Managing Irrelevant Contextual Categories in a Movie Recommender System

Ante Odić*
University of Ljubljana, Faculty of Electrical Engineering
Tržaška cesta 25
Ljubljana, Slovenia
ante.odic@ldos.fe.uni-lj.si

Marko Tkalčič
Johannes Kepler University
Department for Computational Perception
Altenberger Str. 69
Linz,Austria
marko.tkalcic@jku.at

Andrej Košir
University of Ljubljana, Faculty of Electrical Engineering
Tržaška cesta 25
Ljubljana, Slovenia
andrej.kosir@ldos.fe.uni-lj.si

ABSTRACT

Since the users' decision making depends on the situation the user is in, contextual information has shown to improve the recommendation procedure in context-aware recommender systems (RS). In our previous work we have shown that relevant contextual factors have significantly improved the quality of rating prediction in RS, while the irrelevant ones have degraded the prediction. In this work we focus on the detection of relevant contextual conditions (i.e., values of contextual factors) which influence the users' decision making process. The goals are (i) to lower the intrusion for the end user by simplifying the acquisition process, and (ii) to reduce the sparsity of the acquired data during the contextual modeling. The results showed significant improvement in the rating prediction task, when managing the irrelevant contextual conditions by the approach that we propose in this paper.

Keywords

context-aware, recommender systems, user modeling

1. INTRODUCTION

Over the past decade, employing contextual information in recommender systems (RS) has been a popular research topic. Contextual information is defined as the information that can be used to describe the situation and the environment of the entities involved in such systems [5]. Since users' decision making depends on the situation the user is in, contextual information has shown to improve the recommendation results in context-aware recommender systems (CARS) [1, 3, 10], as well as other personalized services [11].

In this work we follow the terminology described in [4]: *contextual factor* refers to a specific type of contextual information (e.g. weather), *contextual condition* refers to a specific value for a contextual factor (e.g. sunny), and *contextual situation* refers to a specific set of these contextual conditions that describe the context in which the user consumed the item.

In our previous work [10] we have proposed a methodology for detecting the relevancy of contextual factors, and have shown that relevant contextual factors significantly improved the quality of rating prediction in RS, while the irrelevant ones degraded the prediction. Similar results were achieved in [3] by assessing the relevancy of contextual factors.

In this work we focus on the detection of relevant contextual conditions, i.e., the values of contextual factors, which influence the users' decision making process, with the goal of lowering the intrusion for the end user by simplifying the acquisition process, and to reduce the sparsity of the acquired data during the contextual modeling.

1.1 The Problems of Many Contextual Conditions: Sparsity and Acquisition

One of the main problems with contextual factors with many contextual conditions is the sparsity of rating data. For example, let us say a specific user rated 20 items in different contextual situations. For uncontextualized modeling that would be a fair amount of ratings from that specific user. However, let us say some contextual factor contains ten contextual conditions and users ratings are equally distributed across those conditions. That would mean that for each condition we only have two ratings from that user. For this reason it would be better to have a lower number of contextual conditions per contextual factor.

In addition, since the contextual data is often explicitly acquired through questionnaires (e.g. in [2] or [10]), lowering the number of questions and possible conditions shortens the questionnaire. This is important for lowering the amount of time required from users to provide ratings and the associated context.

To summarize, the acquisition and usage of contextual factors with many contextual conditions has two negative sides:

- questionnaire size (effort required from a user)
- sparsity (ratings are distributed in many categories)

Therefore, it would be beneficial to reduce the number of

*Corresponding author.

RecSys'13, October 12–16, 2013, Hong Kong, China.
Paper presented at the 2013 Decisions@RecSys workshop in conjunction with the 7th ACM conference on Recommender Systems.

contextual conditions of the relevant contextual factors.

1.2 Problem Statement

The problem with the reduction of the number of contextual condition is how to select the conditions to remove and how to merge the contextual conditions in order to reduce their number.

By avoiding the relevant contextual conditions we might lose valuable information. Hence, we need to detect irrelevant conditions, identify how they should be merged and handled during the acquisition, and during the training and the preparation of recommendations.

In this article we propose an approach by which we achieve the following goals:

- we identify contextual conditions which should be avoided or merged in questionnaires
- we manage irrelevant categories during training to utilize provided ratings and decrease the sparsity

In the following sections we describe the approach, dataset used and the experimental results.

1.3 Experimental Design

In this subsection we describe the experimental design used in this study. For each contextual variable available in the dataset we do the following steps in order to manage irrelevant categories.

First we do the **contextual-condition-relevancy detection**. At this stage we use statistical testing in order to detect which contextual conditions of a specific contextual factor are irrelevant and do not have the impact on the ratings. We consider a contextual condition to be relevant if the users' behavior (how users rate items) is different for that condition than for other conditions. If the users do not rate items differently for that contextual condition than otherwise, we consider the condition to be irrelevant.

The next step is to determine whether these irrelevant conditions could be merged with the relevant ones. For example, if *rainy weather* would be detected as irrelevant, but *cloudy weather* as relevant, perhaps they could be merged into a combined category *cloudy/rainy weather*. Hence, we call this step the **context-categories-merging determination**. Once the merging possibilities are determined, we may use them for two separate tasks: (i) improving the questionnaire, and (ii) improving the contextualized model of users decisions.

Improving the questionnaire. If in a system, after a sufficient amount of data was collected, it is determined that several contextual-factors' conditions are irrelevant and could be merged with others, the questionnaire used for the data acquisition should be modified. (Similarly, if the data is being collected implicitly through sensors, the acquisition procedure should be modified). In this way the number of questions in the questionnaire could be reduced and thus the time required from users to fill-in these questionnaires would be reduced. However, it might be the case that the merges are too complex to employ them in the questionnaire as we will show in the following sections.

Improving the model. In addition to improving the questionnaire, merges should be employed in the model as well. By using the irrelevant conditions in the model during training, the rating data is being used to train the contextualized parameters which depend on the irrelevant contextual conditions. Instead these ratings should be used for training the parameters that depend on the relevant contextual conditions. Hence, by merging categories we are able to use the rating data for the more meaningful task (which consequently reduces the sparsity of ratings), which would result in a better trained model. We will evaluate this task by comparing the root mean square error (RMSE) of the rating prediction with and without merging of context categories, and the results form random merging of contextual conditions as a baseline.

Figure 1 shows the whole procedure described in the article.

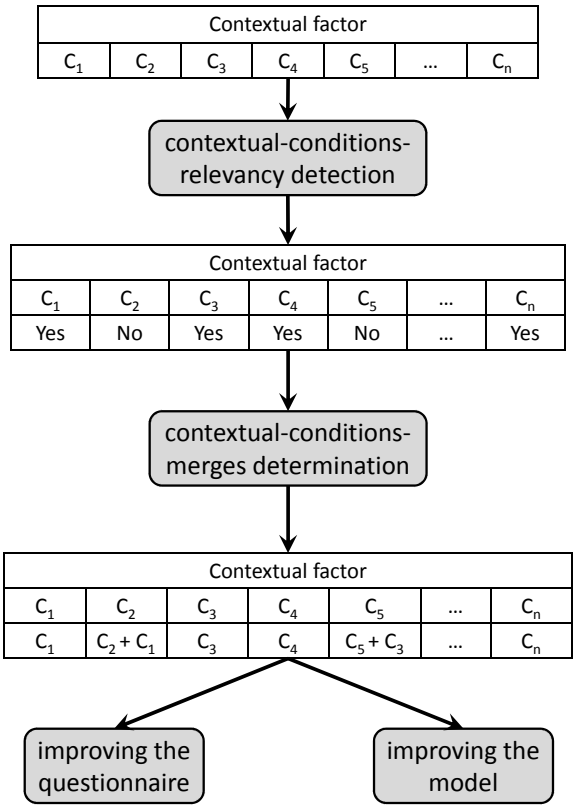

Figure 1: Experimental design. For each contextual variable, the relevancy of categories is detected, merging possibilities are determined and used to improve both the data acquisition and the modeling procedure.

2. MATERIALS AND METHODS

In this section we describe the dataset used in this study and describe each step of the experimental design in more details.

2.1 Dataset

For the purposes of this work we have used the *Context Movie Dataset* (LDOS-CoMoDa), that we have acquired in our previous work [10].

We have created an online application for rating movies which users are using in order to track the movies they watched and obtain the recommendations (`www.ldos.si/recommender.html`). Users are instructed to log into the system after watching a movie, enter a rating for a movie and fill in a simple questionnaire created to explicitly acquire the contextual information describing the situation during the consumption.

The part of the dataset used in this study consists of 1611 ratings from 89 users to 946 items with 12 associated contextual factors. Additional information about our *Context Movies Database* (LDOS-CoMoDa) can be found in [9] and [10].

All the contextual factors and conditions acquired are listed in Table 1.

Table 1: Contextual factors in the LDOS-CoMoDa dataset.

Contextual variable	Description
time	morning, afternoon, evening, night
daytype	working day, weekend, holiday
season	spring, summer, autumn, winter
location	home, public place, friend's house
weather	sunny/clear, rainy, stormy, snowy, cloudy
social	alone, partner, friends, colleagues, parents, public, family
endEmo	sad, happy, scared, surprised, angry, disgusted, neutral
dominantEmo	sad, happy, scared, surprised, angry, disgusted, neutral
mood	positive, neutral, negative
physical	healthy, ill
decision	user's choice, given by other
interaction	first, n-th

2.2 Contextual-Condition-Relevancy Detection

In order to determine if a contextual condition of a specific contextual factor is relevant, we use the **Wilcoxon rank-sum test** in the following way. For each condition (e.g., sunny weather) of a specific contextual factor (e.g., weather), we observe two populations of ratings: ratings associated with that condition only (e.g., sunny weather), and ratings associated with any other condition of the same contextual factor (e.g.,rainy, cloudy, snowy and stormy). We use the **Wilcoxon rank-sum test** to compare these two populations. More specifically, we test the null hypothesis that the ratings from these two populations are sampled from a continuous distributions with equal medians. If we reject the null hypothesis, the medians are different, which means that the users tend to rate items differently during the tested condition (e.g., sunny) compared to the other conditions (e.g. rainy, cloudy, snowy and stormy). If this is the case, we determine that the tested contextual condition is relevant. Otherwise we determine that since there is no difference in ratings, such condition has no impact and is thus irrelevant. The Wilcoxon rank-sum test was chosen over the t-test because the compared samples were not normally distributed.

The described approach was done on the population level, i.e., on the data from the whole population and not for each user separately. Hence, contextual conditions are detected as relevant or irrelevant with regards the whole population of users.

2.3 Contextual-Condition-Merges Determination

Once the irrelevant conditions of each contextual factor are detected, we proceed to merge them with relevant categories. In order to determine which categories should be merged, we compare the distribution of ratings for each irrelevant condition with the distribution for each relevant condition separately (e.g. sunny vs. rainy, sunny vs. cloudy, sunny vs. snowy and sunny vs. stormy). Once again, this is tested with the **Wilcoxon rank-sum test**. In this case, if the test determined that the medians of the ratings distributions for the irrelevant and relevant conditions are equal, we determine that these conditions can be merged. This is because there is no difference in rating when users were in these two separate conditions.

However, the proposed methodology might yield a type of error during merges. It might occur that we determine the two conditions could be merged when in fact they should not. This exception might occur if the distributions were similar, yet, for different user-item pairs ratings were drastically different on different contextual conditions. This is an open issue we plan to address in the future work.

2.4 Merging Contextual Conditions

Once we determine which contextual condition should be merged we implement merging in two separate tasks: (i) improving the questionnaire and (ii) improving the model.

2.4.1 Merging in Questionnaires

In our system we acquire the contextual information explicitly through questionnaire. Hence, we implement merging in the questionnaire by modifying the list of possible contextual conditions users choose from. For example, in our system we have the contextual factor *season*, which contains the following contextual conditions: *spring*, *summer*, *autumn*, *winter*. Let us say that we have determined *summer* to be an irrelevant condition and that it should be merged with the relevant condition *autumn*. We would simply change possible answers in the questionnaire into: *spring*, *summer/autumn*, *winter*. In this way we lower the amount of possible answers, and stop associating ratings with irrelevant contextual condition. Of course, if possible, a new name for the combined condition could be used in the questionnaire.

If contextual information would be acquired implicitly through sensors, merging would be implemented in the step of processing sensor data into contextual conditions.

2.4.2 Merging during Modeling

In this study we used the contextualized matrix factorization algorithm for modeling the interaction between the users and the movie items. Matrix factorization (MF) is a latent-factor model that is widely used in RS ([8, 3, 6, 7]). We implement the contextualization by making users' rating biases context dependent as in [10].

The contextualized users' biases with the matrix factorization **(CUB-MF)** approach uses the contextual information for the contextualized users' biases. Only the users' biases are context dependent. This approach follows the idea that the users' rating behaviour is different on different occasions. The matrix factorization in CUB-MF was made using the

following equation:

$$\hat{r}\,(u,h) = \mu + b_h + b_u(c) + \vec{q}_h^{\,T} \cdot \vec{p}_u, \qquad (1)$$

where $\hat{r}\,(u,h)$ is the predicted rating for user u and item h, $\vec{q}_h$ is the item's latent-feature vector, $\vec{p}_u$ is the user's latent-feature vector. The user's bias b_u and the item's bias b_h measure the deviations of the user's u and the item's h ratings from the rating average μ.

To inspect the impact of merging contextual conditions of contextual factors on the rating prediction, we trained our model for each contextual factor separately, i.e. using only a single contextual factor at the time.

The standard way, of training (without merging) the contextualized model is done in the following way: the algorithm loops through all the ratings in the training set, and calculates the prediction error $e(u,h,c) = r(u,h,c) - \hat{r}(u,h,c)$ for each predicted rating $\hat{r}(u,h,c)$ and real rating $r(u,h,c)$, for user u, item h and contextual condition c. Among other uncontextualized parameters, we modify the contextualized user's u bias by the equation:

$$b_u(c) \leftarrow b_u(c) + \gamma \cdot (e(u,h,c) - \lambda \cdot b_u(c)). \qquad (2)$$

Hence, if the contextual condition was, for example *summer*, we would update $b_u(sunny)$.

When we implement merging during modeling, for each calculated error of prediction, we update the contextualized parameters of all merged conditions, if such exist. Therefore, if, for example, the contextual condition *summer* has to be merged with the condition *autumn*, we would use $e(u,i,summer)$ to update $b_u(sunny)$ and $b_u(autumn)$ simultaneously. In this way we reduce the negative impact of sparsity by utilizing ratings associated with irrelevant conditions to train parameters contextualized by the relevant ones. In addition, during training, for each calculated error of prediction, we also train the uncontextualized users' biases. Once the model is trained, on the testing set, the uncontextualized users' biases are used to predict the ratings associated with the irrelevant contextual conditions. In this way, the algorithm simply avoids the contextualized rating prediction in the case of the irrelevant contextual condition.

2.5 Random Merging as a Baseline

In order to test the positive impact of our procedure for detecting irrelevant contextual conditions, and determining merges, it is important to compare the results from our approach with the fair baseline. It could be that the improvement in the rating prediction is not due to our merging technique, but due to any type of merging simply because we lower the sparsity. In another words, it is important to test if we would get equally improved results by randomly merging several conditions.

Therefore we have implemented a random merging method in the following way: for every contextual factor we count the exact number of irrelevant conditions and determined merges, and select the same amount of random conditions and random merges. In this way we replicate the same amount of merges but select the conditions to be merged randomly.

The results for our approach and the random merges are achieved on 10 different folds.

3. RESULTS

In the cases of the *time*, *daytype* and *location* contextual factors, all conditions were found irrelevant, hence no merges are possible. In the cases of the *decision*, *interaction*, and *physical* contextual factors, all conditions were found relevant, hence no merges are needed. For the remaining contextual factors, table 2 contains the results of the contextual-condition-relevancy detection, and merges determination.

The figures 2, 3, 4, 5, 6 and 7 show the results of the matrix factorization rating prediction.

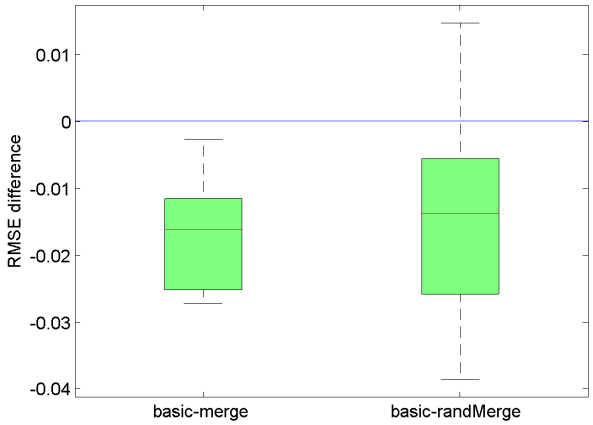

Figure 2: Rating prediction results for dominant emotion.

On each figure, boxplots are presented: one from our merging method (*merge*) and the second one from the random merge baseline (*randMerge*). Both boxplots represent the RMSE difference between the basic model without merging (*basic*), and the *merge* and *randMerge* approaches. Therefore, if the result is above zero, the merging approach performed better (lower RMSE) than the basic approach without merging.

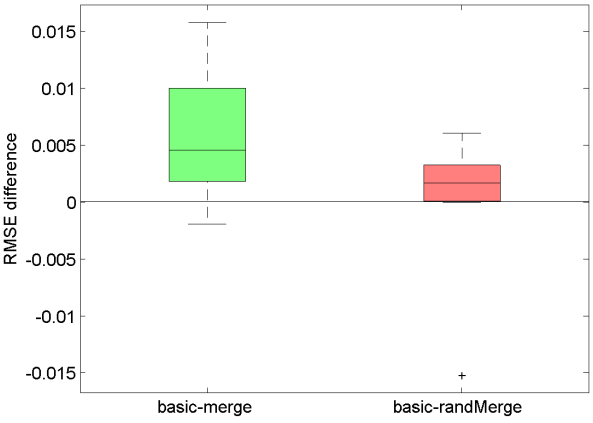

Figure 3: Rating prediction results for end emotion.

The **Wilcoxon signed-rank test** was used to test the statistical significance of the differences between basic and merging approaches. If the difference was statistically significant the box plot is colored green, otherwise it is colored red.

Table 2: Results of the contextual-condition-relevancy detection, and merges determination.

SEASON		
condition	**relevancy**	**merges**
spring	yes	
summer	no	autumn
autumn	yes	
winter	yes	

WEATHER		
condition	**relevancy**	**merges**
sunny	no	
rainy	no	
stormy	no	snowy
snowy	yes	
cloudy	no	

SOCIAL		
condition	**relevancy**	**merges**
alone	yes	
partner	yes	
friends	no	alone partner family
colleagues	no	alone partner family
parents	no	alone family
public	no	alone partner family
family	yes	

END EMOTION		
condition	**relevancy**	**merges**
sad	yes	
happy	yes	
fear	no	sad happy surprised
surprised	yes	
angry	yes	
disgusted	yes	
neutral	yes	

DOMINANT EMOTION		
condition	**relevancy**	**merges**
sad	yes	
happy	yes	
fear	no	sad happy surprised
surprised	yes	
angry	no	neutral
disgusted	yes	
neutral	yes	

MOOD		
condition	**relevancy**	**merges**
positive	yes	
neutral	yes	
negative	no	neutral

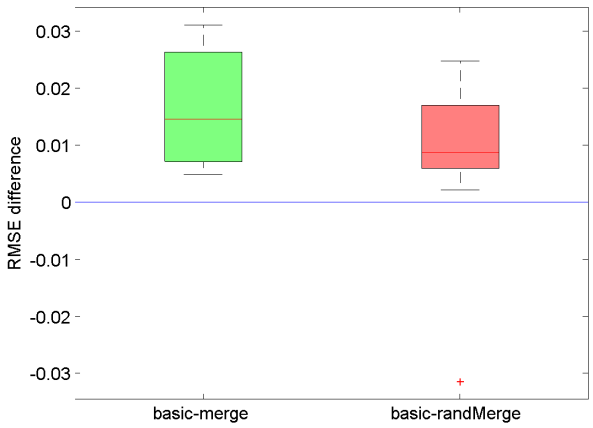

Figure 4: Rating prediction results for mood.

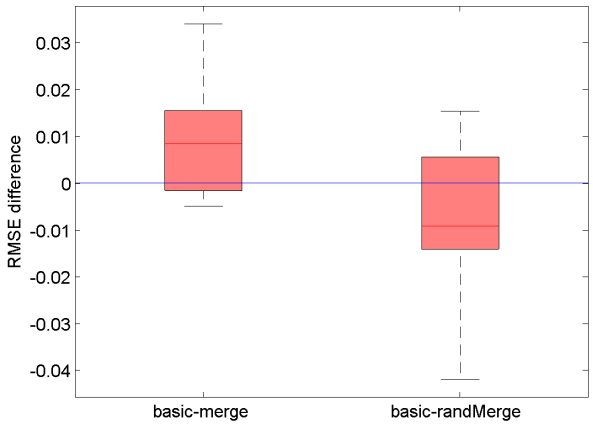

Figure 5: Rating prediction results for season.

3.1 Discussion

In the previous section we could observe different results for different contextual factors. It is interesting to note that contextual factors for which all the contextual conditions were detected as irrelevant (*time*, *daytype* and *location*) are those that were detected irrelevant themselves in our previous work [10]. Similarly, the contextual factors for which all the contextual conditions were detected as relevant (*decision*, *interaction*, and *physical*) are those that were detected as relevant themselves in our previous work. Therefore, we might conclude that such contextual factor for which all the contextual conditions are detected as irrelevant, can be observed as irrelevant and left out from the contextualized modeling altogether. For the remaining contextual factors we summarize the results in Table 3.

Implementing merges in questionnaire can be easily achieved for *season*, *weather* and *mood*, by simply merging conditions between possible answers. However, for *social* contextual condition, as it is shown in Table 2, there are conflicts which prevent us for merging. For example, the condition *parents* can be merged with *alone* and *family*, but not with *partner*, as it is the case with the conditions *friends*, *colleagues* and *public*.

Furthermore, for *end emotion* and *dominant emotion*, the irrelevant condition *fear* can be merged with multiple condi-

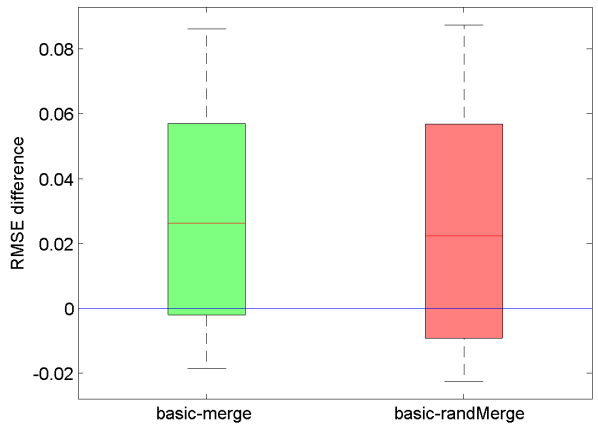

Figure 6: Rating prediction results for social.

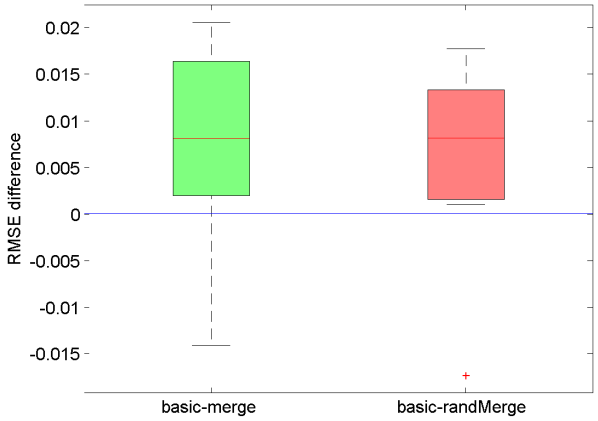

Figure 7: Rating prediction results for weather.

tions (*sad*, *happy*, *surprised*), however each of them is relevant and should be used alone as it is. Therefore, an opened issue remains - how such cases should be handled in questionnaires.

By implementing the proposed procedure for the detection of irrelevant contextual categories, and the proposed way to manage merges during modeling, we achieved significantly better results than without merging for the contextual factors *weather*, *social*, *end emotion* and *mood*. For the contextual factor *season* we achieved an improvement, however it was not statistically significant (Figure 5). In each case our procedure outperformed random-merging baseline, which did not lead to significantly improved results in any case. However, even in the case of random merging there is tendency towards better results with fewer conditions which confirms our assumption from the introduction: many contextual conditions have a large impact on the sparsity of ratings in the contextualized models.

The only contextual factor for which we observed unexpected results is the *dominant emotion*. In this case we achieved significantly worse results for both our approach and the random-merging baseline. We believe that this is an interesting open issue that we plan to address further in the future.

Table 3: Summary of the results. The table tells whether there is an improvement in the questionnaire or in the model, for each contextual factor separately.

	improvement	
context	**questionnaire**	**rating prediction**
season	yes	no
weather	yes	yes
social	no	yes
endEmo	?	yes
domEmo	?	no
mood	yes	yes

4. CONCLUSION AND FUTURE WORK

In this paper we proposed a procedure for detecting the relevancy of contextual conditions and how to manage such conditions by merging them with relevant ones. We implemented merging of contextual conditions on the questionnaire for acquiring contextual data, and into contextualized modeling based on matrix factorization. The results showed significantly improved results by our method, except in the case of one specific contextual factor.

For the future work we plan on researching further why anomalies can occur and how to predict and avoid them. Also, we are interested in solving conflicts described in this paper regarding the implementation of merges in questionnaires.

5. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, June 2005.

[2] L. Baltrunas, B. Ludwig, S. Peer, and F. Ricci. Context-Aware Places of Interest Recommendations for Mobile Users. *Design, User Experience, and Usability. Theory, Methods, Tools and Practice*, pages 531–540, 2011.

[3] L. Baltrunas, B. Ludwig, S. Peer, and F. Ricci. Context relevance assessment and exploitation in mobile recommender systems. *Personal and Ubiquitous Computing*, pages 1–20, June 2011.

[4] V. Codina, F. Ricci, and L. Ceccaroni. Semantically-enhanced pre-filtering for context-aware recommender systems. In *Proceedings of the 3rd Workshop on Context-awareness in Retrieval and Recommendation*, pages 15–18. ACM, 2013.

[5] A. Dey and G. Abowd. Towards a better understanding of context and context-awareness. *Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*, pages 304–307, 1999.

[6] Z. Gantner, S. Rendle, and L. Schmidt-Thieme. Factorization Models for Context- / Time-Aware Movie Recommendations Encoding Time as Context. In *Proceedings of the Workshop on Context-Aware Movie Recommendation*, pages 14–19, 2010.

[7] B. Hidasi and D. Tikk. Enhancing matrix factorization through initialization for implicit

feedback databases. In *Proceedings of the 2nd Workshop on Context-awareness in Retrieval and Recommendation*, CaRR '12, pages 2–9, New York, NY, USA, 2012. ACM.

[8] Y. Koren. Factorization Meets the Neighborhood : a Multifaceted Collaborative Filtering Model. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434. 2008.

[9] A. Košir, A. Odić, M. Kunaver, M. Tkalcic, and J. Tasic. Database for contextual personalization. *ELEKTROTEHNIŠKI VESTNIK*, 78(5):270–274, 2011.

[10] A. Odić, M. Tkalčič, J. F. Tasič, and A. Košir. Predicting and detecting the relevant contextual information in a movie-recommender system. *Interacting with Computers*, 25(1):74–90, 2013.

[11] F. Toutain, A. Bouabdallah, R. Zemek, and C. Daloz. Interpersonal Context-Aware Communication Services. *IEEE Communications Magazine*, (January):68–74, 2011.