Factorization Models for Context-/Time-Aware Movie Recommendations

Zeno Gantner
Machine Learning Group
University of Hildesheim
Hildesheim, Germany
gantner@ismll.de

Steffen Rendle[*]
Machine Learning Group
University of Hildesheim
Hildesheim, Germany
srendle@ismll.de

Lars Schmidt-Thieme
Machine Learning Group
University of Hildesheim
Hildesheim, Germany
schmidt-thieme@ismll.de

ABSTRACT

In the scope of the Challenge on Context-aware Movie Recommendation (CAMRa2010), context can mean temporal context (Task 1), mood (Task 2), or social context (Task 3).

We suggest to use Pairwise Interaction Tensor Factorization (PITF), a method used for personalized tag recommendation, to model the temporal (week) context in Task 1 of the challenge. We also present an extended version of PITF that handles the week context in a smoother way.

In the experiments, we compare PITF against different item recommendation baselines that do not take context into account, and a non-personalized context-aware baseline.

General Terms

Algorithms, Performance

Keywords

context-aware recommender systems, item recommendation, temporal, tensor factorization, matrix factorization

1. INTRODUCTION

In the scope of the Challenge on Context-aware Movie Recommendation (CAMRa2010) [10], context can mean temporal context (Task 1), mood (Task 2), or social context (Task 3). For evaluation, two movie recommendation datasets from MoviePilot[1] and Filmtipset[2] were provided by the challenge organizers.

We focus on methods for Task 1 – temporal context. Goal of Task 1 is to predict which movies a set of users has rated in two specific weeks, given all prior rating events. The specific weeks are the Christmas week (week 52) in 2009, and the week leading up to the Academy Awards ceremony (week 9) in 2010.

We see context-aware recommendation as a special case of classical item recommendation. In classical item recommendation, the context is just the user for whom we want to predict items (e.g. movies). In context-aware recommendation, the context contains usually more information than just the user; for instance, the context in Task 2 of CAMRa2010 is the user and the current mood of the user, and in Task 1, the context is the user and the current date.

Another instance of context-aware recommendation is tag recommendation [3, 9], where the context is given by the current user and the resource (e.g. websites, movies, songs, videos) the user wants to tag. The "items" to be predicted are the tags themselves. We think that the problem of Task 1 and the problem of tag recommendation are similar. This similarity allows us to employ Pairwise Interaction Tensor Factorization (PITF) to solve Task 1, with some modifications to deal with the subtle differences between the tasks.

One main difference between time-aware prediction and tag prediction is the fact that time is sequential, is episodic (at least in our application domain), and can be divided into arbitrarily big/small intervals, while items (the "context" in tag recommendation) do not have these properties. To exploit these properties, we suggest O-PITF, which is an ensemble of several PITF models with overlapping temporal context windows.

The contributions of this work are

1. the modelling of the temporal context-aware movie recommendation problem in Task 1 of the CAMRa Challenge as a general context-aware item recommendation problem,

2. the transfer from tag recommendation based on (User, Resource, Tag) triples to temporal context-aware movie recommendation based on (User, Episode, Movie), which allows us to use Pairwise Interaction Tensor Factorization (PITF) for Task 1 of the Challenge,

3. the design of an ensemble model based on PITF to exploit the temporal structure of the problem (O-PITF),

4. the design of an integrated, compact representation of O-PITF, where the ensemble components share certain components,

5. some simple non-personalized but context-aware baseline methods (for comparison purposes),

[*]Steffen Rendle was with the Department of Reasoning for Intelligence at Osaka University, Japan, supported by a research fellowship of the Japan Society for the Promotion of Science (JSPS), when contributing to this work.
[1]`http://www.moviepilot.de/`
[2]`http://www.filmtipset.se/`

6. and preliminary experiments to investigate the usefulness of our methods.

2. PROBLEM STATEMENT

2.1 Implicit Feedback Item Recommendation

The task of (classical) *item recommendation* from implicit, positive-only feedback [5, 6, 1, 7, 2] is to rank the items from a candidate set I^{cand} according to the probability of being viewed/purchased by a given user, based on the feedback matrix $\mathbf{S} \in \{0,1\}^{|U|\times|I|}$ and possibly additional data $\mathbf{A^U}, \mathbf{A^I}$.

The challenge datasets contain more information than implicit feedback; actually their feedback is explicit (movie ratings on scales like $\{0,\dots,100\}$ and $\{1,2,3,4,5\}$), and there is additional $\mathbf{A^I}$ data about items, e.g. movie keywords, and $\mathbf{A^U}$ about users, e.g. age and gender. Nevertheless, we do not make use of this data. Other algorithms could of course use such data, and one could enhance the methods presented in this paper to exploit the additional data.

Note that item recommendation is related to, but distinct from, *rating prediction*, where the task is to predict how much a user will like an item – or rather, what explicit rating the user will assign to the item.

2.2 Context-Aware Item Prediction

As mentioned in the introduction, classical item prediction[3] can be seen as a special case of *context-aware item prediction*. The task here is to rank the items from a candidate set I^{cand} according to the probability of being viewed/purchased by a given user in a particular context $c \in C$, based on the context-aware feedback tensor $\mathbf{S_C} \in \{0,1\}^{|U|\times|C|\times|I|}$ and possibly additional data $\mathbf{A^U}, \mathbf{A^I}$.

The aforementioned problem of tag prediction has the user and the resource as context, while in the present application, the context is the given user and the particular target week for the predictions; see Table 1 for some more examples of context-aware recommendation.

2.2.1 Differences to Tag Recommendation

While tag recommendation and time-aware movie recommendation look almost the same if one views them both as context-aware recommendation problems, there are two crucial differences:

1. *Repeated events:* Usually, a movie recommender is used to discover new movies, i.e. the system should present only movies to a user that they do not already know, even if we are in a new context (week). On the other hand, a tag recommender may recommend the same tag again and again if it fits the current context (resource), e.g. there is no problem suggesting the tag "news" to the same user for two different resources like news websites.
2. *Temporal structure* vs. *"lexical" structure:* Time has an obvious inherent structure:
   (a) It can be divided into parts,
   (b) those parts have a natural ordering (*sequentiality*), and
   (c) some parts (hours, days, weeks, ...) repeat again and again (*episodes*), and some of the repeating episodes are similar to each other, which can possibly be exploited by a recommender system.

[3]Independently from whether the input data is implicit or explicit feedback.

Items, which are the context in tag recommendation, lack such an obvious structure, e.g. there is not "natural" ordering that can be exploited by a recommender system. In fact, there are many successful tag recommendation models that do not assume any item structure beyond the relations to users and tags. Nevertheless, items do have some structure, for instance the similarity over views/ratings or attributes like actors, directors, etc. in the case of movies.

3. METHODS

3.1 Encoding Time as Context

There are different possible granularities for encoding temporal episodes as context. In this work, the week is the principal entity. Besides the normal calendar weeks that start on Monday, it is also possible to let other kinds of "weeks" start on other days. By combining models of different "weeks", it is possible to encode week context with day granularity.

3.2 Pairwise Interaction Tensor Factorization

Pairwise Interaction Tensor Factorization (PITF) [9] is a tensor factorization model initially developed for tag prediction, but it is also applicable to other kinds of context-aware recommendation tasks. The model is a special case of both Tucker Decomposition and Canonical Decomposition (see Figure 1); it sets parts of the involved parameters (the core tensor and parts of the feature matrices) to fixed values, and thus trades expressivity for learnability. PITF for tag prediction models the interaction between users and tags (contexts), and between resources[4] and tags.

To adapt PITF to time-aware movie recommendations, we take into account some of the differences between tags and time in section 2.2.1: we treat the entities according to Table 1, and filter the predictions with the known ratings to avoid the recommendation of already known movies.

The transfer from tag recommendation to time-aware item recommendation is not limited to PITF. Using this simple transformation, every tag recommendation technique that relies on (User, Resource, Tag) triples to make predictions can be used for time-aware item recommendation as well.

3.3 Ensemble Methods

Combining factor models [8] with different regularization and dimensionality is supposed to remove variance from the ranking estimates. There are basically two simple approaches of combining predictions $\hat{y}^l_{u,c,i}$ of l models:

1. Ensemble of the **value estimates** $\hat{y}^l_{u,c,i}$:
$$\hat{y}^{\text{val}}_{u,c,i} := \sum_l w_l \cdot \hat{y}^l_{u,c,i}, \qquad (1)$$
   where w_l is the weighting parameter for each model.

[4]Resources are called "items" in [9]; we do not use the term here to avoid confusion with our prediction target, the movies, which are also generally called "items" in the recommender systems literature.

Table 1: Context-Aware Item Prediction

Application	Context	Event Type	Prediction Target	Repeated Events
Personalized Tag Recommendation	User, Resource	Tagging	Tag	yes
CAMRa Task 1	User, Week	Rating	Movie	no
CAMRa Task 2	User, Mood	Rating	Movie	no
Restaurant Recommendation	User, Location, Mood	Rating, Visit	Restaurant	maybe
Online Book Store	User, Basket	View, Purchase	Product	no
Online Grocery Store	User, Basket	View, Purchase	Product	yes

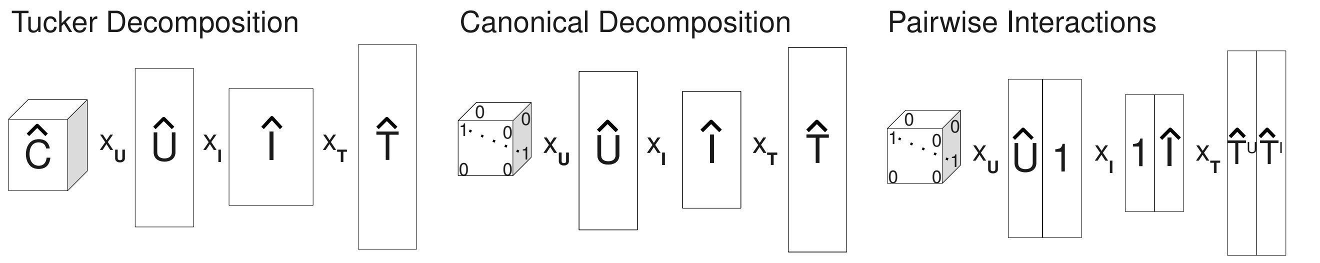

Figure 1: Tensor factorization models for tag recommendation: $\hat{C}$, $\hat{U}$, $\hat{I}$ and $\hat{T}$ are the model parameters (one tensor, three matrices). In Tucker Decomposition the core $\hat{C}$ is variable and the factorization dimensions can differ. For Canonical Decomposition and Pairwise Interactions the core is a fixed diagonal tensor. In Pairwise Interaction parts of the feature matrices are fixed which corresponds to modelling pairwise interactions. Taken from [9].

2. Ensemble of the **rank estimates** $\hat{r}^l_{u,c,i}$:

$$\hat{y}^{\text{rank}}_{u,c,i} := \sum_l w_l \cdot (|I| - \hat{r}^l_{u,c,i}) \qquad (2)$$

   That means tags with a high rank (low $\hat{r}$) will get a high score $\hat{y}$.

Whereas combining value estimates is effective for models with predictions on the same scale, rank estimates are favorable in cases where the $\hat{y}$ values of the different models have no direct relationship.

3.3.1 Combining Different Factor Models

For our factor models the scales of $\hat{y}$ depend both on the dimensionality and the regularization parameter. Thus we use the rank estimates for combining factor models with different dimensionality and regularization. As the prediction quality of all of our factor models are comparable, we have chosen identical weights $w_l = 1$.

3.3.2 Combining the Same Model at Different Iterations

Within each factor model we use a second combination strategy to remove variance. We stop after a predefined number of iterations (1000). In our experiments the models usually converged already after about 500 iterations, but in the following iterations the ranking alternates still a little bit. To remove the variance, we create many value estimates from different iterations and ensemble them. I.e. after the first 500 iterations we create a value estimate for each item in all test posts every 50 iterations and ensemble these estimates with (1). Again there is no reason to favor an iteration over another, so we use identical weights $w_l = 1$. This gives the final estimates for each model. The models with different dimensionality and regularization are combined as described above.

3.4 Overlapping Period Ensembles

Having a model with a fixed division of the year into calendar weeks is not the most flexible option. For a smoother week context, we train several models with datasets that have a different notion of a week, as described in section 3.1. This also brings days that are not in the target week, but very close – e.g. December 20 for the Christmas prediction – into the context of that week. We call that model Overlapping Period PITF Ensemble (O-PITF).

Figure 2 gives an example: In the standard encoding (0), the week 52 corresponds to the calendar week 52. For the other encodings, the week's starting day is shifted, e.g. for (+1), it is shifted one day forward, and for (-1) it is shifted one day backward. For predicting movies in calendar week 52 of 2009, 6 different PITF models are trained, each with a different week definition. For each model, we predict movies for the given users and the context that covers December 21 to 27. For (0), this is context 52, for all schemes with a positive number (here +1 and +2), these are contexts 51 and 52, and for all schemes with a negative number, these are contexts 52 and 53. Note that we have training data exactly until the beginning of calendar week 52 of 2009, which means that for encoding (0), all ratings in context 52 take place in 2008. For encoding (-1), this is different: Here we also have training data for "week 52" in 2009, on December 20. For the encoding (+1), we have training data for context 51 in 2009, which will also be one of the target contexts.

3.5 Compact Overlapping Period Ensembles

A single PITF model — either as a stand-alone recommender, or as a component of an O-PITF ensemble — mod-

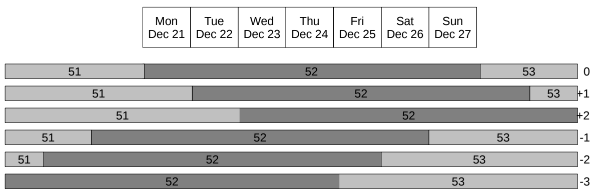

Figure 2: Encoding different week contexts for December 2009.

Table 2: Datasets

Dataset	Events	Users	Items	Sparsity
mp-xmas	3,341,230	81,183	24,057	99.8289
(test)	7,114	160	1,882	97.6375
mp-oscar	4,421,098	100,752	24,894	99.8237
(test)	7,583	160	1,980	97.6064

els the user-context interactions as well as the user-item interactions. When combining several such components to an O-PITF model, we train the different components with respect to different contexts, but the items remain the same. This means we could use the same user-item interaction factors for all components.

Our proposed model *Compact Overlapping Period Ensembles (cO-PITF)* does exactly this: Instead of training several independent components, one user-item matrix and several user-context matrices are trained simultaneously by the same procedure.

4. EXPERIMENTS

In this section, we present our preliminary experiments.

4.1 Datasets

The original MoviePilot training dataset contains 4,544,386 ratings by 105,131 users, with timestamps ranging from January 2008 to March 2010. For the two sub-tasks (Christmas and Oscar week), we filtered out – according to the rules – all ratings starting with the beginning of the prediction period.[5] The resulting datasets can be found in Table 2. "Sparsity" refers to the percentage of zero entries in the feedback matrix **S**. Note that between December and February, about 20,000 new users seem to have joined MoviePilot[6]

4.2 Compared Methods

BPR-MF.

BPR was recently proposed [7] as a generic optimization method for item prediction from implicit feedback. We use a matrix factorization model trained using the BPR framework to compare out context-based approach to an item prediction method that does not take the context into account. We trained BPR-MF models for up to 1000 iterations with learn rate $\alpha = 0.05$ for different numbers of factors ($k \in \{16, 32\}$) and different regularization constants ($\lambda_u, \lambda_{ij} \in \{0.01, 0.005, 0.001, 0.0005, 0.0001, 0.00001\}$), and picked the models with the best prec@5 on the test data.

[5]Dec 21 for xmas, Feb 27 for oscar. For oscar we actually removed two days more than necessary – we will re-train our models for the "quiz set" evaluation by the organizers.

[6]This is just one possible explanation, of course this could also be an artifact of the export procedure, or due to deliberate modification by the challenge organizers.

Table 3: mp (date/week/event/prior) periods.

Dataset	Period	Start	End
mp-xmas	date	2008-12-21	2008-12-27
	week	2008-12-22	2008-12-28
	event	2008-12-20	2008-12-28
	prior	2009-12-14	2009-12-20
mp-oscar	date	2009-02-27	2009-03-07
	week	2009-02-23	2009-03-01
	event	2009-02-13	2009-02-22
	prior	2010-02-17	2010-02-26

PITF.

Pairwise Interaction Tensor Factorization for context-aware movie recommendation as described in section 3.2. For number of factors $k \in \{8, 16, 32\}$, we tried learn rate α and regularization constant λ from the sets {0.001, 0.01, 0.025, 0.05} and {0, 0.00001, 0.00005, 0.0001}. All models were trained for up to 1000 iterations, except for the ones with $k = 32$ (due to time constraints), which we trained for only 650/900 iterations for the mp-oscar and mp-xmas subproblems. Again, we picked the models with the best prec@5 on the test data.

item-knn.

Item-based k-nearest neighbor prediction with cosine similarity.

mp.

A simple, non-personalized and non-context-aware baseline that just recommends the most popular (most rated, not best rated, in the training data) movies to each user.

mp (date/week/event/prior).

Another baseline, not personalized, but a little bit context-aware: Instead of taking the globally most popular movies, we recommend the movies that had most ratings in the corresponding period of the previous year. "Corresponding period" can have different interpretations, so we tried out exactly the same dates (which does not take weekdays into account), exactly the same calendar week (which does not take the exact date of holidays like Christmas into account), and a period in the preceding year with which we tried to cover the event as good as possible (e.g. by also taking in the weekend before week 52 for Christmas, and taking the exact 10 days prior to the Oscar ceremony). We also tried the most popular movies in a period directly before the prediction period. Table 3 gives some details.

One thing that is obvious from considering this baseline method is that for handling events like the Oscar ceremony, which do not happen at fixed dates, using the calendar alone is not sufficient.

random.

Recommending random items is an even simpler baseline.

4.3 Evaluation Metrics

Table 4: Results Task1 — MoviePilot Christmas

Method	AUC	prec@5	prec@10
random	0.5131	0	0.00125
mp	0.9568	0.1075	0.085
mp (date)	0.8634	0.09875	0.079375
mp (week)	0.8649	0.1	0.081875
mp (event)	0.8677	0.09875	0.07875
mp (prior)	0.9533	0.11125	0.106875
item-knn	0.9555	0.1325	0.12
BPR-MF-16	0.9680	**0.1418**	**0.1281**
BPR-MF-32	**0.9711**	0.1397	0.1231
PITF-8	0.9511	0.13125	0.1125
PITF-16	0.9490	0.125	0.103125
PITF-32	0.9501	0.12875	0.109375

prec@n (precision at n) measures the number of correctly predicted items in the top-n recommendations. It is commonly used in the area of information retrieval [4], is relevant to practice, and easy to interpret. We report results for prec@5 and prec@10. Additionally, we report AUC (the area under the ROC curve), which is a more general ranking measure.

$$\mathrm{AUC} = \sum_{\substack{(u,i,j)\in U\times \\ I_u^{\text{test}}\times(I^{\text{cand}}-I_u^{\text{test}})}} z_u\,\delta(\hat{x}_{uij}), \qquad (3)$$

where z_u and δ are defined as

$$z_u := \frac{1}{|U||I_u^{\text{test}}||I^{\text{cand}} - I_u^{\text{test}}|}, \quad \delta(x) := \begin{cases} 1, & x \geq 0 \\ 0, & \text{else} \end{cases}.$$

4.4 Results

4.4.1 *Christmas*

The results for the Christmas prediction problem can be seen in Table 4. Considering precision, BPR-MFis the strongest method, which shows that there is still room for improvement (in terms of hyperparameter search) for the factorization models BPR-MF and PITF. PITF-8 is rather strong, and slightly better than the larger models PITF-16 and PITF-32, which is a hint that we need to look for better regularization parameters for the larger models. BPR-MF with 32 factors is strongest in terms of AUC, which is not surprising, as the BPR method optimizes for a smooth approximation of AUC. The context-aware most-popular baselines do not perform better than the normal most-popular predictions, with the exception of the most popular movies of the week exactly before Christmas week. This seems to imply that there is at least some contextual effect. Possibly it would be worthwhile to investigate different window sizes for mp (prior), to have a bigger impact on accuracy.

4.4.2 *Oscar*

The results for the Oscar prediction problem can be seen in Table 5. Here, PITF behaves more in the expected way, the predictions improve with larger models. Still, we would expect PITF to outperform BPR-MF, which is not context-aware; this again is a hint that we have not find practical hyperparameters for PITF yet. The strongest prediction model in terms of precision is mp (prior). This is somewhat surprising — it is better to recommend the movies that were most popular in the 10 days before the Oscar ceremonies than to use item-based collaborative filtering on the full dataset! The results marked with "recent" use the same methods as above, but ignore training data before July 2009. Such results with respect to more recent data indicate that contextual effects are particularly strong for the mp-oscar problem.

Table 5: Results Task1 — MoviePilot Oscar

Method	AUC	prec@5	prec@10
random	0.5018	0.00136	0.00066
most-popular	0.9611	0.07895	0.08026
mp (date)	0.9048	0.0684	0.0697
mp (week)	0.8979	0.0803	0.0631
mp (event)	0.9001	0.075	0.0743
mp (prior)	0.9623	**0.2039**	**0.1822**
item-knn	0.9597	0.1289	0.1243
BPR-MF-16	0.9695	0.1609	0.1442
BPR-MF-32	**0.9755**	0.1660	0.1498
most-popular (recent)	0.9656	0.0888	0.0825
item-knn	0.9644	0.135	0.1325
BPR-MF-16 (recent)	0.9728	0.1634	0.1472
BPR-MF-32 (recent)	0.9735	0.1574	0.1464
PITF-8	0.9495	0.1026	0.0993
PITF-16	0.9481	0.1105	0.0993
PITF-32	0.9327	0.1276	0.1033

4.5 Discussion

Due to time constraints, we have only trained models with a small number of factors, and there has been only limited effort to find suitable hyperparameters for the advanced factorization methods like BPR-MF, PITF, and O-PITF. Experience from past studies [9, 7] has shown that using a greater number of factors and finding the right hyperparameters contributes greatly to the methods' accuracy. We expect to get improved results from further experiments.

4.6 Reproducibility of the Experiments

Some algorithms used for this paper are implemented in the MyMedia Recommender Framework[7], and will be included in the next public release of the software. The programs used for the evaluations, as well as the data preparation scripts, are available on request.

5. CONCLUSION

We modelled the temporal context-aware movie recommendation problem in Task 1 of CAMRa Challenge as a general context-aware item recommendation problem, which allows us to use the Pairwise Interaction Tensor Factorization (PITF) for temporal context-aware movie recommendation. Furthermore, we suggested an ensemble model based on PITF to exploit the temporal structure of the problem (O-PITF), and a more integrated, compact representation of O-PITF (iO-PITF). We presented preliminary experiments in which we compare PITF against different item prediction algorithms and some simple non-personalized but context-aware baseline methods.

The methods we present in this paper are by no means finished, but rather work-in-progress. There are several directions for future work. First of all, the existing methods need more evaluation, i.e. a finer hyperparameter search and more factors for all factorization methods, experiments on the Filmtipset dataset, experiments on different splits to get results that allow the application of significance tests; we will measure the impact in run-time performance and accuracy of the integrated version of O-PITF (iO-PITF).

Methodological enhancements will be different versions of O-PITF — possibly analog to the different versions of mp (date/week/event/prior) — and approaches to take additional data like rating values, user and item attributes into account.

6. ACKNOWLEDGMENTS

This work was co-funded by the European Commission FP7 project MyMedia under the grant agreement no. 215006.

7. REFERENCES

[1] A. Gunawardana and C. Meek. Tied boltzmann machines for cold start recommendations. In *RecSys '08: Proceedings of the 2008 ACM Conference on Recommender Systems*, pages 19–26, New York, NY, USA, 2008. ACM.

[2] A. Gunawardana and C. Meek. A unified approach to building hybrid recommender systems. In L. D. Bergman, A. Tuzhilin, R. D. Burke, A. Felfernig, and L. Schmidt-Thieme, editors, *RecSys*, pages 117–124. ACM, 2009.

[3] R. Jaeschke, L. Marinho, A. Hotho, L. Schmidt-Thieme, and G. Stumme. Tag recommendations in social bookmarking systems. *AI Commun.*, (21(4)), 2008.

[4] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval.* Cambridge University Press, Cambridge, UK, 2008.

[5] D. Oard and J. Kim. Implicit feedback for recommender systems. In *in Proceedings of the AAAI Workshop on Recommender Systems*, pages 81–83, 1998.

[6] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. M. Lukose, M. Scholz, and Q. Yang. One-class collaborative filtering. In *ICDM 2008*, 2008.

[7] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In *UAI*, 2009.

[8] S. Rendle and L. Schmidt-Thieme. Factor models for tag recommendation in bibsonomy. In *Discovery Challenge at ECML PKDD 2009*, 2009.

[9] S. Rendle and L. Schmidt-Thieme. Pairwise interaction tensor factorization for personalized tag recommendation. In *WSDM.* ACM, 2010.

[10] A. Said, S. Berkovsky, and E. W. De Luca. Putting things in context: Challenge on context-aware movie recommendation. In *CAMRa2010: Proceedings of the RecSys '10 Challenge on Context-aware Movie Recommendation*, New York, NY, USA, 2010. ACM.

[7] `http://mymediaproject.codeplex.com/`