Challenges on Evaluating Venue Recommendation Approaches

Position paper

Pablo Sánchez
Universidad Autónoma de Madrid
Madrid, Spain
pablo.sanchezp@uam.es

Alejandro Bellogín
Universidad Autónoma de Madrid
Madrid, Spain
alejandro.bellogin@uam.es

ABSTRACT

Recommender systems are widely used tools in a large number of online applications due to their ability to learn the tastes and needs of the users. Venue recommendation approaches have recently become particularly useful, and even though these techniques have certain characteristics that differ from traditional recommendation, they deserve special attention from the research community due to the increase on the number of applications using tourism information to perform venue suggestions. In particular, how to properly evaluate (in an offline setting) this type of recommenders needs to be better analyzed, as they are normally evaluated using standard evaluation methodologies, neglecting their unique features. In this paper, we discuss and propose some solutions to two specific aspects around this problem: how to deal with already interacted venues in the test set and how to incorporate the sequence of visited venues by the user when measuring the performance of an algorithm (i.e., in an evaluation metric).

1 INTRODUCTION

The large development of location-based social networks (LBSNs) in recent years has encouraged research on the problem of Point-of-Interest (POI) or venue recommendation, i.e., suggesting new places for users to visit by analysing different contexts such as interaction patterns, friendship relationships, or geographical influence [13, 14]. Foursquare, Gowalla, or GeoLife, and many more, are examples of this kind of social networks, where users record check-ins they make to certain POIs (restaurants, cinemas, hotels, etc.) and share their opinions about them in the application [19, 20]. Because of this, many recommendation techniques have been proposed that exploit these information sources, see for example [7, 11, 12, 14, 19, 21]; however, a critical step to decide whether these algorithms are valuable or could be usable in the real world is the evaluation process, which should be realistic and performed with great care.

With this idea in mind, in this work we analyze some important aspects we have detected related to how POI recommendation tends to be evaluated in offline settings. Our driving hypothesis is that a recommender system should be evaluated in a situation as close as that where it would be used. Because of this, we consider that offline evaluation should be performed by running a temporal split, where the recommender should predict the present (or future) user interactions based on her past interactions [5]. However, independently of whether a temporal split was used, we have detected two challenges that shall be the focus of this paper: first, how should we deal with those venues the user already visited in the past?, and second, can we incorporate the actual order followed by the user (in the test set) to assess the accuracy of the provided recommendations?

In the next sections we motivate and present these two challenges in more detail, and present later some preliminary experiments we have obtained, together with some conclusions and future ideas related to these issues.

2 EVALUATION METHODOLOGIES: KNOWN VS NEW VENUES

In classical recommender systems, no repetitions are typically allowed or considered in the datasets, probably inherited by the domains of the first available datasets (movies) [8, 9]; however, in the venue recommendation context users often visit the same place more than once, and hence, it may make sense to consider how these repetitions should be incorporated in the models and in the evaluation process. This behavior is not limited to venue recommendation, it also happens in music or e-commerce recommendation, and tasks such as session-based recommendation or automatic playlist continuation [16]. Nonetheless, to the best of our knowledge, there is no thorough research about the effects of this paradigm shift, especially regarding the evaluation of the recommendation techniques.

Some papers explicitly state that they separate venues, instead of check-ins, hence, in those cases it is clear that there are no known or visited items in the test set by that user (see [12–14]). However, in other situations it is not obvious how the test set is created, for instance, when temporal splits are created, where repetitions may naturally occur and it is not clear if already known venues were removed from the test set of the user [20]. As we shall see in our experiments, these experimental settings may have a profound impact on the performance of the recommenders and on the observed trends, not only from a reproducibility perspective; hence, the community would benefit from a careful analysis about this issue.

We argue that, by evaluating with items already interacted by the user we are aiming at a different kind of algorithm than when those items are removed. In other terms, a recommender system that performs very well in the first scenario (with known items) is expected to distinguish well which of the previously visited venues the user will visit next. In this context, its final goal is to generate recommendations already known by the user, probably the opposite of a recommender evaluated with only new items in the test set, thus aiming at recommending new, novel venues for each particular user – in fact, some authors define explicitly such a task as *recommending new places* [4].

Our assumption – that we would like to study in the future, since it is out of the scope of this position paper – is that those recommenders that better predict in the case of known venues, would probably generate less novel recommendations in general.

Table 1: Description of the temporal partition evaluated created based on the Foursquare dataset, where U, I, and C denote the number of users, items, and check-ins.

Check-in period	U	I	C	Density	C/U	C/I
Apr'12-Sep'13	267k	3.6M	33M	0.0034%	123.596	9.16
Training: May-Oct '12	202k	1.1M	4.7M	0.0021%	23.267	4.278
Test: Nov '12	150k	352k	831k	0.0017%	5.540	2.361

We believe it would be interesting to understand this effect, in part, to improve current recommendation algorithms that perform well in either of these tasks by creating a hybrid algorithm useful in a real use-case scenario, in such a way that it would detect if new or already visited recommendations should be returned to a user, based on her previous interactions.

3 INTEGRATING SEQUENCES IN EVALUATION

Another specific feature of venue recommendation that differs from the more traditional recommendation problem is that the order in which users visit the venues provides a lot of information. Recently, some methods have been proposed that provide recommendations based on temporal or sequential aspects, such as [6, 21]. However, this information has been neglected, so far, when evaluating these algorithms. Except for the work presented in [6], where the authors propose a metric based on F1 that takes into account the pairwise order between POIs, we have not found other approaches where the evaluation metrics explicitly compare the order of the recommendations against the visited venues.

Furthermore, and related to the discussion presented in the previous section, classical ranking metrics fit the scenario with no repeated items, however, they cannot be adapted to the case where repetitions exist (at least, not to the case where there are repetitions in the test set). Because of this, we believe sequences should be formally integrated and considered when evaluating recommender systems in the venue recommendation context.

With this idea in mind, herein we propose an evaluation metric based on the Longest Common Subsequence (LCS) algorithm, a technique used to find a subsequence of elements (no necessary consecutive) whose length is the maximum possible between two sequences [1]. In our context, one of the sequences will be the recommendation list (R_u) and the other the actual visited venues that appear in the test set of the user (T_u, ordered by ascending timestamp); in this way, the LCS algorithm will measure how many items were recommended in the same order as the user visited them. For instance, if the sequence of items ABCDE is found in the test set of a user, and one recommender suggests ABXCD, whereas another provides ABDXC, the LCS algorithm will score higher the first one, since the subsequence found (exploiting not consecutive items) is larger in that case (4 against 3).

Finally, since the LCS between two sequences is not bounded, we need to normalize this value (lcs). We propose the following three variations when measuring rankings at cutoff N of both recommended and test sequences: $LCSP(R_u, T_u) = lcs(R_u, T_u)/N$ (based on precision), $LCSR(R_u, T_u) = lcs(R_u, T_u)/|T_u|$ (based on recall), and $LCS(R_u, T_u) = lcs(R_u, T_u)^2/(N \cdot |R_u|)$.

4 PRELIMINARY EXPERIMENTS

The experiments have been performed using the global-scale check-in dataset of Foursquare[1] made public by the authors of [17, 18]. Starting from more than 33M check-ins, we created one temporal split containing 6 months of data in its training split and one month for testing (more statistics are shown in Table 1). As a pre-processing step, we performed a 2-core before splitting the data into training and test, so that we force that every user and item has at least 2 check-ins.

We report results obtained by the following recommenders:

- Random (Rnd): random recommender.
- Popularity (Pop): recommender that suggests the most popular items, i.e., items with more check-ins.
- AvgDis: baseline that recommends the closest POIs to the user's average location. The average is computed by calculating the midpoint of the coordinates of the POIs visited by the user.
- PGN: a hybrid approach similar to the USG model proposed in [19] that combines a user-based method (UB), Pop, and AvgDis recommenders. It basically aggregates the scores of every item provided by each of the recommenders, after normalizing each score by the maximum score of each method.
- UB: a k-NN recommender with a user-based approach [15].
- IB: a k-NN recommender with an item-based approach [15].
- HKV: a matrix factorization (MF) approach as described in [10] that uses Alternate Least Squares in the minimization formula.
- IRenMF: weighted MF method proposed by [14]. We selected this approach because, according to the comparison presented in [13], IRenMF was very competitive with a lower execution time with respect to other models, such as GeoMF, Rank-GeoFM, or LFBCA, which agrees with some preliminary experiments we performed in our dataset.

Based on the temporal split presented in Table 1, we decided to focus on the 2 largest cities in terms of number of check-ins (Jakarta and Istanbul) and create 2 independent training-test datasets. Furthermore, in order to make a fair comparison among all the evaluated baselines, we removed repetitions in a user basis for the classical collaborative filtering algorithms; we kept two versions of the training set (with and without check-in frequencies) so that some POI recommendation algorithms, in our case AvgDis and IRenMF, could exploit the frequency of users when visiting a specific venue (denoted as AvgDisFreq and IRenMFFreq). Additionally, to test the experimental conditions discussed in Section 2, we created two test sets: one where those venues the user already interacted in the past (training set) are removed (*with new venues*) and another where they are kept (*with known venues*).

To evaluate the recommenders under the *with known venues* strategy we selected as candidates for each user all the venues that appear in the complete training set of each target city, while when working *with new venues* we remove the ones already rated by that user.

[1] https://sites.google.com/site/yangdingqi/home/foursquare-dataset

We use different ranking metrics to measure accuracy of the recommenders: precision (P), recall (R), mean average precision (MAP), and normalized discounted cumulative gain (NDCG) [2]. We also report the proposed metrics based on LCS, as presented in Section 3. The parameters of the recommenders have been selected by maximizing P@5. Unless stated otherwise, the reported values are computed at a cutoff of 10. Source code to replicate these experiments can be found in the following Bitbucket repository: PabloSanchezP/TempCDSeqEval.

4.1 Comparison of evaluation methodologies

Table 2 shows the results for the cities mentioned before evaluated under the two methodologies presented in Section 2: where only new items for a user appear in her test set (*with new venues*) and where venues already interacted by the user are allowed in the test set (*with known venues*). Nevertheless, for the test set, we always removed the duplicated check-ins (i.e., the users only made one check-in in a POI). As a simple baseline, we have included a method that returns the venues observed in training for each user (Training), ordered by their score and popularity. This baseline, as expected, does not obtain any relevant result in the first scenario, however, when known items are allowed, it is a strong baseline to beat, and some of the more complex algorithms such as IRenMF tend to obtain performance values very close to the ones from this method.

We also notice that in the *with new venues* scenario, the well performing methods tend to be very close to each other (see PGN, UB, IRenMF, and IRenMFFreq in Jakarta), however, in the other scenario the differences increase and some methods take more advantage than others of the different experimental condition.

Another interesting observation is that, as already happens in classical recommendation [3], a popularity bias is found when evaluating in the *with new venues* scenario; however, this bias is strongly reduced in the *with known venues* scenario, favoring the Training baseline, evidencing that in such scenario well-known, popular venues are not as important as previously visited venues by each user, confirming that these two scenarios are actually modeling two different recommendation situations and hypotheses.

4.2 Sequence-aware evaluation metric

To test the evaluation metric proposed in Section 3 based on the LCS algorithm, in Table 2 we have included two methods as skylines (named like this as opposed to the baselines, since their performance is almost impossible to achieve because they look into the test set): TestOrder, that returns the test set in the (ideal) observed order visited by the user (from lowest to highest timestamp), and TestInvOrder, that also returns the test but in the reverse order (from highest to lowest timestamp). The use of these recommenders will serve to justify the LCS-based metric, since besides taking into account the relevance, it also considers the order of visits (note that none of the other recommenders explicitly generates sequences of items, we aim to address this issue in the future). We observe that the LCS-based metrics (LCS, LCSP, LCSR) produce lower values for TestInvOrder than for TestOrder, as TestInvOrder only finds one item in the correct sequence when using these metrics; however, since TestInvOrder obtains much better results than traditional recommenders, we conclude that, for many users, the other algorithms are not able to obtain a single relevant item. At the same time, the skylines obtain the same values by any of the other ranking-based metrics (P, R, MAP, NDCG) since they do not consider the visiting order of the recommended list.

Based on these results, we can provide additional insights about how the different recommendation algorithms behave. For instance, in the *with known venues* scenario, the Training baseline seems to provide recommendations more often in the same order as the one observed in the test set, since the values for the LCS metric is always higher than for any of the sequence-agnostic evaluation metrics. In the other scenario, on the other hand, we do not observe too many variations on how the recommenders are being ranked by each evaluation metric, hence, further analysis and experiments should be performed to better understand this effect. One possible reason for this lack of variability in the performance could be related to the very small number of relevant items returned by the algorithms, in the future we would like to study this problem in more detail.

5 CONCLUSIONS AND FUTURE WORK

In this work, we discuss two aspects regarding how the community should address the evaluation of venue recommendation approaches. First, we analyze whether repeated interactions should be included in the test splits, observing how state-of-the-art recommendation algorithms change under these different experimental conditions. Considering this type of behavior is common in the tourism domain – and inherent to some type of tourists – the presented observations could open up for discussion about how this issue should be addressed in the community, especially, which scenario is more interesting from an offline point-of-view of the evaluation process, without forgetting that some recommendations might be obvious (hence, less useful) for the users [4], as evidenced by the good performance achieved when returning those venues already visited by the user. We aim to continue investigating about this problem in the future, especially about the connection between (lack of) novelty and observed accuracy under experimental conditions with known items. An important issue we aim to address is the best way to exploit check-in datasets such as the one used here, since there is no difference between tourists and locals (which may check-in in nearby places or visit locations as part of their daily life) and, hence, we want (and need) to understand if the derived conclusions concern to locals, tourists, or both.

The second aspect we have presented here is related to the use of sequences in the evaluation of POI recommendation approaches. We have defined an evaluation metric based on the Longest Common Subsequence that takes into account how similar the recommended list is with respect to the order the user checked in the venues. In the future, we would like to explore how this metric behaves on different tasks related to tourism recommendation, such as next-POI recommendation and tour recommendation, where the recommendation order plays an important role. Furthermore, we aim to incorporate in our analysis algorithms that explicitly recommend sequences of items [16].

ACKNOWLEDGMENTS

This work was funded by the project TIN2016-80630-P (MINECO).

Table 2: Performance comparison on 2 different cities including already interacted items by the user in the test set (*with known venues*) and excluding such items (*with new venues*). Our proposal for a sequence-aware evaluation metric is also included (LCS, LCSP, LCSR). Best results are denoted in bold: with † when TestInvOrder and TestOrder are not considered, without † when also the baselines (Rnd, Pop, Training) are not considered.

(a) Istanbul

Recommender	Test with new venues							Test with known venues						
	P	R	NDCG	MAP	LCS	LCSP	LCSR	P	R	NDCG	MAP	LCS	LCSP	LCSR
Rnd	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Pop	0.039	0.076	0.063	0.030	0.008	0.034	0.071	0.054	0.082	0.079	0.036	0.009	0.046	0.075
Training	0.000	0.000	0.000	0.000	0.000	0.000	0.000	†**0.120**	†**0.190**	0.186	0.100	†**0.034**	0.090	†**0.157**
AvgDis	0.001	0.001	0.001	0.001	0.000	0.001	0.001	0.003	0.006	0.007	0.005	0.001	0.002	0.006
AvgDisFreq	0.001	0.002	0.001	0.001	0.000	0.001	0.001	0.003	0.007	0.008	0.005	0.001	0.003	0.006
PGN	0.041	0.082	0.073	0.036	0.009	0.037	0.077	0.070	0.112	0.124	0.065	0.013	0.059	0.101
UB	0.045	0.088	0.078	0.039	0.009	0.039	0.081	0.110	0.167	0.178	0.098	0.021	0.086	0.142
IB	0.036	0.069	0.063	0.032	0.008	0.032	0.064	0.108	0.156	0.175	0.098	0.019	0.082	0.130
HKV	0.043	0.087	0.076	0.039	0.009	0.038	0.080	0.105	0.158	0.170	0.093	0.019	0.082	0.135
IRenMF	0.044	0.089	0.077	0.039	0.010	0.039	0.083	0.100	0.151	0.164	0.090	0.018	0.079	0.130
IRenMFFreq	†**0.047**	†**0.094**	†**0.082**	†**0.042**	†**0.010**	†**0.041**	†**0.087**	**0.117**	**0.181**	†**0.194**	†**0.109**	**0.023**	†**0.092**	**0.154**
TestInvOrder	0.468	0.932	0.978	0.967	0.225	0.100	0.356	0.569	0.910	0.985	0.978	0.162	0.100	0.287
TestOrder	0.468	0.932	0.978	0.967	0.932	0.468	0.932	0.569	0.910	0.985	0.978	0.910	0.569	0.910

(b) Jakarta

Recommender	Test with new venues							Test with known venues						
	P	R	NDCG	MAP	LCS	LCSP	LCSR	P	R	NDCG	MAP	LCS	LCSP	LCSR
Rnd	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Pop	0.029	0.076	0.070	0.044	0.008	0.026	0.073	0.044	0.087	0.091	0.056	0.009	0.038	0.082
Training	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.102	0.196	0.171	0.096	†**0.034**	0.078	0.165
AvgDis	0.001	0.002	0.002	0.001	0.000	0.001	0.002	0.003	0.008	0.007	0.005	0.001	0.002	0.007
AvgDisFreq	0.001	0.002	0.001	0.001	0.000	0.001	0.002	0.004	0.010	0.009	0.006	0.001	0.003	0.009
PGN	0.030	0.078	0.072	†**0.045**	0.008	0.027	0.075	0.056	0.108	0.114	0.069	0.012	0.047	0.100
UB	0.036	0.085	0.075	0.043	0.009	0.032	0.081	0.081	0.141	0.146	0.083	0.019	0.065	0.124
IB	0.019	0.045	0.038	0.021	0.005	0.017	0.043	†**0.120**	†**0.212**	†**0.222**	†**0.141**	**0.026**	†**0.088**	†**0.172**
HKV	0.035	0.084	0.071	0.039	0.009	0.032	0.080	0.078	0.137	0.138	0.078	0.016	0.063	0.121
IRenMF	0.033	0.081	0.071	0.041	0.009	0.030	0.078	0.076	0.135	0.136	0.078	0.016	0.062	0.121
IRenMFFreq	†**0.036**	†**0.092**	†**0.077**	0.044	†**0.010**	†**0.033**	†**0.088**	0.110	0.199	0.193	0.115	0.024	0.084	0.170
TestInvOrder	0.387	0.923	0.963	0.947	0.299	0.100	0.427	0.492	0.912	0.977	0.966	0.223	0.100	0.348
TestOrder	0.387	0.923	0.963	0.947	0.923	0.387	0.923	0.492	0.912	0.977	0.966	0.912	0.492	0.912

REFERENCES

[1] Alberto Apostolico. 1997. String Editing and Longest Common Subsequences. In *Handbook of Formal Languages 2*, Grzegorz Rozenberg and Arto Salomaa (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 361–398.

[2] Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. 2011. *Modern Information Retrieval - the concepts and technology behind search, Second edition.* Pearson Education Ltd., Harlow, England.

[3] Alejandro Bellogín, Pablo Castells, and Iván Cantador. 2017. Statistical biases in Information Retrieval metrics for recommender systems. *Inf. Retr. Journal* 20, 6 (2017), 606–634.

[4] Cécile Bothorel, Neal Lathia, Romain Picot-Clémente, and Anastasios Noulas. 2018. Location Recommendation with Social Media Data. In *Social Information Access*. Lecture Notes in Computer Science, Vol. 10100. Springer, 624–653.

[5] Pedro G. Campos, Fernando Díez, and Iván Cantador. 2014. Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Model. User-Adapt. Interact.* 24, 1-2 (2014), 67–119.

[6] Dawei Chen, Cheng Soon Ong, and Lexing Xie. 2016. Learning Points and Routes to Recommend Trajectories. In *CIKM*. ACM, 2227–2232.

[7] Huiji Gao, Jiliang Tang, Xia Hu, and Huan Liu. 2013. Exploring temporal effects for location recommendation on location-based social networks. In *RecSys*. ACM, 93–100.

[8] Asela Gunawardana and Guy Shani. 2015. Evaluating Recommender Systems. In *Recommender Systems Handbook*. Springer, 265–308.

[9] F. Maxwell Harper and Joseph A. Konstan. 2016. The MovieLens Datasets: History and Context. *TiiS* 5, 4 (2016), 19:1–19:19.

[10] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *ICDM*. IEEE Computer Society, 263–272.

[11] Xutao Li, Gao Cong, Xiaoli Li, Tuan-Anh Nguyen Pham, and Shonali Krishnaswamy. 2015. Rank-GeoFM: A Ranking based Geographical Factorization Method for Point of Interest Recommendation. In *SIGIR*. ACM, 433–442.

[12] Defu Lian, Cong Zhao, Xing Xie, Guangzhong Sun, Enhong Chen, and Yong Rui. 2014. GeoMF: joint geographical modeling and matrix factorization for point-of-interest recommendation. In *KDD*. ACM, 831–840.

[13] Yiding Liu, Tuan-Anh Pham, Gao Cong, and Quan Yuan. 2017. An Experimental Evaluation of Point-of-interest Recommendation in Location-based Social Networks. *PVLDB* 10, 10 (2017), 1010–1021.

[14] Yong Liu, Wei Wei, Aixin Sun, and Chunyan Miao. 2014. Exploiting Geographical Neighborhood Characteristics for Location Recommendation. In *CIKM*. ACM, 739–748.

[15] Xia Ning, Christian Desrosiers, and George Karypis. 2015. A Comprehensive Survey of Neighborhood-Based Recommendation Methods. In *Recommender Systems Handbook*. Springer, 37–76.

[16] Massimo Quadrana, Paolo Cremonesi, and Dietmar Jannach. 2018. Sequence-Aware Recommender Systems. *ACM Comput. Surv.* 51, 4 (2018), 66:1–66:36.

[17] Dingqi Yang, Daqing Zhang, Longbiao Chen, and Bingqing Qu. 2015. NationTelescope: Monitoring and visualizing large-scale collective behavior in LBSNs. *J. Network and Computer Applications* 55 (2015), 170–180.

[18] Dingqi Yang, Daqing Zhang, and Bingqing Qu. 2016. Participatory Cultural Mapping Based on Collective Behavior Data in Location-Based Social Networks. *ACM TIST* 7, 3 (2016), 30:1–30:23.

[19] Mao Ye, Peifeng Yin, Wang-Chien Lee, and Dik Lun Lee. 2011. Exploiting geographical influence for collaborative point-of-interest recommendation. In *SIGIR*. ACM, 325–334.

[20] Jia-Dong Zhang and Chi-Yin Chow. 2013. iGSLR: personalized geo-social location recommendation: a kernel density estimation approach. In *SIGSPATIAL/GIS*. ACM, 324–333.

[21] Jia-Dong Zhang, Chi-Yin Chow, and Yanhua Li. 2014. LORE: exploiting sequential influence for location recommendations. In *SIGSPATIAL/GIS*. ACM, 103–112.

