Title	Passive Profiling from Server Logs in an Online Recruitment Environment
Authors(s)	Rafter, Rachael; Smyth, Barry
Publication date	2001-08
Conference details	Workshop on Intelligent Techniques for Web Personalization at the the 17th International Joint Conference on Artificial Intelligence, Seattle, Washington, USA, August, 2001
Item record/more information	http://hdl.handle.net/10197/4638

Passive Profiling from Server Logs in an Online Recruitment Environment

Rachael Rafter, Barry Smyth
Smart Media Institute,
Dept. Computer Science, University College Dublin,
Belfield, Dublin 4, Ireland
{rachael.rafter, barry.smyth}@ucd.ie

Abstract

The success of recommender systems ultimately depends on the availability of comprehensive user profiles that accurately capture the interests of end-users. However, the automatic compilation of such profiles represents a complex learning task. In this paper, we focus on how accurate user profiles can be generated directly from analysing the behaviours of Web users in the CASPER project. In CASPER user profiles are constructed by passively monitoring the click-stream and read-time behaviour of users. We will argue that building accurate profiles from such data is far from straightforward. In particular, we will describe the techniques that are used in CASPER to generate high-quality graded user profiles from server logs. Finally, we will describe a comparative evaluation of these different techniques on real user data.

1 Introduction

Information overload is now a well-acknowledged problem for Internet users today who often have to search through a vast amount of irrelevant information in order to find the information they actually want. This problem is of course set to rise as the Internet continues to grow at an exponential rate. In response, researchers are looking towards new solutions like *automatic content personalisation* techniques and *recommender systems* to prioritise the delivery of information for individual users based on their learned preferences.

Ultimately, the success of content personalisation technologies depends critically on the availability of comprehensive user profiles that accurately capture the interests of end-users. However, the automatic compilation of accurate profiles represents a complex learning task. In this paper we address the issues involved in automatically learning user profiles and focus on a key assumption that has been adopted by some systems: that accurate user profiles can be generated directly from analysing certain behaviours such as the click-stream (the links that users click) and read-time data (how long a user spends reading a given page) of Web users. Although this idea has been proposed for many recommender systems already, it has rarely been rigorously tested.

In this paper we focus on the CASPER system which investigates applying personalisation technologies such as case-based reasoning [Bradley *et al.*, 2000] and collaborative filtering [Rafter *et al.*, 2000; Rafter and Smyth, 2001] to JobFinder (www.stepstone.ie) [1] - an online recruitment service. Currently, JobFinder relies on traditional database technologies which make it prone to the information overload problem. Specifically, our goal here is to evaluate the validity of the aforementioned assumption: that accurate user profiles can be generated by analysing user behaviour in the CASPER system. CASPER constructs user profiles by passively monitoring the click-stream and read-time behaviour of regular users. Specifically, we will argue that building accurate user profiles from click-stream and read-time data is far from straightforward. In particular, we will describe some of the techniques and metrics that are used in CASPER to generate high-quality graded user profiles from raw server logs. Finally, we will describe a comparative evaluation of these different techniques on real user data.

2 Background

In general, recommender systems rely on user profiles in some shape or form, and one of the most common strategies is to profile users by recording their level of interest in specific information items, thus generating user profiles that consist of content item identifiers and their associated ratings. Current research can be usefully classified along two dimensions: the content filtering strategy used versus the type of user profiling employed. Content filtering methods are generally either *content-based* or *collaborative*, while user profiling techniques are either *active* or *passive*.

The different styles of content filtering methods employed in recommender systems both adopt familiar social recommendation policies albeit from different perspectives. Content-based techniques (e.g. case-based reasoning) seek to recommend similar items to the items that a user has liked in the past. This necessarily involves comparing candidate content items to the content items listed in the user profile, preferring those targets that are similar to items that the user has rated positively and dissimilar to the items that the user has rated negatively. In contrast, content-free techniques (e.g.

---

[1] JobFinder has since been bought by StepStone - was originally www.jobfinder.ie

collaborative filtering) seek to select items for a given user that *other* users (with similar tastes) have liked. This typically involves identifying users that have similar preferences to the target user, in the sense that they have rated some of the same content items in the same way; that is, users whose ratings correlate positively with the target user.

Active and passive user profiling techniques differ in the manner in which they collect profile information from users. Active user profiling techniques essentially transfer the profiling problem onto the user in the sense that the user is required to provide explicit profiling information, such as content ratings. For example, PTV [Smyth and Cotter, 1999] generates personalised TV guides for individual users based on their profiled preferences which are gathered by encouraging users to rate the programs in their guides as positive or negative. Thus, PTV only learns about its users by actively engaging them in the profiling process.

In contrast, passive user profiling techniques remove the burden associated with explicitly rating items from the user, and rather try to infer this information implicitly. To this end, these techniques monitor certain browsing/searching behaviours of the user which are thought to be indicative of that user's interest. For example, the GroupLens Project [Konstan *et al.*, 1997; Sarwar *et al.*, 1998], which performs automated collaborative filtering for Usenet news articles, adopts the amount of time that a user spends reading a news article as an indication of their interest in that item.

Our main focus in this paper is the user profiling component of the CASPER recommender system which has been designed as a personalised information assistant for users of the JobFinder (www.stepstone.ie) recruitment site. One of the core motivations in CASPER is to investigate the implicit profiling of users based on their click-stream and read-time data. In this sense CASPER is related to other recommender systems such as [Konstan *et al.*, 1997; Sarwar *et al.*, 1998; Terveen *et al.*, 1997].

3 Mining User Interests in CASPER

User interests in CASPER are gathered by mining the JobFinder server logs which record details of the user interactions within the website, see Figure 1. Essentially, each line of the log file records a single *job access* (user clicks on a hyperlink to see a job description) by a user, and encodes details like the time of access, the job and user IDs. In addition to this, any action that the user performed with respect to that job is recorded - JobFinder allows users to email a job to themselves for later consideration, or apply for it directly online.

The most basic form of a profile is simply a history of jobs that the user has looked at over time. In general though, such a list does not constitute a reliable picture of the user's interests, for example, a job that has been clicked on may turn out to be uninteresting in the end, once the user has seen its description. Such a basic profile can be misleading as it depicts those jobs that the user *thinks* she might be interested in, prior to viewing. Moreover, a given user is likely to have *different levels* of interest in the jobs that she genuinely likes. Therefore, a more detailed profile representation is needed that also records relevancy information to discriminate between those jobs that the user looks at or considers, and those that she is truly interested in.

Graded profiles, which are used in many collaborative recommender systems (e.g. [Smyth and Cotter, 1999]), supplement the basic profile representation with relevancy indicators. These indicators are essentially the set of grades that measure how relevant each item is to that user. However, these indicators can take many shapes and forms, and it is here that the difference between explicit and implicit profiling becomes interesting. A system like PTV [Smyth and Cotter, 1999] explicitly requires users to grade profile items (in PTV - TV programs) according to their taste, and therefore the grades given to the profile items are the actual grades assigned by the user. In contrast, an implicit profiling system like CASPER's recommender system observes the users browsing behaviour to predict the set of grades that the user would have given had she done so explicitly, and hence an immediate advantage is that the system does not interfere with the user's browsing and the user does not have to perform any extra task. It is often the case that users are unwilling to have to perform extra grading of items - especially in an online situation. It has also been documented [Sarwar *et al.*, 1998] that attaining extra ratings implicitly can at least supplement explicit ratings, especially when the profile space is *sparse* [Konstan *et al.*, 1997; Rafter *et al.*, 2000] and there is a general lack of ratings within it. The 'grades' in an implicitly generated profile may correspond to the amount of time the user spent reading a particular item, whether she purchased the item, or whether she bookmarked it etc, [Nichols, 1997]. In CASPER, these grades correspond to three main types of information: the number of revisits made to a job description, the amount of time spent reading a job description, and whether the user applied for a job or mailed it back to herself.

3.1 Revisit Data

The number of times that a user clicks on a job is thought to be an indicator of her interest in that job, in the sense that users will often return for a second or third read of an interesting job description while they are unlikely to revisit uninteresting jobs after an initial read, (see also, [Goecks and Shavlik, 1999]). For this reason CASPER logs the number of times that each user clicks on a job. This measure of *raw revisits* gives us an idea of how often the user has clicked on the job to read its description. However, the number of times a user *clicks* on a job may not correlate with the number of times that user *revisits* the job. In fact, due to slow bandwidth problems, etc. many of these clicks are so-called "irritation clicks" due to a user repeatedly clicking on a job in frustration, while waiting for the description to download, and therefore do not constitute accurate revisit data. In order to deal with this misleading revisit data, CASPER employs a thresholding technique that counts repeated clicks on the same job as irritation clicks. For example, in Figure 1, according to the server log, the user *Rachael* has repeatedly clicked on the job *richjobs*869* three times in quick succession, presumably in irritation due to a slow download time. Thus, these clicks are collapsed into one. In contrast, the user has clicked on *csjobs*0* twice with
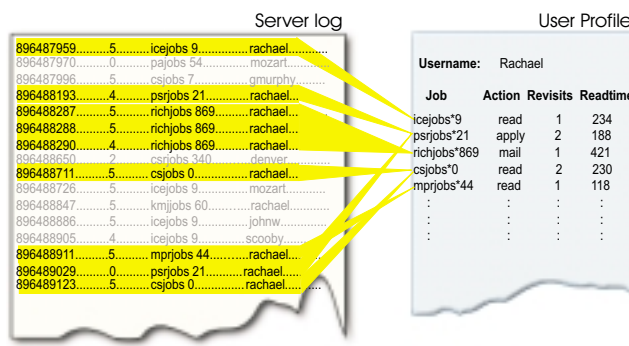
Figure 1: From Server Logs to Graded Profiles

a 16 minute gap between clicks during which time she looked at other jobs. This is interpreted as a genuine revisit and thus both clicks are counted.

3.2 Read-Time Data

Coarse-grained revisit data can be augmented with a more fine-grained measure of relevancy obtained from read-time data. The time a user spends reading a job description has been shown to correlate with that user's degree of interest, [Goecks and Shavlik, 1999; Konstan *et al.*, 1997; Morita and Shinoda, 1994; Nichols, 1997; Oard and Kim, 1998; Sarwar *et al.*, 1998]. For this reason, CASPER also calculates read-time information from the server logs by noting the time difference between successive requests by the same user. Again, a suitable thresholding technique is necessary to eliminate spurious read-times due to a user logging off or leaving her terminal. However, the nature of read-time data makes this task more difficult to manage than the thresholding of revisit data. There is no simple way of identifying misleading read-time data which can arise in the server log for many reasons, for example end-of-sessions, distracted users, multiple windows, or search-bots. This is compounded by factors such as the nature of connection speeds.

In order to prevent spurious read-times (some with values of days or weeks) interfering with the identification of relevant jobs within a profile we adopt a two-step process. This process is designed to identify some *average* value for the time it takes to read a job, and then replace any read-times that deviate wildly from the average. Obviously this approach is not perfect, but the assumption is that it will not interfere with the identification of relevant jobs within the profiles as it is a middling value. However, due to the extent of spurious read-times within the profiles, the actual mean read-time for a user or for a particular job was strongly biased by these outliers and was thus impractical - many users had a mean read-time of days or weeks.

Our approach instead involved using the median of median read-time values per individual *job access* (as opposed to the total read-time over a number of visits to the job) for both users and jobs to calculate a *normal* read-time for the system (Equation 1). The median of medians for both users and jobs was 48 seconds which we took as a reasonable value for a *normal* length of time it takes to read a job description.

$$
\begin{aligned}
SystemNormRT \\
= (average \\
median(median((p_1), .., median(p_n)), \\
median(median((j_1), .., median(j_m)))) \\
\text{where: } p_i \in \text{Profiles}, j_i \in \text{Jobs}
\end{aligned}
$$
(1)

The second step in refining the read-time data in the profiles was to find any read-times (per job access) within the profiles that had a read-time greater or equal to twice the system median (48 secs.). This produced a set of *adjusted read-times* (Equation 2) where all the read-time values are *reasonable*.

$$
\forall \, job\ accesses, \quad adjustedRT = \\
\begin{cases}
SystemNormRT & \text{if rawRT} > 2 * \text{SystemNormRT} \\
rawRT & \text{otherwise}
\end{cases}
$$
(2)

Each of these individual adjusted read-times corresponds to each individual job access by a given user. Of course, many jobs are revisited by a user a number of times, therefore, each job in a profile, received a *total* adjustedRT, which was the sum of all individual adjusted read-times for that job.

Graded read-times (Equation 3) per job were then produced by calculating in each profile, the number of standard deviations each job's total adjusted read-time was above or below the user's mean adjusted read-time.

$$
\forall \, jobs, \\
gradedRT = \frac{adjustedRT - averageRT}{stdevRT}
$$
(3)

Figure 2 shows the detail of a profile with the following fields from left to right: jobID, activity data - whether the user only read the job description (5), or whether she mailed it to herself or applied for it online (4, 0 resp.), raw revisit data, thresholded revisit data (spurious revisits removed), raw read-time and graded read-time (spurious read-times removed and grading method applied).

User A	activity	raw revisit	thresh revisit	raw rt	graded rt
howjobs*317	5	2	2	1753136	3.0793989
resconjobs*300	0	3	3	830	5.922691
icejobs*6	5	1	1	43	-0.11205088
howjobs*247	5	1	1	51	0.35216
resconjobs*285	5	2	2	161	3.0213726
wpgijobs*90	0	5	5	1779164	10.332694
cpljobs*1431	5	2	2	55	0.5842654
wpgijobs*54	0	3	3	2410168	7.0251913
cmcngaleng*15	0	2	2	491	5.057167
howjobs*336	5	1	1	77	1.8608453
sbjobs*277	5	1	1	41	-0.2281036
howjobs*316	5	1	1	715985	0.23610727
hennessyjobs*162	5	1	1	7500782	-0.32104632
sbjobs*275	5	2	1	1107430	1.3386081
firjobs*506	5	3	1	19134	-0.02981327
prsijobs*391	5	1	1	17	-1.6207362
prsijobs*465	4	2	2	65	1.164529
prsijobs*442	4	2	2	106	3.5436099

End-of-sessions

Figure 2: A Graded Profile with Session Boundaries

The diagram depicts a common read-time pattern where a user goes through a few sessions with JobFinder, each ending with a large (misleading) raw read-time due to the time difference between the sessions. In Figure 2, it appears that user A has probably had approximately 7 sessions with JobFinder during this period of time, and jobs with large raw read-times like *wpgijobs*90* and *wpgijobs*54* are the last jobs the user looked at within the different sessions.

Figure 3 focuses on the read-time data within the profile for user A, and shows the jobs sorted by raw read-time (left) and graded read-time (right). The diagram shows that when the jobs in the profile are ordered by the raw read-time data the precision of retrieving the most relevant jobs (those that are applied for, or to a lesser degree those that the user has mailed back to herself) is affected by other jobs with high spurious read-times, for example *hennesseyjobs*162*, appearing at the top of the list. The recall is also affected, because jobs like *resconjobs*300* and *cmcngaleng*15* are pushed down the list by the jobs with high misleading read-times.

However, the improvement is clear when the read-time data is transformed into graded read-time data which helps to remove erroneous read-times. In fact, all the relevant jobs are correctly captured by this data, except for one (*prsijobs*465*).

3.3 Activity Data

The final and perhaps most reliable way to judge user interest in a particular job is to make use of JobFinder's online application or email facility. Briefly, a JobFinder user can either email a job description to herself, for later consideration, or apply for the job directly online. These actions indicate a more serious interest in a job than a simple reading of the job description and for this reason CASPER takes note of the user activity (read(5), apply(0) or email(4)). For example, in Figure 3, we can see user A has read 12 job descriptions, mailed 2 back to herself, and applied for 4 jobs online.

Obviously, a job *will* be highly relevant to a user if she applies for it online. However, users tend to do this infrequently, or not at all, resulting in insufficient data to exclusively base relevancy predictions on. It is therefore necessary to consider these other relevancy measures, (read-time, revisits) to supplement this data, and it is interesting as these measures are common across a wide range of other web-based information filtering domains too. As a result in this paper we prefer not to use the activity data for our profiling, and rather use it as a way of measuring the accuracy of the other (revisit and read-time) measures.

4 Experimental Evaluation

So far, we have described the techniques used in CASPER to produce graded profiles that capture the job preferences of users. One of our central objectives, is to evaluate how these profiles (containing the implicit relevancy information) perform within the collaborative recommendation task [Rafter *et al.*, 2000], against simple profiles with no relevancy information, and ideally against profiles with relevancy information that has been obtained explicitly from the users. Given the difficulty however, in performing such an evaluation without access to a large group of users, we restrict ourselves here to a set of preliminary evaluations to test the value of CASPER's implicit relevancy information, and in particular the measures of revisit data and read-time data.

The experimental study is based on the user profiles generated from server logs between 2/6/98 and 22/9/98, which contained 233,011 job accesses by 5132 different users. These profiles spanned 8248 unique jobs with an average profile size of 14.6 jobs.

Our evaluation of read-time and revisit data as relevancy predictors is based on the activity data of users. We assume that the action of a user applying for a particular job online is a reliable indicator of her interest in that job, and evaluate the read-time and revisit data based on how well it correlates with this information. In other words we are examining how well revisit and read-time data perform at predicting whether a user applied for a job or not. We are also testing in these experiments whether our *improved* measures of revisit and read-time (using thresholding techniques) data actually improve prediction performance. The experiments were therefore restricted to the set of those users who had applied for at least one job. Furthermore, we only took users with a profile size (number of jobs in profile) of 15 or greater. These users numbered 412 in total and were used as the profile base for

User A	activity	raw rt	graded rt
hennessyjobs*162	5	7500782	-0.32104632
wpgijobs*54	0	2410168	7.0251913
wpgijobs*90	0	1779164	10.332694
howjobs*317	5	1753136	3.0793989
sbjobs*275	5	1107430	1.3386081
howjobs*316	5	715985	0.23610727
firjobs*506	5	19134	-0.029813273
resconjobs*300	0	830	5.922691
cmcngaleng*15	0	491	5.057167
resconjobs*285	5	161	3.0213726
prsijobs*442	4	106	3.5436099
howjobs*336	5	77	1.8608453
prsijobs*465	4	65	1.164529
cpljobs*1431	5	55	0.5842654
howjobs*247	5	51	0.35216
icejobs*6	5	43	-0.112050876
sbjobs*277	5	41	-0.2281036
prsijobs*391	5	17	-1.6207362

User A	activity	raw rt	graded rt
wpgijobs*90	0	1779164	10.332694
wpgijobs*54	0	2410168	7.0251913
resconjobs*300	0	830	5.922691
cmcngaleng*15	0	491	5.057167
prsijobs*442	4	106	3.5436099
howjobs*317	5	1753136	3.0793989
resconjobs*285	5	161	3.0213726
howjobs*336	5	77	1.8608453
sbjobs*275	5	1107430	1.3386081
prsijobs*465	4	65	1.164529
cpljobs*1431	5	55	0.5842654
howjobs*247	5	51	0.35216
howjobs*316	5	715985	0.23610727
firjobs*506	5	19134	-0.029813273
icejobs*6	5	43	-0.112050876
sbjobs*277	5	41	-0.2281036
hennessyjobs*162	5	7500782	-0.32104632
prsijobs*391	5	17	-1.6207362

Figure 3: Read-time Details

the experiments.

4.1 Predictions with Revisit Data

For each user in the profile base we produced two sets of predictions for the jobs that the user applied for, based on the two kinds of revisit data: raw revisit predictions, and thresholded revisit predictions. Basically, the jobs in the profile were ordered with jobs that the user had visited most appearing at the top of the lists. For each set of predictions then, we produced 5 lists of the top k predicted jobs, for $k = \{1, 2, 5, 10, 15\}$ for each user. We then measured the precision (4) and recall (5) of each list:

$$precision = \frac{no.\ predicted\ jobs\ applied\ for}{no.\ predicted\ jobs} \quad (4)$$

$$recall = \frac{no.\ predicted\ jobs\ applied\ for}{no.\ jobs\ applied\ for} \quad (5)$$

Figure 4 shows the results as precision against recall where points further away from the origin indicate better overall performance. Each node on the trendline represents the different values for k (indicated by the number beside the node), so we can see how these values vary as the size of the recommendation list (k) increases. Usually as k increases we would expect to see better recall and poorer precision. For each value of k the results are averaged across the 412 users.
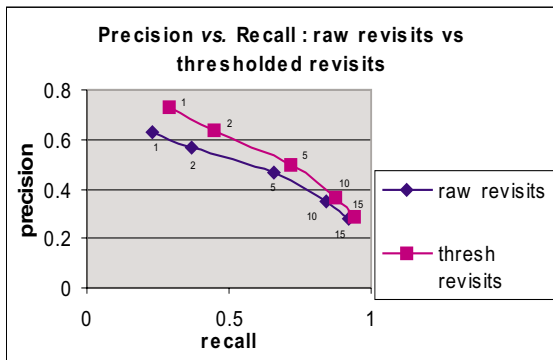

Figure 4: Relevancy Prediction Quality of Revisit Data

The graph shows that there is a clear correlation between revisit data and activity data (specifically jobs that the user has applied for online). It also shows that this data can be more finely tuned by removing the spurious irritation clicks to produce the thresholded version.

4.2 Predictions with Read-Time Data

The read-time prediction experiments proceeded in a similar way to those for the revisit data - two sets of predictions were made, one based on the raw read-time data, and the other based on the graded read-time data. For each set of predictions, 5 lists of the top k (= $\{1, 2, 5, 10, 15\}$) predicted jobs were produced again, and the results averaged over the 412 users. The prediction quality was again measured by precision (4) and recall (5).

The results are shown in Figure 5 as precision against recall for the different values of k.
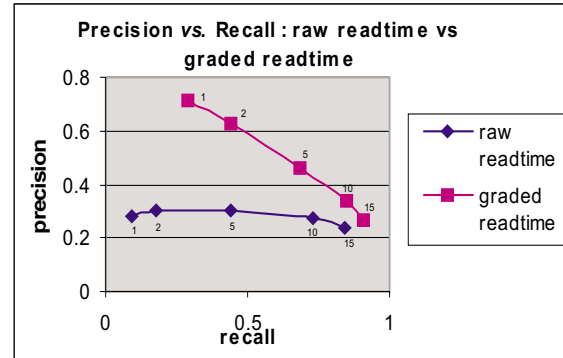

Figure 5: Relevancy Prediction Quality of Read-Time Data

The graph shows that there is only a loose relationship between raw read-time and activity data, as there is no improvement in precision with decreasing values of k. This is because of the large amount of noise that this type of data is subject to, such as a user logging out or leaving her terminal. However, a significant improvement in the correlation between read-time and applying for a job is gained, by refining the data into

graded read-times that eliminate some of the erroneous information. We believe the revisit data performs better because it is less subject to noise than the read-time data which shows little correlation to the activity data in its most raw form.

4.3 Lessons Learned

The idea of attaining quality user profiles by implicitly mining user interests is an attractive one. It has been well documented that the continuous rating of items is a considerable burden on the user. Intuitively, analysing data such as the amount of time a user spends on a particular page or the number of revisits she make to a page should be an indicator of her interest in that page. However in practise the task of turning such observations into quality measures of user interest is difficult.

In our experimentation we test these measures with a case study in online recruitment. It can be argued that the measures we employ are rather heuristic and application specific and the key question is not so much whether they work in our domain, but rather whether these measures are indicative of user interest in general across a wider range of applications. We believe the main contribution of our work in this field is a demonstration of how measures like read-time and revisit information - which are common across many domains - can provide information about a user's taste for particular items in our domain. Importantly, the information in CASPER is attained *for free* simply by mining the original server logs. Our experimentation therefore shows that we can attain good results even in the face of rather noisy data and it follows that improvements should come with better structured or more reliable user information. Although we have not fully demonstrated that our methods are generic or robust enough to work across a wide range of different domains, we are confident that our future research will show that they are generically applicable in a wide range of circumstances.

We believe that similar domains may also benefit from using techniques like those used in CASPER, for example e-commerce domains where there is a homogenous set of items. Obviously in a broader web-browsing domain, metrics like revisits may simply refer to users returning to a key navigation page for example, rather than a page where the content is of serious interest to the user. In any case we believe the situation merits further investigation.

5 Future Work

So far we have shown how the most relevant items in a user profile can be identified using measures of revisit data and read-time data. Ultimately, the goal is to use this information in the collaborative recommendation task [Rafter *et al.*, 2000], i.e. where we are not only concerned with the most relevant jobs for a single user, but rather with the similarity between two users based on this relevancy information. This presents another important issue to be considered when analysing the value of revisit and read-time data. For example, if a user has accessed *jobA* 10 times, is that job 5 times as relevant to that user as *jobB* that she has looked at twice? It is our hypothesis that although *jobA is* more relevant to the user, it is not 5 times more relevant. Therefore, we believe

that a logarithmic model of relevancy may be more appropriate than the current linear model we use - an illustrative example is given in Figure 6. In the future, we plan to investigate this hypothesis, and fully integrate and evaluate the relevancy measures of revisit and read-time data into CASPER's recommender system.
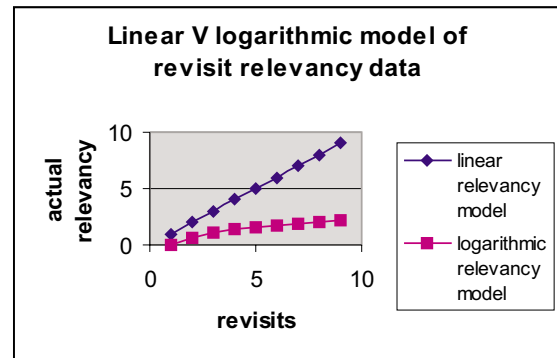

Figure 6: Linear Revisit Model

6 Conclusions

One of the basic assumptions underlying recent research focused on developing user profiling applications for the Internet, is that it is possible to derive user preferences by monitoring measures like the click-stream and read-time data generated in server logs from user sessions. However, although this assumption seems broadly accepted there is relatively little published work that tests the validity of this assumption, [Nichols, 1997].

In this paper we have describe such a study in the context of the CASPER project, which is concerned with developing user profiling techniques for online services like the JobFinder recruitment service. We have described how JobFinder's server logs are mined to generate different types of user profiling information derived from normalised click-stream and read-time data. The results although preliminary, are promising as they indicate that both read-time and revisit data are useful in predicting jobs that individual users have applied for, which we believe is indicative of serious user interest. Furthermore, we have provided a comparative evaluation to demonstrate the differential performance of these techniques.

References

[Bradley *et al.*, 2000] K. Bradley, R. Rafter, and B. Smyth. Case-based user profiling for content personalisation. In *Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-based Systems*, Trento, Italy, August 2000.

[Goecks and Shavlik, 1999] J. Goecks and J Shavlik. Automatically labeling web pages based on normal user interactions. In *Proceedings of the IJCAI Workshop on Machine Learning for Information Filtering*, Stockholm, Sweden, July 1999.

[Konstan *et al.*, 1997] J. Konstan, B. Miller, D. Maltz, J. Herlocker, L. Gordon, and J. Reidl. Grouplens: Applying collaborative filtering to usenet news. *Communications of ACM*, 40(3):77–87, March 1997.

[Morita and Shinoda, 1994] M. Morita and Y Shinoda. Information filtering based on user behaviour analysis and best match text retrieval. In *Proceedings of the 19th annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Dublin, Ireland, July 1994.

[Nichols, 1997] D. Nichols. Implicit rating and filtering. In *Proceedings of 5th DELOS Workshop on Filtering and Collaborative Filtering*, Budapest, Hungary, November 1997.

[Oard and Kim, 1998] D. Oard and J Kim. Implicit feedback for recommender systems. In *Proceedings of AAAI Workshop on Recommender Systems*, Madison, Wisconsin, USA, July 1998.

[Rafter and Smyth, 2001] Rachael Rafter and Barry Smyth. A domain analysis methodology for collaborative filtering. In *Proceedings of 23rd BCS European Annual Colloquium on Information Retrieval Research*, Darmstadt, Germany, April 2001.

[Rafter *et al.*, 2000] R. Rafter, K. Bradley, and B. Smyth. Automated collaborative filtering applications for online recruitment services. In *Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-based Systems*, Trento, Italy, August 2000.

[Sarwar *et al.*, 1998] B. Sarwar, J. Konstan, J. Borchers, J. Herlocker, B. Miller, and J. Reidl. Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, Seattle, Washington, USA, November 1998.

[Smyth and Cotter, 1999] B. Smyth and P. Cotter. The sky's the limit: A personalised tv listings service for the digital tv age. In *Proceedings of Expert Systems '99 (ES99)*, Cambridge, UK, December 1999.

[Terveen *et al.*, 1997] L. Terveen, W. Hill, B. Amento, D. McDonald, and J. Creter. Phoaks: A system for sharing recommendations. *Communications of ACM*, 40(3):59–62, March 1997.