Automated Collaborative Filtering Applications for Online Recruitment Services

Rachael Rafter, Keith Bradley, Barry Smyth

Smart Media Institute,
Department of Computer Science, University College Dublin, Belfield, Dublin 4, Ireland

{rachael.rafter, keith.bradley, barry.smyth}@ucd.ie

Abstract. Online recruitment services suffer from shortcomings due to traditional search techniques. Most users fail to construct queries that provide an adequate and accurate description of their (job) requirements, leading to imprecise search results. We investigate one potential solution that combines implicit profiling methods and automated collaborative filtering (ACF) techniques to build personalised query-less job recommendations. Two ACF strategies are implemented and evaluated in the JobFinder domain.

1 Introduction

Online recruitment services have emerged as one of the most successful and popular information services on the Internet, providing job seekers with a comprehensive database of jobs and a dedicated search engine. For example, the award-winning Irish site, JobFinder (www.jobfinder.ie). However, like many similar Internet applications JobFinder suffers from shortcomings, due to its reliance on traditional database technology and the client-pull information access model, which places the burden of representing job requirements (in the form of a query) on the user. Furthermore, most users submit poorly constructed queries, resulting in unfocused search results.

To address these problems the CASPER project seeks to investigate the role of Artificial Intelligence techniques in providing a more proactive, personalised and intelligent model of information access. CASPER focuses on enhancing the existing JobFinder system in two different ways: by improving the existing database search facility using case-based reasoning (CBR) and fuzzy matching capabilities, and, using personalised automated collaborative filtering (ACF) techniques to proactively recommend new jobs to users. For an overview of CASPER as a whole see [5].

In this paper, we focus on the second of these enhancements. Specifically, we investigate the role of ACF within CASPER as the core of its query-less recommendation service. Briefly, ACF is a feature-free recommendation technique that recommends information items to a given user based on what similar users have previously liked (e.g. [2, 6, 7]). The success of ACF is critically dependant on the availability of high-quality user profiles, which contain graded lists of information items, and has been shown to work well when there is a high expected overlap between related profiles. However, JobFinder's dynamic database of jobs, each with a

limited life span, means that the expected overlap between related users can be very small (see also [1]). In this paper we will discuss how this is problematic for one common form of ACF, and describe the solution investigated in CASPER.

2 User Profiling in CASPER

User profiles are the primary knowledge source in an ACF system. In CASPER, user profiles are automatically and passively constructed by mining JobFinder's server logs. Therefore, profiles are built without interfering with the user's interaction with JobFinder, unlike related systems that require explicit feedback from users [6].

Fig. 1 shows part of a JobFinder server-log. Each line records a single job access by a user and encodes details like the time and type of access, and the job and user ids. The basic form of a profile is simply a list of jobs that the user has looked at over time. In general though, a user is likely to have different levels of interest in different jobs, and this implicit relevancy information should also be in the profile.

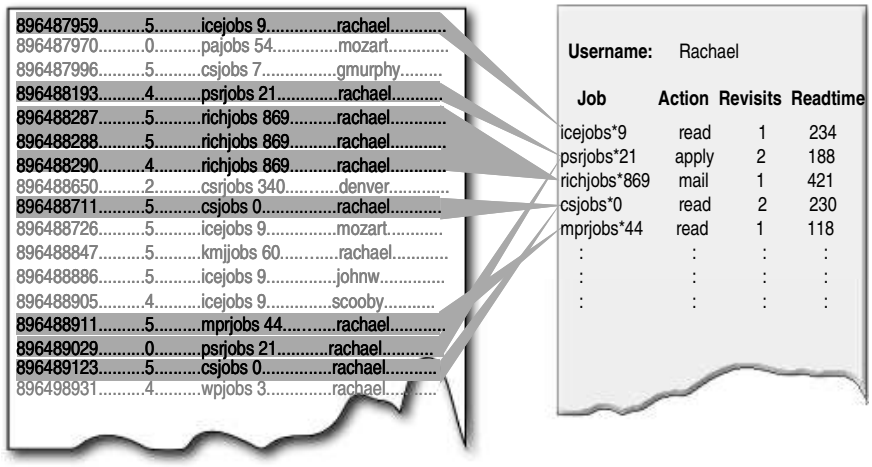

Fig. 1. A partial user profile showing basic and graded profiling information.

CASPER collects three types of relevancy information:

Revisit Data: The amount of times that a user has accessed an information item has been shown to indicate their interest in that item [3]. This information is collected in CASPER and misleading data like "irritation clicks" (repeated clicks on a job description while it is downloading), are removed e.g. job *richjobs*869* in Fig. 1.

ReadTime Data: A more sensitive measure of relevancy gained from read-time data, has also been reported to correlate well with a user's interest [2, 3]. Hence, CASPER also calculates read-time information by noting the time difference between successive requests by the same user, (Fig. 1). Again, a suitable thresholding technique is needed to eliminate spurious read-times, for example a user logging off.

Activity Data: The final and most reliable way to judge user interest in a given job is to avail of JobFinder's online application or email facility. Briefly, a JobFinder user can either email a job description to herself, or apply for the job directly. These actions indicate a more serious interest in a job than a reading of the job description and are recorded in CASPER For example, Fig. 1 shows the jobs that user "Rachael" has read, emailed to herself, and applied for, (5, 4, and 0 respectively in log file).

In our research we have examined each of these factors as measures of relevancy. 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 base relevancy predictions on exclusively. It is therefore necessary to consider these other relevancy measures, (readtime, revisits) to supplement this data, and it is interesting as these measures are common to other web-based information filtering domains too.

Although we have found that both revisit and readtime data correlate with the activity data in profiles of users that applied for jobs, this has not resulted in any significant improvements over the simple ungraded profile representation when incorporated into a recommendation task [1]. This was due to a number of reasons, perhaps the most important of which is the nature of CASPER's profile space where the expected overlap (shared jobs) between profiles is very low. We will argue in the next section that this presents a significant problem for one common ACF technique.

3 ACF and the Problem with Relationships

ACF is a recommendation strategy that draws its recommendations from the profiles of similar users in two steps. A set of users related to the *target user* is identified, and profile items from these users that are not in the target profile, are ranked for recommendation to the target user. The success of ACF depends on recognising quality recommendation partners that are genuinely related to a target user. In this section we will introduce two different ACF algorithms (memory-based ACF and cluster-based ACF) that rely on different methods for identifying relevant users. In addition, we will argue that only one of these is likely to be useful in CASPER.

3.1 Memory-Based ACF and Direct Relationships

Memory-based ACF is probably the simplest form of the general ACF approach. Users are related on the basis of a direct similarity between their profiles, for example, by measuring the degree of overlap between their profile items, or by measuring the correlation coefficient between their grading lists [1, 6]. This leads to a lazy form of ACF whereby a *k-nearest neighbour* (K-NN) strategy is used. Currently CASPER uses a simple overlap metric (1) to calculate similarity.

$$Overlap\ (t, p) = \frac{|Items\ (t) \cap Items\ (p)|}{|Items\ (t) \cup Items\ (p)|} \quad \textit{for: t, p profiles: (t being the target profile)} \qquad \textbf{(1)}$$

[1] Relevancy information was used with Pearson's Correlation Coefficient [4, 6] and compared as a similarity measure to a simple overlap metric (1) that had no relevancy information.

Once the k-nearest users (base profiles) have been identified, the jobs not already present in the target profile are ranked, favouring jobs that occur more frequently in the base profiles, and jobs that occur in more similar profiles to the target profile.

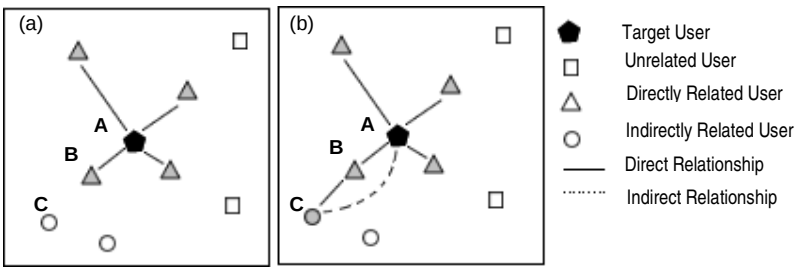

Fig. 2 Direct vs. Indirect User Relationships.

The important thing to note about this form of ACF is its reliance on direct user relationships, for example between A and B in Fig. 2(a), and its ignorance of potential indirect relationships between users. In short, C may have the same job taste as A, but as C has seen a different set of jobs, this will not be recognised in this form of ACF.

In many sparse-profile domains, such as CASPER's, the expected overlap (similarity) between two user profiles is likely to be very small, and the number of other users with significant overlap with a given target user is also likely to be very low [1, 4]. As a result, the target user's recommendations may be based on a small number of profiles with low degrees of similarity. This may compromise the quality of the recommendations, or may even result in no recommendations at all for a user.

3.2 Cluster-Based ACF and Indirect Relationships

Fortunately there is an alternative ACF method that identifies both direct and indirect user relationships. This method uses (eager) clustering techniques to group users prior to recommendation – profiles are clustered into virtual communities such that all of the users in a given community are related, (e.g., [2]).

Fig. 2(b) demonstrates a common scenario where user B is directly related to users A and C. There is no direct relationship between A and C, however there may be an indirect transitive relationship between them through B that should be identified too. Therefore, users A, B, and C should belong to the same virtual community with all other users that are directly or indirectly related to them. In order to specifically exploit this form of indirect relationship the single-link clustering technique can be used with a thresholded version of the similarity metric (1); essentially each community is a maximal set of users such that every user has a similarity value greater than the threshold with at least one other community member.

Once communities of related users are built the recommendation process proceeds in an analogous way to the memory-based approach, except that instead of selecting k neighbours, we select the members of the target profile's community. The benefit then, is the possibility of identifying larger groups of users that are related to the

target user and thus provide a richer recommendation base. The disadvantage is that it is no longer possible to judge direct similarities between all pairs of profiles in a community, as there may be no direct relationship between them, and recommendable items are ranked based only on their frequency in the community.

4 Experimental Analysis

In this section, we describe a preliminary evaluation to test the quality of job recommendation produced by each method of ACF described above.

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, and only 2195 profiles contained 10 or more jobs.

As we had no way of automatically evaluating the recommendations produced by the two ACF versions, the evaluation had to be carried out by hand and was thus restricted to a small set of users. Ten target users were randomly selected, each from a different virtual community. Furthermore, the communities to which these target users belonged covered a range of sizes. Each target user, received two recommendation lists of ten jobs:

- **Memory-Based ACF (K-NN):** Each target user is associated with its k nearest users (k=10 in this experiment) and a ranked recommendation list is produced.
- **Cluster-Based ACF (Clustering):** Each target user is recommended the most frequently occurring jobs in its virtual community.

Both sets of results for each target user were then manually graded (by the researchers involved in the project) as good, satisfactory, or poor (mapped on to a numeric value of 3,2, or 1 respectively), based on the similarity between the recommended jobs and the existing jobs in the target profile. Therefore, every target user received a cumulative grading score across the 10 recommended jobs from each ACF technique. Each score was normalised by dividing by the maximum cumulative grade of 30 and presented in Fig. 3 for each target user. Fig. 3 also encodes a measure of cluster size so that we can see how the recommendation quality behaves for different cluster sizes using the cluster-based ACF method.

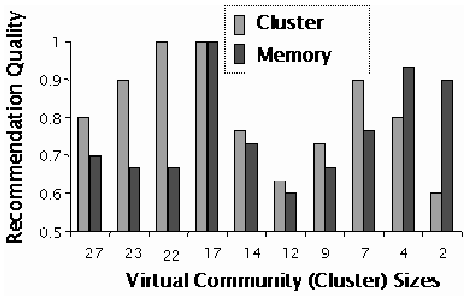

Fig. 3. Recommendation quality for memory-based ACF and cluster-based ACF.

The results clearly show that the cluster-based method outperforms the memory-based version, (except for very small clusters). We believe that this is due to the sparse profile space. As explained in Section 3.1, the expected overlap between users is very small, and therefore many of the 10 nearest users to a target user, chosen by ACF-NN may only overlap with the target profile by 2 or 3 jobs and may not constitute reliable recommendation partners. Thus we can expect some of the recommendations for the ACF-NN method to come from unreliable sources. In contrast, the ACF-Cluster method is basing its recommendation on potentially more reliable measures of indirect relationships between profiles. In this experiment the similarity threshold of the virtual communities was 10, so implicit relationships were based on a transitive overlap of 10 jobs between community members.

5 Discussion

In this paper we have looked at enhancing the JobFinder system with an ACF component. Specifically, we have described how implicit user profiles can be automatically generated from JobFinder server logs to form personalised recommendations using two different ACF approaches, a memory-based method and a clustering-based method. We have argued that the sparseness of CASPER's profile space presents significant problems for the memory-based approach to ACF because of the low expected overlap between profiles in CASPER. The cluster-based approach is presented as a solution, as it can exploit transitive relationships between profiles that otherwise do not overlap, and we argue that this property makes this technique more appropriate in CASPER. This hypothesis is supported by preliminary experimental results and we expect to add a more comprehensive evaluation in the future.

References

1. Billsus, D., Pazzani M.: Learning Collaborative Information Filters. Proceedings of the International Conference on Machine Learning. Morgan Kaufmann, Madison Wisc. (1998)
2. Konstan, J., Miller, B., Maltz, D., Herlocker, J., Gordon, L., Reidl, J.: Applying Collaborative Filtering to Usenet news. Communications of ACM (1997), 40: 3: 77-87
3. Nichols, D.: Implicit Rating and Filtering. Proceedings of 5th DELOS Workshop on Filtering and Collaborative Filtering, Budapest Hungary, November, (1997)
4. Pazzani, M.: A Framework for Collaborative, Content-Based and Demographic Filtering. Artificial Intelligence Review. (in press)
5. Rafter, R., Bradley, K., Smyth, B.: Personalised Retrieval for Online Recruitment Services. Proceedings of the 22nd Annual Colloquium on Information Retrieval, (BCS-IRSG 2000), Cambridge UK, April, (2000)
6. Smyth, B., Cotter. P.: Surfing the Digital Wave: Generating Personalised Television Guides using Collaborative, Case-based Recommendation. Proceedings of the 3rd International Conference on Case-based Reasoning, Munich Germany, (1999)
7. Terveen, L., Hill, W., Amento, B., McDonald, D., Creter, J.: Phoaks: A System for Sharing Recommendations. Communications of the ACM (1997), 40(3), 59 – 65