Personalization: Improving Ease-of-Use, Trust and Accuracy of a TV Show Recommender

Anna L. Buczak, John Zimmerman, Kaushal Kurapati

Philips Research USA
345 Scarborough Rd.
Briarcliff Manor, NY 10510
(anna.buczak, john.zimmerman, kaushal.kurapati)@philips.com
tel. 914-945-6169

Abstract. The plethora of content available to TV viewers has become overwhelming creating a need to help the viewers to find the programs that are the most interesting for them to watch. Towards this end we are developing a personalization system that recommends TV shows to users based on the knowledge of their preferences. For a quicker adoption of the personalization system by users, there is a need for the system to be easy to use, provide recommendations with high accuracy and build trust in the recommendations delivered. The user interface and recommender engine work hand in hand in order to provide all three items. In this paper we describe our system and show how it addresses each of the three issues mentioned.

Introduction

The arrival of Personal Video Recorders (PVRs) has begun to change the way people watch TV. PVRs enable consumers to record TV shows via an electronic program guide (EPG) in a digital format on a huge hard disk. In observing users with PVRs (TiVo and ReplayTV) in their homes, we noticed that within two to three days people shifted from watching live TV to watching recorded TV programs. This change is motivated by both the freedom from broadcasters' schedules and the ability to fast-forward through commercials. However, this change in viewing behavior increases the difficulty of selecting TV programs. Instead of selecting a single program from 100+ channels, PVR users must select a small set of programs to record from 15,000+ on each week. A TV show recommender can aid PVR users by prioritizing these large number of shows. The PTV work of Cotter at al. [1], EPG work of Ardissono et al. [2] and our Multi-agent system [3] stand out in this area as the earliest TV recommender systems.

From our earlier user tests [3], we have learnt that a personalized EPG should provide:

- Ease-of-Use: An intuitive, easy-to-use, interface to browse and search through TV-show listings.
- Trust: Explanations of recommendations in a simple, conversational manner to enable building of consumers' trust in the recommender system.

- Accuracy: An accurate recommender system that can track users' viewing patterns and automatically personalize the interface presentation so as to aid the viewer in making viewing or recording decisions.

One of the key contributions of our work reported in this paper is that our system integrates all the above three requirements for a personalized EPG and improves upon our earlier TV-recommender system. The issues of ease-of-use, trust and accuracy of recommendations are not separate but rather all of them need to be addressed in order for the viewer to feel comfortable about using the system. Ease-of-use is necessary so that all users, regardless of how much they want to get involved and interact with the system, be able to easily find shows interesting for them and feel at ease with the system's everyday use. The issue of trust came about when we noticed that when an unknown show was highly recommended to users, they tended to believe that the recommender was defective. Our goal was to create a method that would give users the activation energy to try a *new program* when the mood strikes. Thus we developed a novel approach called "reflective view history" that explains in a conversational manner why certain shows, unknown to the user, are highly recommended. Of course in order for the "trust building" mechanism to be able to work, the recommendations given by our system need to be very accurate. If the accuracy is not high enough, the system making even the best explanations of why a program is recommended will still be seen as flawed. We developed a new method for achieving the high accuracy: neural network fusion of results from individual recommender engines that we developed over the years.

System Architecture

The complete recommender system under development is comprised of several components (Fig. 1). The main parts are an explicit (E) recommender and two types of implicit recommenders: Bayesian (B) and Decision Tree (DT). The explicit recommender allows users to directly input their likes and dislikes. The explicit recommender has two different interfaces, the feedback interface and the explicit profile interface. Implicit recommenders use the viewing histories of the subjects in order to make recommendations. For each subject there is an individual viewing history and a household viewing history. Based on the view histories, Bayesian and DT recommenders build user profiles (one for the individual and one for the household). When current shows are presented to the system, each recommender generates a set of recommendations. Those recommendations are fused by an artificial neural network (ANN).

Recommenders

Explicit Recommender

The explicit recommender relies on explicit profiles of TV viewers. These profiles result from a question-answer session with the viewer, wherein the viewers' explicit

likes and dislikes towards particular TV channels, show genres and preferred days and times of TV viewing have been gathered [3]. The feedback prong gathers user feedback on specific TV shows and feeds that information to the implicit and explicit prongs and refines their recommender output.

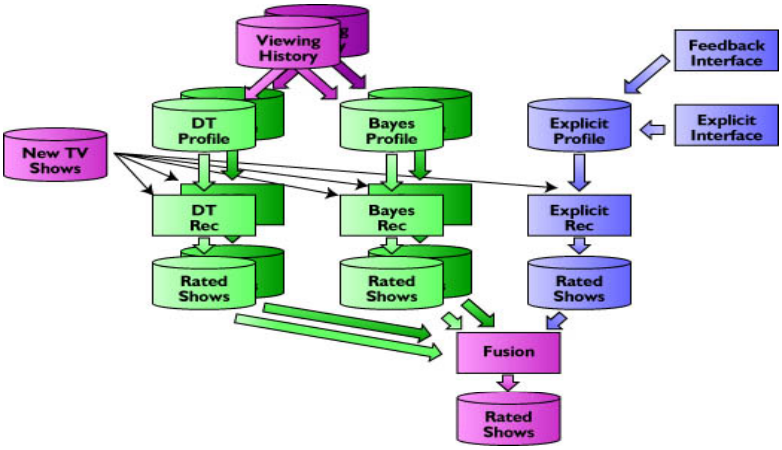

Fig. 1. Recommender System Architecture.

Implicit Recommenders

The implicit recommenders use a viewer's implicit profile, which is built from the viewing history of a TV viewer. The implicit nature of our profiling method stems from the fact that the process does not involve any interaction with TV viewers, regarding their likes and dislikes, other than collecting what shows have been watched. There are two types of viewing histories that these recommenders can use to build the user profile: individual and household. Individual viewing history contains the shows that a given person has watched. Household history consists of shows that the given household has watched. We could see as the history of the TV box - all shows that were watched on a given box are included in the household history. We have developed two implicit recommender agents, one based on Bayesian statistics [3] and one on Decision Trees [4]. Each of the recommender agents can work with the individual or with the household viewing profile resulting in two different sets of recommendations.

Improving Ease of Use with Recommender

During demonstrations and user testing of our recommender interfaces, many users indicated that they wanted either minimal or even no interaction with the system. They did not want to answer a bunch of questions in order to make the recommender work. These users wanted to quickly find something to watch and return to watching

TV. Other users, however, said they wanted to take control of the recommender. They wanted to tweak the system until it produced precise recommendations. Based on this diverse input, we created the following user models:

1. *Do it for me*–These users want a completely automated system. They do not really care how the recommender works; they just want to watch TV.
2. *Let's do it together*–These users want to take some control, but they do not want to spend *too* much time adjusting parameters.
3. *Let me drive*–These users want complete control of the recommender.

Do it For Me Users

The implicit recommender works well for the *Do it for me* users. The system monitors viewing behavior and then makes recommendations. All the users need to do is watch TV. In addition, the interface automatically presents results based on rating. When users activate the system, they see upcoming programs sorted by rating. Placing highly recommended programs at the top of the list reduces the number of shows users need to browse in order to find something they like.

Let's Do it Together Users

The feedback interface (Fig. 2) for the explicit recommender supports the *Let's do it together* users as well as the *Let me drive* users. This interface allows users to modify their explicit profile in small chunks. Instead of displaying attributes for all programs, the feedback interface allows users to rate the program title, channel, genres, actors, and the director for the currently selected program. The interface can also work as a "just in time" explicit profile. When users see a recommendation they disagree with or when they just want to understand why a show has a certain rating, they can quickly see the results and make any changes they want.

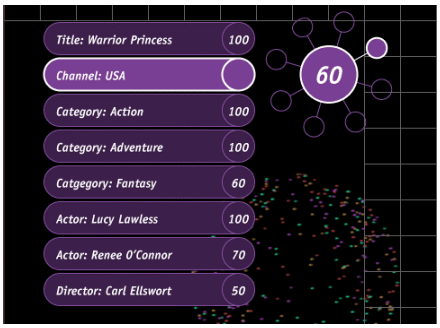

Fig. 2. Feedback Interface expanded.

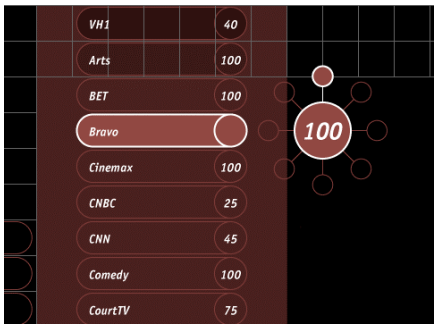

Fig. 3. Explicit profile interface.

Let Me Drive Users

Let me drive users can take more control by accessing their explicit profile interface (Fig. 3) and rating individual program attributes on a 0 to 100 scale. The system initially gives all items a neutral rating of 50. Users can quickly look through the listed items and change as few or as many as they want. Users can adjust all or none of the items, taking as much or as little control of the recommender as they desire.

In the feedback interface, the channel and genre ratings map directly to the explicit profile interface. Users can change the rating of these items without ever visiting the explicit profile interface. Program title, actors, and directors work differently. Initially these items do not appear in the explicit profile interface. The lists would be too long for users to navigate. For example, a complete listing of all actors could easily contain 10,000 names, most of which would be unfamiliar to any specific user. Unlike channel and genre, users do not know enough about all actors, directors and titles to accurately rate them.

When users launch the feedback interface, the program title, actors, and director for the highlighted program are immediately added to their explicit profile. The next time they view their explicit profile interface they can see and modify the rating for these added items.

Improving Trust through Reflective History

During testing of our TV show recommender we encountered a problem. When our system recommended programs users regularly watched, they thought the recommender worked great. However, when the system recommended programs they *did not know*, they felt the recommender was broken. Therefore, we designed a unique feature in our UI/Recommender system, called reflective view history, that explains recommendations in conversational manner thus enabling users to improve trust in the recommendations provided by our system. Previously Herlocker et al. [5] have reported research on explaining recommendations, but they did not focus on a 'conversational' explanation, rather they approached the problem by trying to explain the rationale behind the score derivation.

The *reflective history* displays a conversational sentence justifying highly recommended, new TV shows (Fig. 4). The recommender generates a rating for a week of upcoming shows. The system then looks for highly rated new programs (programs not already in a user's viewing history). Next, it searches for a common person between the new program and programs the user has seen. When it finds an appropriate match, the system generates a conversational sentence:

<NewProgram> <NewTask> <Person> who <OldTask> <OldProgram>.

Example: *Boston Public* stars Jeri Ryan who plays Seven of Nine in *Star Trek: Voyager*.

Table 1. Text strings used in reflective history

Task	NewTask	OldTask
Director	is directed by	directed the TV show
Producer	is produced by	Produced the TV show
Writer	is written by	wrote
Actor	stars	plays <Character> in

The sentence uses a conversational structure, making it sound like something one friend might say to another. This builds on Reeves and Nash's theory [6] that people interact with computers as if they were people. The sentence reveals some of what the system knows about the user. This is a type of self-disclosure that can build trust [7]. The short, conversational sentence works well with our TV recommender. (For a more detailed description of this feature, see [8].)

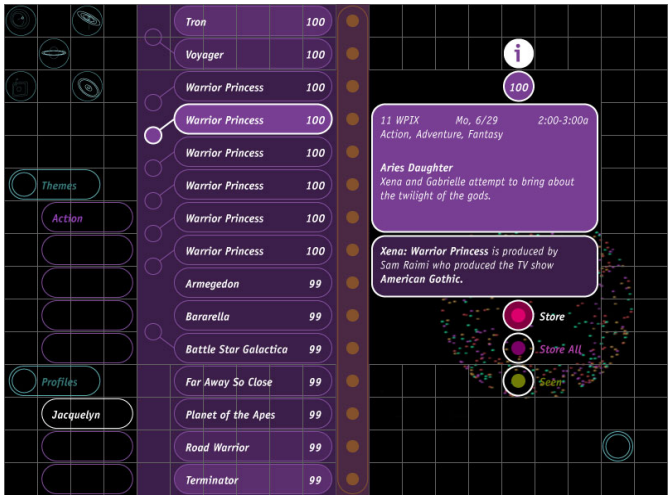

Fig. 4. User Interface with "Reflective View History".

Improving the Accuracy of Recommendations by Fusion

One of necessary elements for user's trust in a recommender system to be achieved is for the system to provide very accurate recommendations, closely matching users likes and dislikes. No explanation of why the show was recommended will build user's trust if the recommendations provided are inappropriate. The third thrust of our research, as described in this paper, was therefore to increase the accuracy of recommendations.

Results of testing our various recommenders (explicit, implicit Bayesian, implicit Decision Tree, heuristic combinations of explicit and implicit) with real users [3] suggest reasonable recommender accuracy. However various recommenders seemed to perform well for various users with no easy way to pre-determine which

recommender might be the right choice for a particular user. Careful consideration of the test results also indicated that different recommenders performed well for very different sets of shows. To improve overall recommender accuracy we *fused* (combined) recommendations from various constituent recommendation algorithms using a neural network that might be able to detect correlations that simple heuristics cannot.

The fusion system combines five individual recommenders in order to obtain the final recommendation. They are:

- Implicit Bayesian based on individual view history
- Implicit Bayesian based on household view history
- Implicit Decision Tree based on individual view history
- Implicit Decision Tree based on household view history
- Explicit

Our approach for fusing recommendations uses a Radial Basis Function (RBF) artificial neural network. The inputs to the network are the outputs from the single recommender mechanisms. This network is trained on partial data from a subset of users. This particular approach has the advantage that it can be developed using ground truth data only from the subjects in our study.

Radial Basis Function Neural Networks

Radial Basis Function networks were chosen for the fusion process. This choice was determined by the fact that RBF nets are universal approximators and train rapidly (usually orders of magnitude faster than backpropagation). Their rapid training makes them suitable for applications where on-line incremental learning is desired. RBF networks usually have three layers: an input layer, a pattern (hidden layer) and an output layer. The nodes in the pattern layer perform a radial basis function transformation, such as Gaussian. The input layer is fully connected to the hidden layer, and the hidden layer units are fully connected to output units. Output units have a linear transfer function. Detailed description of RBF nets and their learning mechanisms can be found in [9].

Data Set

Data from 7 subjects (referred to as A, C, D, F, G, H and I) from our second user test was utilized. Each user rated about 300 shows as "would watch" (1), "wouldn't watch" (0), "may watch" (0.5) and "do not know" (DNK). Users marked shows as "don't know" (DNK) when they were not familiar with the show's title and therefore could not make a decision about being interested/not being interested in watching it. In the study only the shows known to the user were utilized. The total number of known shows with user ratings (1, 0, or 0.5) was 1348.

Metrics

The following three metrics are used in analyzing the fusion results:

- Hit Rate (*HR*)
- False Positive Rate (*FPR*)

- Mean Squared Error (*MSE*)

Hit Rate and False Positive Rate can be computed for all shows that were classified by the user as 0 or 1. For the shows classified as 0.5 it is disputable whether they should be recommended or not. Therefore we computed HR and FPR only on shows that were crisply classified as 1 or 0. However, we included all the shows in the third metric: Mean Squared Error (MSE). For computing HR and FPR, a threshold value needs to be chosen. A higher threshold value will lead to both lower HR and FPR, and a lower threshold will result in higher HR and FPR. The main advantages of MSE metric are that it can be computed for all three types of shows and that it does not require a determination of a threshold value, which can be quite cumbersome. Therefore this is the metric on which we relied the most.

Fusion Results

Several RBF networks with differing number of hidden nodes were trained on data sets from multiple users. 15% to 40% of data from subjects *A*, *C*, and *D* was used as the training set (this represents 26% of the whole data set). For networks' cross-validation 14% to 45% of data from subjects *D*, *F*, and *G* was used (this represents 13% of the whole data set). All the data was used for recall. Data from users *H* and *I* was neither used in training nor in cross-validation (these users are also not a part of households of any other users). The cross-validation was performed using HR and FPR metrics. A threshold of 0.5 was employed to compute HR and FPR. The best performance of RBF networks in terms of HR and FPR was obtained by the cross-validation process for a network with 15 hidden nodes. During cross-validation the best network is chosen from about 10 networks with different number of hidden nodes that were trained with data from the training set.

Fig. 5 shows MSE for all subjects. The RBF net fusion MSE varies for different users from 0.07 to 0.18 per pattern. The average MSE for nine individual recommenders is much higher (it varies from 0.17 to 0.32 per pattern). For all users, with the exception of user *F*, fused MSE is better than the non-fused one. For user *F* non-fused MSE is better by 0.01.

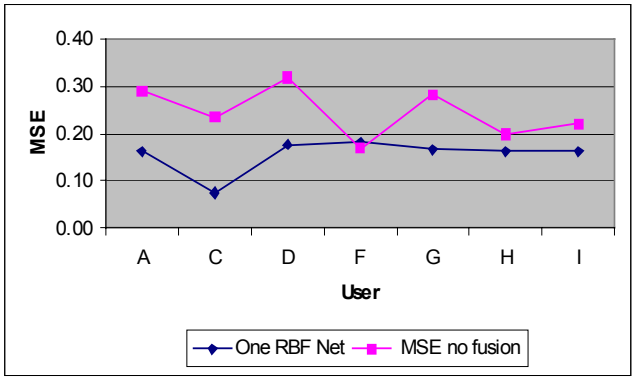

Fig. 5. Comparison of No Fusion MSE and RBF Net MSE.

Overall, these constitute excellent results. They show that the fused system works well both for users who were partially known to the system (some information used in the training set) and users completely new to the fusion mechanism. It is especially important to notice that fused results are superior to the average of non-fused ones for users whose data was neither used in training nor in cross-validation. This proves that the system can be developed using ground truth data only from the subjects in our study and used for other subjects that were not known to the system when developing the fusion mechanism. The fusion network can be viewed as a stereotype network that could be tuned later during system operation to individual user characteristics.

Subjects whose subset of data was used in training of fusion networks can be viewed as those subjects for whom the stereotype network was already almost tuned. The tuning in a real world scenario would take place by using user's feedback to system recommendations.

The behavior of MSE for user *F* needs some explanation: user *F* is an outlier whose behavior is completely different from the other users. This is the only user for which recommenders based on individual watching history were much better than recommenders based on household viewing history. Since user *F*'s data was not used in training, RBF network performing fusion adapted its weights giving more impact to the recommenders based on household viewing history than to the ones based on individual history. These weights were the right choice for all users with the exception of the outlier *F*. The network also adapted its weights for the Explicit Recommender to be high, since Explicit Recommenders usually gave very good recommendations. However *F* is an outlier here as well: Explicit Recommender is the worst recommender for this user.

Conclusions

In this paper, we are addressing the problem of helping TV viewers to navigate through the plethora of content available and finding programs that are the most interesting for them to watch. We are developing a personalized EPG recommender system that recommends TV shows to users based on the knowledge of their preferences. The system consists of several recommender engines and a user interface.

The main thrust of this paper was to develop methodologies for improving three main issues identified in our user tests: ease-of-use, user's trust in the recommendations and their accuracy. We approached the problem from two angles: UI and recommender engine. The ease-of-use is obtained by creating a UI that allows both novice to expert users to achieve the desired amount of interaction with the system, while being able to easily find shows interesting to them.

Increased trust is provided through unique characteristic of our UI & recommender engine called "reflective viewing history" that explains in a conversational manner why certain shows, unknown to the user, are highly recommended. It allows building both trust and forgiveness into the system. In future work the effectiveness of this particular method needs to be numerically determined in a user test.

Trust in recommendations cannot be achieved by their conversational explanation alone. Rather, the system needs to provide a very high accuracy of recommendations and only then explaining them will build the necessary trust in users to try new, highly recommended programs. Improved recommendation accuracy is achieved by performing fusion of results from individual recommender engines by Radial Basis Function neural network. The power of this method is that such a fusion network can be developed based on the data from subjects in our study, whose behavior conforms to the mainstream, and then deployed in the field. When deployed, it will perform well for users that it has not encountered earlier. Later this fusion network could be adapted to individual users by using their feedback to system recommendations. The fusion network can be viewed as a prototype network that could be tuned later during system operation to individual user characteristics.

References

[1] P. Cotter, B. Smyth, "PTV: Intelligent Personalized TV Guides", *Proceedings of the 17th National Conference on Artificial Intelligence*, AAAI 2000, July 30-Aug. 3, pp. 957-964, Austin, Texas, 2000.

[2] L. Ardissono, F. Portis, P. Torasso, "Architecture of a System for the generation of personalized Electronic Program Guides", *Workshop on Personalization in Future TV, User Modeling 2001*, Sonthofen, Germany, July 2001.

[3] K. Kurapati, S. Gutta, D. Schaffer, J. Martino, J. Zimmerman, "A Multi-Agent TV Recommender", *Workshop on Personalization in Future TV, User Modeling 2001*, Sonthofen, Germany, July 2001.

[4] S. Gutta, K. Kurapati, K.P. Lee, J. Martino, J. Milanski, D. Schaffer, J. Zimmerman, "TV Content Recommender System", *Proceedings of the 17th National Conference of AAAI*, Austin, TX, 2000.

[5] J. Herlocker, J. Konstan, J. Riedl, "Explaining Collaborative Filtering Recommendations", *Proceedings of ACM 2000 Conference on Computer Supported Cooperative Work*, pp. 241-250, December 2-6, 2000.

[6] B. Reeves, C. Nass, "The Media Equation: How People Treat Computers, Television, and New Media Like Real People and Places", Cambridge University Press, New York, NY, 1999.

[7] L. Wheeless, J. Grotz,, "The Measurement of Trust and Its Relationship to Self-disclosure", *Communication Research*, 3, 3, Spring 1977, pp. 250-257.

[8] J. Zimmerman, K. Kurapati, " Exposing Profiles to Build Trust in a Recommender", *Proceedings of CHI 2002* (Minneapolis, MN, April 2002), AMC Press, pp. 608-609, 2002.

[9] J. Moody, C.J. Darken, "Fast Learning in Networks of Locally Tuned Processing Units", *Neural Computation*, vol. 1, pp. 281-294, 1989.