Time-based TV programs prediction

Roberto Turrin
Moviri S.p.A.
ContentWise R&D
via Schiaffino, 11 - Milan, Italy
roberto.turrin@moviri.com

Andrea Condorelli
Moviri S.p.A.
ContentWise R&D
via Schiaffino, 11 - Milan, Italy
andrea.condorelli@moviri.com

Paolo Cremonesi
Politecnico Di Milano, DEIB
p.zza Leonardo da Vinci 32
Milan, Italy
paolo.cremonesi@polimi.it

Roberto Pagano
Politecnico Di Milano, DEIB
p.zza Leonardo da Vinci 32
Milan, Italy
roberto.pagano@polimi.it

ABSTRACT

This paper addresses the problem of providing recommendations to TV viewers. Conversely to standard recommender systems operating in the settings of static datasets, recommending TV shows must take into consideration that items are scheduled at specific times. Consequently, the catalog of items is particularly dynamic and users consumption pattern is strongly affected by time context and channel preferences.

In this work, we extend common state-of-the-art recommendation methods to exploit and integrate the current watching context into the user viewing model. Empirical experiments over a large-scale linear TV dataset demonstrate a significant improvement in recommendation quality when context is considered.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Theory

Keywords

TV programs, time, context, recommender systems, audience prediction

1. INTRODUCTION

While recommender systems have seen tremendous achievements in the fields of VoD, limited effort has been conducted to build an effective recommender system for linear TV. Compared to conventional recommender systems, recommending TV programs is more challenging for several aspects:

- **Dynamic catalog of items**. In traditional domains, such as VoD, the available content is updated at a relatively slow rate (e.g., on a daily-basis a few movies are added to/removed from the catalog). Conversely, since TV programs are scheduled at specific times, each item is available only at specific time intervals, leading to a catalog of items that constantly changes over time. This accentuates the *new-item problem*, as many upcoming TV programs have never been watched in the past. Therefore, a TV recommender system cannot rely purely on Collaborative Filtering (CF) techniques to recommended these programs.

- **Time-constrained catalog of items**. In contrast to VoD which provides users with the ability to select and view content at their convenience, in traditional linear TV programs are broadcast according to a preset schedule, and recommendations should consider only programs transmitted at the moment of the recommendation, or within few minutes [13]. We refer to these programs as *live programs*.

- **Strong context-aware consumption patterns**. TV viewers have a number of preferred TV channels (e.g., the first 9 channels on the remote control) and they prefer to watch TV during specific time periods (e.g., prime time) [10]. In linear TV recommender systems, context can be more important than items. If there is an interesting-enough program for the user in her/his preferred channels and time slots, the user will watch it, regardless to the fact that a more interesting program could be scheduled on a different channel or at a different time slot.

- **A user cannot watch different TV channels simultaneously**. Unlike traditional recommender systems, where items are always available and a user can consume more items at the same time, in linear TV a users cannot watch more than one program at the same time and many of the not-watched programs will not be re-transmitted in the near-future. Therefore, when analyzing historical viewing habits, a recommender system should consider that some TV programs which

RecSysTv 2014 Foster City, Silicon Valley USA

are mutually exclusive because scheduled simultaneously [13].

- **Users might watch the same TV program multiple times**, either the same identical program (e.g., an interesting movie the user likes to watch again) or different "episodes" of the same TV series.

In this paper we present a novel algorithm for the recommendation of relevant, live programs to the users.. The algorithm tries to consider all the peculiar aspect of the linear TV domain. The user model adopted by the algorithm is based on three key concepts:

- we assume that the viewing habits of TV users follow regular patterns based on preferred time slots and channels;
- we discretize the continuous contextual time variable into fine-grained time slots and we learn the user preferences within each time slot;
- we merge adjacent time slots based on their proximity to obtain an aggregate user model and we use such model to make predictions.

There are several recommender algorithms that integrate contextual time information into the model to improve accuracy of predictions (see [10] for an overview). The underlying idea when including the time context into the model is to capture the seasonal viewing habits of users: two TV programs are similar in the user's opinion if they are broadcast very close in time to each other. For instance, a user might like watching TV during weekends and not during weekdays. Therefore, in the user's opinion, programs scheduled during weekends are similar to each other in the sense that the user tends to prefer them to programs scheduled during weekdays.

Time-based context domain is usually transformed into a categorical one: the time domain is partitioned into disjoint intervals and each interval is used as a different context. The above approach has one main shortcoming: the context similarity between viewing events is determined by the intervals they belongs to and not on the their real closeness in time. For instance, if the boundary between two intervals is set at 20:00 and there are two events with timestamps 19:58 and 20:02, then those events will be classified as different in context, although they occur very close in time.

The work in [10] presents two approaches for modeling continuous context dimensions: (i) fuzzy event modeling enables events to belong to different contiguous categorical contexts, and (ii) fuzzy context modeling, enables context categories to overlap.

In this paper we propose a new approach for modeling the continuous time context by introducing the concept of *context distance*. We validate our approach on a dataset containing the TV viewing habits of 13,664 users over 217 channels, collected over a period of 4 months, for a total of 21194 distinct programs and 56,101 schedules.

The rest of the paper is organized as follows. Section 1.1 surveys background and related work. Section 2 introduces notations and concretely defines the problem. Section 3 and 4 present the algorithms used to tackle the problem. Section 5 details the experimental setup and Section 6 reports our experimental results. We conclude in Section 7.

1.1 Related works

One of the first paper on TV recommender systems is [7] which presents a personalized Electronic Program Guide (EPG). A number of other works focus on the concept of personalized EPG. Cotter and al. in [14, 4] present a personalized EPG where the selected TV shows are based on a hybrid recommender system which mixes collaborative and content-based recommendations. Users manually input their preferences about channels, genres, and viewing times. This information is combined with the user's viewing activity by means of case-based reasoning and collaborative filtering techniques. Ardissono et al. in [2] present a personalized EPG where recommendations are generated locally on the client side using a hybrid approach on the basis of three information sources: user's implicit preferences represented in terms of program categories and channels (content attributes are downloaded from the satellite stream), (ii) user classes, and (iii) user viewing activity.

Other works address Personal Video Recorders (PVRs). Engelbert et al. in [8] present a recommender system for an extension of PVR based on both explicit user preferences and the user's recording history. A Bayesian Classifier is applied to predict the probability that the user will like a TV program. Srinivas et al. in [17] present an algorithm for PVRs which produces recommendations on the basis of both implicit (using both Bayesian classifier and Detection Trees) and explicit user information, combined together by means of neural networks.

Many works present recommender systems based on hybrid collaborative and content based approaches [1]. The work in [11] describes a TV recommender systems which combines together content-based and collaborative filtering by means of Neural Networks. The system also uses information on the users, such as demographic information, interests, and moods. Martinez et al. in [12] exploit a hybrid approach to solve new-item, cold-start, sparsity, and overspecialization problems. The methods mix together in the same interface the outcome of content-based (computed using the cosine similarity among item feature vectors) and collaborative filtering (using Singular Value Decomposition to reduce the size of item's neighborhood).

Fewer works present TV recommender systems based on social networks [9]. Chang et al. in [3] suggest that TV user preferences can be learned from past viewing experience and from friendship connections in social networks.

Some works focus their attention on implicit and explicit rating elicitation. Uberall et al. in [16] present a recommender system for DVB (Digital Video Broadcasting) based on both the viewing behavior and explicit user preferences on preferred genres, sub-genres, and TV programs. The work in [15] presents a TV recommender system based on a multi-agent approach which combines implicit and explicit user preferences. Implicit preferences are processed with a Bayesian classifier to compute the likelihood that the user will like or dislike a TV program. A decision tree is later used to compute program recommendation scores.

Few works focus their attention on the time evolution of TV recommender systems. Cremonesi et al. in [5] highlight that different CF algorithms have different behaviors according to the state of the recommender system (measured in terms of statistical properties of the user-rating-matrix): during the cold start of the recommender system item-based methods outperform matrix-factorization meth-

ods. The work in [13] highlights the problem of when recommending TV shows. The paper suggests the adoption of a push mechanism, i.e., it is not the user who requests a recommendation, but it is the system that, on the basis of a profit model, triggers recommendations at the right time.

2. DEFINITION OF THE PROBLEM

In this paper we address the problem of recommending relevant TV programs to users based on their viewing habits and on some characteristic of the TV programs. More specifically, we are interested in recommending some of the programs scheduled to be broadcast within a short time period from the recommendation (e.g., within one hour).

The raw input data to the system are:

1. the EPG (Electronic Programming Guide) containing program-id, genre, sub-genre, start-time, end-time, and TV channel for all the scheduled programs;

2. the set of user activities containing user-id, start-time, end-time and TV channel for all the tune events with the granularity of a minute.

In order to have a data structure easier to manage, we pre-process and simplify the raw data. Zapping effect for visions shorter than a minute is neglected: each minute is assigned the channel that is watched most whose vision time is not less that 15 seconds. We first assume that the viewing habits of users have a weekly regular pattern. We divide the time span of a week into time slots of equal duration. If L is the time slot length in minutes, there are $10080/L$ different time slots (where 10080 is the number of minutes in one week). For instance, if $L = 60$ minutes, there are 168 time slots. We denote by $\mathcal{S}$ the set of possible time slots. In our experiments we have used time slots of $L = 30$ minutes.

Additionally, we define the features of a TV program as the union of its genre, sub-genre, and TV channel. We denote by $\mathcal{F}$ the set of all possible features. We denote by $\mathcal{C}$ the set of features representing TV channels only (i.e., $\mathcal{C} \subseteq \mathcal{F}$).

We define the *user-preference tensor* p_{ufs} as the total number of minutes user u spent watching a program with feature f in time slot s during the observation period.

For instance, assume user u watches 80 minutes of a TV program genre "Children" (feature f_1), sub-genre "Cartoon" (feature f_2) on "Disney Channel" (feature f_3) spanning slots s_1 (e.g., Sunday, 2.00pm–3.00pm) and s_2 (e.g., the same day, 3.00pm–4.00pm). This event increments by 80 minutes the 6 elements of the user-preference tensor corresponding to the three features and two slots. More formally, for each $f \in \{f_1, f_2, f_3\}$ and $s \in \{s_1, s_2\}$ we have that $p_{ufs} \leftarrow p_{ufs} + 80$.

Elements of p_{ufs} can be easily computed from the raw data (by joining EPG and tuning events) and can be used to estimate how much a user prefers to watch TV programs of specific genres, on specific channels, or during specific time slots. The user-preference tensor resembles the user-item-context tensor of traditional context-aware recommender systems, the main difference being that the item dimension is replaced with the features dimension.

Each vector p_{u*s} describes the user preferences (in terms of features such as preferred channels, genres and sub-genres) for user u during time slot s.

3. BASELINE ALGORITHMS

Two main factors affect the preferences of a TV user: channel and time. Indeed, the user preferences strongly depends on the temporal context (i.e., day of week and time of day). For instance, during dinner the user might be accustomed to watch newscasts, while after dinner he/she wants to relax and will likely opt for a movie or a reality show. It is worth noting that each user has his own habits, so, as an example, a user might eat dinner at 7pm and another one at 9pm. Furthermore, TV users are strongly affiliated to channels. The typical user switches the TV on and surfs over a very limited number of channels to select the program to watch. In many cases, the choice is restricted to the single channel the user is used to watch. Sometimes, the TV is simply switched on a default channel routinely, regardless the program currently live. As a consequence, we assume that only a minor role is played by the characteristics (e.g., genre and sub-genre) of the broadcast TV program. This assumption motivates the following three baseline algorithms, which are based solely on contextual parameters (channel and time slot).

3.1 Top channel

This baseline algorithm recommends the TV programs that are scheduled during slot s on the most popular channels. Recommendations are the same for each user. The popularity of a channel is defined in terms of the total number of watching minutes accumulated by that channel. For each channel feature $c \in \mathcal{C}$ we compute the popularity r_c of channel c as

$$r_c = \sum_{s,u} p_{ucs} \tag{1}$$

The algorithm recommends first all the TV program scheduled during slot s on the most popular channel. If more recommendations are needed, the algorithm recommends all the TV programs scheduled on the second most popular channel during the same slot s, and so on. Partial ordering of TV programs within a channel is based on their schedule.

3.2 Top channel per user

This baseline algorithm is a refined version of the previous one. The algorithm recommends the TV programs that are scheduled during slot s on the most popular channels, where channel popularity is computed on a per-user basis.

The popularity of a channel is defined in terms of the total number of watching minutes accumulated by user u on channel c. For each user u and for each channel feature $c \in \mathcal{C}$ we compute the popularity r_{uc} as

$$r_{uc} = \sum_{s} p_{ucs} \tag{2}$$

3.3 Top channel per user and slot

This baseline algorithm is a further refinement of the previous ones. The algorithm recommends the TV programs that are scheduled during slot s on the most popular channels, where channel popularity is computed on a per-user and per-slot basis.

The popularity of a channel is defined in terms of the total number of watching minutes accumulated by user u on channel c during time slot s. For each user u, time slot s and channel feature $c \in \mathcal{C}$ we compute the popularity r_{ucs} as

$$r_{ucs} = p_{ucs} \tag{3}$$

4. SMOOTHED CONTEXT

In this section we describe our algorithm which extends the baselines (i) by introducing the *smoothed time context* and (ii) by incorporating in the model all the features (i.e., genre and sub-genre).

The discrete definition of time context described in the previous sections might create unrealistic discontinuities between adjacent time slots. For instance, according to (3), a user might like "Cartoons" on "Disney Channel" from 10am to 11am but have no interest for "Cartoons" on "Disney Channel" from 11am to 12am. This might occur if the viewing history of the user contains many events in the first time slot (10am–11am) and fewer events in the second time slot (11am–12am).

In order to mitigate this effect, we introduce a smoothing function which aggregates the user preferences in each time slot with the preferences in the neighbor time slots.

We first define a distance function which measures the distance between time slots:

$$d(s_1, s_2) = \left[\text{days}(s_1, s_2) + \frac{\text{mins}(s_1, s_2)}{L}\right] \sigma(s_1, s_2) \qquad (4)$$

where

- $\text{days}(s_1, s_2)$ is the difference in days between s_1 and s_2 (e.g., the distance between Tuesday and Thursday is 2, as well as between Tuesday and Sunday)
- $\text{mins}(s_1, s_2)$ is the difference in minutes between the time of day of s_1 and s_2 (e.g., the distance between 00:05 and 00:15 is 10, as well between 00:05 and 23:55)
- $\sigma(s_1, s_2)$ is 1 if s_1 and s_2 are both weekends or both weekdays, 0 otherwise.

Given a time slot s, we define its neighbor set $\mathcal{N}(s)$ as $\mathcal{N}(s) = \{t | d(s,t) \leq D\}$, where D is the maximum aggregation distance. In our experiment we have used $D = 6$. In this way, for instance, the time slot 'Monday 20:00' is in the same neighborhood as 'Friday 20:00' and 'Monday 16:00', but not with 'Friday 16:00'.

Given a time slot s, we can aggregate its neighbors $\mathcal{N}(s)$ in several ways. We explored two main functions: (i) by score and (ii) by rank.

4.1 Score aggregation

The *score-aggregated* user-preference tensor a_{ufs} is computed by smoothly merging the user preferences of time slots in $\mathcal{N}(s)$ by means of a weighted average:

$$a_{ufs} = \frac{\sum_{t \in \mathcal{N}(s)} p_{uft} \cdot \beta(s,t)}{\sum_{t \in \mathcal{N}(s)} \beta(s,t)} \qquad (5)$$

where β is a weighting function that depends on the distance. In our experiments we have defined $\beta(s,t)$ as

$$\beta(s,t) = \frac{1}{d(s,t) + 0.1}$$

.

4.2 Rank aggregation

The aggregated user profile for time slot s is generated as follows. We sort the time slots $t \in \mathcal{N}(s)$ in increasing order of distance from s and we label them with their rank distance $k(t)$, starting with $k(s) = 0$ for slot s, the closest to itself. Time slots t_1 and t_2 with the same distance from s have the same rank value $k(t_1) = k(t_2)$.

We now define the *rank-aggregated* user-preference tensor a_{ufs} as

$$a_{ufs} = \max_{t \in \mathcal{N}(s)} \left[\delta_{uft} \cdot e^{-\alpha k(t)}\right] \qquad (6)$$

where δ_{uft} is 1 if $p_{uft} > 0$ and 0 otherwise, while α is a smoothing constant. This aggregation function assigns the highest score (i.e., 1) to all the features related to TV programs watched by the user during time slot s. Other features, related to programs watched during more distant slots, have a lower score, which decays exponentially to 0 with increasing rank distance from s.

In our experiments we have set $\alpha \approx 0.07$ so that the score of a feature is 50% of the maximum score when $k = 10$.

4.3 Recommendations

Let us assume that a TV program i is scheduled in a time window that spans a set of time slots defined as S_i, and it is characterized by a set of features – among which the channel – defined as F_i. The preferences r_{ui} of user u on live TV program i can be estimated as

$$r_{ui} = \sum_{s \in S_i, f \in F_i} \frac{a_{ufs} \cdot w_f}{w_f \cdot |S_i|} \qquad (7)$$

where $|S_i|$ is the number of slots spanned by program i and w_f is the weight assigned to feature f. In our experiments, we spanned multiple combinations of feature weights, confirming that the best results can be obtained when the channel feature is weighted higher than the other ones (e.g., genre). We have have set $w_f = 1$ for the features describing TV channel (i.e., $f \in \mathcal{C}$).

5. EVALUATION

5.1 Dataset

We used a dataset collecting the TV viewing habits of 13,664 active users over 217 channels, either over-the-air (digital terrestrial broadcasting) or satellite, free or pay-TV. Demographic distribution of the users is reported in Table 1. This dataset has been collected over a period of 4 months in 2013, for a total of 21194 distinct programs and 56,101 schedules (i.e., EPG entries). The available metadata for the programs are title, genre, and sub-genre. The available metadata for the tuning events are channel, start time, and end time.

The dataset is partitioned into two subsets along the time dimension: training set and test set. The *training set* spans the first three months of the dataset. The *test set* spans the last month.

The catalogue is very dynamic: on average there are 82 novel programs each day (about 0.3% of the total number of programs). This number grows if we consider the average number of novel programs in a week: 229 (about 1% of the catalogue size). With such dynamic catalogue traditional collaborative filtering techniques could not be employed, due to the *cold-start* effect.

5.2 Evaluation methodology

The test set is partitioned into disjoint intervals with one-hour duration. For each user u and for each interval in which

Table 1: Demographic distributions of the users

males	41%
females	47%
children	12%
0-19	17%
20-34	15%
35-64	45%
65+	23%
mono component families	9%
multi component families	91%
users with children	33%
users without children	67%

Table 2: Comparison between algorithms. Bolded baselines (topChxx) are the best among baselines for the considered metric. Bolded non-baselines (CxtBlendxx) are statistically better than the best baseline for that metric.

Algorithm	recall @1	recall @5	recall @10	avgRank @50
CxtBlend	0.283	**0.734**	**0.852**	**5.819**
CxtBlend_g	0.290	**0.734**	**0.859**	6.518
CxtBlend_gg	0.290	**0.733**	**0.858**	6.607
topCh	0.125	0.423	0.639	11.942
topCh_u	0.232	0.612	**0.797**	**7.512**
topCh_us	**0.282**	**0.658**	0.759	12.587

the user has watched at least one TV program, we extract the list of TV programs scheduled in that interval in any TV channel. For each program i in the list, we estimate the relevance r_{ui} according to the recommender algorithm. The list is sorted on the basis of r_{ui} and the top-N most relevant elements are presented to the user. Finally, we verify if the TV programs watched by the user in the time frame are in the list of recommended programs and we compute average rank and recall.

The average rank (computed at N) is the average position, in the top-N recommendation list, in which the TV programs effectively watched by the user are suggested. TV programs not recommended in the top-N list are assigned a fictitious rank equals to $N + 1$.

The recall (computed at N) is the percentage of TV programs actually watched by the user that are suggested within the first N positions of the recommendation list.

Both recall and average rank have been computed by weighting the TV programs on the basis of the number of minutes the user watched the program in the 1-hour time interval.

The results compare three versions of our algorithm (denoted as CxtBlend) and three versions of the baseline approach (denoted as topCh).

- **topCh**: baseline algorithm presented in Section 3.1
- **topCh_u**: baseline algorithm presented in Section 3.2
- **topCh_us**: baseline algorithm presented in Section 3.3
- **CxtBlend**: algorithm proposed in Section 4, where genre and sub-genre weights are set to 0.
- **CxtBlend_g**: the same algorithm as CxtBlend, where genre weights are set to 0.1 and sub-genre weights are set to 0.
- **CxtBlend_gg**: the same algorithm as CxtBlend, where both genre and sub-genre weights are set to 0.1.

6. RESULTS

Table 2 presents an overview of the experimental results. We omitted the results of the score algorithm (described in Section 4.1) because they are much worse than all the others (both ContextBlend Rank and TopCh): this can be explained by the fact that the rank aggregation method takes the maximum among the user preferences of nearby slots, thus neglecting the non-visions. The score aggregation method, on the other hand, applies a weighted average of nearby slots, thus giving much more importance to non-visions (or zero time visions) than the rank counterpart. This lead us to think that missing preferences on a TV dataset have to be considered much more like "missing values" than like "negative preferences".

The best baselines are those recommending the most popular channels on a per user-basis. Including the time slot in the baseline slightly increase the accuracy of recommendations. All the three versions of our algorithm outperform the baselines for all of the metrics, with the exception of recall@1. We also observe that the inclusion of genre and subgenre does not increase significantly the accuracy of recommendations. Probably, the genre/sub-genre taxonomy provided together with the EPG is not sufficiently accurate to capture specific user tastes and leverage better predictions. We are further investigating this issue on other datasets.

Figure 1 presents recall@10 for the same algorithms. The recall is plotted as a function of the test week (there are four weeks in the test set). The accuracy of all the algorithms decreases as the week is more distant from the end of the training period (week 1 is the week immediately following the training period, week 4 is the most distant in time from the training period). This behavior is mainly explained by two factors: (i) users' preferences evolve over time, as they find out new types of programs (e.g., a new TV series) or

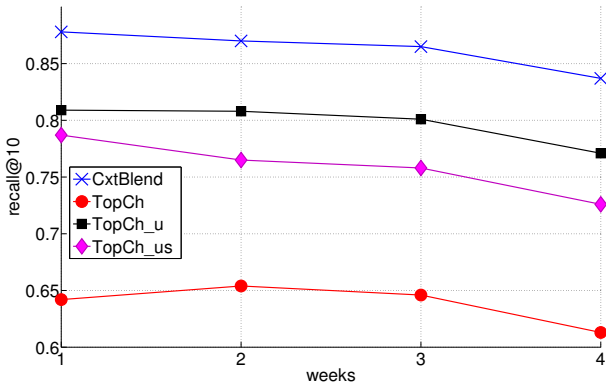

Figure 1: Performance decrement in terms of recall@10 as the weeks are more distant from the training period.

Table 3: Recall@10 for ContextBlend and TopCh (baseline) for different demographic categories of users.

	CxtBlend	CxtBlend_g	CxtBlend_gg	TopCh	TopCh_u	TopCh_us
Male	0.853	0.850	0.849	0.628	0.787	0.751
Female	0.873	0.870	0.870	0.662	0.809	0.778
Children (no gender)	0.836	0.826	0.824	0.517	0.767	0.652
0-19 years old	0.825	0.816	0.814	0.515	0.758	0.640
20-34 years old	0.816	0.808	0.805	0.571	0.755	0.656
35-64 years old	0.833	0.829	0.829	0.621	0.765	0.728
65+ years old	0.917	0.916	0.916	0.703	0.856	0.844
Mono-component family	0.916	0.914	0.913	0.649	0.855	0.837
Multi-component family	0.864	0.862	0.861	0.651	0.800	0.770
User with children	0.801	0.795	0.794	0.577	0.741	0.657
User without children	0.880	0.877	0.877	0.656	0.813	0.788

stop watching certain shows and (ii) the EPG changes over time, resulting in a dynamic catalog where - constantly - some programs appear for the first time and others are removed [5, 6].

Table 3 presents a drill-down of the recall@10 for different demographic categories. With respect to age, older users have a more regular viewing pattern and obtain the best recall. Families with several components are more difficult to predict. Surprisingly, female users are slightly easier to predict than male users.

7. CONCLUSIONS

In this paper we presented a new family of algorithms for the contextual recommendation of TV programs in linear TV services. Users are modeled taking into account the specific consumption patterns typical in the settings of TV, strongly influenced by time context and channel preferences. The drawbacks of representing time context as a discrete variable have been mitigated by introducing the notion of context distance; thus, user model is built by aggregating nearby contextual profiles. Empirical experiments over a large-scale linear TV dataset compared the proposed family of algorithms with a set of baseline approaches (based on channel preferences) and demonstrate a significant improvement in recommendation quality when time context is considered.

8. ACKNOWLEDGMENTS

This work is funded by European Union's Seventh Frame work Programme (FP7/2007-2013) under CrowdRec Grant Agreement n.610594.

9. REFERENCES

[1] K. Ali and W. van Stam. TiVo: Making show recommendations using a distributed collaborative filtering architecture. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 394–401, New York, NY, USA, 2004. ACM.

[2] L. Ardissono, C. Gena, P. Torasso, F. Bellifemine, A. Difino, and B. Negro. User modeling and recommendation techniques for personalized electronic program guides. *Personalized Digital Television*, 6(1):3–26, 2004.

[3] N. Chang, M. Irvan, and T. Terano. A TV program recommender framework. *Procedia Computer Science*, 22(0):561 – 570, 2013.

[4] P. Cotter and B. Smyth. Ptv: Intelligent personalised tv guides. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 957–964. AAAI Press, 2000.

[5] P. Cremonesi and R. Turrin. Analysis of cold-start recommendations in IPTV systems. In *Proceedings of the Third ACM Conference on Recommender Systems*, RecSys '09, pages 233–236, New York, NY, USA, 2009. ACM.

[6] P. Cremonesi and R. Turrin. Time-evolution of iptv recommender systems. In *Proceedings of the 8th International Interactive Conference on Interactive TV&Video*, EuroITV '10, pages 105–114, New York, NY, USA, 2010. ACM.

[7] D. Das and H. ter Horst. Recommender systems for TV. In *Workshop on Recommender Systems, Proceedings of 15th AAAI Conference, Madison, Wisconsin*, pages 35–36, 1998.

[8] B. Engelbert, M. Blanken, R. Kruthoff-Bruwer, and K. Morisse. A user supporting personal video recorder by implementing a generic bayesian classifier based recommendation system. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2011 IEEE International Conference on*, pages 567–571, March 2011.

[9] C. Gurrin, H. Lee, P. Ferguson, A. F. Smeaton, N. E. O'Connor, Y.-H. Choi, and H. Park. Social recommendation and visual analysis on the TV. In A. D. Bimbo, S.-F. Chang, and A. W. M. Smeulders, editors, *ACM Multimedia*, pages 1513–1514. ACM, 2010.

[10] B. Hidasi and D. Tikk. Approximate modeling of continuous context in factorization algorithms. In *Proceedings of the 4th Workshop on Context-Awareness in Retrieval and Recommendation*, CARR '14, pages 3–9, New York, NY, USA, 2014. ACM.

[11] S. H. Hsu, M.-H. Wen, H.-C. Lin, C.-C. Lee, and C.-H. Lee. AIMED: A personalized TV recommendation system. In *Proceedings of the 5th European Conference on Interactive TV: A Shared Experience*, EuroITV'07, pages 166–174, Berlin,

Heidelberg, 2007. Springer-Verlag.

[12] A. Martinez, J. Arias, A. Vilas, J. Garcia Duque, and M. Lopez Nores. What's on TV tonight? an efficient and effective personalized recommender system of TV programs. *IEEE Trans. on Consum. Electron.*, 55(1):286–294, Feb. 2009.

[13] J. Oh, S. Kim, J. Kim, and H. Yu. When to recommend: A new issue on TV show recommendation. *Information Sciences*, 280(0):261 – 274, 2014.

[14] B. Smyth and P. Cotter. Surfing the digital wave, generating personalised TV listings using collaborative, case-based recommendation. In *Althoff K.D., Bergmann R., Branting K. (eds), 'Case-Based Reasoning Research and Development, Proceedings of the Third International Conference on Case-Based Reasoning ICCBR99*, pages 561–571. Springer Verlag, 1999.

[15] K. K. Srinivas, S. Gutta, D. Schaffer, J. Martino, and J. Zimmerman. A multi-agent TV recommender. In *Proceedings of the UM 2001 workshop: Personalization in Future TV*, 2001.

[16] C. Überall, R. Muttukrishnan, V. Rakocevic, R. Jäger, and C. Köhnen. Recommendation index for DVB content using service information. In *Proceedings of the 2009 IEEE International Conference on Multimedia and Expo*, ICME'09, pages 1178–1181, Piscataway, NJ, USA, 2009. IEEE Press.

[17] J. Zimmerman, K. Kauapati, A. Buczak, D. Schaffer, S. Gutta, and J. Martino. TV personalization system. In *Personalized Digital Television*, volume 6 of *Human-Computer Interaction Series*, pages 27–51. Springer Netherlands, 2004.