Proceedings of the of the 17th Conference on Computer Science and Intelligence Systems pp. 67–71
DOI: 10.15439/2022F127
ISSN 2300-5963 ACSIS, Vol. 30

Improving Re-rankCCP with Rules Quality Measures

Piotr Jezusek
Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology, Gdańsk, Poland

Aleksandra Karpus
Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology, Gdańsk, Poland
Email: alekarpu@pg.edu.pl

***Abstract*—Recommender Systems are software tools and techniques which aim at suggesting new items that may possibly be of interest to a user. Context-Aware Recommender Systems exploit contextual information to provide more adequate recommendations. In this paper we described a modification of an existing contextual post-filtering algorithm which uses rules-like user representation called Contextual Conditional Preferences. We extended the algorithm by taking into account rules quality measures while recommending items to a user. We proved that this modification increases the quality of recommendations, measured with *precision*, *recall* and *nDCG*, and has no impact on the execution time of the original algorithm.**

I. Introduction

RECOMMENDER Systems (RS) were created as a response to the information overload problem, which we suffer from nowadays. These software tools and techniques aim at suggesting new items that may possibly be of interest to a user [1]. An item could be a movie (Netflix), a song (Pandora), a job (LinkedIn) or a friend (Facebook). In everyday life we interact with RS when we search for information using Google or when we buy something through the Internet.

Context-aware RS (CARS) are a particular category of RS which exploit contextual information to provide more adequate recommendations [2]. For example, a movie recommendation for a Saturday evening with your friends should be different from one suggested for a Sunday afternoon with your family. It has been proven that adding contextual information in the process of recommendation can highly increase prediction accuracy and user satisfaction [3]. Adomavicius and Tuzhilin [2] distinguish tree main types of context-aware recommender systems, i.e. contextual pre-filtering, contextual post-filtering and contextual modeling. The paradigms differ in the way they incorporate context in the recommendation process. More details are given in Section II.

Karpus et al. [4] proposed a context-aware re-ranking algorithm (re-rankCCP) which utilizes user model called Contextual Conditional Preferences (CCPs). CCPs are special kind of rules which are learned from past user ratings and used to reorder items in a primary recommendation list. This method seems promising in making user explanations for recommendations due to the use of rules that are easy to understand by a human. However, this solution has a big disadvantage. While using CCP, an algorithm only checks its relevance to a current user context, not taking into consideration the quality of an induced preference. Thus, better CCPs can be omitted during reordering what would lead to a reduction in the recommendation accuracy and user satisfaction.

In this paper we propose a method for determining the CCP quality using rules quality measures, i.e. *coverage*, *support* and *confidence* and apply it in the modification of the re-rankCCP. We proved that this modification increases the quality of recommendations, measured with *precision*, *recall* and *nDCG*, and has no impact on the execution time of the re-rankCCP. The main contribution of this paper can be summarized as follows:

- We propose a way to measure quality of CCP with usage of rules quality measures.
- We improve re-rankCCP algorithm to take into account quality of CCPs while generating recommendations.
- We compare effectiveness of modified re-rankCCP on 2 baseline algorithms, 3 rules quality measures and 4 aggregate functions and show that there is a best configuration for the dataset used.

The rest of the paper is organized as follows. Related work and basic re-rankCCP are described in Sections II and III, respectively. Section IV provides technical details of the proposed modification and is followed by a description of the dataset used. Section VI introduces evaluation method while obtained results are presented in Section VII. Conclusions close the paper.

II. Related Work

Adomavicius and Tuzhilin [2] distinguish three main types of CARS, i.e. contextual pre-filtering, contextual post-filtering and contextual modeling. The paradigms differ in the way they incorporate context in the recommendation process.

In contextual pre-filtering, we first do selection of ratings by taking only relevant context into account. Thus, we filter an initial set of ratings and return the contextualized data. After this preparation any known two-dimensional recommendation algorithm could be used to predict user preferences. Baltrunas et al. [5] introduced *micro profiles* which split a user profile into partitions depending on the values of context parameters. They showed that usage of such *micro profiles* gave a significant improvement in the prediction accuracy in the movie domain while considering time as a context variable. Pre-filtering approach which utilizes ontological user profiles was proposed by Karpus et al. [6]. Each user profile consists of many ontologies representing user preferences in

different context and domain. Ferdousi et al. [7], [8] tried to find a correlation between ratings and context in which they were given. They proposed a new context representation based on the Pearson Correlation Coefficient as well as a new pre-filtering technique based on this representation.

Contextual post-filtering applies context after traditional recommendation process. It means that from a predicted set of recommendations we select just those that match current user context. Bahramian et al. [9] proposed a new context-aware tourism recommender system based on an ontology approach where a spreading activation technique is used to contextualize user preferences and learns the user profile dynamically. Negre et al. [10] introduced a context-aware recommender system based on a contextual post-filtering for OLAP queries, where queries recommended by a classic log-based recommender system were contextualized.

Contextual modeling differs radically from previously described paradigms. In this kind of recommenders we incorporate a context in a prediction model. The recommendations are achieved directly from the model, taking into account current user-context situation. Iqbal et al. [11] introduced Kernel Context Recommender System, which is a flexible, fast, and accurate kernel mapping framework that recognizes the importance of context and incorporates the contextual information using kernel trick while making predictions. Zheng et al. [12] proposed method that combines context-aware and multi-criteria recommender systems. They evaluated their solution on an educational data and an extended TripAdvisor dataset. Authors tested different approaches for incorporating context in the recommendation process.

In the recent years, an application of artificial neural networks in CARS is getting more and more attention [13], [14], [15]. Hildebrandt et al. [15] proposed NECTR, a novel recommender system based on a tensor factorization model and an autoencoder-like neural network. A Deep Learning based model which learns customer similarity from the sequence to sequence similarity as well as item to item similarity by considering all features of the item, contexts, and rating components was introduced by Kala et al. [14]. The method uses Dynamic Temporal Warping distance measure for dynamic temporal matching and 2D-GRU (Two Dimensional-Gated Recurrent Unit) architecture.

III. Background - re-rankCCP algorithm

Contextual Conditional Preferences (CCPs) were introduced to provide compact and context-aware representation of user interests for RS [16], [4]. CCP is an expression of the form:

$$(\gamma_1 = c_1) \wedge \ldots \wedge (\gamma_n = c_n) \mid (\alpha_1 = a_1) \succ (\alpha_1 = a_1') \wedge \ldots \wedge (\alpha_m = a_m) \succ (\alpha_m = a_m')$$

with γ_i being contextual variables, α_i item attributes, and $c_1, ..., c_n,\ a_1, a_1', ..., a_m, a_m'$ being exact values of these parameters. Symbol $\succ$ denotes a preference relation, e.g. $x \succ y$ means that someone prefers x over y.

The above CCP is read as *given the context* $(\gamma_1 = c_1) \wedge \ldots \wedge (\gamma_n = c_n)$ *I prefer* a_1 *over* a_1' *for* α_1 *and* $\ldots$ *and* a_m *over* a_m' *for* α_m. An example of the CCP is shown below.

$$\text{time of day} = afternoon \wedge \text{companion} = with\ children \mid \text{genre} \in \{animated, family\} \succ \text{genre} \in \{thriller\}$$

It means that for a given context, i.e. in the afternoon and the company of children, a user prefers movies that belong to the genre "animated" or "family" to those with category "thriller".

CCPs can be learned from explicit user ratings [17]. In order to elicit preference relations the dataset containing ratings, contextual parameters and item features is split into two parts, i.e. positive and negative, based on the value of the ratings. Then, both subsets are divided into smaller sets containing all of the contextual information and one of the item features. Such prepared data are an input for the `Prism`[18] algorithm. Final CCPs are obtained by merging rules with the same context.

An algorithm for generating a list of top k recommendations with CCPs, the re-rankCCP, was introduced by Karpus et al. in [4]. We describe it below.

For a certain user and his current context, first we generate a primary list of top m recommendations with some existing non-context-aware algorithm, e.g., UserKNN. The value of m has to be significantly greater than k, where k is the number of the recommendations in the final list. Then we have to find the best CCPs that will be further used in the reshuffling process.

The best CCPs are those which are most similar to the considered context. In order to count a contextual similarity between a CCP p and a current user context $ctx(u)$ we used the following measure:

$$sim\,(p, ctx(u)) = \sum_{(\gamma_i, c_i) \in p} overlap(ctx(u), (\gamma_i, c_i)) \ . \quad (1)$$

We also used the overlap function defined as:

$$overlap(ctx(u), (\gamma_i, c_i)) = \begin{cases} 1 & (\gamma_i, c_i) \in ctx(u); \\ 0.5 & c_i = -1; \\ 0 & otherwise. \end{cases} \quad (2)$$

The overlap function returns 1 when we are sure that the pair (γ_i, c_i) is contained both in the contextual part of p and in the current user context $ctx(u)$. When it is uncertain, i.e. when the value c_i for the dimension γ_i is equal to -1 (the unknown value), it returns 0.5. Otherwise 0 is returned. Note that the current user context $ctx(u)$ is also a set of pairs (γ_i', c_i'), i.e. the name of the contextual variable and its value.

For each item in the primary recommendations list and each best CCP we have to compute how much an item i satisfies a CCP p. For this purpose, we have to use the *satisfiability* measure:

$$sat(i, p) = \frac{\sum_{\alpha \in a(p)} \left(sim(v_\alpha^m(p), v_\alpha(i)) - sim(v_\alpha^l(p), v_\alpha(i))\right.}{|a(p)|} \ ,$$

where sim denotes Jaccard similarity, α is the name of an item feature, $a(p)$ is the set of item attributes considered in

the CCP p, $v_\alpha(i)$ is the set of values of an attribute α for an item i. Similarly $v^m_\alpha(p)$ and $v^l_\alpha(p)$ denote the sets of values of an attribute α for a CCP p on both sides of the preference relation - m stands for *more preferred* and l for *less preferred*.

The *satisfiability* measure represents the difference between item similarities to the both sides of the CCP preference relation, i.e. the similarity to the most preferred part minus the similarity to the less preferred part. In this way we reward items that fit best to user preferences and penalize items that have features that user does not like. The size of a set of item attributes serves as a normalization factor. Thus, regardless of the number of item features, the value of *satisfiability* is always between 0 and 1.

The next step is to order the primary recommendations list according to the value of average *satisfiability* of the best CCPs. The last part is to cut off unneeded items from resulting recommendations list to receive top 5, top 10 or other top *k* ranking.

IV. Algorithm Modification

One of the key parts of the re-rankCCP algorithm is a selection of best CCPs for current user context based on the similarity measure from Equation 1. However, this measure does not take into account the quality of CCPs. Consequently, recommendation could be made based on less important user preferences. Therefore, we replaced the similarity with a weighted similarity sim_w:

$$sim_w(p, ctx(u)) = q(p) sim\,(p, ctx(u)) \quad ,$$

where sim is the similarity measure from Equation 1 and $q(p)$ is a quality of a CCP p. Now, we need to define the quality of a CCP.

CCPs can be generated from rules induced with Prism algorithm. Thus, we decided to use rules quality measures, like *coverage* or *support*, to define quality of CCP. However, one CCP is created using many different rules. Therefore, we have to decide how to reasonably aggregate many rules quality values into one value characterizing CCP's quality.

In this paper we tested four aggregate functions, which we found the most reasonable, i.e. *minimum*, *maximum*, *sum* and *average*. The last one seems the most obvious because it simply takes quality values from all used rules and returns one normalized value for a CCP. We also obtain standardized quality for the first two functions which simply take the worst and the best rule quality value, respectively. The *sum* function additionally reflects the quantity of rules that are used for creation of a CCP. The more good rules were used, the higher the quality of a CCP would be.

We decided to apply three commonly used rules quality measures, namely: *coverage*, *support* and *confidence* [19]. For this purpose, we had to slightly modify an algorithm for CCPs extraction to compute those measures. However, this algorithm is independent from the re-rankCCP. We also extended a CCP representation to contain information about its quality. Figure 1 shows modified CCP from the above example in the JSON format. The rest of the re-rankCCP remains the same.

```
{"CCPs":[
  { "context":{"time of day":"afternoon", "companion":"with children"},
    "MorePrefered":{"genre":["animated","family"]},
    "lessPrefered":{"genre":["thriller"]},
    "coverage":0.4924},
  ...
  ]}
```

Fig. 1. An example CCP in the JSON format with information about rule quality measured with the coverage.

V. Dataset

We performed experiments on the same dataset as authors of the original re-ranking algorithm, i.e. `LDOS-CoMoDa` dataset. The `LDOS-CoMoDa` dataset [20] was collected by a web application that enables contextual rating of a movie just after watching it. The dataset consists of 2296 ratings given by 121 users to 1232 items. It contains 30 variables among which 12 are contextual parameters. Other variables are basic information about user (user id, `age`, `sex`, `city` and `country`), a rating in a 5-star scale (higher values denote higher preference) and content information about multiple item dimensions (item id, `director`, `country`, `language`, `year`, 3 main `genres`, 3 main `actors` and `budget`). Unknown values are denoted by "-1".

We chose users who rated at least 5 items. Then, we randomly selected 20% of items rated by each of these users to be included in the test set. The remaining data constitute the training set.

VI. Evaluation method

We re-implemented in Python the re-rankCCP algorithm, which was originally implemented in Java. We also performed new training and test sets split. Thus, because of the randomness of the split, we have different data in those sets than in the previous papers [4], [16].

The re-rankCCP is a post-filtering technique which means that it needs other algorithm to work. We decided to test two known methods, i.e. Bayesian Personalized Ranking (BPR)[21] and User *k* Nearest Neighbors (UserKNN)[22]. We had several reasons for this choice. First of all, inventors of re-rankCCP obtained the most promising results with BPR algorithm. Second of all, UserKNN was one of the most (or even the most) popular method in the field of RS. Last but not least is the way how these algorithms treat missing data. BPR tries to minimize its negative impact on the prediction accuracy, while UserKNN completely ignores missing data. Hereby, we obtained a representative sample of base algorithms.

For the re-rankCCP and base algorithms we had to set up some parameters. UserKNN used 50 neighbors to compute recommendations. Base algorithms generate lists of top 100 items while re-rankCCP produces the top 10 list. Rating greater than 3 is considered positive. We decided to choose three commonly used measures of recommendations quality, i.e. *precision*, *recall* and *nDCG* [23].

In addition to the impact of the modification on the quality of recommendations, we wanted to check its impact on the algorithm execution time. Since we slightly modified CCP representation, our method does not affect the time

of generating recommendations. However, it could have impact on the time needed to induce CCPs, since it is where the CCP quality is computed. In order to check it we performed an experiment for which we prepared 4 datasets from the LDOS-CoMoDa. The datasets consists of 3000, 5000, 7000 and 10000 rows respectively. For each dataset we performed CCPs generation for the re-rankCCP algoritm and its modifications. We collected results with *%time* function which is available in *ipython* environment.

VII. Results

Table I shows values of *precision*, *recall* and *nDCG* obtained for different configurations of algorithms, rules quality measures and aggregate functions during our experiments. The best results for each algorithm and rules quality measure is marked with bold (locally best result), while the best results for each algorithm/base algorithm is marked with underline (globally best result considering division into two groups: BPR and UserKNN). It should be noticed that re-rankCCP always improves *precision*, *recall* and *nDCG* of its base algorithm. Nonetheless, re-rankCCP performs weaker than its modifications with rules quality measures.

For most of configurations of algorithms and rules quality measures, the best results were obtained by *minimum* and *average* functions. The first function was the best for *support* and *confidence*, irrespective of a base algorithm, while the latter works well with *coverage* and *support* on re-rankCCP with BPR algorithm. An exception appears in re-rankCCP with UserKNN algorithm and *coverage* measure. The best results for this configuration was obtained by the *maximum* function. The best results for *minimum* and *average* functions should not be surprising. While using *minimum* function, we assure that all other rules used to induce a CCP have greater quality values than the resulting value. We obtain similar effect for the *average*. On the contrary, for the *maximum* we could choose preference which is generated from rules from which one is strong and all others are weak. The same bad effect can happen for *sum* function. Modified re-rankCCP will prefer a CCP from many weak rules than a CCP from two strong rules. The smallest improvement in modified re-rankCCP was obtained with the *coverage* for both base algorithms. It can be justified by the fact that the *coverage* is not a proper quality measure for rules since it considers only antecedent of a rule.

To check the statistical significance of obtained results we performed Wilcoxon signed rank test with $\alpha = 0.05$. For the re-rankCCP with BPR as a baseline algorithm two results were statistically insignificant, i.e. for *coverage-minimum* and *confidence-maximum* pairs with p-value equals to 0.4375 and 0.5282, respectively. The re-rankCCP with UserKNN as a baseline and *support* measure has almost all results insignificant, i.e. for *maximum*, *average* and *sum* functions with p-values equal to 0.5745, 0.0625 and 0.1563 respectively. All other reported results are statistically significant.

Considering results presented in Table I and their statistical significance, we can conclude that objectively the best improvement to the re-rankCCP is obtained using the *support* measure for rules quality and the *minimum* function for an aggregation. It should be noticed that UserKNN performs pretty weak on LDOS-CoMoDa dataset. This could be because of the data sparsity.

Table II shows times of generating CCPs for the re-rankCCP algorithm and its modification with the coverage. We obtained very similar results for support and confidence measures which is why we omitted it here. The differences in execution times are negligible. Thus, we can conclude that our modification does not increase execution time of re-rankCCP, and improves the quality of recommendations.

VIII. Conclusions

In this paper we proposed a way for measuring the CCP quality using rules quality measures and aggregate functions. To the best of our knowledge, this is the first attempt to compute a CCP quality. We also improved re-rankCCP algorithm by incorporating quality of CCPs into recommendation process and proved that this modification outperforms the re-rankCCP as well as both baseline algorithms, i.e. BPR and UserKNN. We compared its effectiveness on two baseline algorithms, three rules quality measures, i.e. *coverage*, *support* and *confidence*, and four aggregate functions, i.e. *minimum*, *maximum*, *sum* and *average*. Our experiments showed that the *support* measure aggregated with the *minimum* function is the best configuration for computing the CCP quality on LDOS-CoMoDa dataset. However, more experiments on other datasets and with more baseline algorithms are needed to check if these results could be generalized.

References

[1] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich, *Recommender Systems: An Introduction*, 1st ed. New York, NY, USA: Cambridge University Press, 2010.

[2] G. Adomavicius and A. Tuzhilin, in *Handbook on Recommender Systems*, S. B. Ricci F., Rokach L. and K. P. B., Eds. Springer, 2011, ch. Context-Aware Recommender Systems, pp. 217–256.

[3] M. Kristoffersen, S. Shepstone, and Z.-H. Tan, "The importance of context when recommending tv content: Dataset and algorithms," *IEEE Transactions on Multimedia*, vol. 22, no. 6, pp. 1531–1541, 2020.

[4] A. Karpus, T. di Noia, and K. Goczyla, "Top k recommendations using contextual conditional preferences model," in *Proceedings of the 2017 Federated Conference on Computer Science and Information Systems, FedCSIS 2017, Prague, Czech Republic, September 3-6, 2017.*, M. Ganzha, L. A. Maciaszek, and M. Paprzycki, Eds., 2017, pp. 19–28. [Online]. Available: https://doi.org/10.15439/2017F258

[5] L. Baltrunas and X. Amatriain, "Towards time-dependant recommendation based on implicit feedback," in *Proceedings of 1st Workshop on Context-Aware Recommender Systems*, 2009.

[6] A. Karpus, I. Vagliano, and K. Goczyła, "Serendipitous recommendations through ontology-based contextual pre-filtering," *Communications in Computer and Information Science*, vol. 716, pp. 246–259, 2017.

[7] Z. V. Ferdousi, D. Colazzo, and E. Negre, "Correlation-based prefiltering for context-aware recommendation," in *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, March 2018, pp. 89–94.

[8] Z. V. Ferdousi, D. Colazzo, and E. Negre, "Cbpf: Leveraging context and content information for better recommendations," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11323 LNAI, pp. 381–391, 2018.

[9] Z. Bahramian, R. Abbaspour, and C. Claramunt, "A context-aware tourism recommender system based on a spreading activation method," K. F. Samadzadegan F., Ed., vol. 42, no. 4W4. International Society for Photogrammetry and Remote Sensing, 2017, pp. 333–339.

TABLE I
RESULTS OBTAINED FOR DIFFERENT ALGORITHMS, RULES QUALITY MEASURES AND AGGREGATE FUNCTIONS.

Algorithm	Base algorithm	Rules quality measure	Aggregate function	Precision	Recall	nDCG
re-rankCCP	BPR	coverage	maximum	0.06043	0.03444	0.19368
			minimum	0.05468	0.03273	0.19309
			average	**0.07098**	**0.04173**	**0.20135**
			sum	0.05875	0.03922	0.18690
		support	maximum	0.08129	0.03909	0.27293
			minimum	**0.10408**	0.04610	0.30538
			average	0.09185	**0.04705**	**0.30710**
			sum	0.08417	0.04142	0.25481
		confidence	maximum	0.05707	0.03022	0.20156
			minimum	**0.08729**	**0.04037**	**0.26302**
			average	0.07266	0.03042	0.23508
			sum	0.07242	0.03370	0.22983
re-rankCCP	BPR			0.05366	0.02668	0.18045
BPR				0.04508	0.02323	0.17693
re-rankCCP	UserKNN	coverage	maximum	**0.01847**	**0.01161**	**0.09903**
			minimum	0.01703	0.01004	0.08884
			average	0.01583	0.00965	0.08128
			sum	0.01703	0.01004	0.08826
		support	maximum	0.02038	0.00856	0.08085
			minimum	**0.03381**	**0.01290**	**0.17228**
			average	0.01894	0.00797	0.08139
			sum	0.02062	0.00830	0.08924
		confidence	maximum	0.02662	0.01085	0.12977
			minimum	**0.03237**	**0.01238**	**0.14131**
			average	0.02542	0.01032	0.12725
			sum	0.02758	0.01076	0.13002
re-rankCCP	UserKNN			0.01570	0.00849	0.07800
UserKNN				0.01103	0.00719	0.03580

TABLE II
TIMES OF GENERATING CCPS FOR THE RE-RANKCCP ALGORITHM AND ITS MODIFICATION WITH COVERAGE.

	Execution time	
Number of rows	With coverage	Without coverage
3000	26.5 s	26.5 s
5000	41.1 s	41.2 s
7000	59.6 s	59.7 s
10000	1min 24s	1min 24s

[10] E. Negre, F. Ravat, and O. Teste, "Olap queries context-aware recommender system," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11030 LNCS, pp. 127–137, 2018.

[11] M. Iqbal, M. Ghazanfar, A. Sattar, M. Maqsood, S. Khan, I. Mehmood, and S. Baik, "Kernel context recommender system (kcr): A scalable context-aware recommender system algorithm," *IEEE Access*, vol. 7, pp. 24 719–24 737, 2019.

[12] Y. Zheng, S. Shekhar, A. A. Jose, and S. K. Rai, "Integrating context-awareness and multi-criteria decision making in educational learning," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, ser. SAC '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 2453–2460.

[13] J. Manotumruksa, "Deep collaborative filtering approaches for context-aware venue recommendation," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 1383. [Online]. Available: https://doi.org/10.1145/3077136.3084159

[14] K. Kala and M. Nandhini, "Gated recurrent unit architecture for context-aware recommendations with improved similarity measures," *KSII Transactions on Internet and Information Systems*, vol. 14, no. 2, pp. 538–561, 2020.

[15] M. Hildebrandt, S. Sunder, S. Mogoreanu, M. Joblin, A. Mehta, I. Thon, and V. Tresp, "A recommender system for complex real-world applications with nonlinear dependencies and knowledge graph context," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11503 LNCS, pp. 179–193, 2019.

[16] A. Karpus, T. di Noia, P. Tomeo, and K. Goczyla, "Rating prediction with contextual conditional preferences," in *Proceedings of the 8th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2016) - Volume 1: KDIR, Porto - Portugal, November 9 - 11, 2016*, A. L. N. Fred, J. L. G. Dietz, D. Aveiro, K. Liu, J. Bernardino, and J. Filipe, Eds. SciTePress, 2016, pp. 419–424. [Online]. Available: http://dx.doi.org/10.5220/0006083904190424

[17] A. Karpus, "Context-aware user modelling and generation of recommendations in recommender systems," Ph.D. dissertation, Gdańsk University of Technology, 2018.

[18] J. Cendrowska, "PRISM: an algorithm for inducing modular rules," *International Journal of Man-Machine Studies*, vol. 27, no. 4, pp. 349–370, 1987.

[19] J. M. Luna, M. Ondra, H. M. Fardoun, and S. Ventura, "Optimization of quality measures in association rule mining: an empirical study," *International Journal of Computational Intelligence Systems*, vol. 12, pp. 59–78, 2018. [Online]. Available: https://doi.org/10.2991/ijcis.2018.25905182

[20] A. Kosir, A. Odic, M. Kunaver, M. Tkalcic, and J. F. Tasic, "Database for contextual personalization," *Elektrotehniski vestnik [English print ed.]*, vol. 78, no. 5, pp. 270–274, 2011.

[21] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," in *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, ser. UAI '09. Arlington, Virginia, United States: AUAI Press, 2009, pp. 452–461. [Online]. Available: http://dl.acm.org/citation.cfm?id=1795114.1795167

[22] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl, "Grouplens: Applying collaborative filtering to usenet news," *Commun. ACM*, vol. 40, no. 3, pp. 77–87, Mar. 1997. [Online]. Available: http://doi.acm.org/10.1145/245108.245126

[23] G. Shani and A. Gunawardana, "Handbook on recommender systems," S. B. K. P. B. Ricci F., Rokach L., Ed. Springer, 2011, ch. Evaluating Recommendation Systems, pp. 257–298.