlap at least with 50% of the points with trajectories in the training set, and with 7% to 14% of test sub-trajectories entirely overlap training sub-trajectories. In other words, based on the standard way training and test sets are split in the literature, a significant portion of the trajectories in the test sets have already been seen during training.

Based on these observations, we propose to evaluate NLs on *stratified test sets based on the overlap between trajectories in the training set*. We find significant variability in model performance, varying the percentage of overlap. Indeed, we find an accuracy $\leq 5\%$ when predicting unseen trajectories (novel mobility) and $\geq 90\%$ when predicting trajectories with high overlaps (known mobility). Surprisingly, we also find that DL-based NLs perform even worse than baseline models (e.g., Mobility Markov Chain or MMC [19]) when tested on novel mobility. Our results are consistent across the datasets analyzed and the NLs selected, demonstrating that current train/test splits are flawed, and more robust methods are needed to evaluate the generalization capabilities of NLs. We also show a way to improve next-location prediction accuracy, especially for the novel mobility scenario, injecting mobility laws into state-of-the-art NLs through a learning-to-rank task. In a nutshell, this paper provides the following novel contributions:

- We show that standard train/test splits of trajectory datasets generate a high trajectory overlap, proposing three metrics to quantify it;
- We evaluate NLs on stratified test sets and show that DL-based NLs do not generalize well on novel mobility, being outperformed by other simpler baselines (e.g., Mobility Markov Chains);
- We show how to improve the accuracy of DL-based NLs, especially for the novel mobility behavior, by performing a rerank of the models' scores based on spatial mobility patterns;
- Based on our findings, we provide a list of recommendations to improve datasets' creation and models' evaluation for next-location prediction.

2 Related Work

Model Generalization

Measuring the generalization capabilities of deep neural networks has recently captured the attention of researchers in artificial intelligence. Lewis et al. [16] find that, in popular Question Answering (QA) datasets, 30% of test-set questions have a near-duplicate in the training sets and that all models perform worse on questions that cannot be memorized from training sets. Sen and Saffari [17] show that QA models do not generalize well on unseen question-context pairs. However, they still perform well on popular QA benchmarks because of their high overlap between train and test data. Liu et al. [20] go beyond the data and study the key factors that impact generalization in QA. An essential impact in generalization is played by cascading errors from retrieval, question pattern frequency, and entity frequency.

Predictability of Human Mobility

Several studies measure the limits of predictability of human mobility [12, 13]. Song et al. [14] analyze mobility traces of anonymized mobile phone users to find that 93% of the movements are potentially predictable. Zhang et al. [21]

show that, when considering the mobility context (e.g., visiting time, kind of place visited), the upper bound of potential predictability in human mobility increases. Other studies show that this upper bound depends on the data scale and the processing techniques adopted [22–24]. In [15, 25], there are shreds of evidence that the so-called explorers (e.g., individuals without a routinary behavior) [26] are less predictable than the others. All the works discussed suggest that models may memorize certain trajectories (e.g., routinary mobility) while not being able to generalize well on novel mobility (i.e., mobility not observed during the training phase).

Next-Location Prediction

Most NLs are based on (gated) recurrent neural networks (RNNs). RNNs [27] can efficiently deal with sequential data such as time series, in which values are ordered by time, or sentences in natural language, in which the order of the words is crucial to shaping its meaning. In Spatial Temporal Recurrent Neural Networks (ST-RNN) [28], RNNs are augmented with time- and space-specific transition matrices. Through linear interpolation, each RNN layer learns an upper and lower bound for the temporal and spatial matrices, which are then used to infer an individual's next visited location. Long Short-Term Memory Projection (LSTPM) [29] use sequential models to capture long- and short-term patterns in mobility data. The authors rely on a non-local network [30] for modeling long-term preferences and on geo-dilated RNNs inspired to capture short-term preferences [31]. More sophisticated models like DeepMove [32] use attention layers to capture the periodicity in mobility data. First, past and current trajectories are sent to a multi-modal embedding module to construct a dense representation of spatio-temporal and individual-specific information. Next, an attention mechanism extracts mobility patterns from historical trajectories, while a Gated Recurrent Unit (GRU) handles current trajectories. Finally, the multi-modal embedding, GRU, and attention layer outputs are concatenated to predict the future location. Recently, Spatio-Temporal Attention Network (STAN) [33] proposes to capture spatio-temporal information to leverage spatial dependencies explicitly. In particular, the authors use a multi-modal embedding layer to model historical trajectories and the GPS locations in the current trajectories. The embeddings are then forwarded to a spatio-temporal attention mechanism that selects a set of potential next locations. Many other works deal with spatio-temporal data using (gated) RNNs and attention mechanisms. Some of them also deal with the semantic meaning associated with locations. Examples of such models are Semantics-Enriched Recurrent Model (SERM) [34], Hierarchical Spatial-Temporal Long-Short Term Memory (HST-LSTM) [35], VANext [36], and Flashback [37]. Other Deep Learning solutions to next-location prediction have been discussed in a recent survey paper [13].

3 Problem Definition

Next-location prediction is commonly defined as the problem of predicting the next location an individual will visit given their historical movements, typically represented as spatio-temporal trajectories [13].

Definition 1 (Trajectory) A spatio-temporal point $p = (t, l)$ is a tuple where t indicates a timestamp and l a geographic location. A trajectory $P = p_1, p_2, \ldots, p_n$ is a time-ordered sequence of n spatio-temporal points visited by an individual, who may have several trajectories, $P_1, \ldots, P_k$, where all the locations in P_i^u are visited before locations in P_{i+1}.

Given this definition, we formalize next-location prediction as follows:

Problem 1 (Next-location prediction) Given the current trajectory of an individual $P_k = p_1, p_2, \ldots, p_n$ and their historical trajectories $\mathcal{H} = P_1, \ldots, P_{k-1}$, next-location prediction is the problem of forecasting the next point $p_{n+1} \in P_k$.

In other terms, a next-location predictor (NL) is a function $\mathcal{M}(P_k, \mathcal{H}) \rightarrow p_{n+1}$, which takes the current trajectory P_k, the set of u's historical trajectories $\mathcal{H}$, and returns a spatio-temporal point p_{n+1} in P_k.

4 Trajectory Overlap

An NL should be able to predict an individual's next location in three scenarios: *(i)* the NL has seen the individual's entire current trajectory during the training phase; *(ii)* it has seen the current trajectory only partially, or it has seen a very similar trajectory of the same individual; *(iii)* the current trajectory was absent from the training set. The latter scenario is essential, as machine learning models' ability to generalize is their capacity of making predictions on data never seen during the training phase [38].

However, in next-location prediction, there may be a significant *overlap* between trajectories in the test set and those in the training set. For example, some test and training trajectories may belong to the same individual. Since human mobility is routinary, an individual's trajectories are similar to each other [12, 39], leading to scenarios *(i)* and *(ii)* above. Given this discussion, we investigate *the extent to which the overlap between trajectories in the test and training sets influences the model's ability to generalize*. We explore three ways to examine overlap: Jaccard Similarity (JS), Longest Common Subsequence (LCST), and Overlap From the End (OFE).

Jaccard Similarity (JS) measures the percentage of locations in the test trajectories that are also in the training trajectories, regardless of the order in which locations appear. Test trajectories with a high JS have many locations in common with training trajectories. In contrast, test trajectories with low JS should be less predictable as they are mainly composed of locations that are

not in the training trajectories. Formally, we define the JS between a trajectory $R \in D_{\text{test}}$ and $P \in D_{\text{train}}$ as:

$$\text{JS}(R, P) = \frac{|P \cup R| - |P \cap R|}{|P \cup R|}$$

We quantify the overlap between R and the training set as the maximum JS over all the trajectories in the training set:

$$\max_{P \in D_{\text{train}}} \text{JS}(R, P).$$

JS $\in [0, 1]$, where 1 indicates a full overlap (all locations in R are at least in a trajectory in D_{train}) and 0 indicates no overlap (none of the locations in R are in the training set).

The Longest Common SubTrajectory (LCST) is the longest subtrajectory in common between two trajectories. Formally, given a training trajectory $P = p_1, p_2, \ldots, p_n$ and a test trajectory $R = r_1, r_2, \ldots, r_m$, we define a recursive function $f(P, R)$ as:

$$f(P,R) = \begin{cases} 0, & \text{if } i = 0 \text{ or } j = 0 \\ f(p_{i-1}, r_{j-1}) + 1, & \text{if } i,j{>}0 \text{ and } p_i{=}r_j \\ \max(f(p_{i-1}, r_j), f(p_i, r_{j-1})) & \text{if } i,j{>}0 \text{ and } p_i{\neq}r_j \end{cases}$$

where P and R indicate the length of the training and test trajectories, respectively, and $f(P, R) \in [0, \min(\mathsf{P}, \mathsf{R})]$. The LCST between P and R is then:

$$\text{LCST}(P, R) = f(P, R)/\ \mathsf{R}.$$

We quantify the overlap between R and the training set as the maximum LCST over all the trajectories in the training set:

$$\max_{P \in D_{\text{train}}} \text{LCST}(R, P).$$

The Overlap From End (OFE) enforces that the common subtrajectory is at the end of the two trajectories. Formally, given a trajectory $P = p_1, p_2, \ldots, p_n$, we define $P' = p_n, \ldots, p_2, p_1$ as its reversed trajectory. We then compute OFE(R, P) with Algorithm 1 and quantify the overlap between R and the training set as the maximum OFE over all the trajectories in the training set:

$$\max_{P \in D_{\text{train}}} \text{OFE}(R, P).$$

In other terms, given a trajectory in the test set, we scan all the trajectories in the training set and we compute, for each pair (P, R), we compute, starting from the last point the number of common points. We than convert this number into a percentage. Finally, the OFE of P is the higher percentage found.

Algorithm 1 OFE Computation

```
overlaps ← dictionary()
for R′ ∈ D_test do
    overlap ← 0
    for P′ ∈ D_train do
        count ← 0
        for k ∈ {0, ..., min(P', R')} do
            if R′[k] = P′[k] then
                count ← count + 1
            else if R′[k] ≠ P′[k] then Break
            end if
        end for
        if count/ R' > overlap then
            overlap ← count/ R'
        end if
    end for
    overlaps[R′] ← overlap
end for
```

5 Experimental Setup

5.1 Datasets

We use five public datasets widely adopted in the literature to evaluate NLs [13] (see Table 1). Three of them (Gowalla, Foursquare New York, Foursquare Tokyo) are collected through social networking platforms, in which mobility traces are generated by the users' georeferenced posts (check-ins). Consequently, these mobility traces are sparse both in time and space. The other two datasets (Taxi Porto and Taxi San Francisco) describe GPS traces from taxis dense in space and time. In detail, Gowalla was a location-based social network platform that, like Foursquare, allowed users to check-in at so-called spots (venues) via a website or an app. The dataset [40] has almost six million check-ins collected over a year and a half, from February 2009 to October 2010. Each check-in contains the user identifier, location identifier, latitude and longitude pair, and timestamp. The dataset also contains information on the users' friendship network, which has around 200,000 nodes and one million edges. Foursquare is another location-based social network platform that allows users to check in into places. Data can be collected through the available APIs. A widely used dataset based on Foursquare is described in [41]. The information contained are the same as Gowalla, with additional information about the category of the venue. Piorkowski et al. [42] collected taxi trajectories in San Francisco in May 2008. Each point in a trajectory includes the taxi's identity, latitude, longitude, timestamp, and occupancy. Points are sampled every 10 seconds on average. Moreira et al. [43] (ECML/PKDD Challenge) collected taxi trajectories in Porto, Portugal. For each trajectory, we have the

taxi's identifier, the latitude, longitude, and timestamp showing when the trip began. For each trajectory, data are sampled every 15 seconds. The dataset also includes auxiliary information for each trip, such as the trip's typology (e.g., sent from the central, demanded to the operator, demanded to the driver), the stand from which the taxi left, and a phone number identification for the passenger.

To extract trajectories from these datasets, we follow the same approach as in [32]: first, we filter out the users with less than ten records; second, we cut the sequence of records into several trajectories for each user based on the time interval between two neighbor records. As in [32], we choose 72 hours as the default interval threshold based on the practice. Finally, we remove the users with less than five trajectories.

5.2 Models

We validate our hypothesis by testing the generalization capability of the following state-of-the-art DL-based NLs.

- **RNN** [27], the building block of the majority of NLs. RNNs are commonly adopted to model sequential data such as time series and natural language, in which the order of the items is crucial to shaping its meaning. RNNs are also widely used as building blocks of NLs to capture spatial and temporal patterns in the trajectories. An RNN is made of a sequence of gates, each one outputting a hidden state h_i based on the current input x_i and the previous gate h_{i-1}. In this work, a gate is implemented as a hyperbolic tangent function (tanh).
- **ST-RNN** [28] enhances RNNs with time- and space-specific transition matrices in this study. Each RNN layer learns an upper and lower bound for the temporal and spatial matrices via linear interpolation. These matrices are then used to predict where a person will go next.
- **Deep Move** [32] uses attention mechanisms to capture spatio-temporal periodicity in the historical trajectories. Also, the model uses GRUs (gated RNNs) to capture patterns in the current trajectory and relies on a multi-modal embedding to capture individual preferences and project trajectories in a low-dimensional space before passing them to the attention mechanisms and GRUs.
- **LSTPM** [29] combines long- and short-term sequential models: long-term patterns are modeled using a non-local network [30], short term preferences are captured using a geographic-augmented version of the concept of dilated RNNs [31].
- **STAN** explicitly captures spatio-temporal information using a multi-modal embedding to represent the trajectories and a spatio-temporal attention mechanism to capture patterns in the data [33]. The role of the attention mechanisms, supported by a balanced sampler, is to rank potential next locations.

		Users	Locations	Trajectories
Gowalla	[40]	5300	125,771	72,593
Foursquare NYC	[41]	4390	13,960	12,519
Foursquare Tokyo	[41]	935	21,394	34,662
Taxi Porto	[43]	500	8524	94,214
Taxi SF	[42]	500	9321	103,120

Table 1 Properties of the datasets adopted in our study. We describe each dataset's time span, number of users, number of locations, and the number of trajectories extracted.

5.3 Training

We split the trajectories into a training set, a validation set, and a test set for each dataset. All sets include trajectories from several users. We sort the trajectories temporally for each user and put the first 70% in the training set, the following 10% in the validation set, and the remaining 20% in the test set.

All models are implemented with PyTorch and are made available through the library LibCity [44]. We follow the same configuration as [32] and use Adam [45] as optimizer.

We ran the experiments on a machine with 126GB of memory and two Nvidia RTX 2080Ti.

6 Testing Generalization Capability

We evaluate the performance of all models using the k-accuracy (ACC@k), the most common evaluation metric in the literature [13]. NLs output a list of all possible locations an individual will visit next ranked from the most to the least likely. ACC@k indicates how many times the true location is among the k top predicted locations. We evaluate all models using ACC@5.

We compare the DL models with Mobility Markov Chains (MMCs) [19], in which the visited locations are the states of a Markov chain and a transition matrix represents the first-order transition probabilities between these locations. The choice of MMCs as a baseline is justified because they cannot generalize as they summarize the training data.

For all datasets and overlap metrics (JS, LCST, and OFE), we compute the number of trajectories in the test set with an overlap with the training set between 0-20%, 20-40%, 40-60%, 60-80%, and 80-100%. Figure 1 shows the results for all the datasets analyzed.

The percentage of trajectories with a high overlap (between 80% and 100%) varies widely with the overlap metric and the dataset. Taxi datasets have more trajectories with a high overlap than the datasets based on check-ins, suggesting that the overlap problem is more severe in GPS traces than in check-ins. We also observe that JS and LCST produce similar overlaps, while with OFE, the number of trajectories with low overlap is remarkably higher. This is due to the severe constraints that OFE imposes by definition (e.g., the overlap is evaluated starting only from the end of the trajectory).

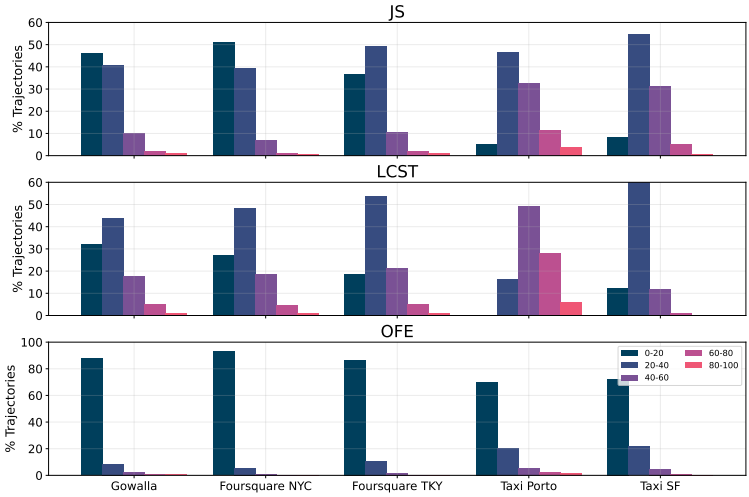

Fig. 1 Fraction of the test trajectories with an overlap of 0-20%, 20.40%, 40-60%, 60-80%, and 80-100% with the training trajectories, for the all datasets, for the evaluation metrics JS, LCST, and OFE.

In any case, Figure 1 highlights that a significant overlap exists between the test and the training set, introducing a bias when evaluating NLs using a random train-test split. Hence, we investigate to what extent this overlap affects model performance.

Figure 2 shows the performances for all the NLs and overlap metrics. Here, increasing the overlap induces a striking improvement in the model performance for both MMC and the NLs, which have similar performance. We present all performances in detail in the Supplementary A.

For example, for Foursquare New York and OFE, the performance of NLs is close to 100% on a test made of trajectories with an overlap with the training set in the range 80-100%. Results for Taxi Porto follow a similar increasing trend, although with less striking performance.

Overall, Figure 2 shows that model performance is strongly affected by trajectory test-train overlap, suggesting that NLs memorize trajectories instead of generalizing. NLs perform well on trajectories with high overlap with the training set but poorly on trajectories with low overlap. These results raise the question of how to improve the accuracy of NLs for low overlap scenarios.

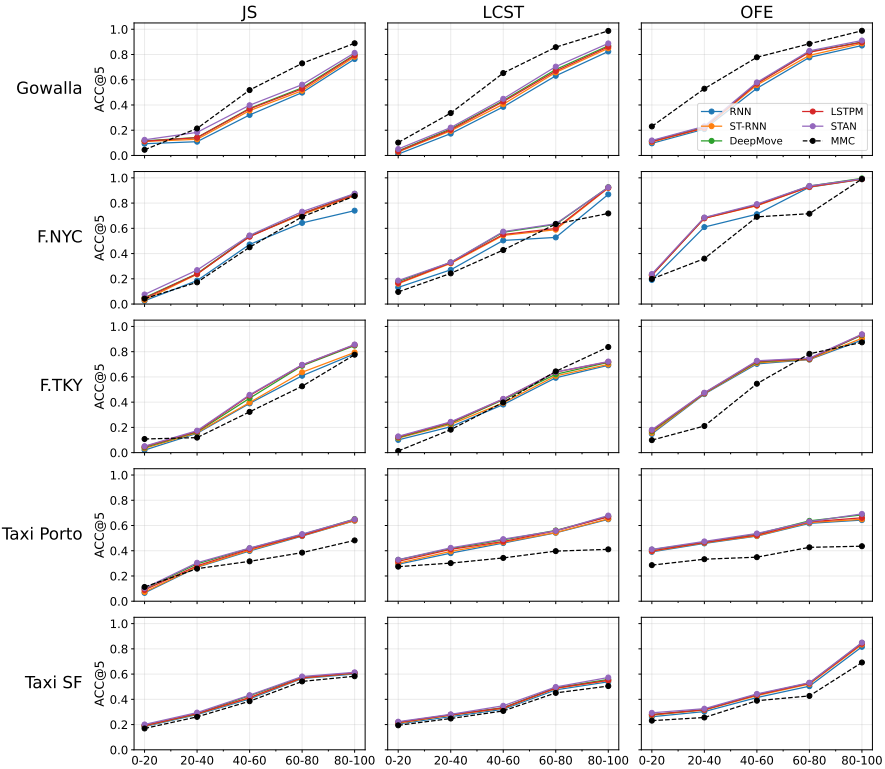

Fig. 2 Results (in terms of ACC@5) for all the datasets and models. We compute the accuracy for the three overlap metrics (JS, LCST, OFE) and for five bins of percentage of trajectory overlap (from 0-20% to 80-100%).

7 Learning to Rank Locations Using Mobility Laws

A possible reason why NLs perform poorly on trajectories with low overlaps lies in the type of DL tools they rely on, i.e., RNNs: they focus on memorizing regularities in long sequences, thus limiting NLs' generalization capabilities. Wrong location predictions happen when the probabilities assigned to each potential location by the NL (i.e., the locations' scores) are low and relatively uniformly distributed. Our intuition is to rerank locations based on new scores obtained, injecting human mobility laws into NLs. We select three prominent human mobility laws [12, 13]:

- the distance law [12]: people prefer travelling short distances. Given an individual's trajectories $P = p_1, p_2, \ldots, p_n$, we compute the Haversine distance between all the consecutive locations p_i, p_{i+1} and consider the average of the distances as a feature $dist_u$;

- the visitation law [39]: the visits to a location decrease as the inverse square of the product of their visiting frequency and travel distance. We denote as f the number of visits to a location (by any individual) and compute how many people visit it within a distance r. An individual's probability to visit location p_{i+1} is given by a power-law of the form $p_{i+1}(r, f) = \mu_i/(rf)^\gamma$, with $\gamma = 1.6$, a parameter fitted with the least squares method. We use the five most probable locations $top_n, n \in 1 \dots 5$ as an input to the reranker.
- the returner and explorer dichotomy [26]: individuals naturally split into two profiles based on their degree of spatial exploration. We compute the average radius of gyration $r_g(u)$ and the 2-radius of gyration $r_g^{(2)}(u)$ for each individual and compute the ratio $\frac{r_g(u)}{r_{2g}(u)}$ using the scikit-mobility [46] library. The profile of the user is then translated into a binary feature: 0 if the individual is a returner and 1 if the individual is an explorer. We denote this feature as re_u.

Our approach consists in predicting the next location using a NL, and then combining into a single scoring model, i.e., a fully connected neural network, both the NL score for the location and the mobility laws. We trained the network using the binary cross-entropy loss $\mathcal{L} = -\sum_{i \in \{0,1\}} y_i \log p(y_i)$, where y_i is the label (i.e., 0 or 1) and $p(y_i)$ is the predicted probability.

The training dataset consists of vectors of the form $[\mathrm{NL}_i(P), dist_u, top_1, \dots, top_5, re_u]$. We denote with $\mathrm{NL}_i(P)$ the score of the NL for a given location i starting from a trajectory P. The label is 1 if the location i is the individual's next-location and 0 otherwise. This means that in a dataset with n locations, for each trajectory we have a positive sample (e.g., the correct next location) and $n - 1$ negative samples for each trajectory. As the number of incorrect samples is much higher than the correct ones, for each correct location, we randomly sampled $k = 20$ wrong locations (e.g., locations that are different from the actual next location the individual will visit), as we found it to be a good trade-off between performance and dataset size.

Table 2 and Figure 3 show how the accuracy changes on the test trajectories with 0-20 overlap on all the datasets and models considered. Our reranking leads to improved accuracy regardless of the dataset and the overlap measures used. The bigger relative improvements are related to the trajectories with an overlap of 0-20. Regarding check-in datasets, on Foursquare New York, the improvement varies from +3.25% (ST-RNN) to +9.38 (LSTPM). Similarly, on Foursquare Tokyo, the improvement varies from +5.69% (DeepMove) to a +9.33% of improvement (STAN). In Gowalla, we have the lowest relative improvement on DeepMove (+4.43%) and the highest on RNN (+29.09%). Concerning taxi datasets, on Taxi Porto, the relative improvement on the average case (i.e., without stratifying the test set) varies from a +2.68% (RNN) to +5.84% (STAN). On Taxi San Francisco, the relative improvement varies from +2.49% (RNN) to +5.74% (DeepMove). Regarding the 0-20 stratification, the largest relative improvement is associated with metrics JS, followed by LCST and OFE. On Foursquare New York, the relative improvement

with JS is up to +96.15%, with LCST being +20.39%, and with OFE being +33.05%. Similarly, on Foursquare Tokyo, we have top relative improvements of +82.35%, +21.78%, and +24.36% with JS, LCTS, and OFE, respectively. Finally, Gowalla's top relative improvements for JS, LCTS, and OFE are +68.82%, +45.45%, and +50.03%. In general, taxi datasets are associated with the lowest relative improvement: with JS, it is up to +7.96%, with LCST +6.68%, and with OFE +7.05% on Taxi Porto. On the other hand, on Taxi San Francisco, the relative improvements for JS, LCST, and OFE are +5.82%, +9.68%, and +8.76%. The largest relative improvement is associated with the 0-20 overlap scenario. For example, the largest relative improvement on the 80-100 bin is 0.12%. In other words, our rerank strategy brings the largest improvement in accuracy, especially where NLs are the least accurate.

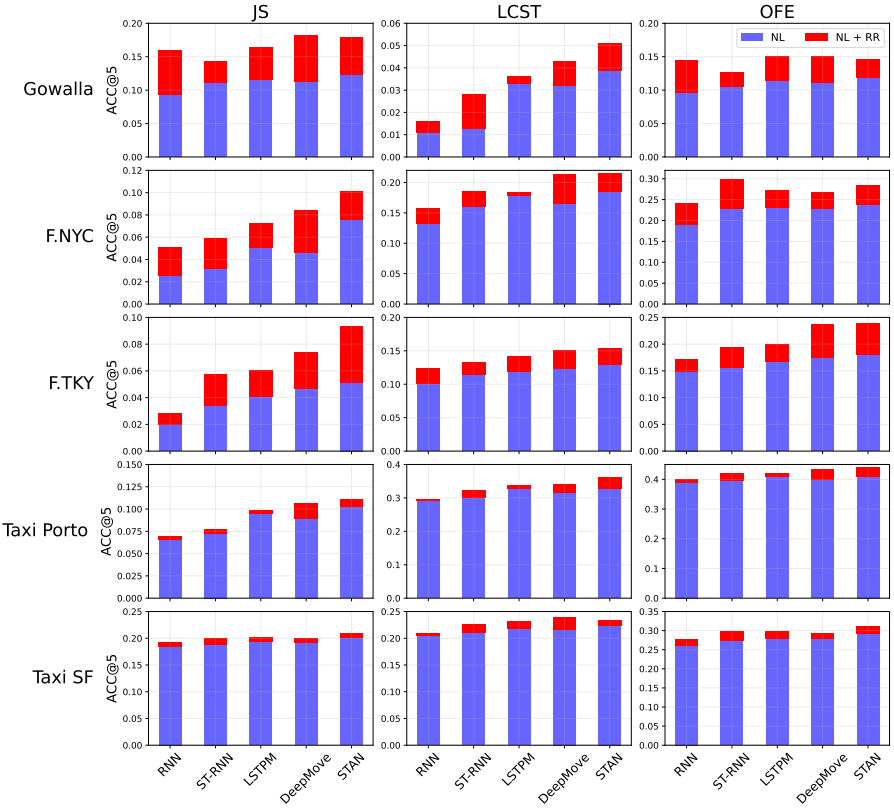

Fig. 3 Results (in terms of ACC@5) for all the datasets for the three overlap metrics (JS, LCST, OFE) for trajectories with a 0-20%. We provide the results for all datasets and the other overlaps in Supplementary B.

	NL + RR	JS		LCST	OFE
Foursquare NYC	RNN	.233 (+9.38%)	**.051 (+96.15%)**	.158 (+19.69%)	.241 (+26.17%)
	ST-RNN	.261 (+5,24%)	**.059 (+84.37%)**	.186 (+15.52%)	.299 (+31.14%)
	Deep Move	.277 (+6,94%)	**.084 (+64.71%)**	.213 (+19.66)	.268 (+16.01%)
	LSTPM	.272 (+8.36%)	**.072 (+56.52%)**	.184 (+10.84%)	.271 (+18.34%)
	STAN	.281 (+6.43%)	**.101 (+32.89%)**	.214 (+15.05%)	.283 (+18.90%)
Foursquare TKY	RNN	.196 (+6.56%)	**.028 (+40.02%)**	.123 (+21.78%)	.171 (+15.54%)
	ST-RNN	.213 (+7,58%)	**.057 (+67.65%)**	.133 (+15.64%)	.194 (+24.36%)
	Deep Move	.223 (+5,69%)	**.060 (+46.34%)**	.142 (+19.33%)	.201 (+19.64%)
	LSTPM	.233 (+6.88%)	**.074 (+57.45%)**	.151 (+21.77%)	.236 (+34.86%)
	STAN	.246 (+9.33%)	**.093 (+82.35%)**	.153 (+18.60%)	.239 (+32.04%)
Gowalla	RNN	.142 (+29.09%)	**.157 (+68.82%)**	.016 (+45.45%)	.144 (+50.03%)
	ST-RNN	.149 (+8.76%)	**.143 (+28.83%)**	.033 (+17.86%)	.127 (+19.81%)
	Deep Move	.165 (+4.43%)	**.164 (+41.38%)**	.041 (+13.89)	.151 (+32.46%)
	LSTPM	.171 (+12.50%)	**.182 (+61.06%)**	.043 (+34.38%)	.151 (+36.04%)
	STAN	.206 (+6.77%)	**.178 (+43.55%)**	.059 (+15.69%)	.146 (+22.69%)
Taxi Porto	RNN	.421 (+2.68%)	**.069 (+4.54%)**	.296 (+1.02%)	.398 (+1.79%)
	ST-RNN	.427 (+2.64%)	.077 (+5.47%)	.313 (+3.98%)	**.418 (+5.55%)**
	DeepMove	.466 (+5.42%)	**.104 (+6.12%)**	.341 (+3.96%)	.434 (+6.11%)
	LSTPM	.457 (+6.52%)	**.095 (+6.74%)**	.336 (+6.32%)	.419 (+5.01%)
	STAN	.483 (+6.62%)	**.111 (+7.96%)**	.351 (+6.68%)	.440 (+7.05%)
Taxi SF	RNN	.288 (+2.49%)	.193 (+4.89%)	.208 (+1.46%)	**.276 (+5.34%)**
	ST-RNN	.297 (+4,95%)	.200 (+5.82%)	.225 (+6.64%)	**.298 (+8.76%)**
	Deep Move	.313 (+5,74%)	.202 (+4.12%)	.227 (+4.13%)	**.297 (+6.45%)**
	LSTPM	.301 (+5.24%)	.199 (+3.65%)	.238 (+9.68%)	**.293 (+5.02%)**
	STAN	.330 (+5.11%)	.208 (+3.48%)	.233 (+4.48%)	**.309 (+5.79%)**

Table 2 ACC@5 of all the models on all the datasets. We find a significant relative improvement, especially on the trajectories with a 0-20 overlap. Regarding check-in datasets, we have the greatest relative improvement on the stratification based on JS (in bold). In taxi datasets, we have similar improvements on JS and OFE and while in Taxi Porto, we have the best improvements on JS, on Taxi San Francisco, we reach the best improvements on OFE. In general, the improvements in check-in datasets are higher with respect to taxi datasets. There is not a specific model on which we have the best improvements.

8 Discussion and Recommendations

This work finds that the models' performances are deeply affected by the level of overlap present in the test trajectories. Based on the amount of trajectory overlap, we identify three scenarios:

- **Known Mobility**: the NL sees the entire trajectory in the training phase (overlap between 80% and 100%). Predictive performance is much higher than the performance on a non-stratified test set (close to 100%) as the test trajectories are almost identical to the training trajectories.
- **Fragmentary Mobility**: the NL sees a significant portion of the trajectory (overlap between 20% and 80%). The majority of trajectories in the test set lies in this scenario. There is a drop in the model performance compared to the previous scenario, decreasing up to ∼80%.
- **Novel Mobility**: the NL sees a tiny or no portion of the trajectory (overlap below 20%). A significant number of trajectories lie in this scenario. However, since NLs cannot rely on the trajectories already seen in the training phase, these are the most difficult trajectories to predict. Indeed, the performance of NLs on test sets with low overlap is considerably lower than the performance on a non-stratified test set.

While predicting known mobility is a simple task, inferring mobility patterns for fragmentary mobility and novel mobility presents challenges (e.g., dealing with under-represented locations or not represented at all in the training set). From a modeling perspective, this may suggest that current models are excellent in memorizing already seen trajectories but cannot generalize well. Some works suggest that reranking techniques or few-short learning algorithms may help solve this problem [47]. Also, results indicated that NLs might not be evaluated adequately. In this sense, here we provide a set of recommendations for the evaluation of NLs:

1. MMCs achieve performance similar to NLs. Therefore, we claim that MMCs and other Markov chains approaches should always be used as a baseline.
2. Although NLs achieve good overall performance, they are significantly biased due to trajectory overlap. Besides the NLs' average performance, researchers should report the performance for the known mobility and novel mobility scenarios. It is indeed crucial to understand whether the improved performance of the proposed NL is actually due to its increasing generalization capability or because it is memorizing better the trajectories in the training set;
3. NLs achieve the worst performance on the 0-20 overlap bin. We can improve the performance on this bin, hence increasing NLs' generalization capability with the support of well-known spatial mobility laws, which are loosely captured by state-of-the-art NLs given their reliance on RNNs.

From other perspectives (e.g., urban planning, sustainability, and others), having models that can generalize well is fundamentally important. First, NLs that generalize can be used to perform better simulations and to analyze what-if scenarios more realistically. For instance, we may be able to see how attractive a new POI in a specific place would be. We cannot solve such problems with a model that only memorizes seen trajectories. Also, it can help urban planners make decisions about traffic and transportation and, thus, reduce pollution. We can also use an NL that can generalize to predict better and understand the mobility of individuals who have never been seen in a region (e.g., a tourist). Also, a generalized model may be geographically transferable (e.g., trained in an area and tested on a new territory). This may represent a significant step toward solutions to some of the United Nations' Sustainable Development Goals. In particular, we may use such models to run simulations or investigate pollution, inclusion, and the design of better cities in territories where we do not have data or have a scarcity of data.

9 Conclusions

In this work, we investigate the generalization capabilities of next-location prediction datasets. We find that model performance is considerably affected by trajectory test-train overlap, suggesting that NLs memorize training trajectories rather than generalizing. We show we mitigate this issue by injecting

mobility laws into state-of-the-art NLs, achieving relative improvement on test sets with low overlap with the training ones. We aim to consider other mobility laws and use more sophisticated models to rerank the results in future work. It would also be helpful to use explainable AI techniques to understand better the role of mobility laws and the relations between the DL modules composing NLs.

Declarations

Funding Luca Pappalardo has been partially supported by EU project SoBigData++ grant agreement 871042.

Conflict of Interest The authors have no competing interests to declare that are relevant to the content of this article.

Ethics approval not applicable

Consent to participate not applicable

Consent for publication not applicable

Availability of data and materials All the data are publicly available and can be downloaded using the links at github.com/scikit-mobility/DeepLearning4HumanMobility.

Code availability The code used to compute the overlap can be found at github.com/MassimilianoLuca/overlap-processing the code of the models can be found at github.com/LibCity/Bigscity-LibCity

Authors' contributions M.L. designed the methodology to compute the overlap and the rerank methodology. G.B. directed the study. All the authors contributed to interpreting the results and writing the paper. G.B. developed this work prior joining Amazon.

Appendix A NL's performances

			JC					LCST					OFE				
			0-20	20-40	40-60	60-80	80-100	0-20	20-40	40-60	60-80	80-100	0-20	20-40	40-60	60-80	80-100
Gowalla	MMC	0.129	0.108	0.119	0.323	0.526	0.776	0.013	0.182	0.398	0.644	0.837	0.099	0.211	0.546	0.783	0.874
	RNN	0.110	0.093	0.109	0.321	0.498	0.764	0.011	0.173	0.385	0.631	0.824	0.096	0.209	0.532	0.777	0.871
	ST-RNN	0.137	0.111	0.128	0.355	0.513	0.782	0.028	0.195	0.406	0.658	0.849	0.106	0.217	0.558	0.793	0.888
	DeepMove	0.158	0.116	0.142	0.373	0.536	0.798	0.036	0.211	0.434	0.681	0.869	0.114	0.226	0.573	0.822	0.899
	LSTPM	0.152	0.113	0.141	0.369	0.528	0.793	0.032	0.203	0.427	0.669	0.861	0.111	0.219	0.569	0.818	0.897
	STAN	0.192	0.124	0.183	0.398	0.561	0.813	0.051	0.221	0.449	0.704	0.888	0.119	0.231	0.579	0.829	0.911
Foursquare NYC	MMC	0.245	0.045	0.214	0.518	0.730	0.889	0.102	0.336	0.653	0.858	0.987	0.230	0.529	0.778	0.885	0.988
	RNN	0.213	0.026	0.186	0.472	0.643	0.740	0.132	0.271	0.504	0.528	0.869	0.191	0.610	0.712	0.926	0.988
	ST-RNN	0.248	0.032	0.236	0.538	0.709	0.859	0.161	0.322	0.544	0.589	0.920	0.228	0.679	0.779	0.928	0.989
	DeepMove	0.259	0.051	0.241	0.539	0.716	0.873	0.178	0.330	0.569	0.631	0.926	0.231	0.683	0.786	0.935	0.996
	LSTPM	0.251	0.046	0.239	0.533	0.717	0.865	0.166	0.327	0.551	0.599	0.922	0.229	0.680	0.782	0.929	0.991
	STAN	0.264	0.076	0.269	0.544	0.732	0.875	0.186	0.331	0.574	0.636	0.925	0.238	0.686	0.792	0.936	0.992
Foursquare TKY	MMC	0.216	0.043	0.172	0.450	0.691	0.856	0.096	0.243	0.428	0.634	0.718	0.199	0.360	0.691	0.716	0.991
	RNN	0.183	0.020	0.153	0.390	0.609	0.783	0.101	0.205	0.381	0.593	0.692	0.148	0.464	0.703	0.737	0.892
	ST-RNN	0.198	0.034	0.158	0.397	0.637	0.794	0.115	0.224	0.394	0.608	0.699	0.156	0.468	0.711	0.739	0.906
	LSTPM	0.211	0.041	0.164	0.431	0.688	0.848	0.119	0.233	0.419	0.621	0.715	0.168	0.471	0.718	0.743	0.931
	DeepMove	0.218	0.047	0.168	0.452	0.694	0.854	0.124	0.239	0.424	0.638	0.719	0.175	0.472	0.723	0.744	0.935
	STAN	0.225	0.051	0.174	0.458	0.696	0.857	0.129	0.244	0.426	0.639	0.722	0.181	0.475	0.728	0.749	0.938
Taxi Porto	MMC	0.309	0.113	0.258	0.316	0.385	0.482	0.274	0.302	0.343	0.397	0.411	0.286	0.333	0.349	0.427	0.436
	RNN	0.410	0.066	0.269	0.398	0.516	0.638	0.293	0.381	0.461	0.542	0.649	0.391	0.458	0.516	0.617	0.643
	ST-RNN	0.416	0.073	0.272	0.404	0.519	0.639	0.301	0.395	0.467	0.544	0.653	0.396	0.463	0.518	0.623	0.649
	DeepMove	0.442	0.098	0.296	0.419	0.528	0.651	0.328	0.417	0.486	0.561	0.672	0.409	0.467	0.532	0.637	0.684
	LSTPM	0.429	0.089	0.281	0.413	0.521	0.647	0.316	0.411	0.473	0.558	0.668	0.399	0.466	0.524	0.629	0.661
	STAN	0.453	0.103	0.305	0.421	0.532	0.649	0.329	0.423	0.492	0.558	0.679	0.411	0.474	0.537	0.631	0.692
Taxi SF	MMC	0.242	0.169	0.261	0.385	0.542	0.583	0.194	0.248	0.308	0.451	0.505	0.231	0.256	0.389	0.427	0.692
	RNN	0.281	0.184	0.279	0.403	0.565	0.601	0.205	0.263	0.321	0.475	0.539	0.262	0.303	0.414	0.504	0.816
	ST-RNN	0.283	0.189	0.284	0.411	0.568	0.607	0.211	0.274	0.329	0.486	0.549	0.274	0.315	0.429	0.521	0.834
	LSTPM	0.296	0.194	0.286	0.429	0.574	0.609	0.218	0.278	0.334	0.492	0.558	0.279	0.319	0.438	0.529	0.847
	DeepMove	0.286	0.192	0.287	0.414	0.571	0.609	0.217	0.277	0.332	0.489	0.551	0.279	0.318	0.435	0.527	0.841
	STAN	0.314	0.201	0.295	0.433	0.581	0.614	0.223	0.281	0.349	0.498	0.573	0.293	0.326	0.443	0.532	0.850

Table A1 ACC@5 of all the models on all the datasets without a stratification (first column) and with the train-test stratification based on overlap metric and percentage of overlap

Appendix B NL's performances after re-ranking

			JC					LCST					OFE				
			0-20	20-40	40-60	60-80	80-100	0-20	20-40	40-60	60-80	80-100	0-20	20-40	40-60	60-80	80-100
Gowalla	RNN	0.140	0.160	0.148	0.347	0.533	0.777	0.016	0.192	0.403	0.647	0.837	0.144	0.257	0.546	0.783	0.874
	ST-RNN	0.117	0.143	0.148	0.326	0.506	0.765	0.013	0.184	0.393	0.634	0.825	0.127	0.263	0.532	0.777	0.871
	DeepMove	0.145	0.164	0.156	0.371	0.521	0.783	0.033	0.204	0.415	0.661	0.850	0.151	0.265	0.558	0.793	0.888
	LSTPM	0.171	0.182	0.178	0.379	0.544	0.799	0.043	0.225	0.452	0.684	0.869	0.151	0.276	0.573	0.822	0.899
	STAN	0.163	0.178	0.182	0.385	0.615	0.797	0.039	0.209	0.440	0.670	0.862	0.146	0.267	0.569	0.818	0.897
Foursquare NYC	RNN	0.233	0.051	0.265	0.479	0.651	0.741	0.158	0.276	0.598	0.531	0.869	0.241	0.742	0.712	0.926	0.988
	ST-RNN	0.261	0.059	0.287	0.563	0.722	0.861	0.186	0.328	0.552	0.590	0.920	0.299	0.787	0.779	0.928	0.989
	DeepMove	0.277	0.084	0.300	0.571	0.724	0.875	0.213	0.358	0.593	0.633	0.927	0.268	0.766	0.786	0.935	0.996
	LSTPM	0.272	0.072	0.290	0.562	0.728	0.873	0.184	0.356	0.576	0.601	0.923	0.271	0.763	0.782	0.929	0.991
	STAN	0.281	0.101	0.327	0.570	0.740	0.877	0.214	0.359	0.592	0.638	0.925	0.283	0.778	0.792	0.936	0.992
Foursquare TKY	RNN	0.195	0.028	0.186	0.408	0.618	0.785	0.123	0.317	0.394	0.596	0.692	0.171	0.520	0.703	0.737	0.892
	ST-RNN	0.213	0.057	0.213	0.411	0.647	0.800	0.133	0.235	0.408	0.614	0.699	0.194	0.525	0.711	0.739	0.906
	LSTPM	0.223	0.060	0.208	0.452	0.698	0.856	0.142	0.246	0.437	0.622	0.715	0.200	0.540	0.718	0.743	0.931
	DeepMove	0.230	0.074	0.209	0.488	0.704	0.858	0.151	0.338	0.444	0.643	0.719	0.236	0.564	0.723	0.744	0.935
	STAN	0.236	0.093	0.278	0.501	0.707	0.865	0.153	0.329	0.447	0.642	0.726	0.239	0.553	0.728	0.749	0.938
Taxi Porto	RNN	0.421	0.069	0.275	0.436	0.525	0.643	0.306	0.387	0.468	0.545	0.649	0.398	0.458	0.516	0.617	0.643
	ST-RNN	0.427	0.077	0.279	0.415	0.529	0.641	0.317	0.401	0.473	0.548	0.654	0.418	0.463	0.518	0.623	0.649
	LSTPM	0.466	0.104	0.301	0.429	0.536	0.652	0.348	0.426	0.498	0.563	0.673	0.434	0.467	0.532	0.637	0.684
	DeepMove	0.457	0.095	0.287	0.423	0.529	0.650	0.337	0.422	0.482	0.559	0.668	0.419	0.466	0.524	0.629	0.661
	STAN	0.483	0.111	0.316	0.436	0.542	0.652	0.355	0.431	0.502	0.563	0.680	0.440	0.474	0.537	0.631	0.692
Taxi SF	RNN	0.288	0.193	0.285	0.422	0.571	0.604	0.208	0.271	0.327	0.477	0.539	0.276	0.303	0.414	0.504	0.816
	ST-RNN	0.297	0.200	0.294	0.433	0.575	0.608	0.225	0.278	0.333	0.488	0.549	0.298	0.315	0.429	0.521	0.834
	DeepMove	0.313	0.202	0.292	0.449	0.581	0.610	0.230	0.282	0.338	0.493	0.558	0.297	0.319	0.438	0.529	0.847
	LSTPM	0.301	0.199	0.297	0.433	0.578	0.613	0.238	0.282	0.338	0.493	0.552	0.293	0.318	0.435	0.527	0.841
	STAN	0.330	0.208	0.304	0.461	0.592	0.617	0.233	0.288	0.355	0.503	0.574	0.310	0.326	0.443	0.532	0.850

Table B2 ACC@5 of all the models after the re-ranking on all the datasets without a stratification (first column) and with the train-test stratification based on overlap metric and percentage of overlap

References

[1] Shi, Y., Feng, H., Geng, X., Tang, X., Wang, Y.: A survey of hybrid deep learning methods for traffic flow prediction. In: Proceedings of the 2019 3rd International Conference on Advances in Image Processing, pp. 133–138. Association for Computing Machinery, ??? (2019)

[2] Khaidem, L., Luca, M., Yang, F., Anand, A., Lepri, B., Dong, W.: Optimizing transportation dynamics at a city-scale using a reinforcement learning framework. IEEE Access **8**, 171528–171541 (2020)

[3] Barlacchi, G., Perentis, C., Mehrotra, A., Musolesi, M., Lepri, B.: Are you getting sick? predicting influenza-like symptoms using human mobility behaviors. EPJ Data Science, 27 (2017)

[4] Canzian, L., Musolesi, M.: Trajectories of depression: unobtrusive monitoring of depressive states by means of smartphone mobility traces analysis. In: Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing, pp. 1293–1304 (2015)

[5] Pappalardo, L., Vanhoof, M., Gabrielli, L., Smoreda, Z., Pedreschi, D., Giannotti, F.: An analytical framework to nowcast well-being using mobile phone data. International Journal of Data Science and Analytics, 75–92 (2016)

[6] Voukelatou, V., Gabrielli, L., Miliou, I., Cresci, S., Sharma, R., Tesconi, M., Pappalardo, L.: Measuring objective and subjective well-being: dimensions and data sources. International Journal of Data Science and Analytics (2020)

[7] Zhu, W.-Y., Peng, W.-C., Chen, L.-J., Zheng, K., Zhou, X.: Modeling user mobility for location promotion in location-based social networks. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1573–1582 (2015)

[8] Burbey, I., Martin, T.L.: A survey on predicting personal mobility. International Journal of Pervasive Computing and Communications (2012)

[9] Wu, R., Luo, G., Shao, J., Tian, L., Peng, C.: Location prediction on trajectory data: A review. Big Data Min. Anal. **1**, 108–127 (2018)

[10] Zheng, X., Han, J., Sun, A.: A survey of location prediction on twitter. IEEE Transactions on Knowledge and Data Engineering **30**(9), 1652–1671 (2018)

[11] Zhao, L.: Event prediction in big data era: A systematic survey. arXiv preprint arXiv:2007.09815 (2020)

[12] Barbosa, H., Barthelemy, M., Ghoshal, G., James, C.R., Lenormand, M., Louail, T., Menezes, R., Ramasco, J.J., Simini, F., Tomasini, M.: Human mobility: Models and applications. Physics Reports **734**, 1–74 (2018)

[13] Luca, M., Barlacchi, G., Lepri, B., Pappalardo, L.: A survey on deep learning for human mobility. ACM Comput. Surv. **55**(1) (2021). https://doi.org/10.1145/3485125

[14] Song, C., Qu, Z., Blumm, N., Barabási, A.-L.: Limits of predictability in human mobility. Science, 1018–1021 (2010)

[15] Amichi, L., Viana, A.C., Crovella, M., Loureiro, A.A.: Understanding individuals' proclivity for novelty seeking. In: Proceedings of the 28th International Conference on Advances in Geographic Information Systems, pp. 314–324 (2020)

[16] Lewis, P., Stenetorp, P., Riedel, S.: Question and answer test-train overlap in open-domain question answering datasets. arXiv preprint arXiv:2008.02637 (2020)

[17] Sen, P., Saffari, A.: What do models learn from question answering datasets? arXiv preprint arXiv:2004.03490 (2020)

[18] Zhou, K., Liu, Z., Qiao, Y., Xiang, T., Loy, C.C.: Domain generalization: A survey. arXiv preprint arXiv:2103.02503 (2021)

[19] Gambs, S., Killijian, M.-O., del Prado Cortez, M.N.: Next place prediction using mobility markov chains. In: Proceedings of the First Workshop on Measurement, Privacy, and Mobility, pp. 1–6 (2012)

[20] Liu, L., Lewis, P., Riedel, S., Stenetorp, P.: Challenges in Generalization in Open Domain Question Answering (2021)

[21] Zhang, C., Zhao, K., Chen, M.: Beyond the limits of predictability in human mobility prediction: Context-transition predictability. IEEE Transactions on Knowledge and Data Engineering (2022)

[22] Smolak, K., Siła-Nowicka, K., Delvenne, J.-C., Wierzbiński, M., Rohm, W.: The impact of human mobility data scales and processing on movement predictability. Scientific Reports **11**(1), 1–10 (2021)

[23] Kulkarni, V., Mahalunkar, A., Garbinato, B., Kelleher, J.D.: Examining the limits of predictability of human mobility. Entropy **21**(4), 432 (2019)

[24] Hofman, J.M., Sharma, A., Watts, D.J.: Prediction and explanation in social systems. Science **355**(6324), 486–488 (2017)

[25] do Couto Teixeira, D., Almeida, J.M., Viana, A.C.: On estimating the

predictability of human mobility: the role of routine. EPJ Data Science **10**(1), 49 (2021)

[26] Pappalardo, L., Simini, F., Rinzivillo, S., Pedreschi, D., Giannotti, F., Barabási, A.-L.: Returners and explorers dichotomy in human mobility. Nature communications **6**(1), 1–8 (2015)

[27] Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning Internal Representations by Error Propagation, pp. 318–362. MIT Press, ??? (1986)

[28] Liu, Q., Wu, S., Wang, L., Tan, T.: Predicting the next location: A recurrent model with spatial and temporal contexts. In: Thirtieth AAAI Conference on Artificial Intelligence (2016)

[29] Sun, K., Qian, T., Chen, T., Liang, Y., Nguyen, Q.V.H., Yin, H.: 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, pp. 214–221 (2020)

[30] Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7794–7803 (2018)

[31] Chang, S., Zhang, Y., Han, W., Yu, M., Guo, X., Tan, W., Cui, X., Witbrock, M., Hasegawa-Johnson, M.A., Huang, T.S.: Dilated recurrent neural networks. Advances in neural information processing systems **30** (2017)

[32] Feng, J., Li, Y., Zhang, C., Sun, F., Meng, F., Guo, A., Jin, D.: Deepmove: Predicting human mobility with attentional recurrent networks. In: Proceedings of the 2018 World Wide Web Conference, pp. 1459–1468 (2018)

[33] Luo, Y., Liu, Q., Liu, Z.: Stan: Spatio-temporal attention network for next location recommendation. In: Proceedings of the Web Conference 2021, pp. 2177–2185 (2021)

[34] Yao, D., Zhang, C., Huang, J., Bi, J.: Serm: A recurrent model for next location prediction in semantic trajectories. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 2411–2414 (2017)

[35] Kong, D., Wu, F.: Hst-lstm: A hierarchical spatial-temporal long-short term memory network for location prediction. In: IJCAI, pp. 2341–2347 (2018)

[36] Gao, Q., Zhou, F., Trajcevski, G., Zhang, K., Zhong, T., Zhang, F.: Predicting human mobility via variational attention. In: The World Wide Web Conference, pp. 2750–2756 (2019)

[37] Yang, D., Fankhauser, B., Rosso, P., Cudre-Mauroux, P.: Location prediction over sparse user mobility traces using rnns: Flashback in hidden states! In: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20, pp. 2184–2190 (2020)

[38] Kawaguchi, K., Kaelbling, L.P., Bengio, Y.: Generalization in deep learning. arXiv preprint arXiv:1710.05468 (2017)

[39] Schläpfer, M., Dong, L., O'Keeffe, K., Santi, P., Szell, M., Salat, H., Anklesaria, S., Vazifeh, M., Ratti, C., West, G.B.: The universal visitation law of human mobility. Nature **593**(7860), 522–527 (2021)

[40] Cho, E., Myers, S.A., Leskovec, J.: 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, pp. 1082–1090 (2011)

[41] Yang, D., Zhang, D., Zheng, V.W., Yu, Z.: Modeling user activity preference by leveraging user spatial temporal characteristics in lbsns. IEEE Transactions on Systems, Man, and Cybernetics: Systems **45**(1), 129–142 (2014)

[42] Piorkowski, M., Sarafijanovic-Djukic, N., Grossglauser, M.: CRAWDAD dataset epfl/mobility (v. 2009-02-24). Downloaded from https://crawdad.org/epfl/mobility/20090224 (2009). https://doi.org/10.15783/C7J010

[43] Moreira-Matias, L., Gama, J., Ferreira, M., Mendes-Moreira, J., Damas, L.: Predicting taxi–passenger demand using streaming data. IEEE Transactions on Intelligent Transportation Systems **14**(3), 1393–1402 (2013)

[44] Wang, J., Jiang, J., Jiang, W., Li, C., Zhao, W.X.: Libcity: An open library for traffic prediction. In: Proceedings of the 29th International Conference on Advances in Geographic Information Systems. SIGSPATIAL '21, pp. 145–148. Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3474717.3483923. https://doi.org/10.1145/3474717.3483923

[45] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

[46] Pappalardo, L., Simini, F., Barlacchi, G., Pellungrini, R.: scikit-mobility: A python library for the analysis, generation and risk assessment of mobility data. arXiv preprint arXiv:1907.07062 (2019)

[47] Wang, Y., Yao, Q., Kwok, J.T., Ni, L.M.: Generalizing from a few examples: A survey on few-shot learning. ACM Computing Surveys (CSUR) **53**(3), 1–34 (2020)