A Factored Relevance Model for Contextual Point-of-Interest Recommendation

Anirban Chakraborty
ADAPT Centre, School of Computer Science & Statistics
Trinity College Dublin, Ireland
anirban.chakraborty@adaptcentre.ie

Debasis Ganguly
IBM Research
Dublin, Ireland
debasis.ganguly1@ie.ibm.com

Annalina Caputo
ADAPT Centre, School of Computer Science & Statistics
Trinity College Dublin, Ireland
annalina.caputo@adaptcentre.ie

Séamus Lawless
ADAPT Centre, School of Computer Science & Statistics
Trinity College Dublin, Ireland
seamus.lawless@scss.tcd.ie

ABSTRACT

The challenge of providing personalized and contextually appropriate recommendations to a user is faced in a range of use-cases, e.g., recommendations for movies, places to visit, articles to read etc. In this paper, we focus on one such application, namely that of suggesting 'points of interest' (POIs) to a user given her current location, by leveraging relevant information from her past preferences. An automated contextual recommendation algorithm is likely to work well if it can extract information from the preference history of a user (*exploitation*) and effectively combine it with information from the user's current context (*exploration*) to predict an item's 'usefulness' in the new context. To balance this trade-off between exploration and exploitation, we propose a generic unsupervised framework involving a factored relevance model (FRLM), comprising two distinct components, one corresponding to the historical information from past contexts, and the other pertaining to the information from the local context. Our experiments are conducted on the TREC contextual suggestion (TREC-CS) 2016 dataset. The results of our experiments demonstrate the effectiveness of our proposed approach in comparison to a number of standard IR and recommender-based baselines.

CCS CONCEPTS

• **Information systems** → **Personalization**; *Information retrieval diversity*; *Recommender systems*; **Probabilistic retrieval models**.

KEYWORDS

Relevance Model, Contextual Recommendation, User Model

ACM Reference Format:
Anirban Chakraborty, Debasis Ganguly, Annalina Caputo, and Séamus Lawless. 2019. A Factored Relevance Model for Contextual Point-of-Interest Recommendation. In *The 2019 ACM SIGIR International Conference on the Theory of Information Retrieval (ICTIR '19), October 2–5, 2019, Santa Clara, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3341981.3344230

ICTIR '19, October 2–5, 2019, Santa Clara, CA, USA

ACM ISBN 978-1-4503-6881-0/19/10…$15.00
https://doi.org/10.1145/3341981.3344230

1 INTRODUCTION

Owing to the enormous volume of online data, contextual recommendation is growing in importance since there is an increasing need for contextually appropriate recommendations in a variety of domains. Among different use-cases of contextual recommendation, we focus on the problem of context-aware 'point of interest' (POI) recommendation.

The POI recommendation is of particular interest to a user travelling to a new part of the world, which she has not visited before. An effective contextual recommender system would then seek to *match* her previous preferences in other contexts (e.g. travel in other locations) with the highly rated POIs of the current context to recommend potentially relevant POIs. Consider for example the situation when a user with her group of friends visits Dublin in summer. Depending on the user's past preferences in other locations (e.g. the user generally loves to hangout with friends and she is an avid stout lover and that she loves hiking etc.), a context-aware system should try to *match* the information from her past preferences with the POI descriptors in her current location (Dublin). For example, recommending the popular tourist destinations and activities in Dublin such as the cliff-walk in Howth, the Guinness Storehouse, Temple Bar etc. is likely to be appropriate for the example user.

It is motivating to approach contextual recommendation from an unsupervised perspective similar to traditional IR, where POIs can be considered analogous to documents, and the preference history is analogous to a query. The main advantage of unsupervised approaches is that instead of relying on training a model with labelled data, they rather seek to utilize the inherent relationships between latent features of the data itself to make predictions. A major challenge in approaching contextual recommendation from an IR perspective is that there is no presence of an explicit, user-entered query. In this case, the query needs to be automatically formulated from the information in the profile of the user, such as pieces of text describing her preferences and non-preferences. The query then needs to be effectively *matched* with the descriptions of the POIs (documents) in the current context.

A number of studies have investigated the problem of contextual recommendation from the point of view of matching the content between the POI (document) representation and the user profile

(query) representation. Among these, the studies in [11, 18] combined POI-category and bag-of-words similarities between POIs and user profiles. Generally speaking, as POI-categories, these approaches made use of external information from location-based social networks (LBSNs), such as Foursquare[1] or Yelp[2], to match previous user preferences and POIs in the current location. Note that contextual recommendation systems based on this thread of work mainly rely on *exploiting* the existing preferential knowledge of users from their profiles.

On the other hand, a different thread of work [4, 8] utilizes rating-based collaborative filtering, i.e. information from other users to estimate the popularity of a POI in a local context with the hypothesis that POIs with frequent positive ratings from other users could also be relevant to the current user. In contrast to *exploitation*, this collaborative filtering based thread of work primarily relies on *exploring* the POIs using the current context.

IR or Recommender (RecSys) Systems?. We argue that it is more suitable to formulate the POI recommendation problem as a (constrained) IR problem, which is characteristically different from the scope of traditional Recommender systems (RecSys). In a traditional RecSys setting, such as a movie recommendation, the popularity of an item depends on user ratings, or other contextual features, e.g. current user location etc. In our problem setup, a POI (item) is represented as a document (bag-of-words) with a unique document ID and the city where it is located. Thus the problem essentially boils down to that of a content-matching one between POIs in other cities (preference history) and the POIs in the current city.

It may also be argued that it is potentially possible to apply a recommendation system (RecSys) based approach to address the problem. However, a careful consideration reveals that it is not possible to directly apply a matrix factorization based methodology on a user-item matrix [7] by drawing an analogy that items and users are respectively equivalent to POIs and preferences (contexts). This is mainly because of the lack of sufficient data for training standard RecSys approaches to learn the user-item associations. Specifically, in the context of our problem, there are no ratings available for the POIs in query locations (contexts).

Our Contributions. Firstly, in contrast to the existing approaches, which either mainly rely on *exploitation* of user profile information, or *exploration* of POIs (based on ratings of other users) in the current user context, we propose a relevance feedback based unsupervised method of investigating the trade-off between the two. To the best of our knowledge, no other existing research has applied a relevance feedback based framework to address this balanced matching problem between a user's previous preferences and the descriptions of items in her current context. Secondly, unlike some of the existing approaches that rely on the use of external information (e.g. in the form of Foursquare categories or tags), our method only makes use of a data collection of user profiles and current contexts for the purpose of contextual suggestion. Thirdly, as we do not use other users' ratings from external resources, our proposed framework is particularly suitable in an extreme cold-start scenario where no user ratings are available for the candidate POIs, which is a practical problem in many cases.

2 RELATED WORK

Contextual Recommendation. The Contextual Suggestion track[3] (TREC-CS) was introduced by TREC to provide a common evaluation platform for participants to deal with the problem of improving contextual suggestion. TREC-CS participants explored different approaches to solve the contextual suggestion problem. A very popular approach among researchers is to retrieve POIs from different LBSNs such as Yelp or Google Place based on geographical context and then apply some heuristics such as "museum will remain closed at night" or "night club will not be recommended in morning" to filter out the POIs that do not match the given temporal context [5]. Then the task is to retrieve appropriate POIs based on user preferences.

Most of the TREC participants framed the task as a content-based recommendation problem [11, 14, 16, 18]. A common strategy is to estimate a user profile based on the preferred POIs and then rank the candidate POIs based on their similarities to the estimated profile, assuming that a user would like POIs that are similar to those liked by the user before. There are studies that used the description of the POIs and/or the web pages of the example POIs to build user profiles, and then several similarity measures are used to rank the POIs [11, 18]. The authors of [14, 15] explored the use of category information for user profile construction and POI ranking.

There are some researches that address the contextual recommendation problem from different interesting aspects. Some researchers proposed rating-based collaborative filtering approaches that are based on detecting features that are common among users' interests and then recommending POIs to users with similar preferences. Matrix factorization is generally applied to build these models, exploiting check-in data collected from LBSNs for recommending POIs [4, 8]. However, collaborative filter based approaches often suffer from the data sparsity problem. This problem becomes worse for POI suggestion as there are a lot of POIs in a city and for a single user it may be possible to visit (and rate) only a few of them.

To address the cold-start problem for hotel recommendation, [13] developed a context-aware recommender system. They created context groups based on reviews and considered the user's preferences in trip intent (purpose of the trip) and the similarity of the user in question with other users such as nationality. They also consider the user preferences for the different hotel aspects for their system. There are works that use time as a context. [6] developed a time-aware venue suggestion system. They modeled appropriateness or popularity of POIs (venues) in the immediate future using time series. [1] applied linear interpolation and learning-to-rank to combine different scores for context-aware venue suggestion.

It is also becoming popular among researchers to use online users' reviews for contextual recommendation. The study in [3] leverages the opinions of users about a POI based on online reviews. A single LBSN may not have the information about all POIs and/or all the possible types of information about the POIs. In a recent study, [2] shows that the combined use of a user's current context

[1] https://foursquare.com
[2] https://www.yelp.com

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

and the reviews and ratings of previously rated POIs from different LBSNs improves recommendation accuracy.

In contrast to the existing approaches, which either mainly rely on *exploitation* of user profile information, or *exploration* of POIs (based on ratings of other users) in the current user context, we propose a generic framework where we investigate the trade-off between user preference history in past contexts (exploitation) and the top retrieved POIs for the user's current context (exploration) for contextual POI recommendation. We will show that the systematic infusion of user preference history and the top retrieved POIs improves the retrieval performance.

Relevance Model. To provide a necessary context to understand our proposed model, we review relevance model [12] briefly in this section. Traditional relevance model (RLM) [12] is a relevance feedback method which estimates the importance of terms for relevance feedback by making use of the co-occurrences between a set of given query terms and the ones in a set of top-ranked documents. The original RLM model as proposed in [12] does not estimate the weights of the original query terms. The original query terms are included in the estimated density of term weights by a mixture model [10]. Although the paper [10] used the notation 'RM3' to denote this mixture model of original query and feedback terms, for the sake of simplicity, we refer to it as RLM and employ it as one of the baselines in our experiments.

3 PROPOSED APPROACH

The core idea of our proposed approach for contextual suggestion (CS) is to make use of a pseudo-relevance feedback based framework to effectively balance the exploration-exploitation trade-off. To this end, we first provide a general description of the relevance model, which is a pseudo-relevance feedback method, and discuss how it could be applied in the context of our problem. We then propose a factored version of the relevance model for effectively combining the information from user profiles and current contexts.

3.1 Relevance Model for IR

The traditional relevance model (RLM) [12] is a relevance feedback method which estimates the importance of terms for relevance feedback by making use of the co-occurrences between a set of given query terms and the ones in a set of top-ranked documents. The underlying assumption in RLM is that the terms frequently co-occurring with the query terms are semantically related to the information need and hence could be useful to enrich the query with additional useful information.

Formally speaking, given a query $Q = \{q_1, \ldots, q_n\}$, RLM involves estimating a term weight distribution $P(w|Q)$ from a set of M top-ranked documents $\mathcal{M} = \{D_1, \ldots, D_M\}$, i.e.,

$$P(w|Q) = \sum_{D \in \mathcal{M}} P(w|D) \prod_{q \in Q} P(q|D). \tag{1}$$

Intuitively, it can be seen that higher $P(w|Q)$ values (RLM term weights) are obtained for a term w if it occurs frequently in a top-ranked document (large $P(w|D)$ value), in conjunction with the frequent occurrence of a query term, i.e. a term $q \in Q$ such that $P(q|D)$ is also large.

3.2 IR Setup for Contextual Suggestion

In contrast to traditional IR, in contextual suggestion (CS) there is no notion of a user entered query. The objective in CS is rather to match the past user preferences with the descriptions of the current POIs (which are analogous to documents). Approaching CS from an IR point-of-view, the best analogy to a query corresponds to pieces of information from the user profile.

Generally speaking, for our model we assume that a user profile U is comprised of a set of N_U profile P_i's, where each tuple P_i is a *triple* comprised of a document (which is a part of the overall static collection $\mathcal{D}$), a set of tags from a controlled tag vocabulary $\mathcal{T}$, and a rating value normalized within $[0, 1]$ (higher being better). More formally,

$$U = \cup_{i=1}^{N_U} \{P_i : P_i = (D, T, r) \in \mathcal{D} \times \mathcal{T} \times [0, 1]\}. \tag{2}$$

The purpose of a tag $t \in \mathcal{T}$ is to describe a POI with the help of a set of single words or short phrases, concrete examples of which are 'seafood', 'beer', etc. applied to a restaurant (POI). The document representation of the POI constitutes the text description compiled from reviews of the restaurant, its home-page etc. The definition of each document is a static feature of the collection.

For the purpose of suggesting useful POIs to a user U in her current context $C(U)$ from an IR point-of-view, we define $C(U)$ as a 2-tuple (in other words, a pair) of $(c, L(c))$, where c denotes a context state (e.g. location, time etc.) and $L(c)$ denotes a set of POI descriptions (documents) satisfying the context state (e.g. POIs in and around the specified location etc.).

From a ranking perspective, one then needs to compute the similarity scores between the POI descriptions from the user profile and the list of POIs in her current context. This requires defining a similarity function between a 3-tuple of the user profile U and a document representation from the list of 'local' context instances $L(c)$

$$\phi : ((D, T, r) \in U) \times (d \in L(c)) \mapsto \mathbb{R}. \tag{3}$$

A simple content matching approach is then to apply a standard ranking model, e.g. LM or BM25, computing the similarity between each relevant context instance $d \in L(c)$ and a profile tuple from the user preferences.

$$S(d, U) = \sum_{P=(D,T,r) \in U} \phi(P, d), \quad d \in L(c) \tag{4}$$

Each POI in the current context list can then be sorted in decreasing order of the similarity values and presented to the user.

3.3 User Profile based RLM

The main challenge in matching a profile with a current context descriptor (Equation 4) is to select the contextually appropriate terms from the content and tags of the profile descriptors. Taking the whole bag-of-words representation of text, comprised of content and tag words, while aggregating the scores in Equation 4 could lead to noise in the similarity estimation.

Such a similarity estimation can be ineffective due to two reasons. First, the profile information of a user may be quite diverse with only a particular aspect of it being likely to be relevant in the current context. Second, since the profile descriptors are full-length documents, selecting parts from these documents that are topically

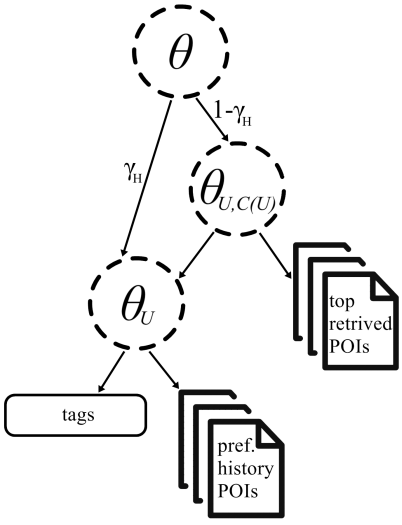


Figure 1: Factored Relevance Model (FRLM)

associated with the query terms may in fact improve similarity estimation. With this motivation, we propose to apply a relevance model (RLM) to define a weighted distribution of terms extracted from the profile information of a user and then use this estimated distribution θ_U to rank the documents (POIs) for a given context, $L(c)$.

To estimate an RLM for a user, we treat the set of tags in a POI descriptor (Equations 4) as the set of observed terms (analogous to a query in the IR setup of RLM). Let T' be the set of tags, i.e. union of all Ts from the set of tuples $(D, T, r) \in U$. The set of top retrieved documents in this case is the given set of documents in the user history, i.e. union of all Ds from the set of tuples $(D, T, r) \in U$. More formally,

$$P(w|\theta_U) = \sum_{(D,T,r)\in U} rP(w|D) \prod_{t\in T'} P(t|D), \tag{5}$$

where the estimated relevance model seeks to capture the semantic association between a user-assigned tag and a content term by co-occurrence evidence from the profile information of a user.

The use of rating scores as confidences for the co-occurrences allows the model to assign more importance to terms that co-occur frequently with the tags in positively rated POI descriptions. Although a reader may be tempted to think that the use of user assigned ratings in an RLM framework makes it supervised, we would like to emphasize that these rating scores are not used as *labels* in a supervised setting to optimize an objective function. If no ratings are available, our RLM-based feedback model simply assumes the confidence values as 1 (uniform across all POIs in each user's profile).

Note that this part of estimating the relevance model corresponds to *exploiting* the available information from the user history to build a weighted term distribution of her preferences.

3.4 Factored RLM for Contextual Relevance

To address the current context of a user, we estimate another relevance model, this time using the RLM estimated from the profile information as the observed set of terms. The set of top documents in this case pertains to the ones present in $L(c)$ (documents constrained to be satisfying the current context). More formally,

$$P(w|\theta_{U,C(U)}) = \sum_{d\in L(c)} P(w|d) \prod_{t\in\theta_U} P(t|d), \tag{6}$$

where $C(U) = (c, L(c))$. Equation 6 represents a factored relevance model because estimating $\theta_{U,C(U)}$ requires estimating θ_U (the factor model). Note that this model *explores* the POIs in the current context (in order to improve on their ranks).

As a further generalization, we propose to use a linear combination of the *exploitation* and *exploration* relevance models of Equations 5 and 6, respectively into a combined model,

$$P(w|\theta) = \gamma_H P(w|\theta_U) + (1-\gamma_H)P(w|\theta_{U,C(U)}), \tag{7}$$

where γ_H acts as the trade-off parameter to balance the two factors. We call our proposed model the factored relevance model (FRLM).

4 ALGORITHMIC DETAILS

As shown in Figure 1, our proposed methodology requires estimating a total of three relevance models. First, the user profile based relevance model, namely θ_U, is estimated by making use of the text present in a user's preference history. Next, to capture the relevance of POIs in a given context (typically a location of a POI), we estimate the factored RLM $\theta_{U,C(U)}$ by using information from both the user preference and the top retrieved POIs, treating the former as equivalent to a query and the latter as top retrieved documents within an RLM framework [12]. Finally, both relevance models θ_U and $\theta_{U,C(U)}$ are linearly combined into a single relevance model θ, the proposed generalized FRLM.

4.1 Proposed Algorithm

Initialization. As described in Algorithm 1, let $\mathcal{U}$ be the set of users and each user $U = \cup_{i=1}^{N_U}\{P_i : P_i = (D, T, r)\} \in \mathcal{U}$ as described in Equation 2. For each user U, we construct a set of all tags used by the user. In other words, a union of all Ts from the set of tuples $(D, T, r) \in U$ is stored in a set T'. Then, we collect all document representations, i.e. union of all Ds from the set of tuples $(D, T, r) \in U$ and store them in set M_U. Then, we execute an initial retrieval with the terms in set T' to select M top-ranked POIs (documents constrained to be satisfying the current context) and store them in set $M_{C(U)}$.

Estimation of User Profile based RLM. Now we estimate the user profile based relevance model θ_U. For each term $w \in M_U$, we compute the probability of sampling the term w from θ_U, denoted by $P(w|\theta_U)$, as in Equation 5 by the joint probability of observing w along with the tags T'. Please note that we take a mixture model of the estimated relevance model θ_U in conjunction with the tags likelihood model i.e. $P(w|T')$, to get the influence of tags T' in the final estimation of θ_U. Tags likelihood model $P(w|T')$ is computed using maximum likelihood estimation (MLE). Smoothing parameter μ_U (tag mixing) controls the relative weight we assign to the relevance model versus the 'tags' model.

Algorithm 1: Proposed Algorithm using FRLM

Input: $\mathcal{U}$, μ_U, $\mu_{C(U)}$, M τ, γ_H
Output: Set of ranklists
while $U = \cup_{i=1}^{N_U} \{P_i : P_i = (D, T, r)\} \in \mathcal{U}$ **do**
  // Initialization
  $T' \leftarrow$ Union of all Ts $\in U$ // 'tags'
  $M_U \leftarrow$ Union of all Ds $\in U$ // pref. history POIs
  $M_{C(U)} \leftarrow$ Top M POIs from retrieve(T') // top-ranked POIs
  // Estimate relevance model θ_U from M_U
  $i = 0$, $L_U \leftarrow null$, $L'_U \leftarrow null$
  for *each term* $w \in M_U$ **do**
    $P(w|\theta_U) = 0$
    for *each* $P_i = (D, T, r) \in U$ **do**
      $P(w|\theta_U)$ += $rP(w|D) \prod_{t \in T'} P(t|D)$
    end
    $P'(w|\theta_U) = \mu_U P(w|\theta_U) + (1 - \mu_U)P(w|T')$
    $L_U[i++] \leftarrow \{w, P'(w|\theta_U)\}$
  end
  Sort L_U in descending order of $P'(w|\theta_U)$
  $L'_U \leftarrow$ top τ terms from L_U
  // Estimate relevance model $\theta_{U,C(U)}$ from $M_{C(U)}$
  $i = 0$, $L_{C(U)} \leftarrow null$, $L'_{C(U)} \leftarrow null$
  for *each term* $w \in M_{C(U)}$ **do**
    $P(w|\theta_{U,C(U)}) = 0$
    for *each document* $d \in M_{C(U)}$ **do**
      $P(w|\theta_{U,C(U)})$ += $P(w|d) \prod_{t \in \theta_U} P(t|d)$
    end
    $P'(w|\theta_{U,C(U)}) =$
     $\mu_{C(U)} P(w|\theta_{U,C(U)}) + (1 - \mu_{C(U)})P(w|\theta_U)$
    $L_{C(U)}[i++] \leftarrow \{w, P'(w|\theta_{U,C(U)})\}$
  end
  Sort $L_{C(U)}$ in descending order of $P'(w|\theta_{U,C(U)})$
  $L'_{C(U)} \leftarrow$ top τ terms from $L_{C(U)}$
  // Estimate generalized factored relevance model θ
  $i = 0$, $L \leftarrow null$
  for *each* $w \in$ *Union of* $(L'_U, L'_{C(U)})$ **do**
    $P(w|\theta) = \gamma_H \cdot P'(w|\theta_U) + (1 - \gamma_H) \cdot P'(w|\theta_{U,C(U)})$
    $L[i++] \leftarrow \{w, P(w|\theta)\}$
  end
  // Query expansion
  $T'' \leftarrow$ all terms in L // weighted term distribution
  retrieve(T'') // ranklist for U
end

$P'(w|\theta_U)$ is the final probability of term w from the mixture model. The term w is added in an initially empty list L_U along with its probability $P'(w|\theta_U)$. When every term in M_U is considered, the list L_U is sorted in descending order of $P'(w|\theta_U)$ and top τ terms are added in another initially empty list L'_U, along with their corresponding $P'(w|\theta_U)$ values.

Estimate Factored RLM. We then estimate the factored RLM $\theta_{U,C(U)}$ from the set $M_{C(U)}$, to capture the contextual relevance in a similar fashion. As described in Algorithm 1, here $P(w|\theta_{U,C(U)})$, i.e. the probability of sampling a term w from $\theta_{U,C(U)}$, is computed by the joint probability of observing w along with the terms in the previously estimated user profile based RLM θ_U. We get a list $L'_{C(U)}$ in the same way we created the list L'_U.

Generalization of FRLM. We then linearly combine two lists L'_U and $L'_{C(U)}$ into the final list L with a smoothing parameter γ_H. Finally all terms in L, along with their corresponding probabilities are put in set T'', an expanded set of tags (terms) which is analogous to expanded query in IR. Note that T'' can be considered as a weighted query as it is a distribution of terms along with their probabilities, i.e. each term in the distribution is boosted by its probability as the weight of the term. Finally, we execute another retrieval with T'' to get the final result and present it to the user U.

5 EXPERIMENT SETUP

5.1 POI and Profile Representation

We represent each document $D \in \mathcal{D}$ as a bag-of-words of descriptive text comprising information about the venue (provided as a part of the TREC open web-crawl) and reviews crawled from LBSN, namely Foursquare. The use of a combination of the open web-crawl and content crawled from LBSN's as a static document collection conforms to the usual experimental settings of most participating systems in the TREC-CS tasks over a number of years [9].

At this point, we would like to mention that since the crawled content can vary across different participating systems (because of the dynamic nature of the LBSN content and the APIs to access it), the results obtained by various systems participating in the TREC-CS task are somewhat difficult to compare against each other. This is also one of the reasons why we experiment with a number of standard baselines instead of comparing against past results from the TREC-CS task overview papers.

We would like to mention that most of the TREC-CS participants used external data (other users' ratings, reviews, category information etc.) for their research and their systems heavily depend on LBSN data such as Foursquare, Yelp etc. Some of them also used handcrafted rules while some of them created another crowdsource data [1] on their own to tune the contextual relevance of a POI. Although crawling additional information from different resources to enrich user model or content of a POI may improve the system performance, this may lead to the reproducibility problem. Instead, this paper focuses on an unsupervised method of POI retrieval on a static collection of POI descriptions. Although it may be possible to apply a combination of post-processing approaches, such as rule-based heuristics crafted from external knowledge sources, to potentially improve the results reported by our proposed methodology further, in our experiments we do not apply any post-processing methods. This is because the objective of the experiments is to investigate the use of a purely data-driven methodology rather than relying on manually formulated rules, which typically tend to be tuned in different ways for different collections.

For our experiments, we only make use of a part of the user profile information, i.e. POIs with ratings higher than a threshold. The ratings of the TREC-CS dataset are integers within a range of -1 to 4. As per our general model profile description of Equation 2, we normalize each score within the range $[0, 1]$. In order to selectively use the positively rated profile information of a user, we then apply a threshold of $\frac{4}{5}$ to constitute the profile for FRLM estimation. In particular, for our experiments, the set U in Section 3.2 corresponds to the 3-tuples of the form (D, T, r), where $r \geq \frac{4}{5}$. The threshold value was optimized after a set of initial experiments.

Information	Value
Total no. of POIs in corpus	1,235,844
No. of cities per user preference history	1 or 2
No. of rated POIs per user preference history	30 or 60
Total no. of candidate cities	164
No. of candidate cities officially used by TREC	48
Maximum no. of POIs per city	23,939
Minimum no. of POIs per city	1,070
Average no. of POIs per city	4543.54
Total no. of user profiles	438
No. of user profiles officially used by TREC	61

Table 1: TREC-CS 2016 collection statistics.

5.2 Dataset

TREC has released a static web crawl of the collection in 2016 [9] for TREC-CS task. In TREC-CS test collection, there are 438 user requests in total, out of which 61 user profiles were used officially for Phase-1 experiments as the test collection has judgments of these 61 user requests. The statistics of the collection is shown in Table 1. In provided user requests, preference history is available for one or two cities with 30 or 60 POIs with ratings (30 POIs each city) and there are 48 candidate cities (for those 61 queries) from where the suggestion has to be made.

As baselines, we employ a number of standard IR-based approaches and recommender system approaches, as descibed in the subsequent sections.

5.3 IR Baselines

(1) **BL1 - BM25**: We use the BM25 retrieval model as the similarity function of Equation 4. We only treat the tag terms from the set of tuples (D, T, r), where $r \geq \frac{4}{5}$, as query terms for computing the similarity. As BM25 parameters, we use $(k, b) = (1.1, 0.3)$ after optimizing them with grid search (with respect to the nDCG@5 evaluation metric).
(2) **BL2 - Term Selection**: Since FRLM estimates a weighted query with varying term importance, we choose as baseline an approach that extracts a subset of terms based on BM25 scores (we use the same settings of k and b) as that in BL1. The number of terms to select was optimized to 25 (after grid search).
(3) **BL3 - BM25 with Term Selection:** Since FRLM uses a combination of both the preference history and the POIs in the current context, we employ a COMBSUM of the two ranked lists obtained with BL1 and BL2, which offers a naive way of combining the history and the current context.
(4) **BL4 - RLM**: Since our proposed method relies on a factored RLM, we choose as a baseline a traditional RLM (Equation 1), where similar to BL1, we use the tag terms from the profile with ratings higher than or equal to $\frac{4}{5}$ as query terms and then estimate RLM to rerank documents in $L(c)$. In contrast to FRLM, this model uses the exploitation step only in the query formulation and not during RLM estimation.

Parameters common to both FRLM and RLM, i.e., the number of feedback documents, M, and that of feedback terms, τ, were optimized to the values 5 and 25 using grid search.

5.4 Recommender System Baselines

We have already mentioned that due to the absence of users' ratings, it is not possible to predict a POI (analogous to an item in RecSys) directly using standard RecSys approaches such as collaborative filtering. However, a different analogy makes it possible to apply standard RecSys approaches as a pre-processing step in this problem. More precisely, one may consider that users in the RecSys terminology are analogous to words in the user reviews (preference descriptions), and that the tags used to describe a POI are analogous to items. This approach enables learning semantic associations between words in user contexts and the tags. It is thus possible to enrich the set of tags given a user preference (analogous to recommending more items for a user in the standard RecSys terminology). With this general description of setting up RecSys based approaches for our experiments, we now describe the details of each baseline.

(1) **BL5 - Most Popular K**: A strong baseline for a RecSys approach is the recommendation of the 'Most Popular' items over the set of all users, with the hope that a new user is likely to find these items helpful as well. Specifically in the context of our problem, we select the most popular K tags from the preference history across all users and use these selected tags as the query for each individual user during test time. As an example, if 'beer' is one of the globally most popular tags across the set of all users, it indicates that retrieving pubs as candidate POIs for any new user is likely not to be a bad recommendation.
After this tag-based query formulation, we use the BM25 retrieval model as the similarity function of Equation 4 with the same settings of k and b as that in BL1. K was tuned on the overall average rating of tags to achieve an optimal setting. Average rating cut-off was set to $\frac{4}{5}$.
(2) **BL6 - Profile Popular K**: In contrast to the previous global approach of finding the most popular set of tags, in this approach we separately restrict the selection of the most popular tags to each user preference history only. This method selects tags in a more personalized manner, e.g., it includes the tag 'beer' in the query if it is one of the most popular tags in the history of the current user only.
Similar to the previous baseline, we use BM25 as the similarity function of Equation 4, with the same settings of k and b as that in BL5. The BM25 parameter, K, was tuned on the user specific average rating of tags to achieve an optimal setting. Average rating cut-off was set to $\frac{4}{5}$.
(3) **BL7 - PMF**: We created a user versus tag matrix and applied standard probabilistic matrix factorization [17]. Top K higher predicted tags are used as query for a user, while we use BM25 retrieval model with the same setup as that in BL5. K has been tuned to achieve an optimum setup. It is an effective way to capture important tags associated to POIs with frequent positive ratings from other similar users, in case the current user did not have those tags in her preference history, that could be relevant.
(4) **BL8 - Bayesian Content-based Recommendation Approach**: Naive Bayes is a standard text classification technique which has been widely used as a content-based recommendation approach as it can be used as a simple but effective content matching function. We trained a Naive Bayes classifier with two classes:

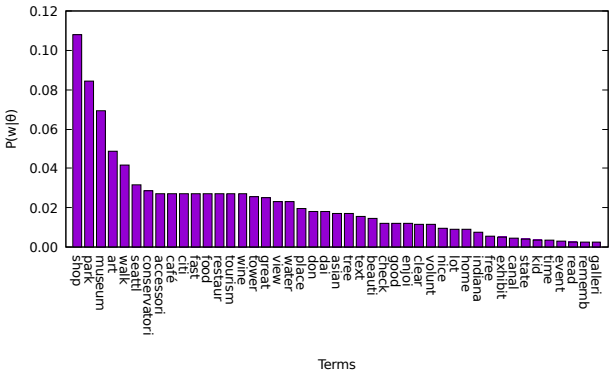


Figure 2: FRLM term distribution for user ID 763

(1) Positive: with all documents i.e. all Ds from the set of tuples (D, T, r), where $r \geq \frac{4}{5}$, and (2) Negative: with all documents i.e. all Ds from the set of tuples (D, T, r), where $r < \frac{4}{5}$. For each candidate POI we take the confidence score returned by the classifier as the score of that POI, if the POI is classified as 'Positive'. We presented the result by sorting all candidate POIs in descending order of their scores.

(5) **BL9 - Classification + Tag Matching**: Our results are not directly comparable with the reported results [9] by the TREC-CS participants, in terms of the absolute values of the evaluation metrics but we employed the proposed methodology used by the best performing TREC-CS 2016 system and their follow up work [1] into our set up with the available data. Here the proposed similarity function is a content matching function with a combination of users' review based (text classification) score and normalized frequency (tag matching) based score.

5.5 Results

Table 2 shows the results obtained by each CS approach. The table shows that FRLM outperforms all other baselines with respect to most standard evaluation metrics. On the P@5 measure, FRLM achieves identical results to BM25. Improvements in nDCG@5, nDCG and MAP are statistically significant at 95% confidence level based on the Wilcoxon signed-rank test.

The better performance of FRLM in comparison with BL2 (Term Selection) indicates that the probability distribution of weighted terms as estimated by FRLM is a more effective way to select candidate terms for query formulation. Although BL3 (BM25 + Term selection) takes both the user's preference history (term selection based exploitation) and the top retrieved POIs (BM25 based exploration) into account, the superior performance of FRLM confirms that the systematic infusion of user preference history and the top retrieved POIs is important for effective retrieval performance. Figure 2 shows FRLM term distribution for a user request (user ID 763) where $T' = \{$`art, cafés, city-walks, fast-food, museums, parks, restaurants, shopping-for-accessories, shopping-for-wine, tourism`$\}$. FRLM assigns higher weights to terms such as 'park', 'museum', which is desirable for this particular case. Indeed, this model is also successful to capture other relevant terms such as 'view', 'tree', 'canal' etc.

	Method	nDCG@5	nDCG	P@5	MAP
	IR-based				
BL1:	BM25	0.2747	0.2889	**0.3934**	0.1326
BL2:	Term Sel.	0.2484	0.3034	0.3639	0.1466
BL3:	BM25 + Term Sel.	0.2411	0.3143	0.3672	0.1530
BL4:	RLM [12]	0.2615	0.3091	0.3574	0.1437
	RecSys-based				
BL5:	Most Popular K	0.1861	0.2580	0.2787	0.1016
BL6:	Profile Popular K	0.2488	0.2811	0.3410	0.1280
BL7:	PMF [17]	0.2287	0.2613	0.3443	0.1165
BL8:	Bayesian	0.2253	0.1495	0.3085	0.0617
BL9:	Classify + Tag. [1]	0.2506	0.2796	0.3410	0.1304
Ours:	FRLM ($\gamma_H = 0.8$)	**0.2919**†‡	**0.3418***†‡	**0.3934**	**0.1616***†‡

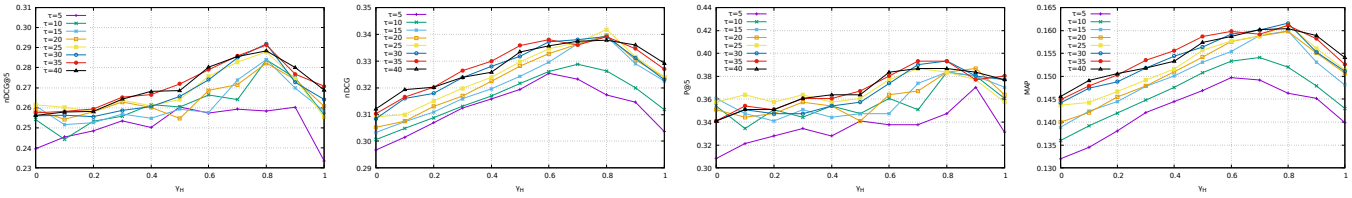
Table 2: Comparisons between FRLM with the baselines. $*$, † and ‡ denote significant improvements over the three strongest baselines - BL1, BL9 and BL4, respectively.

BL5 (Most Popular K) is a very common and sometimes very useful approach employed by recommender systems. Poor performance of this method in our problem setup indicates that a generalized people's choice approach does not really work well where personalization is crucial. Although BL6 (Profile Popular K) performed better than BL5 as it captured the personalized user preference better than BL5, the performance of BL6 is inferior to BL1 (BM25) where we used all tags from the set of tuples (D, T, r), where $r \geq \frac{4}{5}$. We noticed that a POI may have different types of tags. e.g. if a tag 'Seafood' is present as a primary feature, that belongs to a POI with higher rating, then 'Seafood' should be considered as an important tag. On the other hand, if there is another POI with other primary features that the user did not like, but the POI also serves 'Seafood', then the tag 'Seafood' will get a lower rating for that specific POI. As a result discarding tags based on the average rating suffers from information loss.

Both BL7 and BL8 did not perform well due to the lack of sufficient training data. BL9 (Classification + Tag matching), being inferior than FRLM in all metrics, is a precision oriented system. It performed comparatively well in nDCG@5 but in case of recall (nDCG and MAP), it performed noticeably worse.

A higher nDCG@5 measure of FRLM indicates that it is able to retrieve documents with highly judged relevance scores at top ranks in comparison to the baselines. This is particularly beneficial from a user satisfaction point-of-view because a user does not need to scroll-down a list of retrieved suggestions to find her likely best matches. It is particularly worth noting the considerable improvements in the nDCG values, which indicates that FRLM achieves a high recall and will be useful for users who are more patient to explore a list of suggestions to find a set of likely matching venues.

With reference to the best results obtained by FRLM, we would like to mention that although our results are not directly comparable with the official results of TREC-CS tasks in terms of absolute values of the evaluation metrics (e.g. nDCG@5 or MAP), our reported evaluation metric values are not considerably different from the best evaluation metric values of the TREC-CS task.



(a) nDCG@5 variations (b) nDCG variations (c) P@5 variations (d) MAP variations

Figure 3: Effect of precision at top ranks (nDCG@5 and P@5) and recall (nDCG and MAP) with respect to changes in number of terms used in FRLM estimation (τ) and the relative importance assigned to user profile information (γ_H).

Next, we investigate the effects of varying the parameter γ_H (i.e. the trade-off between exploration and exploitation) on the performance of FRLM. Figure 3 shows the sensitivity of FRLM (measured with nDCG@5, nDCG, P@5 and MAP) with respect to the number of terms (τ) used to define the weighted distribution of terms, and the relative importance of the historical context of a user with respect to the POIs in the current context, i.e. γ_H.

An interesting observation is that FRLM performs best with a balanced trade-off between exploration and exploitation. In particular, the optimal results (both in terms of nDCG@5 and nDCG) are obtained when $\gamma_H = 0.8$. Moreover, the effectiveness of FRLM is not good with the user profile history only, which indicates the diverse nature of the historical information itself and demonstrates the usefulness of selectively extracting pieces of information from the history that are contextually relevant in the current situation.

We also observe that too few or too large a number of terms tends to decrease retrieval effectiveness. The former is not able to adequately capture the relevant semantics required to match the profile with the current context, while the latter introduces noise (from parts of profile that are not relevant in the current context) in the estimated FRLM distribution.

6 CONCLUSIONS AND FUTURE WORK

This paper proposes a generic relevance feedback based framework for contextual POI recommendation. A characteristic of our model is that it achieves a sweet-spot between the user's preference history in past contexts (exploitation), and the relevance of top-retrieved POIs in the user's current context (exploration). Our experiments on the TREC-CS 2016 dataset show that our proposed model of a factored relevance model is able to effectively combine these two sources of information, leading to significant improvements in contextual recommendation quality.

In future, we would like to explore ways of further improving the model by incorporating semantic matching (possibly with the use of embedded word representations) between profiles and POI descriptors in the current context.

ACKNOWLEDGMENTS

Funding. This work was supported by the ADAPT Centre for Digital Content Technology, funded under the Science Foundation Ireland Research Centres Programme (Grant 13/RC/2106) and is co-funded under the European Regional Development Fund and by the European Unions Horizon 2020 research and innovation programme under the Marie Skodowska-Curie grant agreement No.: 713567.

Tribute. The first author would like to dedicate this work to the loving memory of his PhD supervisor Prof. Séamus Lawless (Shay), who suffered a tragic death after fulfilling his dream of scaling Mt. Everest.

REFERENCES

[1] Mohammad Aliannejadi and Fabio Crestani. 2017. Venue Appropriateness Prediction for Personalized Context-Aware Venue Suggestion. In *SIGIR 2017*.
[2] Mohammad Aliannejadi, Dimitrios Rafailidis, and Fabio Crestani. 2017. Personalized keyword boosting for venue suggestion based on multiple LBSNs. In *European Conference on Information Retrieval*. 291–303.
[3] Li Chen, Guanliang Chen, and Feng Wang. 2015. Recommender systems based on user reviews: the state of the art. *User Modeling and User-Adapted Interaction* 25, 2 (2015), 99–154.
[4] Chen Cheng, Haiqin Yang, Irwin King, and Michael R Lyu. 2012. Fused Matrix Factorization with Geographical and Social Influence in Location-Based Social Networks.. In *Aaai*, Vol. 12. 17–23.
[5] Adriel Dean-Hall, Charles L. A. Clarke, Jaap Kamps, Paul Thomas, and Ellen M. Voorhees. 2012. Overview of the TREC 2012 Contextual Suggestion Track. In *Proceedings of The Twenty-First Text REtrieval Conference, TREC 2012*.
[6] Romain Deveaud, M-Dyaa Albakour, Craig Macdonald, and Iadh Ounis. 2015. Experiments with a Venue-Centric Model for Personalisedand Time-Aware Venue Suggestion *(CIKM '15)*. ACM, New York, NY, USA, 53–62.
[7] Rainer Gemulla, Erik Nijkamp, Peter J. Haas, and Yannis Sismanis. 2011. Large-scale Matrix Factorization with Distributed Stochastic Gradient Descent. In *Proc. of KDD '11*. 69–77.
[8] Jean-Benoit Griesner, Talel Abdessalem, and Hubert Naacke. 2015. POI Recommendation: Towards Fused Matrix Factorization with Geographical and Temporal Influences. In *ACM RecSys 2015*. 301–304.
[9] Seyyed Hadi Hashemi, Charles LA Clarke, Jaap Kamps, Julia Kiseleva, and Ellen M Voorhees. 2016. Overview of the TREC 2016 contextual suggestion track. In *Proceedings of TREC*, Vol. 2016.
[10] Nasreen Abdul Jaleel, James Allan, W. Bruce Croft, Fernando Diaz, Leah S. Larkey, Xiaoyan Li, Mark D. Smucker, and Courtney Wade. 2004. UMass at TREC 2004: Novelty and HARD. In *TREC 2004*.
[11] Ming Jiang and Daqing He. 2013. PITT at TREC 2013 Contextual Suggestion Track.. In *TREC*.
[12] Victor Lavrenko and W. Bruce Croft. 2001. Relevance Based Language Models. In *ACM SIGIR 2001 (SIGIR '01)*. ACM, New York, NY, USA, 120–127.
[13] Asher Levi, Osnat Mokryn, Christophe Diot, and Nina Taft. 2012. Finding a Needle in a Haystack of Reviews: Cold Start Context-based Hotel Recommender System. In *RecSys '12*. 115–122.
[14] Hua Li and Rafael Alonso. 2014. *User modeling for contextual suggestion*. Technical Report. LEIDOS HOLDINGS INC RESTON VA.
[15] Hanchen Li, Zhen Yang, Yingxu Lai, Lijuan Duan, and Kefeng Fan. 2014. *BJUT at TREC 2014 contextual suggestion track: Hybrid recommendation based on open-web information*. Technical Report.
[16] Dwaipayan Roy, Ayan Bandyopadhyay, and Mandar Mitra. 2013. A Simple Context Dependent Suggestion System. In *TREC*.
[17] Ruslan Salakhutdinov and Andriy Mnih. 2008. Bayesian Probabilistic Matrix Factorization Using Markov Chain Monte Carlo. In *ICML '08*. 880–887.
[18] Peilin Yang and Hui Fang. 2012. *An exploration of ranking-based strategy for contextual suggestion*. Technical Report.