LearNext: Learning to Predict Tourists Movements

Ranieri Baraglia[1], Cristina Ioana Muntean[1],
Franco Maria Nardini[1], Fabrizio Silvestri[2]

[1]ISTI-CNR, Pisa, Italy, {name.surname}@isti.cnr.it
[2]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.

1 Introduction

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 from Flickr;
- we introduce an unsupervised method for mining common patters of movements of tourists starting from geo-tagged pictures downloaded from Flickr, matched and enriched with PoIs from Wikipedia;
- we test our methods against important competitors and a strong baseline on three datasets *Pisa*, *Florence*, and *Rome*. Experiments show that, in all cases, our methods based on Machine Learning techniques consistently outperform our baselines in terms of accuracy.

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 by the online one.

2 Our Solution

The 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, initially a sequence of PoIs visited by a user, is represented in a 68-dimension feature space, described in Table 1. The aim is to learn from data the function that minimizes the error of a given loss function. 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 next. We build the ranking models by relying on two well-know techniques: Ranking SVM [2] and Gradient Boosted Regression Trees (GBRT) [5].

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. 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, broadly classified 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.

	Session features	Candidate PoI features
User preferences & type	`userSessLen_Avg/Max/Min/Tot` `userSessRatio`	`ratioPoIInUserPhotos` `photosPerUser`

Distance & time features	`actualTransferTime`
	`actualVisitTime`
	`euclidDist_Avg/Max/Min/Tot`
		`distFromFirstPoI_Eucl`
		`distFromLastPoI_Eucl`
		`visitTime_Avg/Max/Min`

Session characteristics	`uniqueCategsPerSess` `phPoISess_Avg/Max/Min/Tot`	
PoI characteristics		`start/stop/middleProbab` `entropy`
Frequent sequences		`freqBigrams` `freqTrigrams`
Popularity		`noOfVisists` `ratioUsersVisitingPoI`

Table 1. “Candidate PoI” and ”Session” feature examples

Session features 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.

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).

3 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, a photo sharing portal, and Wikipedia pages. We build three datasets containing tourist movements covering three Italian cities, chosen so as to guarantee a variety of topologies and sizes: small (Pisa), medium (Florence), and large (Rome, i.e., a capital city). The datasets have been made available for download[1].

To devise tourist trails in an 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,

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

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.

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 [1]. Moreover, we compare our solutions against a probability baseline and two important state-of-the-art techniques:

- "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.Given the PoI currently visited by a tourist, Prob predicts the most likely PoI to be visited next.
- "WhereNext" [4] 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.
- "Random Walk" [3] 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.

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.

Table 2 shows the results of the experiments. 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, as global MRR shows.

GBRT is the technique showing the best performance, while Ranking SVM is second, and both techniques perform considerably better than the baselines we

City	Predictor	Success (MRR)			MRR
		@1	@2	@3	
Pisa	Prob	16.08%	-	-	-
	WhereNext [4]	12.56%	-	-	-
	Random Walk [3]	15.07% (0.15)	20.60% (0.17)	25.12% (0.19)	-
	Ranking SVM	32.66% (0.32)	49.74% (0.41)	55.77% (0.43)	0.47
	GBRT	40.70% (0.40)	55.27% (0.47)	63.81% (0.50)	0.56
Florence	Prob	4.59%	-	-	-
	WhereNext [4]	2.90%	-	-	-
	Random Walk [3]	3.25% (0.03)	6.09% (0.04)	8.77% (0.05)	-
	Ranking SVM	33.91% (0.33)	41.01% (0.37)	44.27% (0.38)	0.41
	GBRT	37.76% (0.37)	46.78% (0.42)	53.04% (0.44)	0.48
Rome	Prob	12.93%	-	-	-
	WhereNext [4]	6.37%	-	-	-
	Random Walk [3]	8.43% (0.08)	13.76% (0.11)	19.22% (0.12)	-
	Ranking SVM	21.88% (0.21)	30.24% (0.26)	36.37% (0.28)	0.33
	GBRT	30.95% (0.30)	40.07% (0.34)	47.15% (0.38)	0.42

Table 2. Effectiveness (%) in terms of Success@k, MRR@k, and total MRR of the proposed techniques along with the competitors.

chose. The same behavior can be observed for *Florence* and *Rome* where both Ranking SVM and GBRT are always outperforming the baselines.

4 Conclusions

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 outperform baselines in terms of prediction accuracy.

References

1. Baeza-Yates, R., Ribeiro-Neto, B., et al.: Modern information retrieval, vol. 463. ACM press New York. (1999)
2. Joachims, T.: Training linear svms in linear time. In: Proc. SIGKDD. ACM, New York, NY, USA (2006)
3. Lucchese, C., Perego, R., Silvestri, F., Vahabi, H., Venturini, R.: How random walks can help tourism. In: Proc. ECIR. LNCS (2012)
4. Monreale, A., Pinelli, F., Trasarti, R., Giannotti, F.: Wherenext: a location predictor on trajectory pattern mining. In: Proc. SIGKDD. ACM (2009)
5. Zheng, Z., Zha, H., Zhang, T., Chapelle, O., Chen, K., Sun, G.: A general boosting method and its application to learning ranking functions for web search. ANIPS 20, 1697–1704 (2007)