Exploiting text mining techniques for contextual recommendations

http://www.producao.usp.br/handle/BDPI/48649

Downloaded from: Biblioteca Digital da Produção Intelectual - BDPI, Universidade de São Paulo
Exploiting Text Mining Techniques for Contextual Recommendations

Marcos Aurélio Domingues*, Camila Vaccari Sundermann*, Marcelo Garcia Manzato*, Ricardo Marcondes Marcacini†, Solange Oliveira Rezende*

*Institute of Mathematics and Computer Science - University of São Paulo
São Carlos, SP, Brazil
{mad, camila, mmanzato, solange}@icmc.usp.br

†Federal University of Mato Grosso do Sul
Três Lagoas, MS, Brazil
ricardo.marcacini@ufms.br

Abstract—Unlike traditional recommender systems, which make recommendations only by using the relation between users and items, a context-aware recommender system makes recommendations by incorporating available contextual information into the recommendation process. One problem of context-aware approaches is that it is required techniques to extract such additional information in an automatic manner. In this paper, we propose to use two text mining techniques which are applied to textual data to infer contextual information automatically: named entities recognition and topic hierarchies. We evaluate the proposed technique in four context-aware recommender systems. The empirical results demonstrate that by using named entities and topic hierarchies we can provide better recommendations.

Keywords—Recommender Systems; Context-Aware Recommender Systems; Contextual Information; Text Mining

I. INTRODUCTION

Nowadays, most web sites offer a large number of items (e.g., movies, music, web pages, etc) to their users. Finding relevant content according to each individual’s tastes has, thus, become a challenge. Recommender systems have emerged in response to this problem. A recommender system is an information filtering technology which can be used to predict preference ratings of items, not currently rated by the user, and/or to output a personalized ranking of items that are likely to be of interest to the user [1]. These systems have flourished on the Internet, and web sites such as Amazon1, Netflix2 and Last.fm3 are good examples of recommenders that adapt recommendations to particular user’s tastes.

Traditionally, the data that are most often available for recommender systems are web access logs which represent the interaction activity between users and items. Therefore, the most common systems focus on these two entities to build a model which is used to recommend an ordered list of $N$ items that are expected to be of interest to a certain user.

Unlike the traditional systems, that make recommendations only by using the relation $User \times Item$, a context-aware recommender system makes recommendations by incorporating available contextual information into the recommendation process as explicit additional categories of data [2]:

$$User \times Item \times Context \rightarrow Recommendation,$$

where $Context$ specifies the contextual information associated with the application.

There are many definitions of context in the literature depending on the field of application and the available customer data [1]. In this paper, context is defined as any information that can be used to characterize the situation of an entity (e.g., a web page) [3].

Thus, a promising way to improve the accuracy of recommender systems is to incorporate additional information, such as context, besides the typical information about users and items. However, it is usually necessary an intense and time-consuming human effort to identify, collect and label this additional information about the items in order to be properly employed in recommender systems. Moreover, manually label the content becomes impracticable for large databases.

In this paper, we exploit two text mining techniques to capture the context of textual data. The first technique consists of extracting named entities (i.e., terms related to time and location) from the textual data. The second one consists of using the contextual information extracted from topic hierarchies to improve the accuracy of context-aware recommender systems. Topic hierarchies are efficient models to capture the context of textual data in order to organize them [4]. We empirically evaluate both techniques and the results demonstrate that they can provide better recommendations.

This paper is structured as follows: Section II discusses some definitions of context. In Section III, we present our proposal. The extraction of named entities is described in Section III-A. The learning of topic hierarchies is described in Section III-B. The context-aware recommender systems used to evaluate the contextual information are presented in Section IV. We evaluate our proposal in Section V. In Section VI we depict some related work.
Finally, in Section VII, we present conclusion and future work.

II. CONTEXTUAL INFORMATION

According to [5], the concept of context has been studied extensively in areas of computing and other disciplines. As it has already been mentioned before, context can be defined in many ways, depending on the field of application. After examining 150 different definitions of context from different fields, Bazire and Brezillon [6] concluded that it’s difficult to find a unifying definition. They raised some questions, such as: “Is context a frame for a given object? Is it the set of elements that have any influence on the object? Is something static or dynamic?”.

For Dourish [7], there are two different views of context: the interactional view and the representational view. In the interactional view the context is defined dynamically and there is a relationship between context and activity, in which the activity gives rise to context and context influences activity. In contrast, in representational view, context can be described as a set of known attributes, whose structure does not change through the time.

The most widely accepted definition of context and that is used in this paper was proposed by Dey [3]: “Context is any information that can be used to characterize the situation of an entity”. The entities are, in our work, web pages. According to Adomavicius et al. [8], the contextual information can be of different types. As an example, they mentioned an application for recommending movies to users. Besides the attributes of the users and of the movies, there is also the contextual information. This information consists of three types: “Theater”, “Time” and “Companion”. Each type has some attributes/values. The type “Companion”, for example, has the attributes “alone”, “family”, “co-workers” and so on. This is a way of representing the context of the application. The contextual information can also be organized as a hierarchical structure that can be represented as trees [2], [9], [10]. In this way, the contextual information is a set of contextual dimensions $C$, where each dimension $C$ is defined by a set of $f$ attributes/values, i.e., $C = c_1, c_2, ..., c_f$. These attributes have a hierarchical structure. The values taken by attribute $c_f$ define more granular levels, while $c_1$ less granular levels of the contextual information. For example, in [10], Panniello and Gorgoglione represent the contextual attribute “period of the year” as a hierarchical structure illustrated in Figure 1.

As mentioned in the introduction of this paper, one challenge regarding context-aware recommenders is how to extract this additional data from users, items and their relation. Information such as illustrated in Figure 1 has to be obtained either manually or automatically. While manual methods are time-consuming and error-prone, automatic approaches require the development of algorithms and strategies to extract contextual information from the content. Next section, thus, describes our proposal, which suggests using text mining techniques applied to textual data (i.e., web pages).

III. OUR PROPOSAL

In this section, we present the two text mining techniques to capture the contextual information from textual data (i.e., web pages).

A. Contextual Information from Named Entities

The term Named Entity, widely used in Natural Language Processing applications, was born, according to Sekine [11], in the Message Understanding Conferences (MUC). Named entities are information units like terms, including person, organization and location, and numeric expressions including time, date, money and percent expressions [11]. For instance, in the sentence, from [12], “Flavel Donne works as an analyst in the General Trends, which has been base in Little Spring since July 1998”, “Flavel Donne”, “General Trends”, “Little Spring” and “July 1998” are person, organization, location and time entities, respectively.

The named entity recognition is a task that involves identifying words or expressions that belong to categories of named entities [12]. This process is divided into two subtasks [13]: identification of possible entities and categorization of entities. News articles, web pages, blogs usually contain named entities. In the top 10 search terms by GoogleSearch$^4$ in 2013, most of the terms are named entities. According to the jargon of journalists, the content of a news article must contain answers to six questions: “What”, “Who”, “When”, “Where”, “Why” and “How” [14]. In general, these issues are involved with named entities like persons and organizations, that answer the question “Who”; places, that answer the question “Where”; temporal expressions, that answer the question “When”; and so on.

Our first proposal consist of using entities from web pages as contextual information to improve the accuracy of context-aware recommender systems. We use REMBRANDT [15], a tool for named entities recognition and for detection of relationship between entities. This system was designed to recognize classes of named entities, like thing, location, organization, people, in texts written in Portuguese. REMBRANDT uses Wikipedia$^5$ as knowledge base for the classification of entities and it has its own interface, the SASKIA, to interact with this base. The goal of this interface, according to Cardoso [15], is

Figure 1: Hierarchical structure of the contextual attribute “period of the year” [10]

---


$^5$http://www.wikipedia.org
to facilitate the navigation in the structure of categories, links and redirects of Wikipedia.

The REMBRANDT follows three steps [15]: (1) Recognition of numeric expressions and generation of candidates for named entities, (2) Classification of named entities, and (3) Reclassification of named entities without classes. In step (1), the atomizer of Linguateca\(^6\) (Linguateca is a resource center to the computational processing of Portuguese) is used to split the text into sentences and units, which makes possible to recognize numeric expressions and to identify possible candidates for named entities. In step (2), candidates for named entities are first classified by SASKIA and then classified again by using grammar rules. Finally, in the last step, rules are used to detect relationships among named entities and, with these relationships, some named entities without classes may receive the same class of classified named entities related to them.

In this paper, we extract time and location entities from web pages, and use them as contextual information for context-aware recommender systems. Here, we assume the terms related to location and organization as location entities.

B. Contextual Information from Topic Hierarchies

Our second proposal consists of using unsupervised learning methods to generate topic hierarchies from textual data, that can be viewed as contextual information that characterize the items. In Figure 2 we see a dendrogram (i.e., a topic hierarchy), which is a binary tree where each node represents a set of documents and there are contextual descriptors/topics that indicate the context of these documents. Then, we use the contextual information (topics) to characterize the items in a context-aware recommender system.

![Figure 2: Example of dendrogram [16]](Image)

Although the use of textual information available about the items is a promising way to improve the accuracy of recommendation systems, there are many challenges on how to extract useful knowledge from these textual information [17]. Textual data are inherently unstructured, thereby requiring the application of techniques for text pre-processing to represent textual data in a concise and representative manner. Moreover, choosing the appropriate algorithm for extracting and organizing knowledge from texts, such as algorithms for topic hierarchies construction, is an important task for context-aware recommender systems.

Several approaches have been proposed in literature for topic hierarchies construction, such as algorithms based on term-clustering [18], [19] and clustering labeling [17]. Despite the large number of existing algorithms, no single algorithm is able to extract all possible topic structures from texts [20]. Each algorithm has a bias regarding the coverage and number of topics, making it difficult to decide which one is the best algorithm for each possible domain. For example, even a single topic hierarchy construction algorithm, with different initializations and parameters, can produce very different results.

In this sense, we use an approach for topic hierarchy construction based on the consensus clustering called BC\(^2\) (Buckshot Consensus Clustering) [4]. In BC\(^2\) approach, it is possible to combine solutions of different topic extraction algorithms in a single consensual solution. The results obtained with consensus clustering are promising in many aspects. Combining different structures usually results in a final solution of better quality than the individual solutions. Moreover, consensus clustering is easily parallelizable, promoting the scalability of applications.

In BC\(^2\) approach, several topics are initially extracted from textual data by executing different topic extraction algorithms. Each topic has a set of associated text documents (textual information about the items). Assuming that similar documents will be allocated on the same topics in several of the different solutions, then we compute a co-association matrix. The basic idea is to summarize a set of topics \(L\) by means of a matrix where each element has the value \(M(w,v) = \frac{a_{wv}}{\sum_{v}a_{v}}\) where \(a_{wv}\) is the number of times that the textual information about the items \(w\) and \(v\) are allocated in the same topic. The co-association matrix represents a new proximity relationship for the textual information about the items.

The “consensus” topic hierarchy is constructed from the relations of co-association matrix. The BC\(^2\) approach uses an agglomerative clustering strategy to compute the dendrogram. In this case all the documents are initially considered singletons (unitary clusters). Then the most similar pairs of clusters are iteratively merged until all the documents are allocated into a single cluster. Finally, it is associated a set of descriptors for each (sub)cluster of the dendrogram, thereby obtaining a topic hierarchy. For the descriptors extraction, we can use the most frequent terms (keywords, phrases, or expressions) of each cluster or even apply feature selection techniques to select relevant terms of each cluster.

IV. CONTEXT-AWARE RECOMMENDER SYSTEMS

As already stated, a context-aware recommender system makes recommendations by incorporating available contextual information into the recommendation process as explicit additional categories of data [2].
A context-aware recommender system can be classified according to the use of context in the recommendation process [21]. In Figure 3, we can observe that a system can be classified in pre-filtering, modeling and post-filtering. In pre-filtering, the contextual information is used to filter out irrelevant items before building the recommendation model. Modeling consists of using the context within the recommendation models. Finally, in post-filtering, the contextual information is used after building a traditional recommendation model to reorder or filter out recommendations.

![Figure 3: How to use context in the recommendation process](Image)

In this paper, we evaluate the effects of using the contextual information, obtained from the proposed techniques, in four different context-aware recommender systems. The recommenders, representing different use of context (Figure 3), are described in the next sections.

A. The Pre-Filtering Approach

In a pre-filtering approach, the contextual information is used as a label for filtering out those data that do not correspond to the specified contextual information. This filtering is done before the main recommendation method is launched on the remaining data that passed the filter (contextualized data) to generate the model.

In [2], the combined reduction approach (C. Reduction) uses the contextual information as label to segment the data. A segment is defined as a subset of the overall data selected according to the context or combination of its values.

Briefly, this approach consists of the following two phases. First, using the training data, a recommendation method is run for each contextual segment (e.g., accesses on Mondays would be a segment) to determine which ones outperform the traditional recommendation model (using only user and item data). Second, taking into account the context of the active session, we choose the best contextual model to make the recommendation. Here the best model is the one which has the highest F1 measure [2].

B. The Contextual Modeling Approach

The contextual modeling approach consists of using contextual information directly in the recommendation model. Here, the contextual information is part of the model in addition to the user and item data. In [22], we proposed a contextual modeling approach, called DaV1-BEST, that treats contextual information as virtual items, using them along with the regular items in a recommender system.

Let $m$ be the number of users $U = \{u_1, u_2, ..., u_m\}$ and $n$ the number of all possible items that can be recommended $I = \{i_1, i_2, ..., i_n\}$. In addition, we have other dimensions (i.e., contextual information), $C = \{C_1, C_2, ..., C_t\}$, where each dimension $C$ comprehends a set of values, i.e., $C = \{c_1, c_2, ..., c_f\}$. For example, the contextual information Hour can define a set of integer values from 1 to 24. Now, let $j$ be the number of multidimensional sessions in a web site $S = \{s_1, s_2, ..., s_j\}$. Each session $s$ is a tuple defined by a user $u \in U$, a set of accessed items $I_s \subseteq I$ and a set $C_s \subseteq C$ containing the contextual values associated with the session $s$, i.e., $s = (u, I_s, C_s)$.

The DaV1-BEST approach consists of transforming each multidimensional session $s = (u, I_s, C_s)$ into an extended two dimensional session $s' = (u, I_s \cup C_s)$, where the values of the additional dimension (i.e., contextual information) in $C_s$ are used as virtual items together with the regular items in $I_s$.

Once we have a set of extended two dimensional sessions $S'$, building/learning a contextual recommendation model consists of applying a traditional recommender algorithm on $S'$. Note that regular items are used to build the model and make recommendations. On the other hand, virtual items are used in addition to build/improve the recommendation model but they can not be recommended. We implemented a filter to guarantee this condition.

C. The Post-Filtering Approach

In this approach, we first ignore all the contextual information in the data and apply a traditional User $\times$ Item recommendation method on the whole un-contextual data set. Once we have the traditional model, we use the contextual information to contextualize (i.e., reorder or filter out) the recommendations generated by the model.

In [10], Panniello and Gorgoglione proposed two approaches, Weight PoF and Filter PoF, that use context to reorder and filter out the recommendations, respectively. The approaches first ignore the contextual information in the data and apply a traditional algorithm to build the recommendation model. Then, it computes the probabilities of user’s access items under a given context. The probability $P_c(u, i)$, which a user $u$ accesses an item $i$ under the context $c$, can be computed as the number of users who access the candidate item under a particular context divided by the number of users who access any item under that context. Finally, the score of the items are multiplied by the probabilities to reorder or to filter out the recommendations.
V. EMPIRICAL EVALUATION

To evaluate our proposal, we first combine the context-aware recommendation strategies, described in the previous sections, with the Item-based Collaborative Filtering [23]. Then, we compare the context-aware strategies (i.e., C. Reduction, DaVl-BEST, Weight PoF and Filter PoF) against the un-contextual item-based collaborative filtering (i.e., User×Item) in order to demonstrate how much the results are influenced if we adopt named entities or hierarchical topics as contextual information.

A. Data Set

The experiments were executed with a data set from an agrobusiness web site. The data set consists of 4,659 users and 1,543 different web pages about agrobusiness written in Portuguese language. This textual data is used directly to obtain the set of entities and topics. The users generated a total of 15,037 accesses to these pages.

For the evaluation of named entities, we considered as contextual information the terms related to location, time, and its combination extracted from the textual data by using the tool REMBRANDT [15]. Here, we have 877 different terms related to location, 1334 related to time, and the combination of both named entities generates a total of 2211 terms. For the topic hierarchies, we considered the topics generated by the BC² method as contextual information. For the topic hierarchy construction, we used different runs of the well-known k-means algorithm (with random centers initializations and cosine similarity) to obtain several data partitions for the consensus clustering. To analyze the effect of the number of topics used as context in the recommendation task, we selected subsets of topics using seven different granularities: \{50,100\}, \{15,20\}, \{10,15\}, \{10,50\}, \{5,10\}, \{5,100\} and \{2,7\}. In the granularity configuration \{x,y\}, the parameter x identifies the minimum number of items allowed in the topic, while the parameter y identifies the maximum number of items per topic. When a topic has a few items associated, usually the topic represents more specific contextual information. On the other hand, topics with many items associated represent more general contextual information about the items. Thus, the seven configurations presented above generate subsets of 26, 44, 101, 210, 305, 510 and 1230 topics, respectively, for the data set.

B. Experimental Setup and Evaluation Measures

We combined the contextual recommendation strategies with the Item-based Collaborative Filtering Algorithm [23]. In this algorithm, the recommender model is a matrix representing the similarities between all the pairs of items according to a similarity measure (in our case, the cosine angle). The top N recommendations are generated based on the 4 most similar items (the 4 nearest neighbors). To tune the algorithm, we ran a first set of experiments using different numbers of neighbors and analyzed the Precision measure. We observed that the Precision values tend to increase from 2 to 4 neighbors. For 5 neighbors, the values were a bit worse than for 4 neighbors. Therefore, we have chosen the 4 most similar items to make the recommendations. For the Filter PoF algorithm, we used 0.1 as a threshold to filter out the recommendations, since this value provided the best results in our experiments.

To measure the predictive ability of the recommender systems, we use the All But One protocol [24] with 10-fold cross validation, and calculate the metrics Precision and Mean Average Precision (MAP) [25]. To do this, the sessions in the data set are randomly partitioned into 10 subsets. For each fold, we use n – 1 of those subsets of data for training and the remaining one for testing. The training set \(T_c\) is used to build the recommendation model. For each user in the test set \(T_c\), we randomly hide one item, referred to as the singleton set \(H\). The remaining items represent the set of observables \(O\), based on which the recommendation is made. Then, we compute Precision and Mean Average Precision as follows:

**Precision** calculates the percentage of recommended items that are relevant. This metric is calculated by comparing, for each user in the test set \(T_c\), the set of recommendations \(R\) that the system makes, given the set of observables \(O\), against the set \(H\):

\[
Precision(T_c) = \frac{1}{|T_c|} \sum_{j=1}^{|T_c|} \frac{|R_j \cap H_j|}{|R_j|}. \tag{1}
\]

**Mean Average Precision** computes the precision considering the respective position in the ordered list of recommended items. With this metric, we obtain a single value accuracy score for a set of test users \(T_c\):

\[
MAP(T_c) = \frac{1}{|T_c|} \sum_{j=1}^{|T_c|} AveP(R_j, H_j), \tag{2}
\]

where the average precision (AveP) is given by

\[
AveP(R_j, H_j) = \frac{1}{|H_j|} \sum_{r=1}^{|H_j|} \frac{|Prec(R_j, r) \times \delta(R_j(r), H_j)|}{|Prec(R_j, r)|}, \tag{3}
\]

where \(Prec(R_j, r)\) is the precision for all recommended items up to rank \(r\) and \(\delta(R_j(r), H_j) = 1\), if the predicted item at rank \(r\) is a relevant item \((R_j(r) \in H_j)\) or zero otherwise.

In the empirical evaluation, we computed Precision@N, for \(N\) equal to 1, 2, 3, 5 and 10 recommendations; and MAP@N, for \(N\) equal to 5 and 10 recommendations. For each configuration and measure, the 10-fold values are summarized by using mean and standard deviation. To compare two recommendation algorithms, we apply the two-sided paired t-test with a 95% confidence level [26].

C. Results

In Table I, we present the ranking evaluation by means of MAP@5 and MAP@10 for the four context-aware recommendation algorithms (C. Reduction, DaVl-BEST, Weight PoF and Filter PoF), and also for the User×Item collaborative approach, which is used as baseline.
The results were obtained with three different named entities: location, time and the combination of both entities. In the table, we see that 3 out of 4 algorithms were able to obtain a statistically significant improvement over the baseline. The Filter PoF algorithm was not able to improve over the baseline. The explanation for this fact is that the threshold used to filter out the recommendations is too high for this particular case. With a high threshold, a huge amount of recommendations are filtered out, decreasing the accuracy of the recommender system. The baseline, in turn, obtained the same results regardless of the named entities because it does not use contextual information.

In Figure 4, we compared the precision accuracy for a varying number of recommendations. Analyzing Figure 4, we see that the C. Reduction, DaVI-BEST and Weight PoF algorithms provide a better precision than the baseline. Additionally, the precision value is quite similar for the three algorithms. Regarding Filter PoF, the contextual information obtained from the named entities was not able to improve the precision of the recommendations.

For the topic hierarchies, in Table II, we present the ranking evaluation by means of MAP@5 and MAP@10 for two context-aware recommendation algorithms (Weight PoF and Filter PoF), and also for the User × Item collaborative approach. We do not present the values for C. Reduction and DaVI-BEST because these algorithms present the same performance of the User × Item approach, which is used as baseline. The results were obtained at different levels of granularity: as previously mentioned, higher amount of topics means that more specific context types are used by the algorithm. In the table, it is possible to note that, for most granularity levels, both context-aware techniques were able to obtain a statistically significant improvement over the baseline. Again, the baseline, in turn, obtained the same results regardless of the amount of topics because it does not use contextual information. In addition, the best results of both context-aware recommenders were obtained when the highest number of topics was considered, which means that the more the system knows about the context, the better is the accuracy. This conclusion implies that the design of a recommender algorithm should consider gathering contextual information as much as possible; the advantage of our proposal, in this argument, is that this information extraction procedure is accomplished by an unsupervised technique.

In Figure 5 it is possible to get a better insight of the algorithms’ accuracy according to the number of topics. We selected three granularity levels (most general, most specific and one mid-term), and compared the precision accuracy for a varying number of recommendations. Analyzing Figure 5, we note that Filter PoF obtained low precision when using general context types, whereas its accuracy increased to the best of the three algorithms when more specific context types were considered. On the other hand, Weight PoF also achieved better results when more specific context types were considered, but such improvement was not as significant as the one obtained by Filter PoF. This difference can be explained by the design of these two context-aware algorithms: in the case of Filter PoF, it will recommend items under a given context only if such context is relevant for those candidate items, i.e., the probability $P_e(u,i)$ is higher than the threshold. Indeed, as explained in Section IV, such probability is computed by the number of users who access the candidate item under a particular context divided by the number of users who access any item under that context. Thus, given that more specific context types mean fewer items, such probability will have a stronger influence over the decision process.

Regarding Weight PoF, the importance of a given context is used to weight the score of a candidate item by means of multiplying its probability with the similarity values computed between the candidate and the observable items. In other words, items which are very similar to the observable ones may be recommended regardless of the considered context, which is used, in turn, only to weight the score function.

VI. RELATED WORK

Some state-of-the-art approaches for contextual information extraction from web content have been proposed in the literature. In [27], [28], the authors obtain contextual information from online reviews in order to improve item recommendation. In particular, Li et al. [27] compile a list of lexicons and use a string matching method to extract different types of contextual metadata from reviews. In [28], Hariri et al. propose a multi-labeled text classifier based on Labeled Latent Dirichlet Allocation. They assume that there are explicit labels representing contextual information, and such information is obtained for each review by mapping it to the labels. Our proposal differs from [27], [28] because it does not need a lexicon or a set of labels to extract metadata, which usually are unavailable for web content.

In [29], Semeraro et al. propose to use a spreading activation algorithm in order to compute the correlation between terms from the web document and from a set of external knowledge sources related to linguistic, world and social domains. They use the most correlated external terms as meaningful contextual features in a content-based recommendation process. An important issue related to this approach is that it can only be used when external knowledge sources are available. Thus, our proposal takes some advantage over this approach since it can be used with internal and external data sources.

VII. CONCLUSIONS

In this paper, we proposed to use contextual information from named entities and topic hierarchies to improve the accuracy of context-aware recommender systems.

Using named entities and topic hierarchies, the context-aware recommender systems provided better recommendations in our empirical evaluation. The contextual information obtained from the named entities improved the
Table I: Comparing the context-aware recommendation algorithms against the User × Item algorithm. Values for C. Reduction, DaV1-BEST, Weight PoF and Filter PoF are statistically significant (p-value < 0.05). The highest values are presented in boldface.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Location</th>
<th>MAP@05</th>
<th>Location + Time</th>
<th>Location</th>
<th>MAP@010</th>
<th>Location + Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>User × Item</td>
<td>0.298 ± 0.015</td>
<td>0.298 ± 0.015</td>
<td>0.298 ± 0.015</td>
<td>0.307 ± 0.015</td>
<td>0.307 ± 0.015</td>
<td>0.307 ± 0.015</td>
</tr>
<tr>
<td>C. Reduction</td>
<td>0.432 ± 0.044</td>
<td><strong>0.453 ± 0.035</strong></td>
<td>0.443 ± 0.037</td>
<td>0.446 ± 0.034</td>
<td>0.461 ± 0.035</td>
<td>0.453 ± 0.037</td>
</tr>
<tr>
<td>DaV1-BEST</td>
<td>0.439 ± 0.031</td>
<td><strong>0.453 ± 0.035</strong></td>
<td>0.443 ± 0.036</td>
<td>0.448 ± 0.031</td>
<td>0.461 ± 0.035</td>
<td>0.454 ± 0.036</td>
</tr>
<tr>
<td>Weight PoF</td>
<td><strong>0.440 ± 0.031</strong></td>
<td>0.445 ± 0.035</td>
<td><strong>0.445 ± 0.035</strong></td>
<td><strong>0.449 ± 0.031</strong></td>
<td><strong>0.464 ± 0.035</strong></td>
<td><strong>0.453 ± 0.034</strong></td>
</tr>
<tr>
<td>Filter PoF</td>
<td>0.062 ± 0.017</td>
<td>0.077 ± 0.014</td>
<td>0.056 ± 0.014</td>
<td>0.069 ± 0.017</td>
<td>0.085 ± 0.015</td>
<td>0.064 ± 0.014</td>
</tr>
</tbody>
</table>

Figure 4: Comparison of considered recommendation algorithms with different named entities and top-N recommendations: the graphic on the left-hand side shows the obtained results using the entities of location; on the center using as context the entities of time; and on the right-hand side using the combination of location and time.

Table II: Comparing the context-aware recommendation algorithms against the User × Item algorithm. Values for Weight PoF and Filter PoF are statistically significant (p-value < 0.05). The highest values are presented in boldface.

<table>
<thead>
<tr>
<th># topics/context</th>
<th>User × Item</th>
<th>Weight PoF</th>
<th>Filter PoF</th>
<th>User × Item</th>
<th>Weight PoF</th>
<th>Filter PoF</th>
</tr>
</thead>
<tbody>
<tr>
<td>26</td>
<td>0.298 ± 0.015</td>
<td>0.516 ± 0.027</td>
<td>0.407 ± 0.034</td>
<td>0.407 ± 0.015</td>
<td>0.527 ± 0.022</td>
<td>0.312 ± 0.035</td>
</tr>
<tr>
<td>44</td>
<td>0.298 ± 0.015</td>
<td>0.542 ± 0.060</td>
<td>0.520 ± 0.082</td>
<td>0.520 ± 0.015</td>
<td>0.551 ± 0.058</td>
<td>0.527 ± 0.079</td>
</tr>
<tr>
<td>101</td>
<td>0.298 ± 0.015</td>
<td>0.526 ± 0.063</td>
<td>0.464 ± 0.075</td>
<td>0.307 ± 0.015</td>
<td>0.532 ± 0.061</td>
<td>0.469 ± 0.075</td>
</tr>
<tr>
<td>210</td>
<td>0.298 ± 0.015</td>
<td>0.517 ± 0.043</td>
<td>0.386 ± 0.074</td>
<td>0.307 ± 0.015</td>
<td>0.524 ± 0.042</td>
<td>0.391 ± 0.073</td>
</tr>
<tr>
<td>305</td>
<td>0.298 ± 0.015</td>
<td>0.544 ± 0.037</td>
<td>0.594 ± 0.057</td>
<td>0.307 ± 0.015</td>
<td>0.552 ± 0.037</td>
<td>0.597 ± 0.057</td>
</tr>
<tr>
<td>510</td>
<td>0.298 ± 0.015</td>
<td>0.503 ± 0.051</td>
<td>0.313 ± 0.056</td>
<td>0.307 ± 0.015</td>
<td>0.511 ± 0.049</td>
<td>0.318 ± 0.055</td>
</tr>
<tr>
<td>1230</td>
<td>0.298 ± 0.015</td>
<td><strong>0.546 ± 0.024</strong></td>
<td><strong>0.618 ± 0.024</strong></td>
<td>0.307 ± 0.015</td>
<td><strong>0.552 ± 0.025</strong></td>
<td><strong>0.621 ± 0.025</strong></td>
</tr>
</tbody>
</table>

Figure 5: Comparison of considered recommendation algorithms at different granularities and top-N recommendations: the graphic on the left-hand side shows the obtained results using more general context types (26 topics); on the center using context types at mid-term granularity (210 topics); and on the right-hand side using more specific context types (1230 topics).

recommendations in 3 out of 4 context-aware recommender systems evaluated in this paper. On the other hand, the contextual information captured from the topic hierarchies only provided better recommendations for the Weight PoF and Filter PoF recommenders. However, this improvement generated the highest gains in terms of Precision and MAP.

As future work, we will expand our findings by using other data sets as well as other context-aware recommender systems in order to evaluate the effects of using named entities and topic hierarchies as contextual information in context-aware recommender systems. We will also compare our proposal against other algorithms for generating contextual information.

ACKNOWLEDGMENT

This work was supported by the grants 2010/20564-8, 2011/19850-9, 2012/13830-9, 2013/16039-3, 2013/22547-
REFERENCES


