<table>
<thead>
<tr>
<th><strong>Title</strong></th>
<th>Let's Get Personal: Personalised TV Listings on the Web</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Authors(s)</strong></td>
<td>Smyth, Barry; Cotter, Paul; O'Hare, G. M. P. (Greg M. P.)</td>
</tr>
<tr>
<td><strong>Publication date</strong></td>
<td>1998</td>
</tr>
<tr>
<td><strong>Item record/more information</strong></td>
<td><a href="http://hdl.handle.net/10197/4411">http://hdl.handle.net/10197/4411</a></td>
</tr>
</tbody>
</table>
Let’s Get Personal

Personalised Television Listings on the Web

Barry Smyth, Paul Cotter, Gregory O’Hare

Department of Computer Science
University College Dublin
Belfield, Dublin 4, IRELAND
{Barry.Smyth, Paul.Cotter, Gregory.OHare}@ucd.ie

Abstract. The future success of the Web as an information resource relies on new technologies to provide a more targeted content delivery system. At the moment users are responsible for locating and selecting content. In the future user modelling and information filtering techniques will partially relieve the user of this burden as new personalised content delivery services come online. PTV is one such service, dedicated to providing users with personalised TV guides. In this paper we describe and evaluate the PTV system, paying particular attention to the techniques used for selecting content that matches the personal preferences of individual users.

1 Introduction

For regular Web users the problem of information overload is all too common. However, it is a problem that is beginning to be addressed by the next generation of Web applications – applications that take the specific needs of individual users into account to deliver a personalised information service. Personalising content necessarily means understanding content consumers (users). Recent research has been devoted to Artificial Intelligence (AI) techniques for automatically modelling and profiling user needs (see example [3,4]) and for matching such models with relevant content items [2, 7]. PTV is one such system. The basic idea behind PTV is the ‘online personalised TV guide’. That is, PTV is a television guide, listing programme viewing details just like any other guide, but with one important difference, the listed programmes are carefully selected to match the viewing preferences of individual subscribers. In short, every subscriber sees a different guide, a guide that has been carefully constructed just for them, taking account of their programme preferences, their preferred viewing times, and their available channels. Crucially, PTV can inform users about programmes that they may be interested in watching.

In the remainder of this paper we discuss the form and function of the PTV system. The next section describes the overall PTV architecture, highlighting the main system modules and basic representation details. Section 3 focuses on the AI component of the system, and describes how case-based reasoning (CBR) and automated
collaborative filtering (ACF) techniques are used for selecting new programmes to recommend to users. Section 4 describes experimental studies carried out to test the accuracy of the PTV personalised guides, and in particular the accuracy of the recommendations made by the system.

2 PTV Architecture

PTV is a client-server system operating over the Web, allowing users to register, login, and view their personalised TV guides as specially customised Web pages. The architecture of PTV is shown in Figure 1. A standard Web browser provides the required client functionality, and all user interaction is handled via the HTML Forms interface. The heart of PTV lies with its server-side components, which handle all the main information processing functions such as user registration and authentication, user profiling, guide compilation, and the all-important programme recommendation and grading.

![PTV Architecture Diagram](image)

Fig. 1. The PTV Architecture.

In the following sections we will concentrate on the user profiling aspects of PTV, focusing on how these profiles are used to deliver personalised content and, in particular, how PTV can make intelligent recommendations to PTV subscribers. However, in this section we will provide a suitable backdrop for these future discussions.
discussions by taking a broad look at the form and function of PTV’s main components.

**Profile Database:** PTV maintains three important databases. The first, the profile database, stores crucial information about each subscriber including: the channels they watch; programme titles; content keywords; preferred viewing times; a list of positively graded programmes; a list of negatively graded programmes; a genre schema; various administrative details such as login times (see profile example in Figure 1). When users register with PTV they are invited to specify some profile details, in particular, viewed channels, programme titles, content keywords, and preferred viewing times. This provides PTV with a starting point for each user, and these preliminary profiles are elaborated upon and refined as PTV learns more about the likes and dislikes of individual users.

**Programme Database:** This database contains programme cases. Each entry describes a particular programme using features such as the programme title, genre information, the creator and director, cast or presenters, the country of origin, and the language; an example programme case for the comedy ‘Friends’ is shown in Figure 1.

**Schedule Database:** This database contains TV listings for all supported channels. Each listing entry includes details such as the programme name, the viewing channel, the start and end time, and typically some text describing the programme in question (see the schedule entry example in Figure 1). The schedule database is constructed automatically from online schedule resources (e.g., online teletext pages and static entertainment guides) by PTV’s schedule agents. Each agent is designed to mine a particular online resource for relevant schedule information and the results of these many parallel searches is the compilation of a rich schedule database.

**Recommender:** The recommendation component is the ‘brains’ of PTV. Its job is to take user profile information and to select new programmes for recommendation to a user. In the next section we will explain how PTV uses a hybrid recommendation approach that combines the feature-less and feature-based recommendation strategies of collaborative filtering and case-based reasoning respectively.

**Guide Compilation:** To compile a personalised TV guide for a given date and user, PTV constructs two programme lists: (1) one consisting of those programmes listed as positive in the user’s profile, along with those programmes selected for recommendation (which of course do not occur in the profile); (2) a list of all programmes to be aired on the specified date by a channel listed in the user’s profile. The intersection of these two lists is the set of programmes that will be used to compile the user’s personalised guide. The guide itself is a HTML page dynamically produced by drawing on programme and schedule information from the appropriate databases.

3 **PTV’s Hybrid Recommendation Strategy**

Historically there have been two separate approaches to recommending personalised content. The more traditional approach is the feature-based approach epitomised by information retrieval (IR) and case-based reasoning research [5; 8]. A more recent
strategy, which is gaining popularity in Web-based applications, is the feature-less approach, epitomised by techniques such as collaborative filtering [2,6,7]. With the former technique, content items and user queries (or profiles) are described using a set of carefully chosen descriptive features. Content recommendation is all about matching content items against user queries (or profiles) in terms of their descriptive features, the best matching items being chosen for recommendation. For PTV this means describing TV programmes (and user tastes) according to features like genre, viewing time and channel, cast, director, etc.

The latter technique does away with the need for explicit feature-based representations, which may be difficult to come-by and expensive to encode. Collaborative Filtering bases its recommendations, for a given user, on the likes and dislikes of other users with similar tastes. Each user can be described as a list of relevant content items (items that have been selected or correctly recommended in the past – TV programmes in the case of PTV). Locating users with similar tastes is then a matter of comparing the content items associated with each user to compute some measure of overlap. In this way users can be clustered into communities with similar interests, and individual users can be recommended items that have proved to be of interest to their virtual community but they have not seen.

In this section we explain how PTV recognises the need for a hybrid recommendation strategy that draws on the strengths of feature-based and feature-less approaches, and we show how PTV implements this hybrid strategy by combining collaborative filtering and case-based approaches.

3.1 Towards a Hybrid Recommendation Strategy

On their own CBR and ACF constitute powerful and successful recommendation strategies. However they also suffer a number of shortcomings [1]. For a start, feature-based approaches such as CBR incur a knowledge-engineering cost. Each recommendable content item must be carefully described by a set of features that can be used as the basis for similarity assessment. The wrong set of features will inevitably lead to poor recommendation performance. Moreover, it is often difficult to determine a good set of features that will serve to accurately describe a wide range of recommendable items. For instance, in PTV’s domain, the obvious features for describing TV programmes include genre, time, channel, cast, director, writer, etc. However, these features, while predictive and useful in their own right, often fail to capture subtle similarities and differences between programmes. In particular, programme genre is a very difficult category to tie down with any great accuracy, and in fact, PTV uses four genre categories in an attempt to expose subtle differences between apparently similar programmes. In practice, even with these carefully crafted programme descriptions, mistakes still do occur. For instance, two programmes that appear very similar according to PTV’s CBR component are ‘Friends’ and ‘One Foot in the Grave’. However, most people familiar with these two modern comedies will recognise that they appeal to very different types of viewer, a fact that is ignored because of features that are missing from their case descriptions; Friends appeals to
the ‘young trendy set’ while One Foot in the Grave is judiciously watched by my parents!

The feature-less approach of ACF does not suffer in the same way as CBR, and in fact the identification of subtle similarities between programmes is one of the strengths of the collaborative strategy. By drawing on the shared experiences of a large community of users, ACF can capture the nuances of peoples’ preferences to offer convincingly appropriate recommendations. However, if the power of ACF rests in its use of large communities of users, so do its weaknesses. First of all, without a sufficient user base accurate ACF is an impossibility, so the ACF approach is only really useful once a large and varied user-base has been established. Secondly, and perhaps more worryingly, even with a suitable user-base ACF can only recommend content items that have been available for long enough to have propagated through the communities; if a new item (or one-off item) becomes available it will not be recommended until enough users have found it (by other means), no matter how relevant the item is to a user.

Fortunately, just as ACF covers the cracks in CBR’s feature-based approach, so too CBR can compensate for ACF’s shortcomings. By creating a hybrid recommendation strategy we can use ACF to recommend common content items that have made their way into large virtual communities, while at the same time using CBR to recommend new or one-off items, or to recommend items to unusual users that fall outside of existing communities.

### 3.2 Collaborative Case-Based Recommendation

PTV’s hybrid recommendation strategy is outlined in Algorithm 1 (for a related approach see [1]). The first part (steps 1-3) deals with regular users and regular programmes; that is, users that are well represented by the current user-base and programmes that have made their way into user profiles. Under these conditions an ACF approach is used. Notably, we employ a lazy-ACF approach rather than the traditional eager-ACF strategy so that explicit user communities are not constructed a priori. Instead, a nearest-neighbour strategy seeks to identify n profiles that best match the current user (step 1 below). These profiles then act as the recommendation community and a ranked collection of recommendable programmes is formed by identifying those programmes that are in the community profiles but missing from the current user profile (step 3) – the issues of profile matching and ranking are discussed briefly in the next section.

The second part of PTV’s recommendation strategy caters for unusual users, that is users that do not match well with existing users so that a good recommendation community cannot be formed – this same method can be used to recommend new or one-off programmes but this aspect of PTV is not discussed further here. If a suitable recommendation community cannot be formed (step 4 below) a CBR approach is used to drive recommendation. Again a nearest-neighbour algorithm is used to identify and order a suitable set of ‘n’ programmes according to their similarity to the current user’s profile (see Section 3.4 for further details).
The end result then is a set of recommendable programmes (ordered according to their suitability). During guide compilation (see Figure 1) PTV will include the best (most suitable) programmes from this list in the current guide. In actual fact certain programmes may be eliminated from this initial list (if, for example they are part of a user’s negative programme list), but this post-processing stage is not discussed further here.

**Inputs**  
- \( u \): Current User Profile  
- \( U \): User Profile Base  
- \( S \): List of programmes scheduled programmes  
- \( C \): Programme Case-Base  

**Outputs**  
- \( P \): A ranked list recommendations  

**Recommend**\( (u, U, S, C) \)

1. Community = Best_Profiles\( (u, U) \)  
2. If Community  
   3. \( P = \) Best_ACF_Programmes\( (u, \) Community\( ) \cap S \)  
3. Else if  
   4. \( P = \) Best_CBR_Programmes\( (u, C \cap S) \)  
5. End if  
6. Return \( P \)

---

**Algorithm 2. PTV’s Recommendation Algorithm**

### 3.3 Profile Similarity & Ranking

The key to PTV’s ACF component is its profile similarity and ranking procedures, Best_Profiles and Best_ACF_Programmes respectively. The former takes nearest-neighbour approach, computing the similarity between the current user’s profile \( u \) and the stored profiles \( u’ \) (see Equation 1), in order to return and rank the top \( n \) profiles according to this similarity.

\[
\text{Profile\_Similarity}(u, u') = \frac{\sum_{i \in p(u)} |p_i^u - p_i^{u'}|}{4 \cdot |p(u) \cup p(u')|}
\]

Where \( p(u) \) and \( p(u') \) are the ranked programmes in each user’s profile, and \( r(p_i^u) \) is the rank of programme \( p_i \) in profile \( u \) – programme rankings range from -2 to +2.
This normalised metric has been designed to take account of both positive and negative programme recommendations with 0 indicating a perfect profile match. Programmes missing from one of the profiles are given a default ranking of 0 to allow for meaningful comparisons with their ranked counterparts in the other profile.

Once a set of similar profiles has been selected, the next stage is to identify and order a set of recommendable programmes from these profiles; this is the job of Best_ACF_Programmes in Algorithm 1. The identification is straightforward, namely those programmes that are listed (positively) in the selected profile but that are absent from the current user profile. The ranking of these programmes is based on the formula shown below in Equation 2, which biases programmes in two ways:

1. Programme Frequency – programmes that occur more frequently in the selected profiles should be preferred over those that do not;
2. Profile Similarity – programmes that occur in very similar profiles should be preferred over those that occur in less similar profiles.

\[
\text{Programme Suitability}(p, u) = \sum_{u \in U} \text{Profile Similarity}(u, u')
\]

Where U is the set of all selected similar profiles containing the programme p.

### 3.4 Programme Similarity & Ranking

The similarity between a programme case and a user is more complex to measure than profile similarities because user profiles are not represented using the same feature vocabulary as programme cases (see Figure 1). For this reason PTV constructs a profile schema for each user to describe, in terms of programme features, the type of programmes that a given user watches.

Unfortunately, the profile schema creation algorithm is beyond the scope of this paper. Briefly, it computes a summary representation of the positive and negative programmes listed in a profile, weighting different programme features (genre, origin, etc.) according to their relative frequency distribution across the profile. The similarity between a programme case and a user can then be computed by a standard weighted similarity metric such as that shown in Equation 3.

\[
\text{Programme Similarity}(\text{Schema}(u), p) = \sum w_i \cdot \text{sim}(f_i^{\text{Schema}(u)}, f_i^p)
\]

Where \(f_i^{\text{Schema}(u)}\) and \(f_i^p\) are the ith features of the schema and the programme case respectively.
4 Experimental Studies

The first online evaluation of PTV began in the last week of March and the results presented in this section were compiled over a seven-day period. The system was advertised to computer science students in University College Dublin and Trinity College and during the experimental period there were approximately 200 registered users and a total of 2000 individual guides were requested. The main objective of this experiment is to test the accuracy of the recommendation system used by PTV. The question to be answered is whether or not users consider PTV’s recommendations useful.

4.1 Experimental Setup

During the experimental period the PTV system contained a case-base of 400 programmes, which provided about 30% coverage of a typical week of television, and about 60% coverage of the prime-time viewing slots. Unfortunately, for this experiment, we could not fully test the collaborative filtering component of the recommendation system because the critical mass of user profiles needed to kick-start ACF could not be achieved by the beginning of the experiment. Therefore, the recommendation system relied heavily on the case-base reasoning component in order to select suitable programmes for recommendation; we expect any results reported here to be improved upon once the collaborative filtering technique begins to take effect.

4.2 Experimental Method

Each time a new programme is recommended as part of a personalised guide, the user is invited to grade the recommendation as either good or bad; on average three programme suggestions are made to each user per guide. These gradings are stored and were used as the raw data for the experiment; each grading encodes the username, the programme name, the date, and the grade itself. Over the experimental period a total of 752 individual gradings were saved from 79 different users; on average each user was submitting 9.5 grades over the one-week period. To provide a comparison benchmark data set we also generated a set of random programme recommendations, and collected a comparable set of rating data.

4.3 Experimental Results & Analysis

The results can be interpreted in a number of ways. One way is to compute the percentage of recommendations that were given a ‘good’ rating – 45% for PTV. However, we feel that this is not an effective way to evaluate the accuracy of PTV’s suggestions. As we have mentioned already, users were requesting an average of three recommendations per daily guide. The problem is that it is rare indeed to find three
programmes a day that you would be prepared to start watching. Therefore, we feel that a better measure of recommendation success is to take each user guide in turn, and to count the percentage of users that received ‘n’ or more good recommendations per day. This will allow us, for example, to count the number of users that received at least 1 good recommendation per day, which to our minds is a far better way of evaluating recommendation success.

The graph shown in Figure 2 displays the results for values of n from 1 to 3; that is, it displays the percentage of users faced with at least 1, 2, and 3 good recommendations per guide. The corresponding values for the random recommendation data are also displayed. Clearly, the results for PTV are very positive with 78% of users receiving at least one good new programme suggestion per day. By comparison only 27% of users found one of the random recommendations to be worth watching. In fact, the results show that PTV recommended 3 good programmes a day more often than the random method recommended 1 good programme (32% versus 27%).

We believe these results to be an excellent starting point for PTV, and we are optimistic that they will be improved upon once the user-base grows to a level that facilitates true collaborative filtering. In addition, further improvements will be forthcoming once the case-base is extended to ensure better programme coverage. At the moment nearly 40% of prime-time programmes are never considered for recommendation because they are absent from the case-base.

5 Conclusions

In this paper we have introduced a recommendation system for producing personalised Web-based TV guides. The system uses a hybrid approach to
recommendation, combining the approaches of collaborative filtering (feature-less recommendation) and case-based reasoning (feature-based recommendation). The main system architecture has been described along with the details of its recommendation technique.

We have also presented an evaluation of the system that demonstrates the effectiveness of PTV’s recommendation components compared. In particular, PTV makes good recommendations nearly 80% of the time, a result that far exceeds the quality of random recommendations (27% success rate).

Future research will continue to develop and evaluate PTV’s recommendation approach and interested readers are encouraged to use the system and contribute to our ongoing evaluation of PTV at http://www.cs.ucd.ie/ptv/.

References