Improved Recommendation of Photo-Taking Locations using Virtual Ratings

Mesut Kaya
Insight Centre for Data Analytics
University College Cork, Ireland
mesut.kaya@insight-centre.org

Derek Bridge
Insight Centre for Data Analytics
University College Cork, Ireland
derek.bridge@insight-centre.org

ABSTRACT

We consider the task of collaborative recommendation of photo-taking locations. We use datasets of geotagged photos. We map their locations to a location grid using a geohashing algorithm, resulting in a *user* × *location* implicit feedback matrix. Our improvements relative to previous work are twofold. First, we create virtual ratings by spreading users' preferences to neighbouring grid locations. This makes the assumption that users have some preference for locations close to the ones in which they take their photos. These virtual ratings help overcome the discrete nature of the geohashing. Second, we normalize the implicit frequency-based ratings to a 1-5 scale using a method that has been found to be useful in music recommendation algorithms. We demonstrate the advantages of our approach with new experiments that show large increases in hit rate and related metrics.

CCS Concepts

•**Information systems** → **Recommender systems;**

Keywords

photography; geohashing; implicit ratings

1. INTRODUCTION

The advent of digital cameras and smart phones has revolutionised photo-taking. We now create more multimedia content than ever before; the content is more heavily contextualised, e.g. with increasingly accurate data from phone sensors such GPS receivers; and sharing the content has never been easier nor more commonplace.

Web sites, such as Flickr and Facebook, where multimedia content is shared, implicitly capture contextualised personal preferences over the places that people like to create content of this kind, i.e. the places where they take photos. The preference data is a resource from which we can build personalized and contextualized recommender systems [10].

RecTour 2016 - Workshop on Recommenders in Tourism held in conjunction with the 10th ACM Conference on Recommender Systems (RecSys), September 15, 2016, Boston, MA, USA.

In this paper, we study the task of recommending geolocations to users, based on the data found in photo-sharing sites. In other words, the *items* in our recommender system are geolocations. The recommender may help a user discover new places where they can enjoy good views or nice settings, suitable for photo-taking.

Collaborative recommendation of geolocations for photo-taking has been explored in recent work by Phan et al.[13] Briefly, they use a cartographic hashing function to map the latitude and longitude coordinates associated with photos to rectangular bins: photos taken from within the same bin have the same hash key. The *user* × *location* implicit feedback matrix uses the hash keys for *location*s. The rating by a user for a location is given by the proportion of her geotagged photos whose coordinates map to that bin; ratings are therefore in $(0, 1]$. Phan et al. compared an item-based nearest-neighbours recommender with three matrix factorization methods. They ran experiments measuring RMSE, with non-negative matrix factorization having lowest RMSE.

In this paper, we, like Phan et al., are concerned with recommending locations. We use geohashing rather than Phan et al.'s cartographic hashing to map the places where users took the photos to buckets. Then we make two main contributions. First, we create virtual ratings by spreading users' preferences to neighbouring grid locations. This makes the assumption that users have some preference for locations close to the ones in which they take their photos. These virtual ratings help overcome the discrete nature of the geohashing. Second, we normalize the implicit frequency-based ratings to a 1-5 scale using a method that has been found to be useful in music recommendation algorithms. We evaluate the effect of these two innovations separately and together using experiments that measure hit rate and related metrics, rather than RMSE.

In Section 2 we review the related work; in Section 3 we present our proposed method; and in Section 4 we give the experimental results.

2. RELATED WORK

There is an amount of previous research in recommending locations to users, e.g. [11, 3, 16, 21]. For the most part, this work is concerned with point-of-interest (POI) recommendation. For photo-taking, by contrast, we are not directly interested in recommending POI locations; instead, we want to be able to recommend locations that may give views of POIs (and nice settings for other photos). The locations that we recommend may even be far from any POIs. Hence, following [13], we recommend rectangular cells in

the coordinate space. Phan et al. map latitude and longitude coordinates to rectangular bins using a method of their own invention, which they call Cartographic Sparse Hashing (CASH) [1]. Their method has a parameter, r, the resolution. At the Equator, bins will be r metres wide and r metres high. Note, however, that bins will be taller than r metres the further they are away from the Equator due to the curvature of the Earth. The resulting hash key is a 64-bit integer whose high bits are the hash of the longitude and whose low bits are the hash of the latitude. In more recent work, they use CASH within an activity recommender [1].

There are other location recommenders that also work in a coordinate space. For example, Liu et al. try to predict the next location that a user will visit [12]. Interestingly, they, along with Yuan et al. [21], also consider the role of time in location recommendation, which may also be relevant to photo-taking, but which we do not investigate further here.

Shared photos have been used as a data source for purposes such as POI detection [20], tag recommendation [17], photo-taking location detection [5], and route recommendation [14]. Some work specifically uses Flickr data, just as we and Phan et al. use in our work; for example, Zheng et al. recommend Flickr interest groups to users [22]. But none of this work, other than Phan et al.'s, uses this kind of data to recommend photo-taking locations.

The literature also contains descriptions of systems that assist with photo composition, e.g. [2, 15]. Bourke et al. describe what they call the *social camera*, which recommends a list of popular photos that were taken near to the user's location in similar lighting conditions. The user can choose one of these recommended photos, which will then be used as the basis for assistance with camera settings and framing [2]. Rawat proposes a system called ClickSmart that can provide real-time advice about scene composition and camera settings using rules learned from social media images [15].

Our use of virtual ratings is similar in spirit to the approach of fuzzy event modelling proposed by Hidasi and Tikk [7]. They use a similar idea to model continuous contexts in factorization algorithms.

3. PROPOSED METHOD

In this section, we explain our proposed approach. The approach consists of geohashing, followed by the spreading of users' preferences by creating virtual ratings in neighbouring buckets, followed by the conversion of implicit feedback to 1-5 ratings. Finally, we use a collaborative recommendation algorithm on the resulting feedback matrix.

3.1 Geohashing

We do not use Phan et al.'s CASH method, preferring to use *geohashing*, which is more common. Both have the same effect: they divide the surface of the Earth into a grid of rectangular cells (called bins in CASH and buckets in geohashing); a hash function takes in latitude and longitude and maps them to one of the cells of the grid. Geohashing maps latitude and longitude into a geohash key of up to 12-characters. Coordinates that map to the same key are in the same cell of the grid (bucket). The scheme is hierarchical: prefixes of the hash key designate larger cells that include those designated by extensions of the prefix. The size of the prefixes is known as the *precision*. In this paper we take 7-character long prefixes. These designate buckets that at the Equator are 152.9 metres wide and 152.4 metres high.

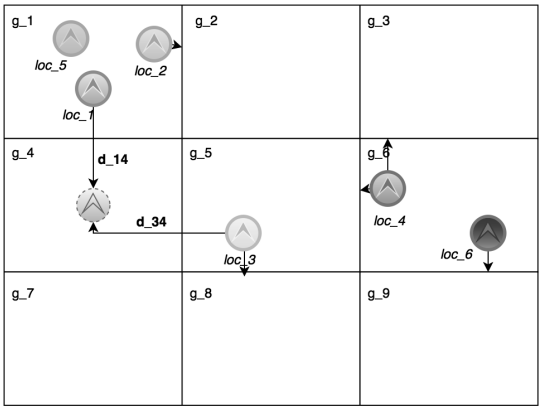

Figure 1: Geohashing

It is interesting to note that Phan et al. run experiments where they vary the resolution of their hashing method. Altering the resolution, however, does not just affect sparsity, it also alters the items (bins) that get recommended. Arguably, this should not be a parameter that one alters to minimise error. It is instead something one should fix at a granularity that users find useful for photo-taking. As we said, we fix precision at 7, which give buckets that are about 150×150 metres, which we think is an appropriate size for photo-taking location recommendation. Had we used a precision of 6, buckets would be approximately 1km by 600 metres, which is clearly too large for useful recommendations. A case can be made for precision of 8 (about 40 by 20 metres) but anything higher is probably too small (e.g. precision of 9 recommends 5 by 5 metre locations).

It is important also to say that the size of the buckets, which we use to recommend *where* to take photos, is unrelated to the size of *what* might be photographed. From a bucket on the south bank of the River Thames, for example, a user might capture a panoramic shot of the north London skyline or she might zoom in on a pigeon eating a discarded hamburger. This emphasises the point too that recommending photo-taking buckets is not the same as recommending POIs. In the same example of a bucket on the south bank of the River Thames, there might be a POI in the same bucket (e.g. the London Eye) but the user might be taking photos of POIs in a different bucket in the distance (e.g. Big Ben) or may not be taking photos of specific POIs at all (e.g. skylines and pigeons).

After hashing, we have an initial *user* $\times$ *location* ratings matrix, where locations are buckets and ratings are based on frequencies. Figure 1 shows an example. Suppose a user u has taken six photos in six different locations. Suppose loc_1, loc_2 and loc_5 are geohashed to the same bucket g_1. The ratings matrix contains triples such as $\langle u, g_1, 3\rangle$, meaning that user u has taken three photos in bucket g_1.

3.2 Creating virtual ratings

One problem with hashing to a rectangular grid is its discretization of coordinate space. In Figure 1, for example, taking a photo at loc_2 is taken as positive feedback for that point in space and others near it. But the rating is recorded only for bucket g_1. The geohashing results in us recording no positive feedback for the nearby points in g_2.

Our solution to this problem is to create virtual ratings in the *user* × *location* matrix by spreading the original frequencies to neighbouring buckets. First, we decide which buckets to spread to. We may spread to zero, one or more of the eight neighbouring buckets. We only spread from a bucket to a neighbour if the bucket contains a photo-taking event that is close enough to the neighbour. We calculate the geodesic distance between the coordinates of the photo-taking events in the bucket and the centre of the neighbour.[1] Only if the minimum of these distances is smaller than a threshold value Δ will we create a virtual rating. For example, in Figure 1, the rating for g_1 will only be spread to g_4 if the distance between loc_1 and the centre of g_4 (this being smaller than the distances from loc_2 and loc_5) is smaller than Δ.

Next, we decide the value of the virtual rating. Its value is a discounted version of the one that is being spread. Following [21], we use a power law distribution to model the preference of a user for a neighbouring bucket as a function of the minimum distance we calculated previously. This maps a distance of 0 to a weight of 1.0 and it maps the maximum distance (Δ) to a weight of 0.0. The neighbour's virtual rating is the product of the weight and rating (frequency) associated with the source bucket.

There is, however, the issue of how to aggregate ratings that 'arrive' in a bucket from different sources. For such cases, we use the simple heuristic that the virtual rating is the *maximum* of the ratings arriving from different sources. For example, in Figure 1, bucket g_4 receives two virtual ratings. One comes from g_1: it is g_1's rating (3) discounted by an amount based on the distance d_{14}. The other comes from g_5: it is g_5's rating (1) discounted by an amount based on distance d_{34}. The larger of these will be taken as the virtual rating for g_4. Note that the same calculation is used even if g_4 already contained a rating of its own: its new rating is the maximum of its original and the two discounted virtual ratings. Since the virtual ratings are discounted by an amount based on distance, only in exceptional cases will they replace an existing rating.

Spreading virtual ratings to neighbouring buckets does not, of course, enlarge what is being recommended. Recommendations continue to be made at the level of individual buckets.

3.3 Rating normalisation

Phan et al. normalize the frequency-based ratings to the range $(0, 1]$ [13]. They do this by dividing the frequency (the number of photos taken by a user in a bin) by the total number of photos taken by that user. However, we found that by their approach 98% of the normalised ratings lie between 0 and 0.1. This has several problems: it implies low preference for 98% of all locations in which a user took a photo; it gives a very skewed distribution; and it means that a recommender that is evaluated using RMSE can do well by always predicting a number between 0 and 0.1.

Instead, we follow Celma's method [4] to convert implicit feedback to a 1-5 rating scale. Following Celma, we compute the Complementary Cumulative Distribution of the frequencies in a user's profile. Then, items (buckets) that fall in the top $80 - 100\%$ of the distribution are given a rating of 5, items that fall into the $60 - 80\%$ range are given a rating of 4, and so on. Celma proposed his method in the context of music listening: to convert how often a user listens to a track into a 5-point scale. Unlike the kinds of 5-point *explicit* rating scales used in book and movie recommenders on the web, for Celma's normalized ratings, a rating of 1 does not necessarily mean that the user *dislikes* the item; rather, the fact that the item was listened to at all implies some level of *positive* feedback, but less enthusiastic positive feedback than that associated with higher points on the scale. It seems appropriate to use Celma's method for the implicit frequency-based ratings that we have in our photo-taking scenario.

[1]We use the `geopy` Python library for this purpose: https://pypi.python.org/pypi/geopy

Table 1: Recommender configurations

	$(0, 1]$ ratings	1-5 ratings
No virtual ratings	1	5
Virtual ratings	1-VR	5-VR

3.4 Recommendation algorithm

At this point, we have a normalized ratings matrix. Our goal, given a user and bucket for which the user has no rating, is to predict the user's rating. For this, we use matrix factorization to transform users and buckets into the same latent factor space. We choose to use matrix factorization since it is widely used for collaborative recommenders and a form of matrix factorization was the best performing approach in [13]. Specifically, for the matrix factorization we use Koren et al.'s SVD [9], solving the objective function using stochastic gradient descent. We have not 'swapped in' different recommender algorithms since our focus is on measuring the contributions made by the virtual ratings and the different forms of normalisation. (We do, however, compare against three baseline recommender systems – see below.)

4. EXPERIMENTS

4.1 Datasets

We collected the data used in this work from the photo-sharing website flickr.com by using its API. We searched for geotagged photos taken in London and Dublin in 2015 to create two datasets. After geohashing, we discarded users who had ratings for fewer than five buckets. The final dataset for London contains 112,671 photos taken by 978 unique users. Users have an average of 115 photos. The final dataset for Dublin contains 54,082 photos taken by 1,567 users and users have an average of 34 photos.

4.2 Recommenders

We compared four configurations of the recommender, depending on whether ratings were normalized to $(0, 1]$ (as in [13]) or to 1-5 (as we propose) and depending on whether virtual ratings are used or not. The names of these four configurations are given in Table 1. It follows that the system called '1' is closest to the one described in [13]: ratings are normalised to $(0, 1]$ and virtual ratings are not used. The main difference with [13], as mentioned in Section 3.1, is that we are using geohashing where they used their CASH method.

We also compare against three baseline recommenders, POP_H, POP_ALL, and HOME, which we now describe.

4.2.1 Popularity-based recommenders

We compare our methodology with a baseline recommender that recommends the most popular items for which the user has no rating. It is well-known that, in recommender systems in general, recommending the most popular items can be a highly competitive baseline [18]. For the kinds of 5-point explicit rating scales used in book and movie recommenders on the web, a popularity-based recommender typically recommends to a user those items that she has not rated and that have the greatest number of high ratings (4 or 5 stars) [18]. In other words, the recommender only recommends items that lots of people like. Accordingly, we include in our experiments a baseline that we refer to as POP_H ('H' for 'high'): in computing popularity, it counts only those users who have given the item a rating of 4 or 5. But, as we have already mentioned in Section 3.3, on the rating scales that we are using, low ratings (e.g. 1 and 2) are not signals of dislike: they show a track was listened to (in the case of music) or a photo was taken (in our case), which is positive feedback, albeit not as positive as a rating of 4 or 5. Hence, we also include another baseline recommender, POP_ALL. In computing popularity, it recommends items rated by the greatest number of users, irrespective of the values of the ratings.

4.2.2 Home location-based recommender

Using a Flickr dataset, Van Laere et al. conclude that a user is more likely to take pictures in locations that are closer to home [19]. It follows then that we can build a baseline recommender that recommends to a user locations for which she has no rating and that are close to her home location.

Some Flickr users provide a textual description of their home location in their Flickr profile. For the users in our datasets, we wrote a crawler to visit their profiles and obtain their home location descriptions, if given. (Of course, not all of these descriptions will be correct, and this may reduce the performance of this baseline recommender.) We convert the textual description to a geolocation (latitude and longitude).[2] For the London dataset, we were able to obtain the home geolocations of 40.52% of the users; for the Dublin dataset, it was 39.5% of the users.

The baseline recommender, which we refer to as HOME, works as follows. For each user whose home location coordinates are known, we calculate the geodesic distance between their home location and the centres of the buckets (items). Then we recommend the closest buckets for which the user has no rating. For users whose home location coordinates are not known, we default to recommending the most popular buckets in the dataset, as POP_ALL would do.

4.3 Methodology and metrics

Phan et al. calculate the RMSE between the predicted and actual ratings for the members of a test set. We chose to base our experiment on a newer methodology, emphasising recommendation over prediction [6]. (In any case, because two of our systems normalise to $(0, 1]$ and two normalise to a discrete 1-5 scale, we cannot directly compare their RMSEs.)

We used 5-fold-cross-validation using 80% for training and 20% for a probe set. From the probe set, we construct a test set. The idea is that the test set will contain items from the probe set that the user liked, and therefore these are ones for which a recommender will be rewarded if it recommends them in the experiments. In [6], where they assume a conventional 1-5 scale, the test set contains only the highly-rated items from the probe set (i.e. ones rated 4 or 5). However, we return to the point that we have made before that, for Celma-style normalisation of frequency data, a rating of 1 or 2 does not necessarily denote dislike. If we want to test recommender performance on all liked items, then we must include in the test set all probe items, irrespective of their values of their ratings.

In fact, we have chosen to run all experiments twice: in one set of experiments, from the probe set we create a test set by retaining only test items where the user's normalised rating is 4 or 5; for the other set of experiments, we take all of the probe set as test set.

There are no virtual ratings in the test sets and, where different recommenders are compared (even ones that normalise to $(0, 1]$), they are compared using the same sets of test items.

For each test item, we randomly select 1000 other buckets for which the user has no rating. We predict the user's ratings for all 1001 buckets and then sort them by descending predicted rating. We then recommend the top-k ($k = 10$), which may or may not include the test item.

This means that experiments are being done on an item by item basis for each item in the test set, rather than on a user by user basis. We must define our metrics accordingly.

For each test item, we measure whether the test item is in the top-k or not. If it is, we call this a hit and record the total number of hits, H. From this, we calculate the *hit rate* (or recall):

$$\text{HR} = \frac{H}{|Test|} \tag{1}$$

where *Test* is the test set.

We also calculate the *average reciprocal hit-rank* (ARHR), which in our setting we define as follows:

$$\text{ARHR} = \frac{1}{|Test|} \sum_{\substack{r_{ui} \in Test, \\ rank_{r_{ui}} \neq 0}} \frac{1}{rank_{r_{ui}}} \tag{2}$$

where $rank_{r_{ui}}$ is the position of this test item in the top-k ($1 \leq rank_{r_{ui}} \leq k$) or zero if this test item was not recommended in the top-k.

Although ARHR considers the positions of the hits, neither ARHR nor HR gives information about the original rating as well as the position. It is common to discount this based on the logarithm of the rank. Hence, inspired by discounted cumulative gain (e.g. [8]), we define *average discounted gain* (since, with only one test item at a time, it is not cumulative), as follows:

$$\text{ADG} = \frac{1}{|Test|} \sum_{\substack{r_{ui} \in Test, \\ rank_{r_{ui}} \neq 0}} \begin{cases} r_{ui} & \text{if} \quad rank_{r_{ui}} = 1 \\ \frac{r_{ui}}{\log_2(rank_{r_{ui}})} & \text{if} \quad rank_{r_{ui}} > 1 \end{cases} \tag{3}$$

4.4 Parameter Δ

When using virtual ratings, there is the parameter Δ, which needs to be set for the experiments. With small values of Δ, there are fewer virtual ratings than with larger values of Δ. We tried values of 0 (which is the same as no virtual ratings), 75, 150, 225 and 300. In Figures 2 and 3,

[2]We do this by using the geopy python library, which uses the Google Maps V3 geocoder: https://developers.google.com/maps/documentation/geocoding/intro

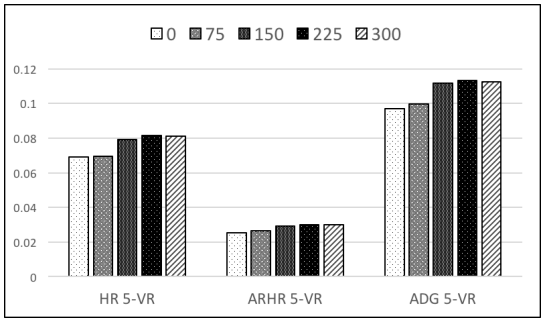

Figure 2: HR, ARHR and ADG for 5-VR with varying Δ, London dataset

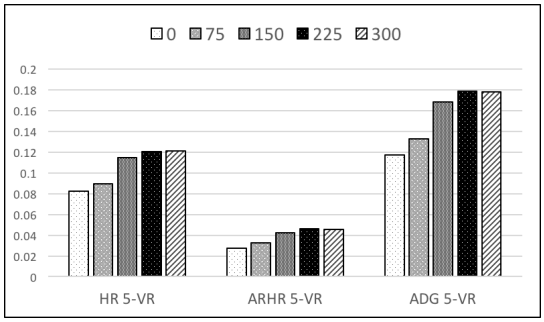

Figure 3: HR, ARHR and ADG for 5-VR with varying Δ, Dublin dataset

we show the hit rate, average reciprocal hit-rank and average discounted gain for one of the system configurations (5-VR) with varying values of Δ for the London and Dublin datasets, respectively.

As can be seen, HR, ARHR and ADG tend to increase as Δ increases but then, as Δ becomes too big and so ratings are being spread too far, HR, ARHR and ADG level off or even fall. The Figures show that the most competitive value for Δ for this system configuration for both datasets is 225.

The results for 1-VR (not shown) follow a similar pattern, but its most competitive value for Δ is 300 for both datasets.

These are the values we use for Δ in the results that we show in the next section.

4.5 Results

We now compare the seven recommenders, i.e. the three baselines (POP_H, POP_ALL and HOME) and the four configurations of our recommender, which depend on the normalisation scheme and whether virtual ratings are used or not (1, 1-VR, 5 and 5-VR). As per the previous section, the two configurations that use virtual ratings use their most competitive values for Δ.

Figures 4, 5 and 6 show the hit rate, average reciprocal hit-rank and average discounted gain respectively.

The worst-performing system overall is the one designated 1, corresponding roughly to the system in [13]. Not only is it out-performed by the other three configurations (1-VR, 5 and 5-VR), it is out-performed by the three baselines (except on the London dataset, where it has very slightly higher HR than HOME does, but even here it is out-performed by the other two baselines).

The best-performing system is 5-VR — the one that combines both our innovations. It out-performs all recommenders on all metrics across both datasets.

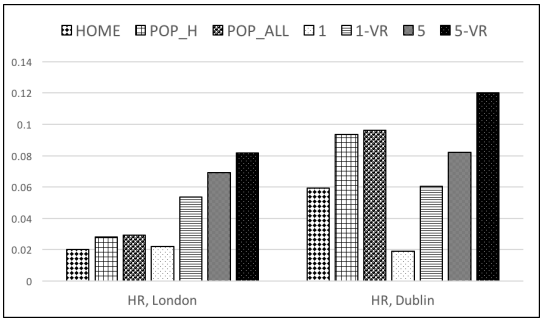

Figure 4: HR

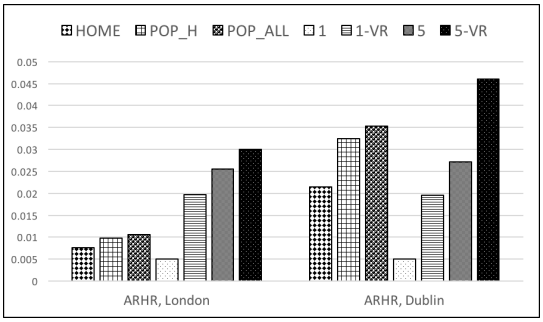

Figure 5: ARHR

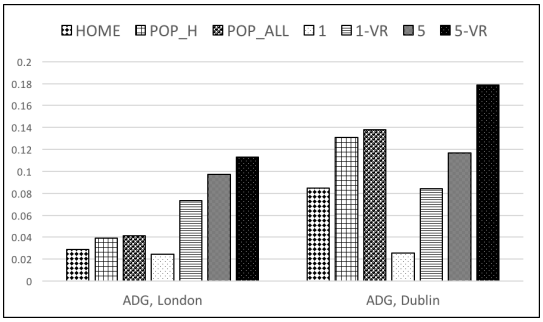

Figure 6: ADG

There are also some other patterns in the results. For both datasets and all evaluation metrics, of the four non-baselines, system 1 is the worst; adding virtual ratings (1-VR) makes an improvement; scaling to 1-5 (system 5) is better again. But the best-performing configuration is 5-VR, where we scale to 1-5 and use virtual ratings.

Another pattern is that the baselines are more competitive on the Dublin dataset than the London dataset. They never out-perform system 5-VR, but on the Dublin dataset, POP_H and POP_ALL both out-perform systems 1, 1-VR and 5. The performance of the HOME baseline is more mixed.

For reference, Table 2 shows the values on which the Figures in this section are based.

In fact, as explained in Section 4.3, we chose to run two set of experiments, differing in the way in which the test set was constructed from the probe set. In one case, all ratings

Table 2: Results for experiments where test set equals probe set

Dataset	Metric	HOME	POP_H	POP_ALL	1	1-VR	5	5-VR
London	HR	0.02	0.0282	0.0294	0.0222	0.0536	0.0691	0.0816
London	ARHR	0.0077	0.0098	0.0106	0.005	0.0197	0.0255	0.0301
London	ADG	0.0287	0.0394	0.0411	0.0243	0.0732	0.0971	0.1131
Dublin	HR	0.0595	0.0936	0.0963	0.0191	0.0603	0.0821	0.1203
Dublin	ARHR	0.0215	0.0324	0.0354	0.005	0.0195	0.0272	0.0461
Dublin	ADG	0.085	0.1308	0.138	0.0255	0.0842	0.1169	0.1788

from the probe set are placed into the test set. These are the experiments whose results were shown in the Figures and in Table 2. In the other case, only highly-rated items from the probe set are included in the test set. The results for this set of experiments are given in Table 3. Comparing the two tables, we see that numerically the results in the second set of experiments are higher. But, for the most part, the story about which systems out-perform each other remains the same. In particular, 5-VR remains far and away the best of the recommenders.

5. DISCUSSION & CONCLUSIONS

We presented an approach to recommending geolocations to users for photo-taking. We geohashed coordinates to cells in a rectangular grid, and used these as the items in an implicit feedback matrix. We investigated two innovations. One was to create virtual ratings in neighbouring cells. The other was to normalise ratings using a method developed for music recommenders. Our experiments, measuring hit rate, average reciprocal hit-rank and average discounted gain, showed that the two innovations together outperformed all other configurations and popularity-based and home location-based baseline recommenders.

There are many avenues for future work. For a start, we can test on other datasets and on recommender algorithms other than SVD. There remains an open question about the precision of the geohashing. Here we are using precision of 7, resulting in buckets that are about 150 metres by 150 metres. The only way to determine whether this is the best choice or whether higher precision (smaller buckets) would be better is through a user trial. It may even be that precision should be personalised or adaptive in some other way, and this could be investigated in future work.

Finally, there is the opportunity to integrate other factors into the work. Presently, we recommend only the photo-taking location. Date and time may also be important: perhaps a location may offer better candidate subjects for photos in certain seasons, on certain dates or at certain times of day. Related to date and time are the photo-taking conditions: a bucket may be a better location, e.g., when the sun has risen, when the weather is not overcast or when the sun is not glaring (although, of course, for some photographers these difficult conditions provide opportunities for the exercise of their photographic talents). Some users might also want assistance with the choice of subject, camera settings or composition. For this, it may be possible to integrate our work with the kind of work done on the social camera [2] and ClickSmart systems [15].

6. ACKNOWLEDGEMENTS

This publication has emanated from research supported in part by a research grant from Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/2289.

7. REFERENCES

[1] P. Bhargava, T. Phan, J. Zhou, and J. Lee. Who, what, when, and where: Multi-dimensional collaborative recommendations using tensor factorization on sparse user-generated data. In *Procs. of the 24th Int. Conf. on the World Wide Web*, pages 130–140, 2015.

[2] S. Bourke, K. McCarthy, and B. Smyth. The social camera: Recommending photo composition using contextual features. In *Procs. of the Second Workshop on Context-Aware Recommender Systems*, 2010.

[3] X. Cao, G. Cong, and C. S. Jensen. Mining Significant Semantic Locations from GPS Data. *Proc. VLDB Endow.*, 3(1-2):1009–1020, 2010.

[4] Ò. Celma. *Music Recommendation and Discovery: The Long Tail, Long Fail, and Long Play in the Digital Music Space*. Springer, 2010.

[5] D. J. Crandall, L. Backstrom, D. Huttenlocher, and J. Kleinberg. Mapping the world's photos. In *Procs. of the 18th Int. Conf. on the World Wide Web*, pages 761–770, 2009.

[6] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Procs. of the 4th ACM Conf. on Recommender Systems*, pages 39–46, 2010.

[7] B. Hidasi and D. Tikk. Approximate modeling of continuous context in factorization algorithms. In *Procs. of the 4th Workshop on Context-Awareness in Retrieval and Recommendation*, pages 3–9, 2014.

[8] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002.

[9] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[10] J. J. Levandoski, M. Sarwat, A. Eldawy, and M. F. Mokbel. Lars: A location-aware recommender system. In *Procs. of the IEEE 28th Int. Conf. on Data Engineering*, pages 450–461, 2012.

[11] B. Liu, Y. Fu, Z. Yao, and H. Xiong. Learning Geographical Preferences for Point-of-interest Recommendation. In *Procs. of the 19th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 1043–1051, 2013.

[12] Q. Liu, S. Wu, L. Wang, and T. Tan. Predicting the Next Location: A Recurrent Model with Spatial and Temporal Contexts. In *Procs. of the 30th AAAI Conf. on Artificial Intelligence*, 2016.

Table 3: Results for experiments where test set contains highly-rated probe items

Dataset	Metric	HOME	POP_H	POP_ALL	1	1-VR	5	5-VR
London	HR	0.0272	0.0413	0.0402	0.0315	0.0698	0.0932	0.1083
London	ARHR	0.0104	0.0137	0.0146	0.0071	0.0261	0.0361	0.0416
London	ADG	0.0657	0.0923	0.0938	0.0577	0.165	0.2258	0.26
Dublin	HR	0.0759	0.1193	0.1241	0.0289	0.081	0.1139	0.1529
Dublin	ARHR	0.0292	0.0442	0.0479	0.0084	0.028	0.0403	0.0629
Dublin	ADG	0.1812	0.2809	0.2962	0.0618	0.185	0.2638	0.3828

[13] T. Phan, J. Zhou, S. Chang, J. Hu, and J. Lee. Collaborative Recommendation of Photo-Taking Geolocations. In *Procs. of the 3rd ACM Multimedia Workshop on Geotagging and its Applications in Multimedia*, pages 11–16, 2014.

[14] D. Quercia, R. Schifanella, and L. M. Aiello. The shortest path to happiness: Recommending beautiful, quiet, and happy routes in the city. In *Procs. of the 25th ACM Conf. on Hypertext and Social Media*, pages 116–125, 2014.

[15] Y. S. Rawat. Real-time assistance in multimedia capture using social media. In *Procs. of the 23rd ACM International Conference on Multimedia*, pages 641–644, 2015.

[16] Y. Shi, P. Serdyukov, A. Hanjalic, and M. Larson. Nontrivial Landmark Recommendation Using Geotagged Photos. *ACM Trans. Intell. Syst. Technol.*, 4(3):47:1–47:27, 2013.

[17] B. Sigurbjörnsson and R. van Zwol. Flickr Tag Recommendation Based on Collective Knowledge. In *Procs. of the 17th Int. Conf. on the World Wide Web*, pages 327–336, 2008.

[18] H. Steck. Item popularity and recommendation accuracy. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 125–132. ACM, 2011.

[19] O. Van Laere, S. Schockaert, and B. Dhoedt. Georeferencing Flickr resources based on textual meta-data. *Information Sciences*, 238:52–74, 2013.

[20] Y. Yang, Z. Gong, and L. H. U. Identifying Points of Interest by Self-tuning Clustering. In *Procs. of the 34th Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 883–892, 2011.

[21] Q. Yuan, G. Cong, Z. Ma, A. Sun, and N. M. Thalmann. Time-aware point-of-interest recommendation. In *Procs. of the 36th Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 363–372, 2013.

[22] N. Zheng, Q. Li, S. Liao, and L. Zhang. Which photo groups should I choose? A comparative study of recommendation algorithms in Flickr. *Journal of Information Science*, 36(6):733–750, 2010.