Personalized Keyword Boosting for Venue Suggestion based on Multiple LBSNs

Mohammad Aliannejadi[1], Dimitrios Rafailidis[2], and Fabio Crestani[1]

[1] Faculty of Informatics, Università della Svizzera Italiana, Lugano, Switzerland
mohammad.alian.nejadi@usi.ch, fabio.crestani@usi.ch
[2] Aristotle University of Thessaloniki, Thessaloniki, Greece
draf@csd.auth.gr

Abstract. Personalized venue suggestion plays a crucial role in satisfying the users needs on location-based social networks (LBSNs). In this study, we present a probabilistic generative model to map user tags to venue taste keywords. We study four approaches to address the data sparsity problem with the aid of such mapping: one model to boost venue taste keywords and three alternative models to predict user tags. Furthermore, we calculate different scores from multiple LBSNs and show how to incorporate new information from the mapping into a venue suggestion approach. The computed scores are then integrated adopting learning to rank techniques. The experimental results on two TREC collections demonstrate that our approach beats state-of-the-art strategies.

Keywords: venue suggestion, user tags, location-based social networks

1 Introduction

With the availability of location-based social networks (LBSNs), such as Yelp, TripAdvisor and Foursquare, users can share check-in data using their mobile devices. LBSNs collect valuable information about users' mobility records with check-in data including user feedback, such as ratings, tags and reviews. Being able to suggest personalized venues to a user plays a key role in satisfying the user needs on LBSNs, for example when exploring a new venue or visiting a city [4].

There is a number of different LBSNs that are widely used. However, a single LBSN does not have a comprehensive coverage over all venues and all types of information. For instance, Booking.com mainly focuses on hotels. Combining multimodal information e.g., ratings, tags, reviews of previously visited venues from different LBSNs improves the accuracy of venue suggestion [1]. For instance, the key idea of our best performing work in the TREC Contextual Suggestion Track 2015 [1] is to exploit multimodal information from multiple LBSNs, and combine them linearly to model the user preferences on venues, thus significantly beating the competitors that exploit information from a single LBSN.

A main challenge for venue suggestion is how to model the user profile, built based on the user feedback on previously visited venues. For example, users may

annotate a venue with predefined tags; even though the tag assignments are very sparse, tags contain valuable information which is worth exploiting when building user profiles. Relevant studies propose to model user profiles and generate recommendations based on the similarity between the user preferences and the venues' descriptions and categories [16]. Other studies leverage the opinions of users about a venue based on online reviews [2]. Some LBSNs, such as Foursquare, extract keywords from users' reviews. We refer to them as *venue taste keywords*. Another challenge for venue suggestion is how to calculate the correlation between user tags and information about a venue such as taste keywords. The correlated information could further be used to model the personalized user behavior for tagging venues and to solve the sparsity problem of user tags that often occurs in LBSNs.

In the effort to face these challenges, in this paper our main contribution can be summarized as follows:

1. We present a probabilistic generative approach to find the mapping between user tags and venue taste keywords, thus modeling more accurately the personalized opinion of users about venues.
2. We address the sparsity problem of user tags by performing personalized boosting of taste keywords of visited venues, so that our model is capable of predicting user tags for unexplored venues.
3. We examine several alternative machine learning models to perform tag prediction and evaluate the impact of our boosting approach on the venue suggestion task, comparing it with the alternative models.
4. We evaluate several learning to rank techniques to incorporate boosting and tag prediction into our venue suggestion model using information from multiple LBSNs.

The experiments on two benchmark datasets show that our proposed approach outperforms state-of-the-art strategies.

2 Related Work

Rikitianskii et al. [16] proposed a content-based approach to apply Part of Speech (POS) tagging to venues' descriptions, to get the most informative terms for a venue, which are then used to create positive and negative profiles when suggesting venues. Several rating-based collaborative filtering approaches have been proposed in the literature, which are based on finding common features among users' preferences and recommending venues to people with similar interests. These models are usually based on matrix factorization, exploiting check-in data for recommending venues, such as the studies reported in [5, 9]. Factorization machines generalize matrix factorization techniques to leverage not only user feedback but also other types of information, such as contextual information in LBSNs [11]. In addition, some studies follow a review-based strategy, building enhanced user profiles based on their reviews. When a user writes a review about

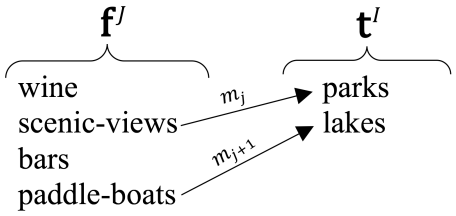


Fig. 1. An example of mapping of $J = 4$ venue taste keywords to $I = 2$ user tags.

a venue there is a wealth of information which reveals the reasons why that particular user is interested in a venue or not. For example, Chen et al. [4] argued that reviews are helpful to deal with the sparsity problem in LBSNs.

There are also many studies that propose to annotate venues with taste keywords. For instance, He et al. [10] presented a latent-class probabilistic generative model to annotate venues with taste keywords. Ye et al. [17] trained a binary classifier for each venue taste keyword with a set of extracted features so that the trained classifiers can predict the taste keywords for a new venue. There are also other studies which use various types of information such as images and audio to annotate venues, such as the study in [6]. However, none of these studies exploit information from multiple LBSNs.

3 Proposed Approach

3.1 Personalized Keyword Boosting

Personalized keyword-tag mapping. We propose a probabilistic framework to map user tags to venue taste keywords in a personalized manner. The goal of the mapping is to find the correlation between venue taste keywords and user tags. The fundamental assumption is that a user opts to assign a particular tag to a venue following a pattern that is related to the venue's content. We assume that a user chooses a particular tag for a particular venue based on its type and/or characteristics. For example, if a user tags a venue with *healthy-food* because the venue is a vegetarian restaurant, then we expect that the same user will tag other vegetarian restaurants with *healthy-food*. To model each user's personalized tag mapping, we propose a probabilistic generative model for mapping user tags and venue taste keywords. An example of such mapping is depicted in Figure 1, with a sequence of two user tags and a set of four taste keywords for a venue. Our goal is to identify for each user the most probable mapping of venue taste keywords to user tags.

Given a sequence without repetition of taste keywords $\mathbf{f}^J = \langle f_1 \dots f_j \dots f_J \rangle$, we have to compute the sequence of user tags $\mathbf{t}^I = \langle t_1 \dots t_i \dots t_I \rangle$ for each individual. Notice that $\mathbf{t}^I$ denotes a sequence named $\mathbf{t}$ whose length is I. Therefore, $\mathbf{t}^i$ would be of length i ($\mathbf{t}^i = \langle t_1 \dots t_i \rangle$), while t_i denotes the i-th element of a sequence ($\mathbf{t}^I$). We aim at finding the sequence of tags, which maximizes $Pr(\mathbf{t}^I | \mathbf{f}^J)$:

$$\hat{\mathbf{t}}^I = \underset{\mathbf{t}^I}{\text{argmax}}\{Pr(\mathbf{t}^I|\mathbf{f}^J)\} = \underset{\mathbf{t}^I}{\text{argmax}}\{Pr(\mathbf{f}^J|\mathbf{t}^I)Pr(\mathbf{t}^I)\}, \tag{1}$$

where $Pr(\mathbf{t}^I)$ is the user tag model which assigns a probability to a given sequence of user tags. We consider user tags as independent of each other. Therefore, we rewrite the user tags model as follows:

$$Pr(\mathbf{t}^I) = p(I)\prod_{i=1}^{I} p(t_i|\mathbf{t}^{i-1}, I) = p(I)\prod_{i=1}^{I} p(t_i|I), \tag{2}$$

where $\mathbf{t}^{i-1} = \langle t_1 \dots t_{i-1}\rangle$. We rewrite the probability $Pr(\mathbf{f}^J|\mathbf{t}^I)$ in Equation 1 by marginalizing it over the latent variable which *maps* taste keywords to user tags: $\mathbf{m}^J = \langle m_1 \dots m_j \dots m_J\rangle$, with $m_j \in \{1, \dots, I\}$:

$$Pr(\mathbf{f}^J|\mathbf{t}^I) = \sum_{\mathbf{m}^J} Pr(\mathbf{f}^J, \mathbf{m}^J|\mathbf{t}^I), \tag{3}$$

where

$$\begin{aligned} Pr(\mathbf{f}^J, \mathbf{m}^J|\mathbf{t}^I) &= p(\mathbf{m}^J|\mathbf{t}^I, I, J)p(\mathbf{f}^J|\mathbf{m}^J, \mathbf{t}^I, I, J) \\ &= p(J|\mathbf{t}^I)\prod_{j=1}^{J}[p(m_j|\mathbf{m}^{j-1}, J, \mathbf{t}^I, I)p(f_j|\mathbf{f}^{j-1}, \mathbf{m}^J, J, \mathbf{t}^I, I)], \end{aligned} \tag{4}$$

where $\mathbf{m}^{i-1} = \langle m_1 \dots m_{i-1}\rangle$. In our model, we consider a zero-order dependence for both mappings m_j and taste keywords f_j, while $p(J|\mathbf{t}^I)$ depends only on J. Notice that m_j only depends on the length of the user tag sequence I and f_j depends only on its corresponding mapped user tag t_{m_j}. Therefore, our task is simplified to the estimation of the following probabilities:

$$Pr(\mathbf{f}^J|\mathbf{t}^I) = p(J)\sum_{\mathbf{m}^J}\prod_{j=1}^{J} p(m_j|I)p(f_j|t_{m_j}). \tag{5}$$

Parameter estimation based on EM. To solve the parameter estimation problem of Equation 5, we follow the Maximum Likelihood (ML) criterion subject to the constraint $\sum_f p(f|t) = 1$, for each user tag t. To transform the constrained optimization problem to an unconstrained one, we use Lagrange multipliers. However, given that we have unobservable variables in our model (Equation 3), there is no closed-form solution. According to the Expectation-Maximization (EM) algorithm, we adopt an iterative algorithm to estimate the parameters of our model. In the first step the model parameters are randomly initialized, while in the second step the initial model is used to optimize the model parameters. This process is repeated until the algorithm convergences.

Boosting taste keywords. Let $\mathbf{f}^I = \langle f_1 \dots f_I \rangle$ be the set of taste keywords of a venue and $\hat{\mathbf{f}} \in \mathbf{f}^I$ be the set of venue taste keywords which are mapped to user tags. According to our probabilistic approach, we consider that $\hat{\mathbf{f}}$ correlates to the user's interest more than the other venue taste keywords. Hence, we boost $\hat{\mathbf{f}}$ to model the user's interests. This helps us to address the data sparsity problem (see Section 1). The personalized boosted venue taste keywords are used for venue suggestion (see Section 3.2).

Predicting user tags - alternative models. We also examine three alternative models for predicting user tags, where we use the statistical mapping ($\mathbf{m}$) between user tags and venue taste keywords to train a model and predict the user tags for a new venue. Given a list of taste keywords of a new venue, each alternative model predicts the most likely list of user tags. We examine the following models:

M1. In this model, we follow the ML criterion for Equation 1 to find $\hat{\mathbf{t}}^I$ as described in Section 3.1.

M2. Assume that we have N sample pairs of user tags and venue taste keywords for each user: $S = \{(\mathbf{f}_{(1)}, \mathbf{t}_{(1)}), \dots, (\mathbf{f}_{(n)}, \mathbf{t}_{(n)}), \dots, (\mathbf{f}_{(N)}, \mathbf{t}_{(N)})\}$. We calculate N corresponding mappings: $M = \{\mathbf{m}_{(1)}, \dots, \mathbf{m}_{(n)}, \dots, \mathbf{m}_{(N)}\}$. M is then used to train a tagging model which is used to predict user tags for a new venue. In this model, we adopt Conditional Random Fields (CRF) [13] to predict user tags.

M3. This model is similar to M2. We adopt a SVM-based tagging model [12] instead of CRF to predict user tags.

3.2 Venue Suggestion based on Multiple LBSNs

In this section, we briefly explain our method for performing venue suggestion, exploiting the scores from multiple LBSNs. We describe two sets of scores, the frequency and review-based scores, and show how to combine them to produce the final venue ranking using several learning to rank techniques.

Frequency-based scores. Frequency-based scores are based on the assumption that a user visits the venues that she likes more frequently than others and rates them positively[3]. We create positive and negative profiles based on contents of venues that a user has visited and calculate their corresponding normalized frequencies. A new venue is then compared with the user's profiles and we compute a similarity score. For simplicity, we only explain how to calculate the frequency-based score using venue categories. The method can be easily generalized to calculate the score for other types of information.

Given a user u and her history of rated venues $h_u = \{v_1, \dots, v_n\}$, each venue has a corresponding list of categories $C(v_i) = \{c_1, \dots, c_k\}$. We define the user category profile as follows:

Definition 1. *A **Positive-Category Profile** is the set of all unique categories belonging to venues that user u has previously rated positively. A **Negative-Category Profile** is defined analogously for venues that are rated negatively.*

[3] We consider reviews with rating [4, 5] as positive, 3 as neutral, and [1, 2] as negative.

We assign a user-level normalized frequency value to each category in the positive/negative category profile. The user-level normalized frequency for a positive/negative category profile is defined as follows:

Definition 2. *A **User-level Normalized Frequency** for an item (e.g., category) in a profile (e.g., positive-category profile) for user u is defined as:*

$$\mathrm{cf}_u^+(c_i) = \frac{\sum_{v_k \in h_u^+} \sum_{c_j \in C(v_k), c_j = c_i} 1}{\sum_{v_k \in h_u} \sum_{c_j \in C(v_k)} 1},$$

where h_u^+ is the set venues that users u has rated positively. A user-level normalized frequency for negative category profile, cf_u^-, is calculated analogously.

Based on Definitions 1 and 2 we create positive/negative category profiles for each user. Let u be a user and v be a candidate venue, then the category-based similarity score $S_{cat}(u, v)$ is calculated as follows:

$$S_{cat}(u, v) = \sum_{c_i \in C(v)} \mathrm{cf}_u^+(c_i) - \mathrm{cf}_u^-(c_i). \tag{6}$$

We use Equation 6 to calculate a venue taste keyword score (S_{key}). As for boosting, for each user we generate positive and negative *boosted venue taste keyword profiles* following Definition 1 considering only the venue taste keywords that are mapped to user tags. Then, we calculate the frequency-based score for boosted keywords (S_{boost}) according to Definition 2 and Equation 6.

We follow Definitions 1 and 2 to create positive and negative *user-tag profiles* for each user. The profiles contain user tags that each user has assigned to previously visited venues. However, since a new venue does not come with user-assigned tags, we predict user tags utilizing the alternative models of ML, CRF or SVM and consider them as user tags for the venue. We calculate a frequency-based score for the user tags predicted by ML, CRF or SVM similar to Equation 6 and refer to them as S_{lm}, S_{crf} and S_{svm}, respectively.

Review-based score. To better understand the reasons a user gives a positive/negative rating to a venue, we need to dig into the reviews associated with the venue and examine what other users say about that same venue. We assume that if a user rated a venue positively, then she shares the same opinions with all other users who also gave a positive rating to that place. The same assumption also holds for negative ratings.

We adopt a binary classifier per user to learn why she likes/dislikes venues and to assign a score for a new venue[4]. The binary classifier is trained using the reviews from the venues a particular user has visited before. We use the positive training samples which are extracted from the positive reviews of liked example suggestions, and negative samples which are from the negative reviews of disliked example suggestions, analogously. Since the users' reviews may contain a lot of

[4] An alternative to binary classification would be a regression model, but in this case it is inappropriate due to the data sparsity, that degrades the accuracy of venue suggestion.

noise and off-topic terms, we calculate a TF-IDF score and use it as the feature vector for training the classifier. As classifier we use Support Vector Machine (SVM) [7] and consider the value of the SVM's decision function as the score (S_{rev}) since it approximates how relevant a venue is to a user profile.

Personalized venue ranking. Summarizing, our proposed model consists of the following scores S_{cat}, S_{key}, S_{rev}, and S_{boost} and is denoted as *PK-Boosting* (**P**ersonalized venue taste **K**eyword Boosting). After calculating the scores (Section 3.1) we combine them to produce a final ranking. We adopt several learning to rank[5] techniques for this purpose as they have proven to be effective before [14]. In particular, we examine the following learning to rank techniques: MART, RankNet, RankBoost, AdaRank, CoordinateAscent, LambdaMART, ListNet, and RandomForest. Regarding the three alternative models, we replace S_{boost} with the three other scores for each model as follows: S_{lm} for *UT-ML* (**U**ser **T**ag prediction using ML), S_{crf} for *UT-CRF*, and S_{svm} for *UT-SVM*.

4 Experimental Evaluation

4.1 Setup

Datasets. We evaluate our approach on two benchmark datasets, published by the Text REtrieval Conference (TREC). The datasets are those used in the *Batch Experiments/Phase 2* of the TREC Contextual Suggestion Track 2015 [8] and 2016[6]. We denote them as *TREC2015* and *TREC2016*, respectively. The task was to rerank a list of candidate venues in a new city for a user given her history of venue preferences in other cities. For both datasets each user has visited from 30 to 60 venues in one or two cities. Each user may have tagged venues to explain why she likes a venue. We crawled Yelp and Foursquare to get more information about venues, such as reviews and taste keywords. More in details, for each venue in the dataset, we created a query from the venue's name and location to find the corresponding Yelp and Foursquare profiles. To avoid noises in the dataset, we further verified the title and location of each result. The crawled data from both LBSNs have a big overlap. Yelp is crawled mainly for reviews, whereas Foursquare mainly for venue taste keywords. More details can be found in Table 1.

Evaluation protocol. To perform a fair comparison, we use the official evaluation protocol and metrics of TREC for this task which are P@5 (precision at 5), nDCG@5 (Normalized Discounted Cumulative Gain at 5), and MRR (Mean Reciprocal Rank). As P@5 is considered the main metric for TREC2015, we also consider it as the main evaluation metric for our work. We conduct a 5-fold cross validation on the training data to tune our model.

Compared methods. We consider our previous work [1] which is the best performing model of TREC 2015 as our baseline and denote it as *Baseline*.

[5] We use the implementation of learning to rank named RankLib:: https://sourceforge.net/p/lemur/wiki/RankLib/

[6] https://sites.google.com/site/treccontext/

Table 1. Statistical details of datasets

	TREC2015	**TREC2016**
Number of users	211	442
Number of venues	8,794	18,808
Number of venues crawled from Yelp	6,290	13,604
Number of venues crawled from Foursquare	5,534	13,704
Average reviews per venue	117.34	66.82
Average categories per venue	1.63	1.57
Average taste keywords per venue	8.73	7.89
Average user tags per user	1.46	3.61
Number of distinct user tags	186	150

Baseline extracts information from different LBSNs and uses them to calculate two sets of scores based on: 1) user reviews and 2) venue content. The scores are then combined using linear interpolation. We choose this baseline for two reasons, firstly because it is the best performing run of TREC 2015, and secondly, because it also utilizes scores calculated from different LBSNs. We also compare our approach with the 3 alternative models of UT-ML, UT-CRF and UT-SVM, based on which the user tags can be predicted for a new venue (see Section 3.1).

4.2 Results

Performance evaluation against compared methods. Tables 2 and 3 demonstrate the performance of our approach against competitors for the TREC2015 and TREC2016 datasets, respectively. For each model we adopt the best performing learning to rank technique (see Tables 4 and 5), where the best learning to rank technique for PK-Boosting is ListNet [3]. Tables 2 and 3 show that PK-Boosting outperforms the competitors with respect to the three evaluation metrics. This shows that the proposed approach for boosting venue taste keywords improves the performance of venue suggestion. This happens because PK-Boosting solves the data sparsity problem, while at the same time it aids the ranking technique by capturing user preferences more accurately. The models UT-ML, UT-CRF and UT-SVM introduce a prediction error, when predicting user tags for a new venue, which is then propagated to venue ranking and subsequently degrades the models' performances.

Impact of number of visited venues. Figure 2 reports P@5 of all models on TREC2015 and TREC2016. In this set of experiments, we vary the number of venues to map the taste keywords to the user tags. We calculate the scores of Section 3 with different number of venues and train the ranking model. Figure 2 shows that PK-Boosting achieves the highest accuracy, when compared with other models for all different number of venues. This result indicates that PK-Boosting is less prone to noise when the training set is smaller, whereas the prediction models ML and SVM are not very well trained using such a small training set. In fact, their performance is worse with smaller training sets.

Table 2. Performance evaluation on TREC2015. Bold values denote the best scores, significant for $p < 0.05$ in paired t-test. Δ values (%) express the relative improvement, compared to the Baseline method. For each model we report the scores using the best learning to rank technique (Table 4).

	P@5	Δ(%)	nDCG@5	Δ(%)	MRR	Δ(%)
Baseline	.5858±.0073	-	.6055±.0061	-	.7404±.0055	-
UT-ML	.6114±.0048	4.37	.6241±.0029	3.07	.7380±.0023	−0.32
UT-CRF	.6161±.0041	5.17	.6212±.0043	2.59	.7302±.0016	−1.38
UT-SVM	.6152±.0058	5.02	.6256±.0033	3.31	.7419±.0014	0.20
PK-Boosting	**.6190±.0044**	**5.67**	**.6312±.0031**	**4.24**	**.7610±.0015**	**2.78**

Table 3. Performance evaluation on TREC2016. For each model we report the scores using the best learning to rank technique (Table 5).

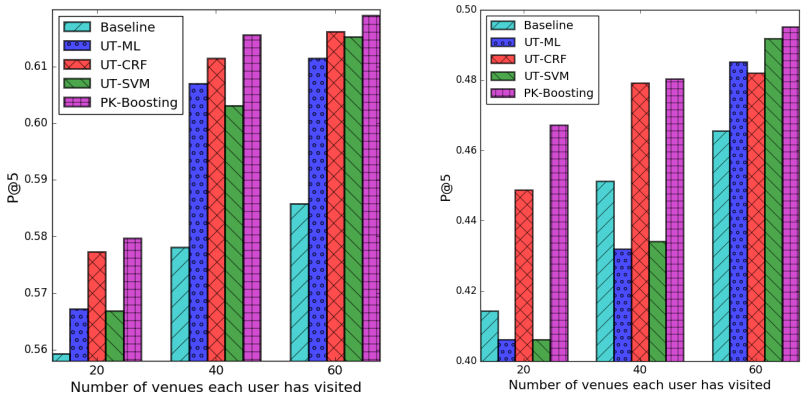
	P@5	Δ(%)	nDCG@5	Δ(%)	MRR	Δ(%)
Baseline	.4656±.0064	-	.3055±.0059	-	.5975±.0050	-
UT-ML	.4852±.0036	4.21	.3239±.0046	6.02	.5824±.0010	−2.23
UT-CRF	.4820±.0056	3.52	.3153±.0038	3.21	.6214±.0013	4.31
UT-SVM	.4918±.0038	5.62	.3259±.0044	6.68	.6338±.0018	6.40
PK-Boosting	**.4951±.0046**	**6.36**	**.3259±.0032**	**6.68**	**.6480±.0015**	**8.78**

Table 4. Effect on P@5 for different learning to rank techniques in TREC2015. Bold values denote the best learning to rank technique per model.

	UT-ML	UT-CRF	UT-SVM	PK-Boosting
MART	.5829±.0026	.5915±.0030	.5886±.0028	.5943±.0024
RankNet	**.6114±.0048**	.6104±.0039	.6085±.0044	.6072±.0027
RankBoost	.5934±.0029	.6019±.0036	.6019±.0039	.6038±.0031
AdaRank	.6028±.0054	.6009±.0062	.6038±.0048	.5782±.0067
CoordinateAscent	.5924±.0044	.5858±.0048	.5848±.0036	.5896±.0063
LambdaMART	.5962±.0018	.5991±.0027	.5981±.0017	.6066±.0034
ListNet	.5991±.0036	**.6161±.0041**	**.6152±.0058**	**.6190±.0044**
RandomForests	.5736±.0043	.5877±.0062	.5810±.0061	.5870±.0043

Table 5. Effect on P@5 for different learning to rank techniques in TREC2016.

	UT-ML	UT-CRF	UT-SVM	PK-Boosting
MART	.4525±.0027	.3788±.0030	.3869±.0025	.4131±.0029
RankNet	.4820±.0042	.3672±.0034	**.4918±.0038**	.4754±.0039
RankBoost	.4000±.0025	.3574±.0020	.3574±.0027	.3574±.0022
AdaRank	.4557±.0064	.4557±.0050	.4557±.0062	.4557±.0059
CoordinateAscent	.4656±.0039	.4721±.0053	.4689±.0048	.4787±.0053
LambdaMART	**.4852±.0036**	.4557±.0018	.4492±.0028	.4820±.0025
ListNet	.4754±.0050	**.4820±.0056**	.4852±.0038	**.4951±.0046**
RandomForests	.4164±.0060	.4033±.0051	.4197±.0057	.3924±.0044



(a) TREC2015 (b) TREC2016

Fig. 2. Effect on P@5 by varying the number of venues that each user has visited for (a) TREC2015 and (b) TREC2016.

Using information from multiple LBSNs. Tables 6 and 7 evaluate the performance of the examined models after removing information from the different LBSNs. In this set of experiments, we report the relative drop in performance of different models when using information from the two different LBSNs. As we can see in almost all cases we observe a drop in the performance when a source of information is removed from the model. The average drop for TREC2015 is -4.96% and for TREC2016 is -10.59% which confirms the effectiveness of exploiting information from different LBSNs. This indicates that using multimodal information from different LBSNs is a key to improve venue suggestion. For all different runs, the winning method is the proposed PK-Boosting approach, that uses a combination of information from both Foursquare and Yelp.

5 Conclusion

In this study, we presented a probabilistic generative model to map user tags to venue taste keywords. The resulted mapping allows to exploit several techniques to address the data sparsity problem for venue suggestion. We studied two directions: 1) the proposed PK-boosting model to boost venue taste keywords and 2) three alternative models to predict user tags for new venues. In addition, we explained how to incorporate the new information into a venue suggestion approach, calculating different scores from information from multiple LBSNs. Following learning to rank strategies, the final venue suggestion ranking is performed based on the computed scores. The experimental results on two benchmark datasets demonstrate that our approach beats state-of-the-art strategies. This confirms that the proposed approach, PK-Boosting, solves the data sparsity problem and captures user preferences more accurately. As future

Table 6. Performance evaluation after removing information provided by Foursquare (F) and Yelp (Y) in the TREC2015 dataset. Δ values (%) express the relative drop, compared to the performance that each model has when using information from the two different LBSNs. (Average drop= −4.96%)

	F	Y	P@5	Δ(%)	nDCG@5	Δ(%)	MRR	Δ(%)
Baseline	✓	✓	.5858±.0073	-	.6055±.0061	-	.7404±.0055	-
	✓	✗	.5649±.0057	−3.57	.5860±.0062	−3.22	.7263±.0060	−1.90
	✗	✓	.5697±.0068	−2.75	.5917±.0056	−2.28	.7341±.0051	−0.85
UT-ML	✓	✓	.6114±.0048	-	.6241±.0029	-	.7380±.0023	-
	✓	✗	.5213±.0029	-14.73	.5220±.0043	-16.35	.6401±.0021	-13.26
	✗	✓	.5621±.0044	-8.06	.5653±.0035	-9.42	.6752±.0017	-8.51
UT-CRF	✓	✓	.6161±.0041	-	.6212±.0043	-	.7302±.0016	-
	✓	✗	.5621±.0023	-8.76	.5826±.0017	-6.21	.7226±.0027	-1.04
	✗	✓	.5991±.0029	-2.75	.6138±.0037	-1.19	.7388±.0029	1.17
UT-SVM	✓	✓	.6152±.0058	-	.6256±.0033	-	.7419±.0014	-
	✓	✗	.5640±.0040	-8.32	.5858±.0035	-6.36	.7277±.0039	-1.91
	✗	✓	.6047±.0037	-1.71	.6173±.0025	-1.33	.7413±.0026	-0.08
PK-Boosting	✓	✓	**.6190±.0044**	-	**.6312±.0031**	-	**.7610±.0015**	-
	✓	✗	.5630±.0039	-9.05	.5902±.0045	-6.50	.7458±.0026	-1.99
	✗	✓	.5934±.0013	-4.13	.6142±.0014	-2.69	.7518±.0013	-1.21

Table 7. Performance evaluation after removing information provided by Foursquare (F) and Yelp (Y) in the TREC2016 dataset. (Average drop= −10.59%)

	F	Y	P@5	Δ(%)	nDCG@5	Δ(%)	MRR	Δ(%)
Baseline	✓	✓	.4656±.0064	-	.3055±.0059	-	.5975±.0050	-
	✓	✗	.3967±.0053	−14.8	.2572±.0061	−15.81	.5916±.0052	−0.99
	✗	✓	.4525±.0051	−2.81	.2921±.0058	−4.39	.5736±.0060	−4.00
UT-ML	✓	✓	.4852±.0036	-	.3239±.0046	-	.5824±.0010	-
	✓	✗	.4557±.0017	-6.08	.3078±.0026	-4.97	.6458±.0037	10.89
	✗	✓	.4525±.0037	-6.74	.2971±.0033	-8.27	.6107±.0017	4.86
UT-CRF	✓	✓	.4820±.0056	-	.3153±.0038	-	.6214±.0013	-
	✓	✗	.3475±.0045	-27.91	.2213±.0030	-29.81	.4869±.0024	-21.65
	✗	✓	.4689±.0018	-2.72	.3162±.0051	0.29	.6254±.0021	0.64
UT-SVM	✓	✓	.4918±.0038	-	.3259±.0044	-	.6338±.0018	-
	✓	✗	.3508±.0042	-28.67	.2350±.0035	-27.89	.5358±.0036	-15.46
	✗	✓	.3836±.0043	-22.01	.2559±.0021	-21.48	.5315±.0038	-16.14
PK-Boosting	✓	✓	**.4951±.0046**	-	**.3259±.0032**	-	**.6480±.0015**	-
	✓	✗	.4459±.0044	-9.94	.3000±.0016	-7.95	.6025±.0017	-7.02
	✗	✓	.4557±.0020	-7.96	.2979±.0035	-8.59	.5960±.0019	-8.02

work, we plan to extend our model to capture the time dimension and perform time-aware venue suggestion, an important issue in LBSNs [15].

Acknowledgements. This work was partially supported by the Swiss National Science Foundation (SNSF) under the project "Relevance Criteria Combination for Mobile IR (RelMobIR)".

References

1. Aliannejadi, M., Bahrainian, S.A., Giachanou, A., Crestani, F.: University of Lugano at TREC 2015: Contextual Suggestion and Temporal Summarization tracks. In: Voorhees, E.M., Ellis, A. (eds.) TREC 2015. NIST (2015)
2. Aliannejadi, M., Mele, I., Crestani, F.: User model enrichment for venue recommendation. In: Ma, S., Wen, J., Liu, Y., Dou, Z., Zhang, M., Chang, Y., Zhao, X. (eds.) AIRS 2016. LNCS, vol. 9994, pp. 212–223. Springer (2016)
3. Cao, Z., Qin, T., Liu, T.Y., Tsai, M.F., Li, H.: Learning to rank: From pairwise approach to listwise approach. In: Ghahramani, Z. (ed.) ICML 2007. vol. 227, pp. 129–136. ACM (2007)
4. Chen, L., Chen, G., Wang, F.: Recommender systems based on user reviews: the state of the art. User Modeling and User-Adapted Interaction 25(2), 99–154 (2015)
5. Cheng, C., Yang, H., King, I., Lyu, M.R.: Fused matrix factorization with geographical and social influence in location-based social networks. In: Hoffmann, J., Selman, B. (eds.) AAAI 2012. pp. 17–23. AAAI Press (2012)
6. Chon, Y., Kim, Y., Cha, H.: Autonomous place naming system using opportunistic crowdsensing and knowledge from crowdsourcing. In: Abdelzaher, T.F., Römer, K., Rajkumar, R. (eds.) IPSN 2013. pp. 19–30. ACM (2013)
7. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20(3), 273–297 (1995)
8. Dean-Hall, A., Clarke, C.L.A., Kamps, J., Kiseleva, J., Voorhees, E.M.: Overview of the TREC 2015 Contextual Suggestion Track. In: Voorhees, E.M., Ellis, A. (eds.) TREC 2015. NIST (2015)
9. Griesner, J., Abdessalem, T., Naacke, H.: POI recommendation: Towards fused matrix factorization with geographical and temporal influences. In: Werthner, H., Zanker, M., Golbeck, J., Semeraro, G. (eds.) RecSys 2015. pp. 301–304. ACM (2015)
10. He, T., Yin, H., Chen, Z., Zhou, X., Sadiq, S., Luo, B.: A spatial-temporal topic model for the semantic annotation of POIs in LBSNs. ACM TIST 8(1), 12:1–12:24 (2016)
11. Koren, Y., Bell, R.M., Volinsky, C.: Matrix factorization techniques for recommender systems. IEEE Computer 42(8), 30–37 (2009)
12. Kudo, T., Matsumoto, Y.: Fast methods for kernel-based text analysis. In: Hinrichs, E.W., Roth, D. (eds.) ACL 2003. pp. 24–31. ACL (2003)
13. Lafferty, J.D., McCallum, A., Pereira, F.C.N.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Brodley, C.E., Danyluk, A.P. (eds.) ICML 2001. pp. 282–289. Morgan Kaufmann (2001)
14. Liu, T.: Learning to rank for information retrieval. Foundations and Trends in Information Retrieval 3(3), 225–331 (2009)
15. Liu, Y., Liu, C., Liu, B., Qu, M., Xiong, H.: Unified point-of-interest recommendation with temporal interval assessment. In: Krishnapuram, B., Shah, M., Smola, A.J., Aggarwal, C., Shen, D., Rastogi, R. (eds.) SIGKDD 2016. pp. 1015–1024. ACM (2016)
16. Rikitianskii, A., Harvey, M., Crestani, F.: A personalised recommendation system for context-aware suggestions. In: de Rijke, M., Kenter, T., de Vries, A.P., Zhai, C., de Jong, F., Radinsky, K., Hofmann, K. (eds.) ECIR 2014. LNCS, vol. 8416, pp. 63–74. Springer (2014)
17. Ye, M., Shou, D., Lee, W., Yin, P., Janowicz, K.: On the semantic annotation of places in location-based social networks. In: Apté, C., Ghosh, J., Smyth, P. (eds.) SIGKDD 2011. pp. 520–528. ACM (2011)