Toward a Personal Recommender System

Bradley N. Miller
Computer Science Dept.
Luther College
Decorah, IA 52101
bmiller@luther.edu

ABSTRACT

This position paper is an outgrowth from work reported in a *Transactions on Information Systems* article entitled PocketLens: Toward a Personal Recommender System, July 2004, in which we propose and compare several architectures for a decentralized recommender system built on top of peer-to-peer infrastructure. In this paper we review the need for personal recommender systems and propose the deployment of personal recommender systems using existing RSS and weblog technologies as the underlying communication infrastructure.

Keywords

Recommender Systems, Collaborative Filtering

INTRODUCTION

As I look at the applications I use on my personal computer each day, more and more of them are becoming net-enabled. My browser shows me web pages, but I am also able to synchronize my bookmarks between the browser on my home machine and my browser at school. My recipe program allows me to publish a personal recipe collection to the web, and is savvy enough to import recipes found on websites published by other people using the same software. My RSS news reader keeps me up to date on hundreds of topics I am interested in reading about. My iTunes music player allows me to sample, purchase, and share music with other members of my family. Version tracker software alerts me to new releases of my favorite applications.

Each of these applications would be enhanced by recommender features. Just as Usenet news used to overwhelm me with new articles each day, my RSS client contains more new headlines than I could ever hope to read. I would love to discover new artists and albums to purchase through iTunes. I would be very pleased to learn about new and interesting recipes that I should make, and I would like to find out about additional software that I could use.

In this paper I propose that the next wave of recommender systems should be integrated with many of the client programs we use for work and play each day. Further, I propose that the engine that drives the recommendations can and should be located on the user's desktop, close to the applications it serves. In [3] we argue that there are two primary issues, *trust* and *control*. In this paper I will summarize our arguments in favor of personal recommender systems, and then briefly present an architecture that would allow for recommenders to run on every desktop.

When a website collects personal information, the user must trust the site to protect the information appropriately. Foner identifies five ways that this trust might be misplaced [2]: **Deception by the recipient**, a site could set out to deliberately trick users into revealing personal information. **Mission creep** refers to a situation in which an organization begins with a clearly defined use for personal data, but expands the original purpose over time. **Accidental disclosure** happens when a website accidentally makes private information available. **Disclosure by malicious intent** happens when personal information that you entrust to an organization is stolen. Finally, information is sometimes released because of **subpoenas**. Even though an organization may take great care to protect personal information, they are obliged under the law to disclose that information when they are subpoenaed.

Most websites try to alleviate their customer's fears by posting a privacy policy on their site that describes how the merchant will use information provided by the customer. However, over time, merchants may rethink their policies, or abandon them altogether for a variety of reasons. This was the case with defunct Toysmart.com. When Toysmart ceased operations in May 2000, the company was forced to sell its assets, including its customer list and customer profiles. In addition, some of the largest websites, like `Yahoo.com` have altered their privacy policy to allow them to sell their customers' email addresses in order to add additional sources of revenue.

Workshop: Beyond Personalization 2005
IUI'05, January 9, 2005, San Diego, California, USA
http://www.cs.umn.edu/Research/GroupLens/beyond2005

The key issue highlighted in these scenarios is that once merchants have control of users' personal information, the users can no longer control the uses to which that information is put. These are exactly the kind of control issues that the users in Ackerman, Cranor, and Reagle's privacy study were concerned about [1]. User control of personal information is a key trust issue for recommenders.

A second key trust issue is the reliability of the recommendations. The credibility of Amazon.com's recommendation services was called into question in 2002 when it admitted to using 'faux recommendations' in order to drive business to its new clothing store partners. The 'faux recommendations' were presented right next to the traditional recommendations for other products, but were not based on a customer's past shopping behavior or preferences. How can users know whether to trust the recommendations that are made?

In summary there are two main concerns with centralized recommenders. First, a centralized recommender might share personal data in inappropriate ways. Second, a recommender owned by a commerce site might make recommendations that serve the site, not the user. Both concerns can be met by a *personal recommender*. The personal recommender holds the user's personal data locally, and only shares data the user explicitly identifies as sharable. The personal recommender is software owned by the user and running on the user's local machine. It answers to noone except the user. My long-term vision is to develop such a personal recommender.

A PERSONAL RECOMMENDER ARCHITECTURE

PocketLens is an algorithm that is designed to support personal recommenders. In [3] we introduce PocketLens and five reference architectures that use the algorithm. The PocketLens algorithm is a variant of the accurate and efficient item–item algorithm [4], with three key features introduced that would allow the algorithm to operate on the user's local machine. First, we have modified the algorithm to construct the model incrementally. Second, the PocketLens model is small, since it is only for one user, which allows us to use the algorithm on desktop computers and palmtops. Third, to preserve privacy, the PocketLens algorithm has the property that none of ratings are saved in conjunction with anything that would identify the user they came from.

The PocketLens algorithm divides the recommendation process into two parts. In part one PocketLens searches the network to find ratings to use in building a similarity model. Ratings may come from randomly selected users, or may come from a defined set of users that the algorithm has learned are 'good' over time. Much of the discussion in [3] is devoted to exploring five peer-to-peer architectures that support the sharing of ratings and discovery of 'good' users. In part two the similarity model is used to make recommendations to the user.

Although the PocketLens algorithm has been implemented and tested in the lab, it has not yet been deployed in an actual peer-to-peer environment. Part of the reason for the delay in deployment is because up until recently there has been no widely available infrastructure to use in publishing a set of ratings. The popularity of RSS (http://blogs.law.harvard.edu/tech/rss) syndication technologies has led me to believe that RSS would provide just such an infrastructure. RSS is an XML standard that allows a user to publish 'channels' of information. Each channel can have many items, and each item is represented by a set of attributes including a globally unique identifier (guid), a title, a description, and a URL.

With RSS as the backbone of a ratings exchange system, users could rate new items and publish the ratings just as a user would add an entry to their weblog. To help new users find sources of ratings, individual users may also choose to publish their list of links to other RSS feeds of ratings.

Although the process of publishing ratings may sound complex, there are already many tools available that would enable users to publish their ratings like a weblog. Some of these tools include local weblog servers like Blosxom and WordPress or web based services, with local APIs like Radio Userland and MoveableType. The fact that these technologies are already in place should help speed user adoption because there is no need to introduce yet another set of technologies into the user's system.

There are many exciting applications being developed today. Some of them are already taking their first steps toward adding recommendation capabilities using a rule based approach. For example iTunes provides 'smart playlists' that allow you specify a set of criteria including the artist, genre, or play count to create a custom list of songs to play. iTunes and others also allow users to rate songs on a scale from 1 – 5 stars. NetNewsWire and others provide 'smart list' capability to do keyword matching on new articles. Personalized recommenders could take all of these applications to new levels, while at the same time opening up recommender systems research in exciting new content areas.

REFERENCES

1. M. S. Ackerman, L. F. Cranor, and J. Reagle. Privacy in e-commerce: Examining user scenarios and privacy preferences. In *ACM Conference on Electronic Commerce*, pages 1–8, 1999.
2. L. Foner. *Political Artifacts and Personal Privacy: The Yenta Multi-Agent Distributed Matchmaking System*. PhD thesis, Massachusetts Institute of Technology, 1999.
3. B. N. Miller, J. A. Konstan, and J. Riedl. Pocketlens: Toward a personal recommender system. *ACM Trans. Inf. Syst.*, 22(3):437–476, 2004.
4. B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International World Wide Web Conference (WWW10)*, Hong Kong, May 2001.