LearNext: Learning to Predict Tourists Movements

Ranieri Baraglia,
Cristina Ioana Muntean,
Franco Maria Nardini
ISTI–CNR
Pisa, Italy
{name.surname}@isti.cnr.it

Fabrizio Silvestri*
Yahoo! Research Labs
Barcelona, Spain
silvestr@yahoo-inc.com

ABSTRACT

In this paper, we tackle the problem of predicting the "next" geographical position of a tourist given her history (i.e., the prediction is done accordingly to the tourist's current trail) by means of supervised learning techniques, namely Gradient Boosted Regression Trees and Ranking SVM. The learning is done on the basis of an object space represented by a 68 dimension feature vector, specifically designed for tourism related data. Furthermore, we propose a thorough comparison of several methods that are considered state-of-the-art in touristic recommender and trail prediction systems as well as a strong popularity baseline. Experiments show that the methods we propose outperform important competitors and baselines thus providing strong evidence of the performance of our solutions.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Search process*

Keywords

Geographical PoI Prediction; Learning to Rank

1. INTRODUCTION

This work presents LearNext, a "next-tourist-place" predictor allowing the provisioning of the "next" most likely place that a tourist will visit in a city. The approach we propose could be used as a building block to build more complex applications such as: devise suggestions regarding places of interest when visiting a city and make effective predictions of the touristic behavior in a city. In the latter case, devising an effective prediction is required to anticipate or "pre-fetch" possible services in the next location. It could be of help in different scenarios, e.g., i) prediction of touristic flows, ii) location advertising. In the first scenario, our predictor can be used to devise how tourists will visit a city from a macroscopic point of view thus helping the management of the touristic resources of the city, while the second scenario relies on exploiting a touristic prediction for understanding the effect of the advertisement in a particular part of the city or, more important, for choosing where to place it in order to maximize its effects.

LearNext works by predicting touristic places according to the current position of a tourist that is visiting a city and a history of previously visited places (i.e., *visit patterns*) from other users. For the selection of tourist sites, the system uses a set of *Points of Interest* (*PoI*s) identified a priori. In particular, the contributions of this paper are the following:

- we propose LearNext: a next-PoI predictor that learns tourists' behavior from common patterns of movements extracted by Flickr by means of two state-of-the-art machine learning approaches. Our models are trained on a set of 68 features using GBRT [17] and Ranking SVM [6] as learning methods;
- we introduce an unsupervised method for mining common patters of movements of tourists starting from geo-tagged pictures downloaded from Flickr. This is a method which uses i) Flickr as the most important online photo service to gather public photos (and their metadata) from users all around the world and ii) Wikipedia to gather information regarding Points of Interest (PoIs) in the given geographic area. The results of this proposed unsupervised method is a set of structured common patterns of movements of tourists that visited (making photos) the given area in the past;
- we test our methods against important competitors and a strong baseline on three datasets built by means of the methodology above. Each collection corresponds to a popular italian touristic area. In particular, we collect data from photos taken in *Pisa*, *Florence*, and *Rome*. Experiments show that, in all cases, our methods based on Machine Learning techniques consistently outperform with up to 300%, in terms of prediction accuracy, our baselines.

LearNext is structured into two modules: one operating offline and one operating online with respect to the current visit of a tourist. The offline module is used to create the knowledge model that is in turn used for predicting tourist behavior. The online module uses information from the current visit of a tourist and the knowledge model to predict the next location. Recommending next PoIs is a challenging task. One would expect that suggesting the most frequently visited set of PoIs would provide high quality recommenda-

*This work was completed before Fabrizio Silvestri joined Yahoo! Inc.

CIKM'13, Oct. 27–Nov. 1, 2013, San Francisco, CA, USA.
Copyright 2013 ACM 978-1-4503-2263-8/13/10 ...$15.00.
http://dx.doi.org/10.1145/2505515.2505656.

tions. In fact, as show in Section 4, such a baseline performs quite poorly with respect to the methods we develop.

2. RELATED WORK

This paper takes on the problem of predicting the most likely "*Point of Interest*" (PoI) to be visited by a tourist during her tour of a given city. It involves two appealing fields of research in the touristic scenario: *data analysis* and *PoI prediction/recommendation*. The first focuses on the analysis of the photo traces left by tourists when visiting a city and the second studies techniques to predict/recommend interesting PoIs exploiting knowledge mined from historical data.

Data Analysis. A significant number of papers relies on mining geo-spatial and textual metadata associated with Flickr images. Important efforts have been spent in analyzing the dynamics of people moving through cities [5]. Rattenbury *et al.* [14] analyze the geo-temporal dynamics of Flickr tags in order to distinguish between tags describing places and events. Popescu and Grefenstette [13] deduce visit times at landmarks based on timestamps of Flickr photos. Moreover, Ahern *et al.* [1] plot aggregated textual metadata associated with geo-referenced Flickr images on a map interface.

PoI Prediction/Recommendation. A first approach to solve the PoI prediction problem uses trajectory pattern mining to devise temporally-annotated common patterns (trajectories) of movements from data. Trajectories are a concise representation of the behavior of moving objects as sequences of regions frequently visited with typical travel time. Trajectory-based models are exploited in [11], [3], [7] to predict the most likely locations that are of interest for a user.

Monreale *et al.* propose "WhereNext", a method predicting the next location of a moving object [11]. A decision tree, named T-pattern Tree, is built and evaluated with a formal training and test process. The tree is learned from the trajectory patterns within a certain area, and it is used as a predictor for the next location of a new trajectory by finding, on the tree, the best matching path. Finally, the authors show an exhaustive set of experiments and results on real-world datasets.

Krumm and Horvitz propose a trajectory-based system, called Predestination [7]. It tries to predict the location of a certain vehicle as a natural progression of a trip, by exploiting previous covered trajectories.

Noulas *et al.* [12] study the problem of predicting the next venue a mobile user will visit (in foursquare-like terminology, the next *check-in*), by exploring the predictive power offered by different aspects of the user behavior. The authors propose a set of 12 features that aims to capture the factors that may drive users' movements. They model transitions between types of places, mobility flows between venues, and spatio-temporal characteristics of user check-in patterns. Furthermore, they exploit such features in two supervised learning models, based on linear regression and M5 model trees, resulting in a higher overall prediction accuracy. They model the task as a binary classification problem, whereas we model it as a next PoI ranking problem that is based on the likelihood of each PoI to be next in the user trail. Moreover, we propose a broader set of features, originated from a Flickr dataset, capturing more dimensions of the touristic behavior. We cast the prediction problem into a "*learning to rank*" task, which allows us to use two effective Machine Learning techniques (Ranking SVM [6] and GBRT [17]) to solve it.

Similar efforts have been spent in solving the PoI recommendation task. Here, the problem deals with generating a list of possible PoIs that are of interest for a tourist. It differs from the prediction task as it aims at maximizing the satisfaction of the user during her tour of the city, while the first one aims at identifying only one PoI as the first candidate to be visited. In [8], a location-aware recommender system (LARS) that uses location-based ratings to produce recommendations is proposed. Ye *et al.* [15] realize location recommendation services for large-scale location-based social networks, by exploiting the social and geographical characteristics of users and locations/places. Zheng *et al.* perform travel recommendations by mining multiple users' GPS traces [16]. They model multiple users' location histories with a tree-based hierarchical graph.

Lucchese *et al.* propose an algorithm which interactively generates personalized recommendations of touristic places based on the knowledge mined from photo albums and Wikipedia [10]. The authors introduce the model as a graph-based representation of the knowledge, and exploits random walks with restart to select the most relevant PoIs for a specific user.

3. OUR SOLUTION

Let $P = \{p_1, p_2, \ldots, p_n\}$ be a set of Points of Interest (PoIs) for a given touristic location. Let $U = \{u_1, u_2, \ldots, u_m\}$ be a set of users. We assume a tourist u_i has visited a subset of the available PoIs, $V_i \subseteq P$, $V_i = \left\{v_1^i, v_2^i, \ldots, v_k^i\right\}$. Without loss of generality, we can assume that PoIs in V are ordered according to their visit ordering. In other words u_i visited v_1^i then v_2^i, etc. When the user id is clear from the context, we drop the superscript and we simply refer to visited PoIs as v_j. Let $Y^V = \langle y_1, y_2, \ldots, y_{|P \setminus V|}\rangle$ be an ordering (i.e., a permutation) for the PoIs not yet visited by u_i, such that y_1 is the PoI that the tourist will likely visit after v_k, y_2 the second one, etc. Finally, let Y be a general permutation of PoIs in $P \setminus V$. We can define the LearNext problem as follows.

The problem is to learn a function $n : V \to Y$ over a class of functions H, such that a loss function $\Delta\left(Y, Y^V\right)$ is minimized. The loss function measures the penalty of having ordered PoIs in $P \setminus V$ as in Y instead of having outputted the correct ordering Y^V. Suppose that our data is sampled from a distribution $P\left(V, Y\right)$, then the goal is to minimize the risk:

$$n = \operatorname*{argmin}_{f \in H} \int \Delta\left(f\left(V\right), Y^V\right) dP\left(V, Y\right) \qquad (1)$$

Rather than evaluating the whole set of possibilities, we restrict our optimization problem to the samples in a training set $S \subseteq 2^P$ and we seek to minimize the empirical error

$$n = \operatorname*{argmin}_{f \in H} \sum_{V \in S} \Delta\left(f\left(V\right), Y^V\right) \qquad (2)$$

Our goal is to find the best function n that can predict the ranking of PoIs, which a user has not yet visited, according to the probability of being the next PoI in the trail.

Machine-learned models. The above LearNext problem can be cast into a learning to rank formulation that allows to build models able to order PoIs following their decreasing likelihood of being visited as the next PoI for a given user. A trail is represented in a 68-dimension feature space. Accordingly, models are trained on a dataset containing feature vectors corresponding to touristic trails. Machine Learning, in fact, allows to learn from data the function n that minimizes the error of a given loss function $\Delta\left(Y, Y^V\right)$. In particular, as already said, we resort to study the problem as a "*Learning To Rank*" one [9]. We adopted a learning to rank solution as it allows the automatic construction of ranking models from training data. Indeed, this model can order new objects according to their degrees of relevance for the tourist. This way, the LearNext problem becomes a supervised Machine Learning problem that is solved by building a model that ranks highest the PoI with the highest likelihood of being visited as next by the tourist. In particular, each example is represented by a high-dimensional feature vector and its label indicates the PoI's degree of relevance to the user. The learning algorithm is trained to predict the relevance from the feature vector.

We build the ranking models by relying on two well-know techniques: Ranking SVM [6] and Gradient Boosted Regression Trees (GBRT) [17]. Ranking SVM is a pairwise learning to rank technique based on the well-known Support Vector Machines. Gradient Boosted Regression Trees (GBRT) work by building an ensemble of regression trees, typically of limited depth. During each iteration a new tree is added to the ensemble, minimizing the specified cost function. GBRT defines the current state-of-the-art approach in learning to rank. In the Yahoo! Learning to Rank Challenge 2010 [4] all winning methods incorporated GBRT.

Features of PoIs and tourist trails. An important aspect to take into account for an accurate solution of the LearNext problem using learning to rank consists of carefully designing the feature space so that the main characteristics of the dataset are captured. This is important as it defines the signals that are the basic step for learning the prediction model. In particular, in our tourism scenario we believe that different dimensions can be useful to determine how tourists choose PoIs in a city. When visiting a city, in fact, a tourist takes into account the popularity of a PoI, the distance of a given PoI with respect to her current position, how much a particular PoI matches her interests, the time needed to reach it, the time needed to visit it, etc. To model all these dimensions of tourist behavior we define a set of 68 different features. Each feature aims at capturing a particular signal available in the data. We broadly classify features in two main categories, namely "Session" and "PoI". Session features are meant to model the tourist behavior and capture concepts like groups of PoI visited, distances among PoIs, etc. It is based on the characteristics of each PoI within that user session (trail). On the other hand, PoI features model the characteristics of a candidate PoI, also taking into account the past activities of the tourist. Accordingly, PoI features model the characteristics of the PoI to be suggested.

Tables 1 and 2 summarize the set of features we introduce. Session features (Table 1) are based on the current trail of the user; they can be, for example, the transfer time and the actual visit time spent by a tourist in her session, the number of unique categories for all PoIs in that session, the euclidean and latitude/longitude distance of consecutively visited PoIs in a session (average, max, min, total), time and length of the current session, number of photos per PoI in a session (average, max, min, total), length of the sessions belonging to the same tourist (average, max, min, total) making the current visit.

Feature Name	Description
actualTransferTime	Total transfer time from a PoI to the next one in a session.
actualVisitTime	The visit time for all PoIs in a session.
categsPerSess	Number of categories per session.
distLat_Avg distLat_Max distLat_Min distLat_Tot distLen_Avg distLen_Max distLen_Min distLen_Tot	Average, Max, Min, Total Latitude and Longitude distance between PoIs in a session.
euclideanDist_Avg euclideanDist_Max euclideanDist_Min euclideanDist_Total	Average, Max, Min, Total Euclidean distance between PoIs in a session.
phPoISess_Avg phPoISess_Max phPoISess_Min phPoISess_Tot	Average, Max, Min, Total number of photos of PoIs in a session.
uniqueCategsPerSess	The number of unique categories per session.
sessLen	Number of PoIs in a session.
sessTime	Total time for a session, from beginning to end.
userSessLen_Avg userSessLen_Max userSessLen_Min userSessLen_Total	Average, Max, Min, Total length of sessions belonging to a user.
userSessRatio	The ratio between the number of sessions made by the user and the maximum number of sessions for a user.

Table 1: List of "session" features used to model the behavior of a tourist in a city.

On the other hand, PoI features are based on the next PoI to be suggested and model the distance of the next PoI from the first PoI of the session, whether the PoI belongs to the top ten categories visited by users, the number of times a tourist visits that PoI in the training set, the conditional probability of observing that PoI given the last PoI visited by a user, the probability of observing the PoI as first (resp. last) PoI in the training set, number of photos of the PoI (average, max, min), number of past photos of the PoI from the same user, and the visit time of the PoI (average, max, min, total).

4. EXPERIMENTAL EVALUATION

To assess the effectiveness of our proposed techniques, we use three different datasets built in a fully automatic process by exploiting both photos from Flickr[1], a photo sharing portal, and Wikipedia pages. We build three datasets containing tourist movements covering three Italian cities, important from a touristic point of view: *Pisa*, *Florence*, and *Rome*. They are chosen so as to guarantee a variety of topologies and sizes: small (Pisa), medium (Florence), and large (Rome, i.e., a capital city). The rationale of the choice

[1] http://www.flickr.com

Feature Name	Description
cat1, cat2, ..., cat10	Top 10 most frequent categories.
distFromFirstPoI_Eucl distFromFirstPoI_Lat distFromFirstPoI_Len distFromLastPoI_Eucl distFromLastPoI_Lat distFromLastPoI_Len	Latitude, Longitude and Euclidean distance from last and first PoI of the session.
entropy	The entropy of the last PoI in the session.
freqBigrams	The frequency of the PoI given the last PoI in session.
freqTrigrams	The frequency of the PoI given the last two PoIs in session.
middleProbab	The probability that the PoI is within a trail and not in the extremes.
numCategories	The number of categories assigned to the PoI.
numPhotos_Avg numPhotos_Max numPhotos_Min numPhotos_Total	Average, Max, Min and Total number of photos of the PoI in the collection.
noOfVisits	The total number of visits of a PoI in the collection.
photosPerUser	The total number of photos of belonging to a user.
photosPoI_userId_Avg photosPoI_userId_Total	Average and total number of photos of a PoI for a user.
ratioPhotosPoI	The ratio between the number of photos for the PoI and the maximum number of photos for a PoI.
ratioPoIInUserPhotos	The ratio between the number of photos of a PoI for a user and all the photos belonging to the user.
ratioSessWithPoI	The ratio between the number of sessions containing the PoI and the total number of sessions.
ratioUsersVisitingPoI	The ratio between the number of users visiting the PoI and the total number of users.
startProb stopProb	The probability that a PoI is first or last in a trail.
visitTimePoI_User	The total visit time of a PoI for a user.
visitTime_Avg visitTime_Max visitTime_Min visitTime_StdDev visitTime_Total	Average, Max, Min, StdDev and Total visit time of the PoI.

Table 2: List of "PoI" features used to model the characteristics of each candidate destination.

is to propose a complete evaluation of our techniques and its competitors by varying the size of the cities we are dealing with. The datasets have been made available for download to encourage its use within the community allowing the reproducibility of results[2].

We build the datasets by identifying the PoIs in a certain geographical region and the corresponding photos available on Flickr. Given an area of interest, we firstly collect all the geo-referenced Wikipedia pages falling within this region. We assume each geo-referenced Wikipedia page, whose geographical coordinates falls into the given area, to be a Point of Interest in the city we are analyzing. For each PoI, we retrieve its descriptive label as the named entity associated with it, its geographic coordinates as the ones specified in the Wikipedia page, and the set of categories the PoI belongs to, listed in the page[3]. The method is thus able to build a list of PoIs within a given geographical bounding box in a fully automatic way by exploiting Wikipedia as an external source of knowledge.

To devise tourist trails in the area of interest we query Flickr to retrieve the metadata (user id, timestamp, tags, geographic coordinates, etc.) of the photos taken in the given area. The assumption we are making is that photo albums made by Flickr users implicitly represent touristic itineraries within a given city. To strengthen the assumption and thus the accuracy of our method, we retrieve only photos having the highest geo-referenced precision in the given area of interest. Then, we collect geo-tagged photo albums from Flickr users. We discard photo albums containing only one photo and those containing photos with no GPS information associated. Eventually, photos are mapped to the set of PoIs previously collected from Wikipedia. This is done by associating a photo to a PoI if that photo is in the ball having the PoI as its center and $r = 100$ meters as its radius. Moreover, since several photos by the same user are usually taken close to the same PoI, we collapse them by considering the timestamps associated with the first and last of these photos as the starting and ending time of the user visit to the PoI. The results of the assignment above produce, for each Flickr user, a stream of PoIs she visited.

Finally, in order to build the trail sets, we need a way to split the stream of PoIs visited by each user in a meaningful and realistic time-wise set of trails. We employ a time-based cutting method that produces the list of trails a user performed, by considering the inter-arrival time of each pair of sequential photos in her stream. To do so, for each city, we compute the distribution of probability of the inter-arrival time x to be less then a given time threshold k, i.e., $P(x \leq k)$. Then for each dataset we devise the time threshold k corresponding to $P(x \leq k) = 0.9$. Regarding Rome, it corresponds to 5 hours, for Florence 6 hours, while for Pisa 3 hours.

Table 3 shows the main properties of the datasets we use to evaluate our techniques. We report the number of PoIs (column "PoIs") that have been found for each of the three cities. Furthermore, columns "Users" and "Photos" report the number of distinct users and public photos we crawled from Flickr. Table 4 shows the main properties of the trails we extracted using pictures in the dataset. We report the number of trails containing two or more PoIs, the number of PoIs visited at least once and the average number of trails going through each PoI.

Dataset	PoIs	Users	Photos
Pisa	124	1,825	18,170
Florence	1,022	7,049	102,888
Rome	671	13,772	234,616

Table 3: Properties of the three datasets we use.

[2] Links to the trail datasets: `http://hpc.isti.cnr.it/~muntean/datasets/LearNext.tar.gz`

[3] Wikipedia provides a fine-grained categorization of its geo-referenced pages.

Dataset	Trails ≥ 2	Visited PoIs	Avg. Trails per PoI
Pisa	992	110	9.01
Florence	5,984	888	6.73
Rome	12,565	490	25.64

Table 4: Properties of the trail datasets we build.

The set of possible destinations, given a PoI, is an important information that we may exploit in order to detect the most likely next PoI a tourist will visit. Figure 1 shows the outlinks entropy of the three datasets computed on the distribution of PoIs reached from previous ones. In this case the lower the entropy the higher the likelihood that a user will select a frequently visited PoI.

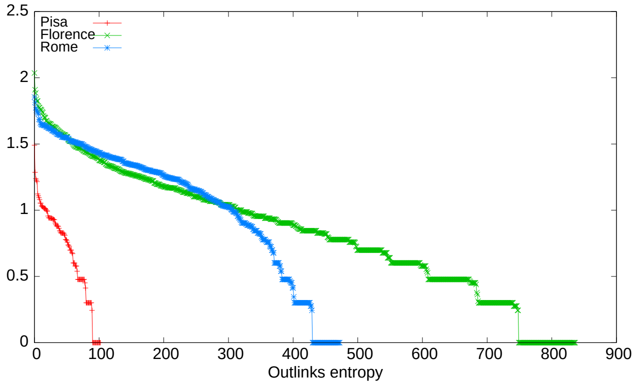

Figure 1: Distribution of the outlinks entropy of the PoIs in the three datasets.

Effectiveness Evaluation. The evaluation of our solution to the LearNext problem is aimed at answering the following research question: **are learning to rank techniques effective for predicting the next PoI?**

We intend to answer the questions above by adopting a standard training/test evaluation strategy over the three datasets of trails available. For each of the three cities, we generate a training set (80%) and a test set (20%) of trails. The effectiveness of the methods is assessed by means of Success@k (i.e., the percentage of times that the correct answer is in the top-k ranked PoIs), MRR (@k), and total MRR [2]. Moreover, we compare our solutions against a probability baseline and two important state-of-the-art techniques, i.e. WhereNext [11] and Random Walk [10]. All the methods have been tested by using the same datasets and the same training/test methodology described above. The results have been validated by means of a standard 10-fold cross validation.

For comparing our solution to the state of the art we use three different baselines.

- "Prob" uses the training set to build a directed graph where nodes are PoIs of the given city and edges are transactions from a source PoI to a destination PoI. Each edge is weighted with the probability to observe the transaction from the source PoI to a destination PoI (if any) in the training set. Given the PoI currently visited by a tourist, Prob predicts the most likely PoI to be visited next by selecting, from the set of the current PoI's outlinks, the one with highest probability.
- "WhereNext" [11] uses trajectory pattern mining to devise T-Patterns, i.e., frequent behaviors of movement in the city, from data. T-Patterns constitute the knowledge model used to compute the prediction. In this paper, we use the original implementation of the predictor presented in [11] kindly provided to us by the authors. We test different combinations of parameters to mine T-Patterns from our datasets.
- "Random Walk" [10] employs a graph-based representation of the PoIs in a city. Authors named it "itinerary graph" and exploit it by using a random walk with restart to select the most relevant PoIs for a given tourist. As for WhereNext, we use the original implementation of the method presented in [10], provided by the authors. We build the itinerary graph over each of the cities we are considering and, for each trail in the test set, we compute the list of the top-10 recommendations. The list of recommended PoIs is then used to evaluate how good is the method at predicting the next candidate PoI. For the two methods above, we report only the best performances we obtain.

The evaluation strategy we use to assess how the proposed techniques behave in terms of effectiveness is the following: each model for the three cities has been trained on the corresponding training set. A **training set** contains positive and negative examples of candidate next PoI, represented by its features. Given a trail of length N, training set contains both session features (computed on the first $N-1$ PoIs of the trail) and PoI features. The latter are computed considering both the actual next PoI visited by the tourist, i.e., the N-th PoI of the trail (as a positive example) and a few negative examples, with PoIs different from the ones seen in the actual trail. Negative examples have been selected on a distance basis. Two negative examples have been selected from PoIs close to the N-th one while one has been selected far from the N-th one. For building the **test set** we adopt the following process. Given a trail of length N in the test set, we use the first $N-1$ PoIs of the trail to profile the tourist history and re-rank all final PoIs observed in the training, according to the prediction model. The resulting sorted list is then evaluated by using the metrics introduced before. The aim of this evaluation is thus to measure how many times our models are able to re-rank correctly, i.e., to rank in the first positions of the whole list of PoIs the actual next PoI.

We measure metrics like: Success@k, MRR@k and total MRR, i.e. MRR computed on the complete list of re-ranked PoIs. Results are computed for all the techniques, our proposed solutions to the LearNext problem along with three methods we choose as baselines. Table 5 shows the results of the experiment. WhereNext and Random Walk never outperform Prob in terms of Success@1. Instead, the techniques we propose consistently outperform all the baselines. For *Pisa*, in terms of Success@1, Ranking SVM scores 32.66% and GBRT scores 40.70%, while Prob scores 16.08%. Important results should be highlighted also for Success@2. Here, our methods are able to score 49.74% (Ranking SVM), and 55.27% (GBRT). Roughly speaking, in half of the cases our methods are able to rank the actual next PoI in the two highest positions of the list. Performance improves when considering higher values for the cut-off parameter. In particular, if we look at the performance in terms of Success@5, Random Walk attains a score of 46.73%, whereas Ranking SVM scores 73.36%, and GBRT

City	Predictor	Success (MRR)					MRR
		@1	@2	@3	@5	@10	
Pisa	Prob	16.08%	-	-	-	-	-
	WhereNext [11]	12.56%	-	-	-	-	-
	Random Walk [10]	15.07% (0.15)	20.60% (0.17)	25.12% (0.19)	31.65% (0.20)	46.73% (0.22)	-
	Ranking SVM	32.66% (0.32)	49.74% (0.41)	55.77% (0.43)	65.82% (0.45)	73.36% (0.46)	0.47
	GBRT	40.70% (0.40)	55.27% (0.47)	63.81% (0.50)	75.87% (0.53)	88.44% (0.55)	0.56
Florence	Prob	4.59%	-	-	-	-	-
	WhereNext [11]	2.90%	-	-	-	-	-
	Random Walk [10]	3.25% (0.03)	6.09% (0.04)	8.77% (0.05)	11.69% (0.06)	20.13% (0.07)	-
	Ranking SVM	33.91% (0.33)	41.01% (0.37)	44.27% (0.38)	48.20% (0.39)	53.29% (0.40)	0.41
	GBRT	37.76% (0.37)	46.78% (0.42)	53.04% (0.44)	59.31% (0.45)	69.34% (0.47)	0.48
Rome	Prob	12.93%	-	-	-	-	-
	WhereNext [11]	6.37%	-	-	-	-	-
	Random Walk [10]	8.43% (0.08)	13.76% (0.11)	19.22% (0.12)	26.38% (0.14)	38.12% (0.16)	-
	Ranking SVM	21.88% (0.21)	30.24% (0.26)	36.37% (0.28)	46.95% (0.30)	59.49% (0.32)	0.33
	GBRT	30.95% (0.30)	40.07% (0.34)	47.15% (0.38)	56.34% (0.40)	67.68% (0.41)	0.42

Table 5: Effectiveness (%) in terms of Success@k, MRR@k, and total MRR of the proposed techniques along with the competitors.

scores 88.44%. GBRT is the technique showing the best performance, while Ranking SVM is second, and both techniques perform considerably better than the baselines we chose. Furthermore, the result for total MRR points out that, even if Ranking SVM and GBRT in some cases are not able to place the next PoI in a high position in the list, the overall ranking does not degrade significantly. The same behavior could be highlighted for *Florence* and *Rome* where both Ranking SVM and GBRT are always outperforming the baselines. In particular, while for *Florence*, GBRT performs about 7 times better w.r.t. Prob, when considering *Rome*, GBRT only doubles the performance of Prob. Looking at the entropy plotted in Figure 1 we would expect that Prob would perform better for *Pisa* than for *Florence* and *Rome*. Indeed, from results in Figure 1 we can observe that Prob behaves as expected. Nevertheless, entropy distribution is very skewed and we expect that in many cases probability features are not enough for a good performance. This is confirmed once again by the effectiveness of our Learning based techniques.

From the results shown above we conclude that *learning to rank techniques are effective for predicting the next PoI in a trail.*

5. CONCLUSIONS AND FUTURE WORK

We proposed to apply machine learning techniques to tackle the problem of predicting the "next" touristic attraction a user will visit on the basis of her visit history (i.e., the prediction is done accordingly to what the user has already visited in the touristic attraction). We modeled the problem as an instance of learning to rank and we defined a feature space composed of 68 features capturing both the touristic behavior and the peculiar characteristics of candidate PoIs. GBRT and Ranking SVM constantly outperform the Prob baseline in terms of prediction accuracy. An immediate extension of this research is to devise a method to plan a visit in a city ahead of time by using LearNext as a building block.

Acknowledgments. This work was partially supported by the EU projects InGeoCLOUDS (no. 297300), MIDAS (no. 318786), E-CLOUD (no. 325091), the Italian PRIN 2011 project "Algoritmica delle Reti Sociali Tecno-Mediate" (2013-2014) and the Regional (Tuscany) project SECURE! (FESR PorCreo 2007-2011).

6. REFERENCES

[1] S. Ahern, M. Naaman, R. Nair, and J. H.-I. Yang. World explorer: visualizing aggregate data from unstructured text in geo-referenced collections. In *Proc. ACM/IEEE-CS DL*, ACM, 2007.

[2] R. Baeza-Yates, B. Ribeiro-Neto, et al. *Modern information retrieval*, volume 463. ACM press, 1999.

[3] R. Baraglia, C. Frattari, C. I. Muntean, F. M. Nardini, and F. Silvestri. A trajectory-based recommender system for tourism. In *Proc. AMT 2012*, 2012.

[4] O. Chapelle and Y. Chang. Yahoo! learning to rank challenge overview. *JMLR*, 14:1–24, 2011.

[5] F. Girardin et al. *Aspects of implicit and explicit human interactions with ubiquitous geographic information.* PhD thesis, 2009.

[6] T. Joachims. Training linear svms in linear time. In *Proc. SIGKDD*, 2006. ACM.

[7] J. Krumm and E. Horvitz. Predestination: inferring destinations from partial trajectories. In *Proc. UbiComp'06*. Springer-Verlag, 2006.

[8] J. J. Levandoski, M. Sarwat, A. Eldawy, and M. F. Mokbel. Lars: A location-aware recommender system. In *Proc. ICDE*, IEEE, 2012.

[9] T.-Y. Liu. Learning to rank for information retrieval. *Found. Trends Inf. Retr.*, 3(3):225–331, Mar. 2009.

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

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

[12] A. Noulas, S. Scellato, N. Lathia, and C. Mascolo. Mining user mobility features for next place prediction in location-based services. In *Proc. ICDM*. IEEE CS, 2012.

[13] A. Popescu and G. Grefenstette. Deducing trip related information from flickr. In *Proc. WWW*. ACM, 2009.

[14] T. Rattenbury, N. Good, and M. Naaman. Towards automatic extraction of event and place semantics from flickr tags. In *Proc. SIGIR*. ACM, 2007.

[15] M. Ye, P. Yin, W.-C. Lee, and D.-L. Lee. Exploiting geographical influence for collaborative point-of-interest recommendation. In *Proc. SIGIR*. ACM, 2011.

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

[17] Z. Zheng, H. Zha, T. Zhang, O. Chapelle, K. Chen, and G. Sun. A general boosting method and its application to learning ranking functions for web search. *ANIPS*, 20:1697–1704, 2007.