Foundations for building a digital sister-in-law: explicitly building scrubtable user models from critical reviews

Sacha Groves
Basser Dept of Computer Science
University of Sydney
AUSTRALIA 2006
sacha@cs.usyd.edu.au

Judy Kay
Basser Dept of Computer Science
University of Sydney
AUSTRALIA 2006
judy@cs.usyd.edu.au

Abstract

Increasing amounts of digital information create the need for high quality recommender systems. This paper explores one critical aspect in building such systems: this is the construction of quality models of influential user’s recommendations. These can serve as a foundation for automated recommender systems.

We describe PEER REVIEW, a system for explicitly eliciting user models based upon critical reviews of movies. The system includes DH, an interface that enables a user to create a set of simple metadata which captures the essence of their reading of a critical review: the features they considered noteworthy in the movie, the things they liked and those they disliked. We describe how these are built into a user model, which represents the reviewer’s preferences for movies and movie-attributes.

Importantly, the process we describe provides a scrubtable user model. This paper describes how we enable the user to ’scrutinise’ the user model to see explanations of the conclusions it holds about the user: these explanations are links back to the relevant parts of the reviews used to build the user model.

We describe two evaluations of this idea. The first assessed whether average users would consistently detect parts of a review related to relevant metadata. This evaluation indicated that users were moderately consistent with their understanding of the review. The second assessed one approach for assessing the codings for a review taken in the design of the DH interface. It indicated that users were well able to use this interface. At the same time, there were minor inconsistencies in codings. We propose a mechanism to overcome these.

Keywords user models, metadata, scrubtable, elicitation interface, usability

1 Introduction

When I want to go out to the movies, rather than reading reviews, I ask my sister-in-law. We all have an equivalent who is both an expert on movies and an expert on us. What we need to build is a digital sister-in-law. [5]

Increasing amounts of digital information create the need for high quality recommender systems, like Negroponte’s sister-in-law in digital form. To build such a system, we need the following elements:

- a user model for the movie preferences of the user who wants a recommendation;
- a similar model for the ’digital sister-in-law’;

This paper describes a system we propose for eliciting subjective codings of reviews of movies. This
is used to construct the second of these models, the user model of a movies expert.

In addition, we would like to build a scrubtable recommender which could explain its recommendations. This means that the user should be able to scrutinise a recommendation to determine how it was made. This means that the user should be able to access the user models which underpin the recommendation as well as the processes involved in the recommendation. So, for example, a recommendation for *Crouching Tiger, Hidden Dragon* might be explained in terms like these:

Your user model shows you like martial arts movies and *Crouching Tiger, Hidden Dragon* is in this genre, as indicated by the following quote by Roger Ebert in a review "'Crouching Tiger, Hidden Dragon' is the most exhilarating martial arts movie I have seen'.

Similarly, Ebert's review has '[the story has] depth and poetry' and your user model indicates you like films with a strong story.

We believe there is an important place for scrubtable in recommenders. Part of this is due to issues of the user's right to access and control the private information that their own user model holds. Equally, we believe that it is important that the relationship between a user and the machine-recommender should allow the user a sense of complete control and the opportunity to really understand the processes driving recommendations.

One source of high quality information about movies is the critical reviews such as those in newspapers and web sites [2, 1]. Typically, these give detailed reports of the movie and a critical assessment of the strengths and weaknesses of the movie. We are constructing a system which can exploit such reviews to make movie recommendations. One obvious appeal of reviews is partly due to the quality of information that they carry. Another, less obvious benefit is that the review itself can serve as the basis for the scrubtable of a recommendation; where the recommender system is asked to explain its advice, it can refer back to the parts of the review which were used to inform the system's model of the movie.

Although this work is in the domain of movies, we are designing the tools so that they should be applicable to other areas where it makes sense to review objects. For example, we propose to make similar use of reviews of software for teaching and learning.

2 The roles of the model

Figure 1 gives an overview of the approach we take to building models of critic's movie preferences. It illustrates how we want to use a collection of reviews of movies to define two sets of models:

- user models for several reviewers;
- a collective model that is built from the individual user models for several users.

Beginning at the left of the figure, we show two movies. First, we will consider *Movie A*. The figure shows that we have two reviews for that movie, one by reviewer 1 and the other by reviewer 2. Each of these is coded using our DH interface, with three different users (which we refer to as coders) performing the coding for the first review and two coders coding the second review. Note that one of the coders doing the coding may well be the person who wrote the review but equally it might be another person. We separate the roles of reviewer and coder so that we can take advantage of the existing collections of reviews and so that this process can operate even if the reviewers do not wish to do the coding of their reviews.

![Figure 1: Overview of the process of building individual reviewer models and collective models.](image)

The coding process involves the coders examining the review and the metadata associated with the movie being reviewed. They match the parts of the review that they feel refer to specific elements of the metadata, and make a subjective decision as to what the reviewer feels about the metadata.
element. So, for example, if reviewer 1 writes in their review "The director shows absolute mastery of filmmaking", coder1 may decide the phrase indicates that the metadata Director: John Smith is something that the reviewer feels was a positive aspect of the movie.

Once we have codings for a review, we can collect all those that apply to reviews of a particular reviewer and build these into a user model. In the figure, we show the case where reviewer 1 has written a review for only Movie A, but reviewer 2 is shown to have written reviews for both Movie A and Movie B. All the codings for both of these movies can be used to construct the user model for reviewer 2. This is indicated by the solid and dashed lines feeding into reviewer 2's user model. In addition, the figure shows dotted lines, representing data from the codings of the reviews of other movies.

Once we have constructed individual user models for several reviewers, we can combine them to create a collective model. When we want to recommend a movie to a user, we might make use of the user model for a particular reviewer or a collective model over several reviewers.

Note that the coders make an entirely subjective decision and the resulting additions to the reviewer’s user model must be considered as such. It is possible that different coders may be inconsistent with each other, but this is expected as any piece of creative writing will invariably be interpreted differently between different people. This inconsistency can be exploited as a further evidence source when making a recommendation. So, for example, a recommendation may state that reviewer 1 must like director John Smith as 20 coders stated that the phrase "The director shows absolute mastery of filmmaking" indicates a liking of the director.

A final and important element in the whole process is that the user receiving a recommendation for a movie should be able to scrutinise the following elements in the process depicted in Figure 1:

- the user model that was involved in constructing the recommendation (where this might be an individual reviewer’s model or a collective model);
- the detailed codings used to build the user model;
- the parts of the actual reviews which were the basis for the codings.

There are many parts to the whole process of constructing the user models, the recommendations and the information for the user to scrutinise.

3 DH interface for explicitly constructing user models from reviews

![Figure 2: Example DH screen shot](image)

Figure 2 shows a sample screen for the DH interface. It has two main parts. The large area on the left is for viewing a review. For the work described in this paper, we are concerned with the coding of existing reviews. In general, the area to the left of the screen can be used to enter or edit reviews.

The right side of the screen provides the support for coding the review. This has two stages. First, one must define the elements to be coded. For example, the actors mentioned in the review are elements that might be coded. Equally, the other movies can be coded, along with other attributes of the movie mentioned in the review. For the present experiment, we have done this stage, defining identifiers for the movies, actors, director and other people mentioned in the review as well as the genres. In the longer term, this stage can be performed either automatically or with some user involvement. It is important that the identifiers be unique so that different user models and recommender tools can be unambiguously refer to these components of user models.

The second stage, and the subject of this paper, is the coding of the elements as follows:
+ for elements classified as good;
- those coded as bad;
* coding an element as present in the review but
not judged as either good or bad;

For example, if in the review the reviewer
mentioned "Actor X was quite good", they would
code the metadata "Actor:X" as '+'. The more
coders that code this element in the same way,
the further evidence a recommender has that the
reviewer feels this way.

Initially, we felt the above would be simple
to code, yet descriptive enough to deconstruct
a review into a form useful for a recommender.
However, after examining numerous movie reviews,
we noticed that most reviews compare the film
being reviewed to other films and felt that the
following would be useful additions:

< not as good as the film named;
> better than the film named;
= about as good (or bad) as the film named.

So, the phrase "Movie A is similar in theme to
Movie B, but nowhere near in quality" would be
coded as "<Similar: Movie B".

It is this coding that the current work evaluated.
Clearly, the coding is a gross simplification of the comments in the review: they often contain
considerable subtlety. However, the later processes
involved in building the user models and the recom-
mender system will be greatly simplified if such
codings can be used effectively.

It should be noted that another assumption we
make is that reviews are written with a broad audi-
cence in mind and therefore aim to give that au-
dience a common understanding of the reviewer’s
views of a film. Thus is it expected, and indeed empirically tested (see section 5), that the codings
will be reasonably consistent.

4 The form of the user model

The user models we construct are kept in the um
representation [3] which models each component of
a user model by holding a collection of evidence
available about it. So, for example, the component
modelling a reviewer’s preference for the actor Russell Crowe might be like that shown in Figure 3.

The first line gives the name of the component, the
actor Russell Crowe. The lines that contain DH
Rating are related to the codings '+', '-' and '"'.
'+' indicates support, '-' indicates negate and '"' is
taken as weak evidence of the overall rating of
the film. So, a film that the reviewer likes would
have '"' codings indicating support and vice versa.

All but the last of these is coded as support meaning
these pieces of evidence suggest the user likes
Russell Crowe. This piece of evidence is coded as
negate because it indicates the user dislikes this actor. The lines that contain DH Comparison relate
to the '<?, >=' codings. Similar to the above,
'>' indicates support, '<' indicates negate and '='
is support if the element being compared is consi-
dered good by the reviewer and negate if bad.

The design of um is driven by the goal of building
scorable systems. So this evidence can be used to
explain why a system might conclude that the user
likes Russell Crowe: the first 7 pieces of evidence
indicate this and only the last, rather odder piece
of evidence indicates otherwise.

| Actor: "Russell Crowe"
| Support given coder1 DH Rating Rev921127-130 (Aug 20 2001)
| Support given coder2 DH Rating Rev921127-130 (Aug 20 2001)
| Support given coder3 DH Rating Rev921131-300 (Aug 20 2001)
| Support given coder4 DH Rating Rev921300-330 (Jan 11 2000)
| Support given coder5 DH Rating Rev921300-330 (Jan 11 2000)
| Support given coder6 DH Rating Rev921300-330 (Jan 11 2000)
| Support given coder7 DH Rating Rev921300-330 (Jan 11 2000)
| Support given self DH Comparison Rev92135-136 (Aug 20 2001)
| Support given self DH Comparison Rev92135-136 (Aug 20 2001)
| Support given self DH Comparison Rev92135-136 (Aug 20 2001)
| Support given self DH Comparison Rev92135-136 (Aug 20 2001)
| Support given self DH Comparison Rev92135-136 (Aug 20 2001)

Figure 3: Example model for a reviewer’s preferences for a particular actor

For the design of DH, the important aspect of
this representation of the user model is that vari-
ous coder’s coding of a review can provide evidence
like that shown in the figure. For example, the first
three pieces of evidence were collected on the same
day (as indicated by the stamp at the end of each)
and all indicate the reviewer modelled likes this ac-
tor. All indicate the review likes Russell Crowe
(hence the support in the first part of the evidence).
All were provided by a direct elicitation interface,
meaning that the information was given directly
(rather than being based on some other source type,
for example, user observation). The actual coders
are indicated by the identifiers coder1, coder2 and

1The um system allows any recommender program to
make its own decisions about how to treat the evidence,
so, if necessary, the codings can be treated differently
coder’s in the third part of each piece of evidence. All were based upon the use of DH to code a particular review, which is identified by Rev921. The numbers after this identifier indicate the position of the relevant text within the review. (In the figure, where this is shown as 00, indicates that the actual position is not known.)

Note that the pieces of evidence labelled self were provided by the user themself. Although this particular example has all evidence coming from DH, in general, there might be other sources of evidence, provided by other programs.

5 Evaluation of the interface

This formative experiment was designed to evaluate the coding of elements in the prototype DH interface. It aimed to assess:

- User Interface Issues: the ease of use of the interface for coding a review, given a brief set of instructions;
- Coding Issues: whether coders could code a critic’s review as a set of structured elements;
- Consistency Issues: the consistency in the coding between coders.

The coders (who we will refer to as participants) were all asked to code a single movie review. Roger Ebert’s review of the movie "The Matrix" [2].

The experiment took the participants through the following steps:

1. The author gave a brief overview of the task to be done, explaining that the codings ‘+’, ‘-’, ‘=’, ‘<’, ‘>’ had the meanings as we described them above (in section 2). The participants were also told that the purpose of the interface was to enable structured coding of the information in a free natural-language critical review of a movie.

2. The participant read the review carefully. Since no participant was the author of this review, all had to come to terms with the review

3. They then went through the supplied metadata terms for the movie, using these to code the review.

4. During the process, the participants commented on what they were doing and, in keeping with the cooperative think-aloud approach [4] suggested improvements to the interface.

5. After the coding had been completed, each participant was asked to review the codings and, for each one, to select and highlight the part of the review which was the basis for this coding.

The participants were three Computer Science postgraduate students. All had completed a course in user interface design and were avid movie buffs. This choice of participants means that we need to take care in interpreting this evaluation. The choice of movie buffs was desirable since we wanted the coding to be done by people who would be likely to appreciate the subtleties of the review and to take account of the broad range of elements that reviewers like Ebert mention. On the other hand, the strong computer science background of the participants makes their assessment of the interface closer to that of an expert. Since our main concern was to assess the effectiveness of the overall coding strategy, these participants were a satisfactory population for a formative assessment.

5.1 Observations

The results are in the three parts noted at the beginning of this section.

5.1.1 User Interface Issues

Overall, the participants found the interface straightforward to use. They completed the task with minimal awareness of the interface, being able to focus on the problems of deciding on the meaning of the review and how to code it.

One problem was found. To select a metadata element for coding, the interface required the participant to double click the element. The participants, however, felt that a single click would be more intuitive.

5.1.2 Coding Issues

In general, the participants had little problem with coding the review. They all understood implicitly
what the coding elements meant and generally how to apply it to the review.

One participant felt that the "black and white" nature of the coding elements made some decisions difficult. For example, they felt the whilst the reviewer enjoyed the action in the movie, the reviewer clearly stated that the action was out of place.

Another participant suggested that the less than, greater than and equal to codings could be extended beyond their original purpose, to assess and attribute in relation to another film, or in relation to previous work by the artist. For example, less than in relation to "Actor:Keanu Reeves" would mean "In the current film, Keanu Reeves was not as good as in his previous work".

5.1.3 Consistency Issues

Table 1 summarizes the results of the coding. Coder 1 is the first author of this paper and the creator of the interface. Coders 2-4 are the three participants in the experiment. A bullet (●) in the column for a participant indicates that they coded the element listed at the end of that row. For example, the first row shows that Coder 2, 3 and 4 coded the metadata element Actor:Carrie-Anne Moss as '+' and that Coder 1 did not code that element.

The highest consistency is indicated by an asterisk in all four coder columns. For example, the "Actor:Keanu Reeves" element was coded as '+' by all four coders. The lowest consistency is indicated by a single bullet in a row. For example, the element "Producer:Joel Silver" was coded by only one coder.

The only elements where all coders were consistent was that the actor Keanu Reeves was a positive aspect of the movie.

In addition, at least three of the coders (including the author) coded the following:

- the actor Laurence Fishburne was a positive aspect of the movie;
- the two directors, Andy Wachowski and Larry Wachowski were a positive aspect of the movie;
- the special effects were a positive aspect;
- Dark City was compared with the movie, although the codings are different, one as < and one as +.

The last of these is particularly interesting and was followed up. Note that participants were instructed that '+' coded positive elements of the film and that '<' meant the film was not as good as the other film. Here is the part of the review mentioning Dark City:

"Dark City"...offered intriguing motivations for villainy. "Matrix" is more like a superhero comic book...It's cruel, really, to put tantalizing ideas on the table and then ask the audience to be satisfied with a shoot-out and a martial arts duel.

We can see that this does indicate that The Matrix was being judged poorer than Dark City and, at the same time, that it is quite positive about The Matrix. Both codings are reasonable.

One definition of inconsistency in codings is: one coder assigned the '+' code and another the '-' code; or '<' was assigned by one and '>' by another. Taking this definition, there were no cases of inconsistency.

At the same time, there were many cases where only some of the elements were coded by the coders.
Consider, for example, the codings associated with the actors. The main comments about the actors is in the following paragraph of the review:

And it has performances that find the right notes. Keanu Reeves goes for the impassive Harrison Ford approach, "acting" as little as possible. I suppose that's the right idea. Laurence Fishburne finds a balance between action hero and Zen master. Carrie-Anne Moss, as Trinity, has a sensational title sequence, before the movie recalls that she's a woman and shuttles her into support mode. Hugo Weaving, as the chief Agent, uses a flat, menacing tone that reminded me of Tommy Lee Jones in passive-aggressive overdrive. There's a well acted scene involving Gloria Foster as the Oracle, who like all Oracles is maddeningly enigmatic.

Another important difference in the coding applies in the case of most of the codings that are relative to other movies. This seems to be because the review both compares and relates The Matrix to other movies. For our user modelling purposes, we can see that the current coding is effective in enabling the user to code the relationship with other movies. However, it seems that it lacks the power to capture some of the most useful user modelling elements.

6 Discussion and conclusions

THIS IS A MESS IN THE VERSION OF THE OZUI THAT I HAVE, SO I SUGGEST WE START FROM SCRATCH.

Need to say:

- '+' and '-' not good enough -> goto slider
- < and > and = are too tricky to capture what we want, and it isn't worth having for time being
- conclusions of the pen and paper
- codings are close enough, and besides we record different peoples codings, so can add another level to the recommendation -> people who code like you.

- next step is to use the codings in a recommender that is scutable etc etc

7 Stuff left over/not used

![Figure 4: This picture is in the paper's directory - why?](image.png)

- Need to talk about pen and paper experiment
- Figure 4 is not used.

References