social   answer

Harnessing collaborative interest for improving question recommendation with context-aware semantic model

Jorge Castro[a,b,*], Jie Lu[b], Guangquan Zhang[b], Luis Martínez[c]

[a]*Department of Computer Science and Artificial Intelligence, University of Granada, Granada (Spain)*
[b]*School of Software, University of Technology Sydney, Sydney (Australia)*
[c]*Computer Science Department, University of Jaén, Jaén (Spain)*

{ Big Data?

Abstract

Content-Based Recommender systems (CBRSs) filter relevant items to users in overloaded search spaces using information about their preferences. There are successful CB RSs for several domains including recommendation ~~of~~ news and question answering (QA), among others. Classical CB ~~The traditional~~ recommendation scheme on ~~consists of~~ analyzes ~~analyzing the terms used in the item description to~~ generate an item profile and a user profile that ~~is~~ are → based on item description → used with recommendation later used to recommend items that match the user profile. This basic scheme can be further improved considering that context influences user preferences. Some examples of contexts are the day of week, time, season of year or companion of the target user. An interesting source of context is the trend in current collaborative interest, where a feed of status updates can be analyzed to model the context. When the information is extracted in such a way, there are several key aspects in the context integration with the user profile, such as context cleaning, aggregation and weighting. This paper explores such aspects and proposes a recommender system that integrates context to improve QA recommendation with contextual information. A case study will evaluate the results on several datasets, showing that the context integration benefits recommendation.

Keywords: recommender systems, context-aware recommendation, user profile contextualization

*Corresponding author
Email addresses: jcastro@decsai.ugr.es (Jorge Castro), jie.lu@uts.edu.au (Jie Lu), guangquan.zhang@uts.edu.au (Guangquan Zhang), martin@ujaen.es (Luis Martínez)

→ where? www e-commerce??

1. Introduction

The increasing amount of information available in current scenarios affects users' satisfaction when they search for items that meet their interests. This situation originates that users need to put significant effort for finding relevant pieces of information for them. In some scenarios it might not be possible for users to explore information items in order to select the most suitable one. Hence, the *information overload* problem impacts users satisfaction. Recommender Systems (RSs) have been a powerful tool for alleviating information overload in large search spaces. RSs have been proved to be successful in several domains, such as e-business [15], e-learning [32, 34], e-tourism [4, 18], e-commerce [27], web pages [17, 33] and financial investment [16], among others. In this paper , we focus on question answering (QA) recommendation with contextual information integration. → citations ??

There are several approaches within RSs. The most widespread ones are Collaborative Filtering (CFRS) and Content Based (CBRS). The main difference between ~~among~~ them is that CFRSs focus on users' interaction with items, i.e., user preferences, while CBRSs focus on the analysis of items descriptions, i.e., item content. Therefore, the performance of these recommendation approaches is subject to the quality and amount of available information in each type.

Within CBRSs we distinguish two kinds of CBRSs: (i) based on item features, and (ii) based on items descriptions. In this paper we focus on the latter, given that QA items have a strong component [citations] of textual information for both explaining the question and answering it. Although CBRSs suffer from user cold start because they need some input from user preferences and lack of diversity in recommendations [7], CBRS have demonstrated their utility when new items are introduced in the system, i.e., in scenarios with strong item cold-start [3]. This feature makes the application of CBRS approaches interesting in domains where new items are constantly introduced, such as web pages or news. In this direction, QA recommendation shares the features to apply CBRSs, hence, we focus on them.

In the QA domain recommendation scenario there are several proposals for recommendation with CBRS approaches. Shao et al.[30] propose to apply Latent Dirichlet

Allocation to label questions in a latent semantic feature category and find the most suitable answerers. In this direction, Zheng et al. [35] combine trust-based analysis of answerers with content analysis. While these approaches aim to reduce the answer time searching for adequate answerer for new posted questions, there are other RSs that aim to expand users' knowledge recommending already answered questions. Odiete et al. [19] analyze users' preferences and build a graph of expertise used to find gaps in their knowledge and suggest relevant questions.

In this work we focus on recommending answered questions that are in the target user's area of interest, and that are also relevant regarding the topics that are being discussed by users in microblogging sites. Therefore, it is interesting to explore Context-Aware CBRSs (CA-CBRSs), which integrate contextual information to the content-based recommendation. De Pessemier et al. [12] consider recommendations in mobile devices as very suitable to integrate context-awareness, and uses devices sensors and time of the day to deliver contextualized news recommendations. In QA recommendation, SeaHawk [24] and Prompter [23] provide CA-CBRSs that support programmers to complete issues and bugs using query completion and recommends StackOverflow questions, where the context is the specific part of the source code in which the recommendations are requested. In this direction, Libra [25] also integrates recommendations but it also considers, in addition to context extracted from the Integrated Development Environment context, resources opened by the user such as URLs or documents to better understand his/her context. Other works consider collaborative interest as current buzzwords to deliver currently relevant recommendations in e-commerce scenarios [22]. As it can be seen, there is no previous approach that focuses on context-aware recommendation regarding collaborative interest in the QA domain.

In this paper, we propose a novel CA-CBRS approach that recommends QA items and introduces context awareness based on topic detection within collaborative interest. In this proposal, the context is extracted from microblogging systems to characterize current trends in collaborative interest. The usage of such a context mainly helps the RS to recommend questions related to topics of interest and indirectly overcomes the overfitting problem. However, the context extracted this way is often noisy or several topics are mixed. With this regard, we propose to cluster context to identify the topics that are being discussed, and after that the context topic most suitable to target user's preferences is selected to build a contextualized user profile that combines preferences and context. This way, the proposal provides contextualized recommendations that are also adjusted to users' individual interests.

The remaining of this paper is structured as follows. Section 2 provides a background of CBRSs and CA-CBRSs. Section 3 introduces in further detail our proposal of CA-CBRS for QA recommendation. Section 4 shows a case study performed to evaluate the proposal and discuss the findings. Finally, Section 5 concludes the paper.

2. Preliminaries

This section provides the required background for the current research, including basics about CBRS, and related works in content based recommendation with context awareness.

2.1. Content-based recommender systems.

An accurate definition of Recommender System (RS) given by R. Burke [9] is *"any system that produces individualized recommendations as output or has the effect of guiding the user in a personalized way to interesting or useful objects in a large space of possible options"*. Within RSs, various techniques can be distinguished based on the knowledge source [12]: demographic, knowledge-based, community-based, content-based, collaborative, and hybrid recommendations. Among them, we focus on CBRSs.

The various CBRSs approaches can be classified regarding the item representation. Here we focus on those CBRSs that use a vector space modeling to represent items. With this regard there are CBRS with (i) feature-based representations, and (ii) free-text representation.

In feature-based representation, items are usually stored in a database table where rows are items and columns are the item features (also called fields or properties) and each item might have a different value for each of these features. The recommender system learns a model that uses such information to approximate the rating function. ~~With this regard~~ there are proposals based on weighting features importance regarding the user's ratings using collaborative information [31] or using entropy and dependence between items' features and user's ratings [10]. Other approaches learn the rating function using linear regression [12].

In free-text representation there is a natural language piece of text that describes the item, such as movie synopsis or that is part of the item itself, such as the content of a web page or a news article. Such a kind of unstructured data provides information about the item (called document in these systems), however, they also have the complexity of dealing with natural language due to polysemous words and synonyms.

citation [ ]
The TFIDF approach is usually applied when dealing with free-text representation items. In TFIDF, the unstructured data is converted in structured data stemming words [26] to keep their root. This process reduces the number of components of documents unifying words such as computer, compute and computing, which are different forms that share meaning. After that, for each document, a vector of weights of each term is generated based on the importance of the term on the document:

$$profile_d^{tfidf} = \{w_{t,d} \quad s.t. \quad t \in d\} \tag{1}$$

$$w_{t,d} = tf_{t,d} * idf_t \tag{2}$$

$$idf_t = -\log\left(\frac{|N|}{|N_t|}\right) \tag{3}$$

where $tf_{t,d}$ is the number of occurrences of term t in document d, N is the set of all documents and N_t is the set of documents that contain the term t at least once.

At this point the system contains a vector space model of items. User profiles can be generated aggregating the profiles of the items that they liked in the past [31]. The recommendation is computed comparing user profiles with item profiles with the cosine correlation and the closest ones are recommended.

While TFIDF method is effective, it cannot deal with polisemy or synonym words. In order to overcome this issue, Latent Semantic $\overset{\text{Analysis}}{\boxed{\text{Indexing}}}$ (LSA) is applied [11]. In LSA, the term-document matrix is factorized with Singular Value Decomposition

(SVD) to reduce it to orthogonal dimensions and keep the f most relevant singular values (see Figure 1).

$$TFIDF_{(|D|\times|T|)} = U_{(|D|\times f)} * s_{(f)} * V^{t}_{(f\times|W|)} \tag{4}$$

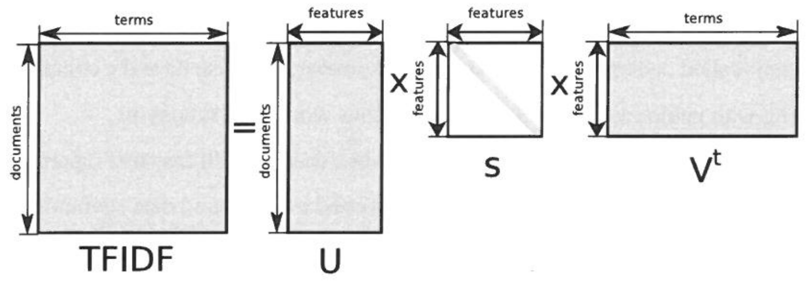

Figure 1: Decomposition of TFIDF matrix with singular Value decomposition. Note that s is a diagonal matrix with the singular values sorted in descending order.

This way, a reduced feature space is defined, which properly manages noise and redundancy of terms. User profiles are generated from this feature-space definition through a linear combination of document profiles that they liked [6]. Then, recommendations are generated comparing user and document profiles with cosine correlation coefficient.

2.2. Context-aware recommender systems.

In addition to the information traditionally used by RSs to recommend, as noted by R.Burke [9], other sources of information can be considered in the recommendation, such as the context in which the recommendation is received by users. F. Ricci [28] stated that the conditions or circumstances in which the recommendation is delivered significantly affect the decision behavior of the users. Therefore, the consideration of users' context is key to provide interesting recommendations.

With this regard, the various context-aware recommendation approaches can be classified into three classes [2]:

- Pre-filtering: The system selects the feedback gathered in the same context in which the recommendation is delivered to the user.

- Post-filtering: The recommendations are generated first without considering contextual information. After that, the item predictions are modified regarding the specific context of the users, possibly filtering out some items.

- Contextual modeling: The contextual information is directly integrated in the model that is used to recommend.

In pre-filtering, the approach is to filter out the information that was not not gathered in the current context. In traditional RSs the information is viewed as a function $R : User \times Item \rightarrow Rating$ that the RS tries to approximate. In CARS, the information can be viewed as a three dimensional cube $User \times Item \times Context \rightarrow Rating$. Contextual pre-filtering selects only the information relative to the context, hence, it tries to approximate function $R_{context} : User \times Item \rightarrow Ratings_{context}$ This way, they only consider ratings generated in the context in which the recommendation is delivered to the user. Contextual pre-filtering is a simple and effective approach, but it is affected by data sparsity when there is not enough information generated in all contexts to provide accurate recommendations. Some researchers have tried to overcome this issue through context-generalization [2] when the information available in the current context is not enough and include information from the broader context.

In post-filtering, the recommendations are first computed overlooking the contextual information, as in traditional RSs. After this initial step, recommendations are adjusted to the current context either removing irrelevant items or through a weighting function that changes predictions of items regarding the suitability of target user's context. A previous work [21] evaluated them in the same scenarios and compared pre- and post-filtering approaches. It determined that neither pre-filtering nor post-filtering completely dominates the other and a study to determine the best approach is needed in each specific case.

Previous approaches try to reduce the CARS problem to a two dimensional one that can be solved with traditional RSs. This is not the case in contextual modeling, in which contextual information is directly integrated in the recommendation model to recommend. With this regard, researchers have explored heuristic-based [20], probabilistic [1], or matrix factorization [5] contextual modeling approaches.

3. Semantic model for recommending questions with context awareness based on topic detection in collaborative interest

Here, a novel proposal, LSAContextCluster, for recommendation based on CA-CBRS is introduced. Recommendations might need to be targeted to specific contexts, e.g., when a system delivers recommendations of answered questions about history and current situation makes some questions more interesting than others, such as in the celebration of the anniversary of some relevant historic event. In these cases, it is possible to modify user profiles to include contextual information in a way such that later recommendations are both targeted to user preferences and current context.

Single and clear example

The proposed model fits into the CARS approach of contextual modeling, because it integrates contextual information in the model built by the RS. The general scheme of the proposal, LSAContextCluster, is depicted in Figure 2, and it is composed of five phases:

(i) QA domain semantic analysis: It applies LSA to reduce the dimensionality of the term-document matrix.

(ii) Build user preference profile: It analyses users' preferences and generates a profile for each of them based on the profiles of the document he/she liked in the past.

(iii) Build context model: It analyses the context, which consists on a stream of status updates within a given time frame, applies clustering to separate the various topics that the context contains, and generates feature-space profiles for each context topic.

Before you did not mention why stream

(iv) Contextualize user profiles: It selects context topic that is most suitable to target user's preferences, combines the preference-based user profile and the context topic profile to generate the contextualized user profile.

(v) Prediction: It compares the document profiles and the contextualized user profile to recommend.

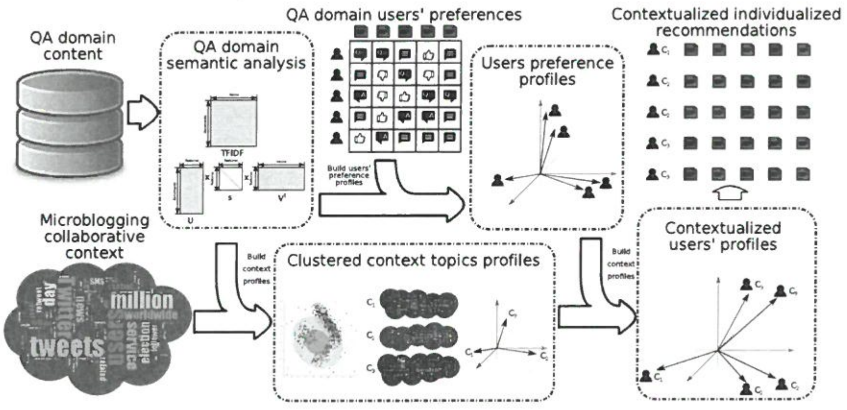
QA domain content
QA domain semantic analysis
TFIDF
U
s
V'
QA domain users' preferences
Build users' preference profiles
Users preference profiles
Contextualized individualized recommendations
Microblogging collaborative context
Build context profiles
Clustered context topics profiles
Build context profiles
Contextualized users' profiles

Figure 2: General scheme of the proposal.

3.1. *QA domain semantic analysis*

Our proposal assumes that the QA dataset contains textual information of the question and their related answers. In this proposal we consider the questions together with all their answers as the document, and the words used in their text as the terms. The terms are stemmed using the Porter Stemmer algorithm [26]. Once terms are stemmed, the TFIDF document profiles $profile_d^{TFIDF}$ are built according to Eq. 1.

Once the TFIDF document profiles are built, LSA is performed to reduce the dimensionality of the matrix. LSA is proven to be effective through the description of both document and terms in a feature space with a reduced number of features. Therefore, the aim of this step is to decompose the initial word-document matrix in a word-features matrix U, a singular value vector s, and a document-features matrix V (see Eq. 4).

An approximated factorization of the TFIDF matrix is performed with Singular Value Decomposition, which allows to reduce the dimensionality of the original matrix keeping the f most relevant singular values of the original matrix. Hence, we obtain the profile of both terms and documents in the feature space, which compose the QA domain semantic model:

$$profile_d^{LSA} = \{u_{t,1}, \ldots, u_{t,f}\} \tag{5}$$

Cuando se usa TFIDF se obtienen 2 vectores, uno del perfil y otro de pesos para cada feature.
No tengo claro lo que estáis haciendo al dar un solo vector

Table 1: Users' preferences over items, the rating matrix.

	d_1	$\dots$	d_k	$\dots$	d_n
u_1	r_{u_1,d_1}	$\cdots$	r_{u_1,d_k}	$\cdots$	r_{u_1,d_n}
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$
u_j	r_{u_j,d_1}	$\cdots$	r_{u_j,d_k}	$\cdots$	r_{u_j,d_n}
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$
u_m	r_{u_m,d_1}	$\cdots$	r_{u_m,d_k}	$\cdots$	r_{u_m,d_n}

$$profile_t^{LSA} = \{v_{t,1}, \dots, v_{t,f}\} \tag{6}$$

3.2. Building user preference profile

At this point LSAContextCluster has built a model with terms and documents profiles. In order to provide personalized recommendations to users, it is needed to build user profiles in the same feature-space. LSAContextCluster holds a unary matrix that states whether a given user has expressed interest in another document (see Table 1) either creating, commenting, or voting it. In this table, the set of documents that user u has expressed interest in is defined as:

$$R_u = \{d \quad s.t. \quad r_{u,d} \in R\} \tag{7}$$

This way, the user's profile is built upon the profiles of the documents that belong to R_u, and describes the user's preferences in terms of the latent space:

$$profile_u^{LSA} = \sum_{d \in R_u} profile_d^{LSA} = \{\sum_{d \in R_u} profile_{d,1}^{LSA}, \dots, \sum_{d \in R_u} profile_{d,f}^{LSA}\} \tag{8}$$

Pesos
Perfil ??

3.3. Context model building

To include contextual information in the recommendation process, it is needed to build the context model. In this proposal, we aim to promote questions that are relevant regarding current happenings. Recent researches [14] have highlighted the immediacy of microblogging services such as Twitter, where users share short sentences or fragments of news, thus, our proposal uses it as the source of collaborative interest.

$\rightarrow$ Deberías mencionar esto desde la introducción

Formally, a status update consist of a free text input generated by a user with certain timestamp, among other meta-data. Analyzing these status updates, LSAContextCluster generates a model of the context that is later used to modify the user profile. The scheme of the context model building phase is depicted in Fig. 3.

Entiendo que Microbl collaborat es filtrado para eliminar términos que no están en QA y que al MCC se le aplica LSA pero la parte indicada en la figura no muestra eso. Estoy equivocado?

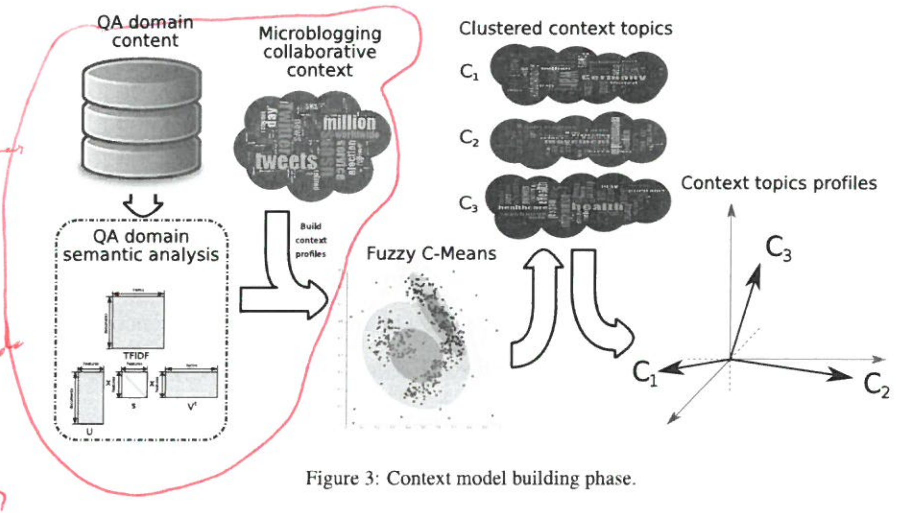

Figure 3: Context model building phase.

The context is composed of the status updates that were generated in a given time window. In this proposal, it is set to 24 hours, although [illegible] this parameter can be modified regarding the desired sensitivity of the context model. First, all terms of the status updates of the current context are stemmed. After that, from all the terms that the context contains, LSAContextCluster filters out the terms that do not appear in the QA semantic model generated in phase one (see Section 3.1).

→ Te referencia por ejemplo a los límites ??

Given that the context is composed of several status updates, there can be a mixture of topics. To determine the context topics, the proposal performs a fuzzy clustering of the terms used in the context. Hence, fuzzy c-means clustering algorithm [8] groups the terms using their feature vector $profile_t^{LSA}$ as the term definition. The distance among terms used in the clustering is based on cosine correlation coefficient. The result is a set of clusters where each cluster c_i defines a context topic.

Once the terms of current context are grouped in context topics, the proposal generates a context profile for each topic combining the profiles of the terms that are included in each cluster. Therefore, LSAContextCluster builds a profile for each context topic using the feature representation of each term from the QA domain (see Eq 9). At the end of this phase, LSAContextCluster has generated a model of the context composed of several context profiles, one for each context topic detected by the clustering.

$$profile_{c_i}^{LSA} = \sum_{t \in c_i} profile_t^{LSA} = \{\sum_{t \in c_i} profile_{t,1}^{LSA}, \ldots, \sum_{t \in c_i} profile_{t,f}^{LSA}\} \tag{9}$$

3.4. User profile contextualization

In this step, the target user's preference profile is combined with the context model to provide contextualized personalized recommendations. To do so in a personalized way, from all the context topic profiles that the context model contains, LSAContextCluster selects the most similar to the user's preference profile. This way, the context topic c_i that has a greater cosine coefficient with the target user preferences is used to modify his/her profile, hence, the contextualization of user profiles is personalized to user preferences.

$$\underset{c_i}{\text{argmax}} \quad cosine(profile_u^{LSA}, profile_{c_i}^{LSA}) \tag{10}$$

After this selection, the profile of the selected context topic c_i and the user's preference profile are combined to obtain the contextualized user profile: Explain the convex combination...

$$profile_{C,u}^{LSA} = \alpha * profile_u^{LSA} + (1 - \alpha) * profile_{c_i}^{LSA} \tag{11}$$

3.5. Prediction

Once we have the contextualized user profile, we can produce a prediction of the suitability for a given item regarding the profile. The recommendation is a list of documents sorted by $p_{u,d}$:

$$p_{u,d.c} = profile_{C,u}^{LSA} * s * profile_d^{LSA} \tag{12}$$

4. Case study

To evaluate the proposal, we performed an experiment that simulates the recommendation of QA items in various contexts. The remainder of the section is structured