From Amateurs to Connoisseurs: Modeling the Evolution of User Expertise through Online Reviews

Julian McAuley
Stanford University
jmcauley@cs.stanford.edu

Jure Leskovec
Stanford University
jure@cs.stanford.edu

ABSTRACT

Recommending products to consumers means not only understanding their *tastes*, but also understanding their level of *experience*. For example, it would be a mistake to recommend the iconic film *Seven Samurai* simply because a user enjoys other action movies; rather, we might conclude that they will *eventually* enjoy it—once they are ready. The same is true for beers, wines, gourmet foods—or any products where users have acquired tastes: the 'best' products may not be the most 'accessible'. Thus our goal in this paper is to recommend products that a user will enjoy *now*, while acknowledging that their tastes may have changed over time, and may change again in the future. We model how tastes change due to the very act of consuming more products—in other words, as users become more *experienced*. We develop a latent factor recommendation system that explicitly accounts for each user's level of experience. We find that such a model not only leads to better recommendations, but also allows us to study the role of user experience and expertise on a novel dataset of fifteen million beer, wine, food, and movie reviews.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information Search and Retrieval

Keywords

recommender systems, expertise, user modeling

1. INTRODUCTION

> *"Even when experts all agree, they may well be mistaken"*
>
> – Bertrand Russell

In order to predict how a user will respond to a product, we must understand the tastes of the user and the properties of the product. We must also understand how these properties change and evolve over time. As an example, consider the *Harry Potter* film series: adults who enjoy the films for their special effects may no longer enjoy them in ten years, once their special effects are obsolete; children who enjoy the films today may simply outgrow them in ten years; *future* children who watch the films in ten years may not enjoy them, once Harry Potter has been supplanted by another wizard.

WWW 2013, May 13–17, 2013, Rio de Janeiro, Brazil.
ACM 978-1-4503-2035-1/13/05.

This example highlights three different mechanisms that cause perceptions of products to change. Firstly, such change may be tied to the age of the *product*. Secondly, it may be tied to the age (or development) of the *user*. Thirdly, it may be tied to the state (or zeitgeist) of the *community* the user belongs to.

These mechanisms motivate different models for temporal dynamics in product recommendation systems. Our goal in this paper is to propose models for such mechanisms, in order to assess which of them best captures the temporal dynamics present in real product rating data.

A variety of existing works have studied the evolution of products and online review communities. The emergence of new products may cause users to change their focus [19]; older movies may be treated more favorably once they are considered 'classics' [20]; and users may be influenced by general trends in the community, or by members of their social networks [26].

However, few works have studied the *personal development* of users, that is, how users' tastes change and evolve as they gain knowledge, maturity, and experience. A user may have to be exposed to many films before they can fully appreciate (by awarding it a high rating) *Citizen Kane*; a user may not appreciate a *Romanée-Conti* (the 'Citizen Kane' of red wine) until they have been exposed to many inferior reds; a user may find a strong cheese, a smokey whiskey, or a bitter ale unpalatable until they have developed a tolerance to such flavors. The very act of consuming products will cause users' tastes to change and evolve. Developing new models that take into account this novel viewpoint of user evolution is one of our main contributions.

We model such 'personal development' through the lens of user *experience*, or *expertise*. Starting with a simple definition, experience is some quality that users gain over time, as they consume, rate, and review additional products. The underlying hypothesis that we aim to model is that users with similar levels of experience will rate products in similar ways, even if their ratings are temporally far apart. In other words, each user evolves on their own 'personal clock'; this differs from other models of temporal dynamics, which model the evolution of user and product parameters on a single timescale [20, 40, 41].

Naturally, some users may already be experienced at the time of their first review, while others may enter many reviews while failing to ever become experienced. By individually learning for each user the rate at which their experience progresses, we are able to account for both types of behavior.

Specifically, we model each user's level of experience using a series of latent parameters that are constrained to be monotonically non-decreasing as a function of time, so that each user becomes more experienced (or stays at least as experienced) as they rate additional products. We learn latent-factor recommender systems for

different experience levels, so that users 'progress' between recommender systems as they gain experience.

'Experience' and 'expertise' are *interpretations* of our model's latent parameters. In our context they simply refer to some unobserved quantity of a user that increases over time as they consume and review more products. Intuitively, our monotonicity requirement constrains users to evolve in the same 'direction', so that what we are really learning is some property of user evolution that is common to all users, regardless of when they arrive in the community. We perform extensive qualitative analysis to argue that 'experience' is a reasonable interpretation of such parameters. In particular, our goal is not to say whether experienced/expert users are 'better' or 'more accurate' at rating products. We simply model the fact that users with the same level of experience/expertise rate products in a *similar* way.

Our experimental findings reveal that modeling the personal evolution of each user is highly fruitful, and often beats alternatives that model evolution at the level of products and communities. Our models of user experience also allow us to study the differences between experienced and novice users, and to discover *acquired tastes* in novel corpora of fifteen million beer, wine, food, and movie reviews.

A Motivating Example

We demonstrate the evolution of user tastes and differences between novices and experts (i.e., experienced users) by considering the beer-rating website *RateBeer*, which consists of around three million beer reviews.

In this data we find a classic example of an acquired taste: hops. The highest-rated beers on the website are typically the hoppiest (American Pale Ales, India Pale Ales, etc.); however, due to their bitterness, such beers may be unpalatable to inexperienced users. Thus we might argue that even if such beers are the *best* (i.e., the highest rated), they will only be *recognized* as the best by the most experienced members of the community.

Figure 1 examines the relationship between product ratings, user experience level, and hoppiness, on *RateBeer* data. Beers of three types are considered (in increasing order of hoppiness): lagers, mild ales, and strong ales. The x-axis shows the average rating of products on the site (out of 5 stars), while the y-axis shows the *difference* between expert and novice ratings. 'Experts' are those users to whom our model assigns the highest values of our latent experience score, while 'novices' are assigned the lowest value; Figure 1 then shows the difference in product bias terms between the two groups.

This simple plot highlights some of our main findings with regard to user evolution: firstly, there is significant variation between the ratings of beginner and expert users, with the two groups differing by up to half a star in either direction. Secondly, there exist entire *genres* of products that are preferred almost entirely by experts or by beginners: beginners give higher ratings to almost all lagers, while experts give higher ratings to almost all strong ales; thus we might conclude that strong ales are an 'acquired taste'.

Finally, we find a strong correlation between the overall popularity of a product, and how much it is preferred by experienced users: experts tend to give the harshest ratings to the lowest-rated products, while they give the most generous ratings to the highest-rated products (continuing our analogy, they have learned to 'fully appreciate' them). Thus while a lager such as *Bud Light* is disliked by everybody, it is *most disliked* by experts; one of the most popular beers in the entire corpus, *Firestone XV*, is liked by everybody, but is *most liked* by experts. We find such observations to be quite general across many datasets that we consider.

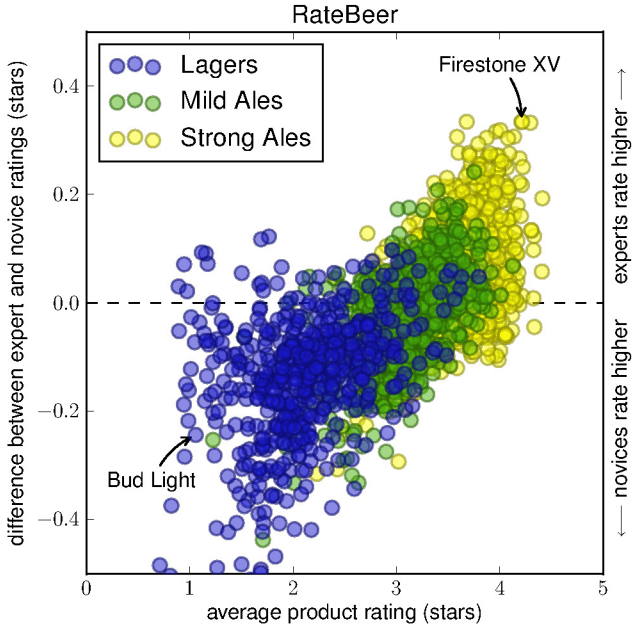


Figure 1: Beloved products are most beloved by experts; hated products are most hated by experts.

Contribution and Findings

We propose a latent-factor model for the evolution of user experience in product recommendation systems, and compare it to other models of user and community evolution. While temporal dynamics and concept drift have been studied in online reviews [13, 20, 29], the critical difference between our model and existing work on temporal dynamics is how we treat experience as a function of time. Existing models for temporal dynamics work under the hypothesis that two users will respond most similarly to a product if they rate the product *at the same time*. In contrast, we model the *personal* evolution of users' tastes over time. We show that two users respond most similarly to a product if they review it *at the same experience level*, even if there is significant separation in time between their reviews. Our model allows us to capture similarities between users, even when their reviews are temporally far apart.

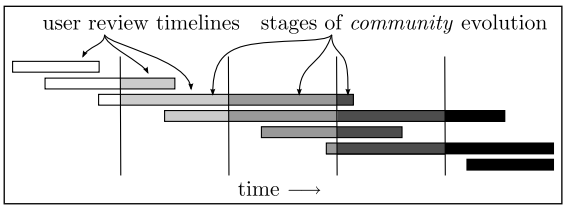
In terms of experiments, we evaluate our model on five product rating websites, using novel corpora consisting of over 15 million reviews. Our ratings come from diverse sources including beers, wines, gourmet foods, and movies.

We find that our model of user *experience* significantly outperforms traditional recommender systems, and similar alternatives that model user evolution at the level of products and communities.

Finally, we find that our latent experience parameters are themselves useful as *features* to facilitate further study of how experts and novices behave. For example, we discover that experienced users rate top products more generously than beginners, and bottom products more harshly (as in Fig. 1); we find that users who fail to gain experience are likely to abandon the community; we find that experts' ratings are easier to predict than those of beginners; and we find that experienced users agree more closely when reviewing the same products. We can also use the notion of experience to discover which products, or categories of products, are preferred by

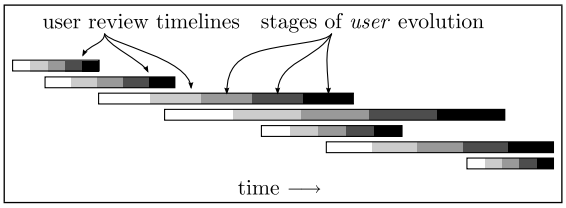
Community evolution at uniform intervals:

(a)



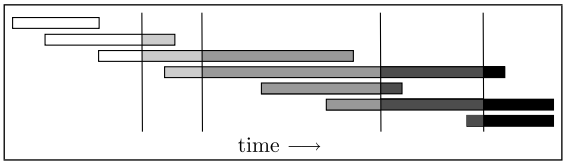
Individual user evolution at uniform intervals:

(b)



Community evolution at learned intervals:

(c)



Individual user evolution at learned intervals:

(d)

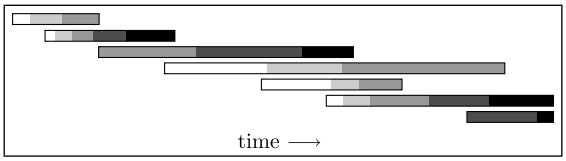


Figure 2: Visualization of the models we consider. Horizontal bars represent user review timelines; colors within each bar represent evolution parameters for each user.

beginners or by experts—that is, we can discover which products are *acquired tastes*.

The rest of this paper is organized as follows: We describe our models for user evolution in Section 2, and we describe how to train these models in Section 3. In Section 4 we describe our novel rating datasets and evaluate our models. In Section 5 we examine the role of our latent experience variables in detail, before reviewing related work in Section 6 and concluding in Section 7.

2. MODELS OF USER EVOLUTION

We design models to evaluate our hypothesis that 'experience' is a critical underlying factor which causes users' ratings to evolve. We do so by considering alternate paradigms of user and community evolution, and determining which of them best fits our data.

Figure 2 visualizes each of the four models we consider. Each horizontal bar represents a single user's review timeline, from their first to their last review; the color within each bar represents the evolution of that user or their community. At each of these stages of evolution, a different recommender system is used to estimate their ratings. Recommender systems for adjacent stages are regularized to be similar, so that transitions between successive stages are smooth.

(a) Community evolution at uniform intervals: First we consider a model where 'stages of evolution' appear at uniform time intervals throughout the history of the community. The model of Figure 2 (a) is in some sense the most similar to existing works [20, 40, 41] that model evolution of users and products using a single global 'clock'. The intuition behind this model is that communities evolve over time, and prefer different products at different time periods.

(b) User evolution at uniform intervals: We extend the idea of community evolution and apply it directly to individual users (Fig. 2 (b)). This model captures the intuition that users go through different life-stages or experience levels and their preferences then depend on their current life stage.

(c) Community evolution at learned intervals: This model extends (a) by *learning* the rates at which communities change over time (Fig. 2 (c)). The model is based on the intuition that a community may not evolve at a uniform rate over time and that it is worth modeling different stages of community evolution.

(d) Individual user evolution at learned intervals: Last, we consider a model where each user individually progresses between experience levels at their own *personal rate* (Fig. 2 (d)). This model is the most expressive of all four and is able to capture interesting phenomena. For example, some users may become experts very quickly while others may never reach the highest level of experience; others may behave like experts from the time they join (e.g. the bottom right user of Fig. 2 (d)). This model is able capture such types of behavior.

Models (a) and (b) are designed to assess whether user evolution is guided by changes at the level of individual users, or by changes in the community at large. We find that, given enough data, both models lead to modest improvements over traditional latent-factor recommender systems, though there is no clear winner between the two. Once we learn the stages at which users and communities evolve, as in (c) and (d), we significantly outperform traditional recommender systems, though the benefit of learning is much higher when we model evolution at the level of each individual user, i.e., when we treat 'evolution' as analogous to 'becoming an expert'.

Put simply, we fit recommender systems for different stages of user evolution, and the models differ only in terms of how users *progress* between stages. Thus the actual recommender systems used by each model have the same number of parameters, though the models of Figure 2 (c) and (d) have additional parameters that control *when* users evolve. The model of Figure 2 (d) is the most expressive, in the sense that it has enough flexibility to represent each of the other models. For example, if no evolution took place at the level of individual users, the model of Figure 2 (d) could learn latent parameters similar to those of Figure 2 (c). In practice, we find that this is not the case; rather we learn dynamics of individual users that are quite different to those at the level of communities.

Model Specification

We shall first describe our most general model, namely that of Figure 2 (d). The other models in Figure 2 can later be treated as special cases.

We start with the 'standard' latent-factor recommender system [33], which predicts ratings for user/item pairs (u, i) according to

$$rec(u, i) = \alpha + \beta_u + \beta_i + \langle \gamma_u, \gamma_i \rangle.$$

Here, α is a global offset, β_u and β_i are user and item biases (respectively), and γ_u and γ_i are latent user and item features.

Although this simple model can capture rich interactions between users and products, it ignores *temporal* information completely, i.e., users' review histories are treated as unordered sets. Even so, such models yield excellent performance in practice [22].

We wish to encode temporal information into such models via the proxy of *user experience*. We do so by designing separate recommender systems for users who have different experience levels. Naturally, a user's experience is not fixed, but rather it evolves over time, as the user consumes (and rates) more and more products.

For each of a user's ratings r_{ui}, let t_{ui} denote the time at which that review was entered (for simplicity we assume that each product is reviewed only once). For each of a user's ratings, we will fit a latent variable, e_{ui}, that represents the 'experience level' of the user u, at the time t_{ui}.

Each user's experience level evolves over time. As a model assumption, we constrain each user's experience level to be a non-decreasing function of time. That is, a user never becomes *less* experienced as they review additional products. We encode this using a simple monotonicity constraint on time and experience:

$$\forall u, i, j \quad t_{ui} \geq t_{uj} \Rightarrow e_{ui} \geq e_{uj}. \tag{1}$$

What this constraint means from a modeling perspective is that different users evolve in similar ways to each other, regardless of the specific time they arrive in the community.

In practice, we model experience as a categorical variable that takes E values, i.e., $e_{ui} \in \{1 \dots E\}$. Note that it is not required that a user achieves all experience levels: some users may already be experienced at the time of their first review, while others may fail to become experienced even after many reviews.

Assuming for the moment that each experience parameter e_{ui} is observed, we proceed by fitting E separate recommender systems to reviews written at different experience levels. That is, we train $rec_1(u,i) \dots rec_E(u,i)$ so that each rating r_{ui} is predicted using $rec_{e_{ui}}(u,i)$. As we show in Section 3, we regularize the parameters of each of these recommender systems so that user and product parameters evolve 'smoothly' between experience levels.

In short, each of the parameters of a standard recommender system is replaced by a parameter that is a function of experience:

$$\begin{aligned} rec(u,i) &= rec_{e_{ui}}(u,i) \\ &= \alpha(e_{ui}) + \beta_u(e_{ui}) + \beta_i(e_{ui}) + \langle \gamma_u(e_{ui}), \gamma_i(e_{ui}) \rangle. \end{aligned} \tag{2}$$

Such a model is quite general: rather than assuming that our model parameters evolve gradually over time, as in [20], we assume that they evolve gradually as a function of experience, which is *itself* a function of time, but is learned *per user*. Thus we are capable of learning whether users' 'experience' parameters simply mimic the evolution of the entire community, as in Figure 2 (c), or whether there are patterns of user evolution that occur independently of when they arrive in the community, as in Figure 2 (d).

Because of this generality, all of the models from Figure 2 can be seen as special cases of (eq. 2). Firstly, to model evolution of communities (rather than of individual users), we change the monotonicity constraint of (eq. 1) so that it constrains all reviews (rather than all reviews per user):

$$\forall u, v, i, j \quad t_{ui} \geq t_{vj} \Rightarrow e_{ui} \geq e_{vj}. \tag{3}$$

Secondly, the 'learned evolution' models (c, d) and the non-learned models (a, b) differ in terms of how we fit the experience parameters e_{ui}. For the non-learned models, experience parameters are set using a fixed schedule: either they are placed at uniformly spaced time points throughout the entire corpus, as in Figure 2 (a), or they are placed at uniformly spaced time points for each individual user's history, as in Figure 2 (b).

In the next section, we describe how to learn these parameters, i.e., to model the points at which a community or an individual user changes.

3. TRAINING THE MODELS

We wish to optimize our model parameters and latent experience variables so as to minimize the mean-squared-error of predictions on some set of training ratings $r_{ui} \in \mathcal{T}$. Suppose each of our E recommender systems has parameters

$$\Theta_e = (\alpha(e); \beta_u(e); \beta_i(e); \gamma_u(e); \gamma_i(e)),$$

and that the set of all experience parameters e_{ui} is denoted as $\mathcal{E}$. Then we wish to choose the optimal $(\hat{\Theta}, \hat{\mathcal{E}})$ according to the objective

$$(\hat{\Theta}, \hat{\mathcal{E}}) = \underset{\Theta, \mathcal{E}}{\operatorname{argmin}} \sum_{r_{ui} \in \mathcal{T}} \frac{1}{|\mathcal{T}|} (rec_{e_{ui}}(u,i) - r_{ui})^2 + \lambda \Omega(\Theta)$$
$$\text{s.t. } t_{ui} \geq t_{uj} \Rightarrow e_{ui} \geq e_{uj}. \tag{4}$$

This equation has three parts: the first is the mean-squared-error of predictions, which is the standard objective used to train recommender systems. The second part, $\Omega(\Theta)$, is a regularizer, which penalizes 'complex' models Θ. Assuming that there are U users, I items, E experience levels, and K latent factors, then our model has $(1 + U + I + U \cdot K + I \cdot K) \times E$ parameters, which will lead to overfitting if we are not careful. In practice, similar experience levels should have similar parameters, so we define $\Omega(\Theta)$ using the smoothness function

$$\Omega(\Theta) = \sum_{e=1}^{E-1} \|\Theta_e - \Theta_{e+1}\|_2^2, \tag{5}$$

where $||\cdot||_2^2$ is the squared ℓ_2 norm. This penalizes abrupt changes between successive experience levels. λ is a regularization hyperparameter, which 'trades-off' the importance of regularization versus prediction accuracy at training time. We select $\lambda \in 10^0 \dots 10^5$ by withholding a fraction of our training data for validation, and choosing the value of λ that minimizes the validation error. The third and final part of (eq. 4) is the constraint of (eq. 1), which ensures that our latent experience parameters are monotonically non-decreasing for each user.

Simultaneously optimizing all of the parameters in (eq. 4) is a difficult problem, in particular it is certainly not convex [22]. We settle for a local optimum, and optimize the parameters Θ and $\mathcal{E}$ using coordinate ascent [27]. That is, we alternately optimize (eq. 4) for Θ given $\mathcal{E}$, and for $\mathcal{E}$ given Θ.

Optimizing (eq. 4) for Θ given $\mathcal{E}$, while itself still a non-convex problem, can be approached using standard techniques, since it essentially reduces to optimizing E separate recommender systems. In practice we optimize model parameters during each iteration using L-BFGS [31], a quasi-Newton method for non-linear optimization of problems with many variables.

Alternately, optimizing (eq. 4) for $\mathcal{E}$ given Θ means assigning each of a user's reviews to a particular recommender system, corresponding to that review's experience level. The best assignment is the one that minimizes the mean-squared-error of the predictions, subject to the monotonicity constraint.

Optimizing a sequence of discrete variables subject to a monotonicity constraint can be solved efficiently using dynamic programming: it is related to the *Longest Common Subsequence* problem [4], which admits a solution whose running time (per user) is bilinear in E and the number of ratings in their history.

This procedure is visualized in Figure 3, for $E = 5$ experience levels, and a user with 8 ratings. Rows represent each of the five experience levels, while columns represent each of a particular user's ratings, ordered by time. The optimal non-decreasing set of experience levels is the shortest path from the 'start' to the 'end' of this

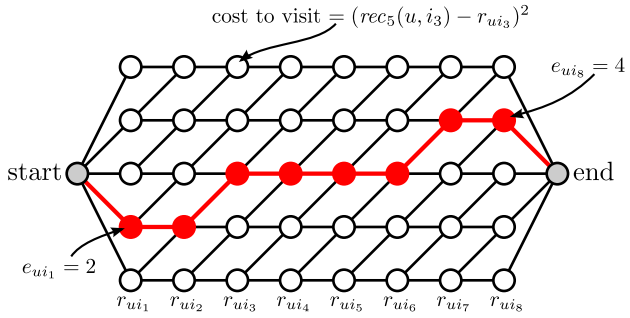


Figure 3: Experience fitting as a dynamic programming problem. Rows represent experience levels, columns represent ratings, ordered by time.

graph, where the cost of visiting a node with rating r_{ui} at experience level k is the prediction error $(rec_k(u, i) - r_{ui})^2$.

These two steps are repeated until convergence, that is, until $\mathcal{E}$ does not change between successive iterations. On our largest datasets, all parameters could be optimized in a few hours on a standard desktop machine.

Again, the above procedure refers to the most general version of our model, in which we learn monotonic evolution parameters *per user*, as depicted in Figure 2 (d). Training the community version of our model (Fig. 2 (c)) simply means replacing the monotonicity constraint of (eq. 1) with that of (eq. 3).

4. EXPERIMENTS

Our goal in this section is to evaluate the models described in Figure 2. We compare the following models:

lf: A standard latent-factor recommender system [21].

a: A model whose parameters evolve for the entire community as a function of time (Fig. 2 (a)).

b: A model whose parameters evolve independently for each user (Fig. 2 (b)).

c: A model whose parameters evolve for the entire community as a function of time, where the 'stages' of evolution are learned (Fig. 2 (c)).

d: A model whose parameters evolve independently for each user, where the stages of evolution are learned (Fig. 2 (d)).

The models of Figure 2 (a) and (c) are most similar to existing models for temporal evolution, e.g. [22]: item parameters are shared by ratings made at the same time. We aim to compare this to models where parameters are shared by users at the same experience level, regardless of the specific time they arrive in the community (as in Fig. 2 (d)).

Experimental Setup

To evaluate each method, we report the Mean Squared Error (MSE) on a fraction of our data withheld for testing, that is, for our test set $\mathcal{U}$ we report

$$\text{MSE}(\mathcal{U}) = \frac{1}{|\mathcal{U}|} \sum_{r_{ui} \in \mathcal{U}} (rec_{e_{ui}}(u, i) - r_{ui})^2. \quad (6)$$

We also use a validation set of the same size to choose the hyperparameter λ. Throughout our experiments we set the number of experience levels, and the number of latent product and item dimensions to $E = 5$ and $K = 5$; larger values did not significantly improve performance in our experience.

Since it is unlikely to be fruitful to model the evolution of users who have rated only a few products, we compare our models on users with at least 50 ratings. Users with fewer than 50 ratings are not discarded, but rather their ratings are combined so that they are treated using a single 'background' model; we then model the evolution of this entire group as though it were a single user.

We use two schemes to build our test sets: our first scheme consists of selecting a *random* sample of reviews from each user. This is the standard way of selecting test data for 'flat' models that do not model temporal dynamics. The second scheme we use to build our test set is to consider the *final* reviews for each user.

The latter setting represents how such a system would be used in practice, in the sense that our goal is to predict how users would respond to a product *now*, rather than to make *post hoc* predictions about how they *would have* responded in the past. However, sampling reviews in this way biases our test set towards reviews written by more experienced users; this is no longer the case when we sample reviews randomly.

Of course, we do not fit latent experience parameters for the ratings in our test set. Thus for each rating used for testing, we assign it the experience level of its chronologically nearest training rating.

dataset	#users	#items	#ratings
Beer (beeradvocate)	33,387	66,051	1,586,259
Beer (ratebeer)	40,213	110,419	2,924,127
Fine Foods (amazon)	218,418	74,442	568,454
Movies (amazon)	759,899	267,320	7,911,684
Wine (cellartracker)	44,268	485,179	2,025,995
TOTAL	1,096,185	1,003,411	15,016,519

Table 1: Dataset statistics.

Datasets

Our choice of rating data reflects a variety of settings where users are likely to have 'acquired tastes'. The datasets we consider are summarized in Table 1. Each of our datasets were obtained from public sources on the web using a crawler, and are made available for others to use.[1] We consider the beer review websites *BeerAdvocate* and *RateBeer*, the wine review website *CellarTracker*, as well as reviews from the *Fine Foods* and *Movies* categories from *Amazon*. In total we obtain over 15 million ratings from these sources. In principle we obtain the *complete* set of reviews from each of these sources; data in each of our corpora spans at least 10 years.

We previously considered *BeerAdvocate* and *RateBeer* data in [28], though not in the context of recommendation. Recommendation on (different) *Amazon* data has been discussed in [15] and [25].

Since each of these rating datasets has a different scale (e.g. beers on *RateBeer* are rated out of 20, wines on *CellarTracker* are rated out of 100, etc.), before computing the MSE we first normalize all ratings to be on the scale $(0, 5]$.

Evaluation

Results in terms of the Mean Squared Error (MSE) are shown in Tables 2 and 3. Table 2 shows the MSE on a test set consisting

[1] `http://snap.stanford.edu/data/`

	BeerAdv. (overall)	BeerAdv. (taste)	BeerAdv. (look)	RateBeer (overall)	Amazon Fine Foods	Amazon Movies	CellarTracker
(lf) latent-factor model	0.452 (.01)	0.442 (.01)	0.313 (.01)	0.496 (.01)	1.582 (.02)	1.379 (.00)	0.055 (.00)
(a) community at uniform rate	0.427 (.01)	0.417 (.01)	0.293 (.01)	0.458 (.01)	1.527 (.02)	1.371 (.01)	0.051 (.00)
(b) user at uniform rate	0.437 (.01)	0.423 (.01)	0.300 (.01)	0.477 (.01)	1.548 (.02)	1.376 (.01)	0.053 (.00)
(c) community at learned rate	0.427 (.01)	0.417 (.01)	0.293 (.01)	0.458 (.01)	1.529 (.02)	1.371 (.01)	0.051 (.00)
(d) user at learned rate	**0.400 (.01)**	**0.399 (.01)**	**0.275 (.01)**	**0.406 (.01)**	**1.475 (.03)**	**1.051 (.01)**	**0.045 (.00)**
benefit of **(d)** over **(lf)**	11.62%	9.73%	12.19%	18.26%	6.79%	23.80%	18.50%
benefit of **(d)** over **(c)**	6.48%	4.12%	6.13%	11.42%	3.53%	23.34%	13.20%

Table 2: Results on users' most recent reviews. MSE and standard error.

	BeerAdv. (overall)	BeerAdv. (taste)	BeerAdv. (look)	RateBeer (overall)	Amazon Fine Foods	Amazon Movies	CellarTracker
(lf) latent-factor model	0.430 (.01)	0.408 (.01)	0.319 (.01)	0.492 (.01)	1.425 (.02)	1.099 (.01)	0.049 (.00)
(a) community at uniform rate	0.415 (.01)	0.387 (.01)	0.298 (.01)	0.463 (.01)	1.382 (.02)	1.082 (.01)	0.048 (.00)
(b) user at uniform rate	0.419 (.01)	0.395 (.01)	0.305 (.01)	0.461 (.01)	1.383 (.02)	1.088 (.01)	0.048 (.00)
(c) community at learned rate	0.415 (.01)	0.386 (.01)	0.298 (.01)	0.461 (.01)	1.374 (.02)	1.082 (.01)	0.048 (.00)
(d) user at learned rate	**0.409 (.01)**	**0.373 (.01)**	**0.276 (.01)**	**0.394 (.01)**	**1.189 (.03)**	**0.711 (.01)**	**0.039 (.00)**
benefit of **(d)** over **(lf)**	5.05%	8.61%	13.45%	19.94%	16.61%	35.31%	20.49%
benefit of **(d)** over **(c)**	1.55%	3.33%	7.20%	14.50%	13.47%	34.32%	18.23%

Table 3: Results on randomly sampled reviews. MSE and standard error.

of the *most recent* reviews for each user, while Table 3 shows the MSE on a *random* subset of users' reviews.

Table 2 shows that our model significantly outperforms alternatives on all datasets. On average, it achieves a 14% reduction in MSE compared to a standard latent factor recommender system, and a 10% reduction compared to its nearest competitor, which models user evolution as a process that takes place at the level of entire communities. Note that due to the large size of our datasets, all reported improvements are significant at the 1% level or better.

By considering only users' most recent reviews, our evaluation may be biased towards reviews written at a high level of experience. To address this possibility, in Table 3 we perform the same evaluation on a *random* subset of reviews for each user. Again, our model significantly outperforms all baselines. Here we reduce the MSE of a standard recommender system by 17%, and the nearest competitor by 13% on average.

Reviews from *BeerAdvocate* and *RateBeer* have multiple dimensions, or 'aspects' to users' evaluations. Specifically, users evaluate beers in terms of their 'taste', 'smell', 'look', and 'feel' in addition to their overall rating [28]. Tables 2 and 3 show results for two such aspects from *BeerAdvocate*, showing that we obtain similar benefits by modeling users' evolution with respect to such aspects. Similar results from *RateBeer* are omitted.

It is perhaps surprising that we gain the most significant benefits on movie data, and the least significant benefits on beer data. However, we should not conclude that movies require more expertise than beer: a more likely explanation is that our movie data has a larger *spectrum* of expertise levels, whereas users who decide to participate on a beer-rating website are likely to already be somewhat 'expert'.

We gain the most significant benefits when considering reviews written at all experience levels (as in Table 3) rather than considering users' most recent reviews (as in Table 2). However we should not conclude from this that experts are unpredictable (indeed in Section 5 we confirm that experts are the *most* predictable). Rather, since inexperienced users are *less* predictable, we gain the most benefit by explicitly modeling them.

We also note that while we obtain significant benefits on *Amazon* data, the mean-squared-errors for this dataset are by far the highest. One reason is that *Amazon* users use a full spectrum of ratings from 1 to 5 stars, whereas *CellarTracker* users (for example) rate wines on a smaller spectrum (after normalizing the ratings, most wines have scores above 4.25); this naturally leads to higher MSEs. Another reason is that our *Amazon* data has many products and users with only a few reviews, so that we cannot do much better than simply modeling their bias terms. As we see in Section 5, bias terms differ significantly between beginners and experts, so that modeling expertise proves extremely beneficial on such data.

5. QUALITATIVE ANALYSIS

So far, we have used our models of user expertise to predict users' ratings, by fitting latent 'experience' parameters to all ratings in our corpora. Now we move on to examine the role of these latent variables in more detail.

Throughout this section we use the term 'expert' to refer to those reviewers (and ratings) that are assigned the highest experience level by our model (i.e., $e_{ui} = E$). We use the term 'beginner' to refer to reviewers and ratings assigned the lowest level (i.e., $e_{ui} = 1$). Again, we acknowledge that 'expertise' is an *interpretation* of our model's latent parameters, and other interpretations may also be valid. However, in this section, we demonstrate that our latent parameters do indeed behave in a way that is consistent with our intuitive notion of expertise.

We begin by examining how a user's experience level impacts our ability to predict their rating behavior. Table 4 compares the prediction accuracy of our model on reviews written at different experience levels. We find in all but one case that users at the highest experience level have the lowest MSE (for *Amazon Movies* they have the second lowest by a small margin). In other words their rating behavior can be most accurately predicted by our model. This is not to say that experts agree with *each other* (which we discuss later); rather, it says that *individual* experts are easier to model than other categories of user. Indeed, one can argue that some notion of

	$e=1$	$e=2$	$e=3$	$e=4$	$e=5$
BeerAdvocate	0.423	0.396	0.471	0.449	**0.358**
RateBeer	0.494	0.469	0.408	0.533	**0.300**
Amazon Fine Foods	1.016	1.914	1.094	2.251	**0.960**
Amazon Movies	0.688	**0.620**	0.685	1.062	0.675
CellarTracker	0.061	0.039	0.041	0.037	**0.028**

Table 4: MSE per experience level e

'predictability' is a necessary condition for such users to be considered 'experts' [12].

While we find that beginners and intermediate users have lower prediction accuracy, it is surprisingly the 'almost experts' ($e_{ui} = E - 1$) who are the *least* predictable; from Table 4 we see that such users have the *highest* MSE in three out of five cases. From this we might argue that users do not become experts via a smooth progression, but rather their evolution consists of several distinct stages.

Experience Progression

Next, we study how users progress through experience levels as a function of time. Figure 4 shows the (cumulative) time taken to progress between experience levels (the final bar represents the entire lifetime of the user, since there is no further level to progress to). The dark blue bars show the progression for those users who progress through all levels of experience, i.e., it ignores those users who arrive to the site already experienced as well as those who never obtain the highest experience levels. The yellow bars show users who reach all but the highest experience level.

How much time is spent at each experience level?

First, we observe that on most datasets, the final experience level is the 'longest', i.e., it covers the longest time period, and includes the largest number of reviews. This makes sense from the modeling perspective, when taken together with our previous finding that experts' ratings are easier to predict: the model is 'finer-grained' during the stages of user evolution that are most difficult to fit accurately. Fewer distinct experience levels are required later on, once users' rating behavior has 'converged'.

Do users who become experts differ from those who don't?

Secondly, Figure 4 compares users who progress through all levels of experience to users who do not. Yellow bars show the progression of users who reach all but the final experience level. Surprisingly, while such users enter roughly the same number of ratings per level (Fig. 4, bottom) as those users who eventually become experts, they do so much slower (Fig. 4, top). Thus it appears as though the *rate* at which users write reviews, and not just the number of reviews they write, is tied to their progression.

Do experts agree with each other?

Thirdly, Figure 5 shows the extent to which users *agree* with each other as they become more experienced. 'Agreement' has been argued to be another necessary condition to define users as experts [12]. To study this, we consider ratings *of the same product, written at the same experience level*. Specifically, for each item i and experience level k, we find the set of users who rated that item at that experience level, i.e., we find all u such that $e_{ui} = k$. We then compute the variance of such ratings for every item and experience level. Our goal is to assess how this quantity changes as a function of users' experience. We do so for all products that were reviewed at least 5 times at the same experience level. Since this limits the amount of data we have to work with, we first linearly interpolate each user's experience function over time (so that their experience function is a piecewise linear function, rather than a step function), and compute this quantity across a sliding window.

Indeed, in Figure 5 we find that users do tend to agree with each other more as they become more experienced, i.e., their ratings have lower variance when they review the same products. This is consistent with our finding that experts' ratings are easier to predict than those of beginners.

User Retention

Next we consider how experience relates to user retention. We want to study how users who leave the community (defined as users who have not entered a review for a period of six months) differ from those who remain in the community. Figure 6 visualizes the experience progression of these two groups. Here we consider the first 10 ratings for all users who have entered at least 10 ratings (Fig. 6, top), and the first 100 ratings for all users who have entered at least 100 ratings (Fig. 6, bottom); this scheme ensures that every datapoint is drawn from the same sample population.

We find that both classes of users *enter* the community at roughly the same level (at the time of their first review, both groups have roughly the same experience on average). However, as the number of reviews increases, users who go on to leave the community have lower experience compared to those who stay. In other words, they gain experience more slowly. This discrepancy is apparent even after their first few reviews. This failure to become experienced may be a factor which causes users to abandon the site, and could be used as a feature in 'churn prediction' problems [11, 18]. We mention the parallel work of [9], which also studies *BeerAdvocate* and *RateBeer* data: there, a user's failure to adopt the *linguistic* norms of a community is considered as a factor that may influence whether they will abandon that community.

Acquired Tastes

In Figure 1, we hinted at the idea that our model could be used to detect *acquired tastes*. More precisely, it can help us to identify products that are preferred by experts over beginners (and *vice versa*).

To do so, we compare the difference in product bias terms between the most expert (experience level 5) and the least expert (experience level 1) users. That is, we compute for each item i the quantity

$$d_i = \beta_i(5) - \beta_i(1).$$

Thus a positive value of d_i indicates that a product is preferred by experts over beginners, while a negative value indicates that a product is preferred by beginners over experts.

How do expert and beginner biases differ?

In Figure 7 we compare the average rating of each product to d_i (for products with at least 50 ratings). Our main finding in this figure is that there exists a positive relationship between products that are highly rated and products that are preferred by experts. In other words, products with high average ratings are rated *more* highly by experts; products with low average ratings are rated more highly by beginners. Recall that in Figure 1 we examined the same relationship on *RateBeer* data in more detail.

One explanation is that the 'best' products tend to be ones that require expertise to enjoy, while novice users may be unable to appreciate them fully. This phenomenon is the most pronounced on

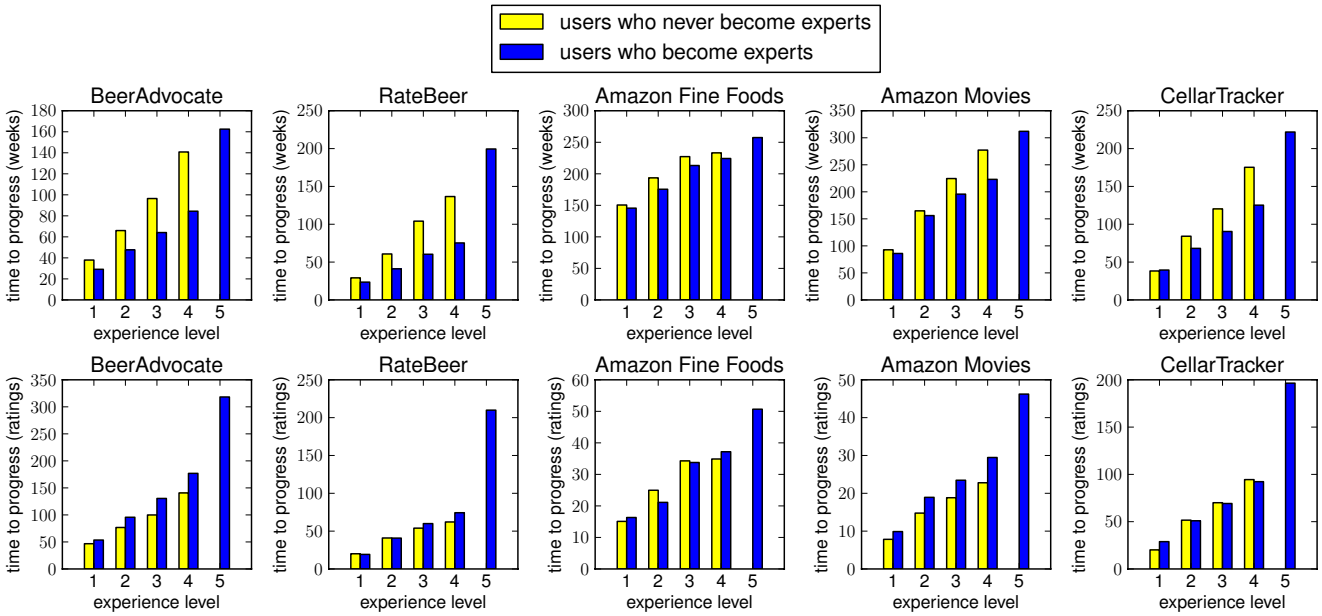


Figure 4: Users who never become 'experts' tend to progress slower than users who do. Cumulative time (top), and number of ratings (bottom), taken to progress between experience levels.

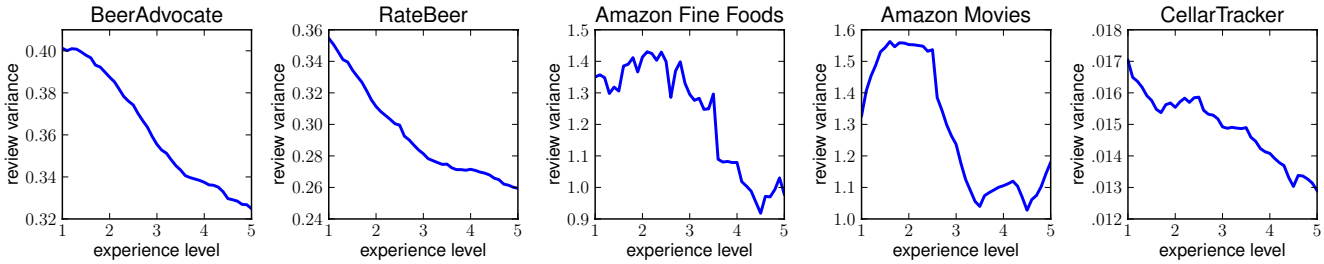


Figure 5: Experienced users agree more about their ratings than beginners. Experience versus rating variance (when rating the same product).

our *Movies* and *RateBeer* data, and exists to a lesser extent on our *BeerAdvocate* and *Fine Foods* data; the phenomenon is absent altogether on our *CellarTracker* data. Again, we should not conclude from this that movies require more 'expertise' than wine, but rather that our *Movies* data has a larger separation between beginners and experts.

Perhaps more surprising is the lack of products that appear in the top left or bottom right quadrants of Figure 7, i.e., products with below average ratings, but positive values of d_i, or products with above average ratings but negative values of d_i. In other words, there are neither products that are disliked by beginners but liked by experts, nor are there products that are liked by beginners but disliked by experts.

It is worth trying to rule out other, more prosaic explanations for this phenomenon: for instance, it could be that beginners give mediocre reviews to all products, while experts have a larger range. We mention two negative results that discount such possibilities: firstly, we found no significant difference between the average ratings given by beginners or experts. Secondly, we did not observe any significant difference in the variance (that is, the variance across all of a user's reviews, not when reviewing the same product as in Figure 5).

Which genres are preferred by experts or beginners?

In Figure 1 we showed that there are entire *genres* of products that tend to be preferred by experts or by beginners. Specifically, we showed that almost all strong ales have positive values of d_i (preferred by experts), while almost all lagers have negative values of d_i (preferred by beginners). Of course, it is not surprising (to a beer drinker) that experts dislike lagers while preferring India Pale Ales (IPAs), though it is more surprising that beginners also have the same polarity with respect to these products—the experts are simply more extreme in their opinions.

Table 5 shows which genres have the lowest and highest values of d_i on average, i.e., which products are most preferred by beginners and experts (respectively). We focus on *BeerAdvocate*, *RateBeer*, and *CellarTracker*, which have the most meaningful genre information. The results are highly consistent across *BeerAdvocate* and *RateBeer*, in spite of the differing product categorizations used by the two sites (Kvass is a form of low-alcohol beer, Kristallweizen is a form of wheat beer, IPA is a form of strong ale, and Gueuze is a type of lambic). Again, there is a clear relationship between products' overall popularity and the extent to which experts prefer them; non-alcoholic beer is naturally not highly rated on a beer rating website, while lambics and IPAs are more in favor.

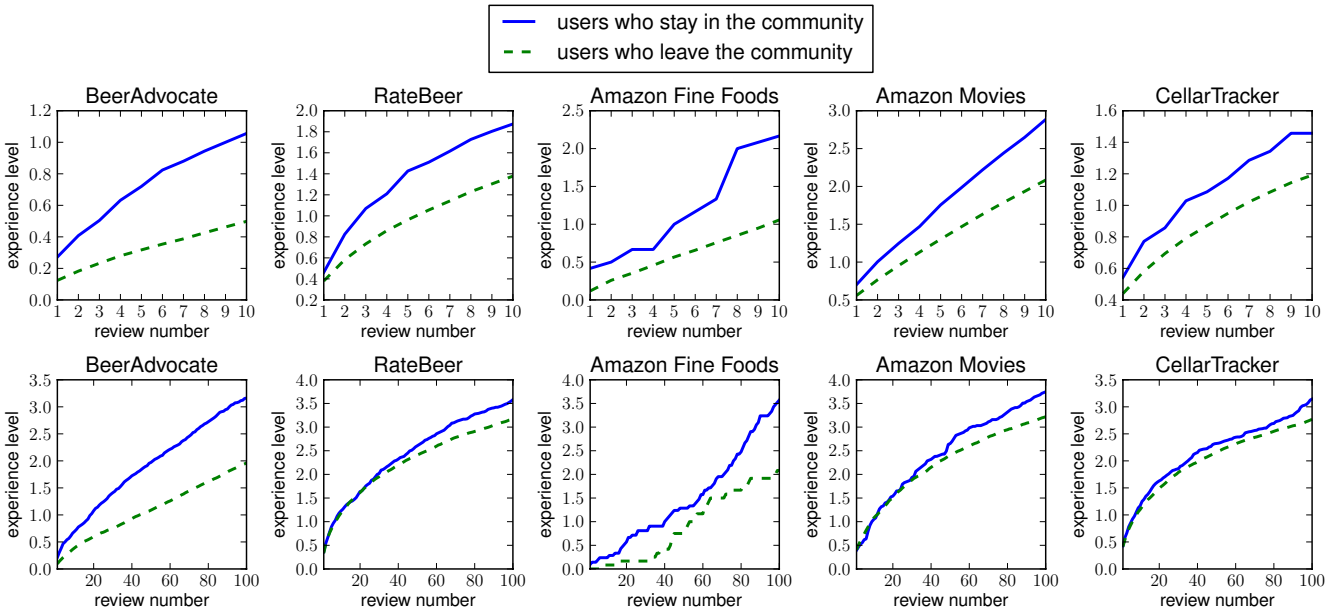


Figure 6: Users whose experience progresses slowly are more likely to abandon the community. First 10 ratings of all users who have at least 10 ratings (top), and first 100 ratings of all users who have at least 100 ratings (bottom).

6. RELATED WORK

Traditional recommender systems treat each user's review history as a series of unordered events, which are simply used to build a model for that user, such as a latent factor model [22]. In spite of the excellent performance of such models in practice, they naturally fail to account for the temporal dynamics involved in recommendation tasks.

Some early works that deal with temporal dynamics do so in terms of *concept drift* [23, 37, 39]. Such models are able to account for short-term temporal effects ('noise'), and long-term changes in user behavior ('drift'), for example due to the presence of new products within a community.

Sophisticated models of such temporal dynamics proved critical in obtaining state-of-the-art performance on the *Netflix* challenge [3], most famously in [20]. As discussed in [20], few previous works had dealt with temporal dynamics, other than a few notable exceptions [2, 10, 35]. Around the same time, 'adaptive neighborhood' models were proposed [24], that address the problem of iteratively training recommender systems whose parameters ought to change over time.

Better performance may be obtained by modeling large-scale global changes at the level of entire communities [41], or by developing separate models for short term changes (e.g. due to external events), and long-term trends [40].

Other works that study temporal dynamics at the level of products and communities include [29], where the authors studied how existing ratings within communities may influence new users, and how community dynamics evolve; and [13], who studied how users are influenced by previous ratings of the same product.

Expertise has been studied in domains other than recommender systems, for example in the literature on education and psychology [5, 34]. One area where 'expertise' has received significant attention is *web search*. The role of expertise with respect to search behavior is a rich and historied topic, whose study predates the emergence of modern search engines [14]. Of particular interest is [38], since the authors study how users *evolve* (with respect to the level of technical content in their queries) as they gain expertise. We also briefly mention the topic of *expertise identification* [1, 6, 16, 32]. This line of work is orthogonal to ours, in that it deals with *discovering experts*, rather than *recommending products based on expertise*; however, such works offer valuable insights, in the sense that like our own work, they attempt to model the behavior of expert users.

7. DISCUSSION AND FUTURE WORK

An interesting finding of our work is that beginners and experts have the same *polarity* in their opinions, but that experts give more 'extreme' ratings: they rate the top products more highly, and the bottom products more harshly. Thus naively, we might conclude that we should simply recommend both groups of users the same products: nobody likes adjunct lagers, so what does it matter if beginners dislike them *less*? The counter to this argument is that in order to *fully* appreciate a product (by giving it the highest rating), a user must first become an expert. Thus perhaps we should focus on *making a user an expert*, rather than simply recommending what they will like *today*.

This viewpoint motivates several novel questions. Can we determine, based only on which products a user reviews, whether they will become an expert? Can we recommend not just products, but *sequences* of products, that will *help* them to become an expert, or maximize their total enjoyment?

Another avenue of research is to study *linguistic* differences between experts and non-experts. 'Expertise' has been studied from the perspective of linguistic development [34], for example in the context of second-language acquisition [8, 36]. Since our rating data comes from *review* corpora, we can use it to study how users' *rating* expertise relates to their *reviewing* expertise. Do experts write longer reviews or use fewer personal pronouns? Are their reviews considered more helpful by others in the community [30]?

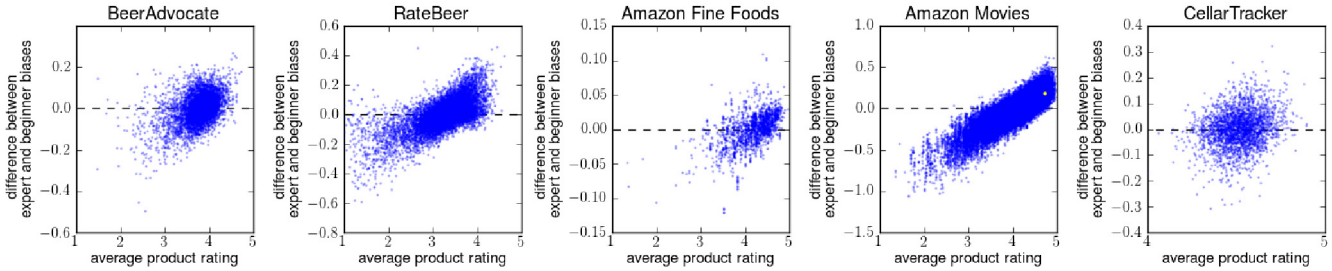


Figure 7: Average product ratings are correlated with the difference between expert and beginner biases. *Seven Samurai* is marked in yellow.

	BeerAdvocate	$\bar{\beta}_i(1)$	$\bar{\beta}_i(5)$	RateBeer	$\bar{\beta}_i(1)$	$\bar{\beta}_i(5)$	CellarTracker	$\bar{\beta}_i(1)$	$\bar{\beta}_i(5)$
Preferred	Low Alcohol Beer	-.423	-.534	Low Alcohol	-.581	-.724	Barbera	-.035	-.119
by beginners:	Kvass	-.316	-.653	Pale Lager	-.519	-.630	Syrah (blend)	-.110	-.182
	Light Lager	-.302	-.487	Premium Lager	-.246	-.290	Cabernet-Syrah	-.048	-.105
	American Adjunct Lager	-.237	-.285	American Dark Lager	-.199	-.287	Zinfandel	-.066	-.111
	European Strong Lager	-.403	-.428	Strong Pale Lager	-.068	-.103	Sémillon	-.092	-.126
	European Pale Lager	-.154	-.216	German Pilsener	-.135	-.202	Syrah	.019	.005
	Japanese Rice Lager	-.144	-.212	Pilsener	-.066	-.095	Port	-.028	-.046
	American Pale Wheat Ale	.033	-.023	Kristallweizen	.060	.016	Grenache (blend)	.011	-.005
	American Blonde Ale	-.047	-.080	Fruit Beer	-.105	-.137	Sauvignon Blanc	-.003	-.020
	English Dark Mild Ale	-.025	-.072	Malt Liquor	-.557	-.675	Gr uner Veltliner	.024	.008
Preferred	Baltic Porter	.091	.128	Strong Porter	.205	.243	Melon de Bourgogne	.261	.412
by experts:	English Barleywine	.007	.055	Barley Wine	.216	.248	Champagne	.080	.193
	American Wild Ale	.150	.196	Wild Ale	.197	.261	Cabernet-Syrah (blend)	.083	.173
	English Pale Mild Ale	-.011	.023	American Strong Ale	.203	.227	Petit Verdot	.054	.143
	Flanders Red Ale	.132	.183	Double IPA	.260	.294	Pinot Gris	.118	.206
	Flanders Oud Bruin	.018	.073	Black IPA	.152	.185	Pinotage	.064	.117
	Unblended Lambic	-.019	.028	Unblended Lambic	.135	.240	Grenache	.009	.048
	Gueuze	.160	.223	Saison	.142	.176	Grenache Blanc	.038	.089
	Chile Beer	-.254	-.223	Imperial Stout	.308	.338	Dolcetto	.063	.105
	Rauchbier	-.143	-.095	Quadrupel	.354	.367	Mourvedre Blend	-.031	.007

Table 5: Acquired tastes: products preferred by beginners and products preferred experts. Average beginner and expert item biases are shown.

Do experts use specialized vocabulary (e.g. beer-specific language such as 'lacing' and 'retention'), and do they better conform to the linguistic norms of the community [9]?

Although we argued that gaining expertise is a form of *personal* development that takes place no matter when a user arrives in a community, it is not the *only* such form of development. When a user arrives in a community, they must adopt its norms with respect to rating behavior [29], for example they must learn that while wines are rated on a scale of 1-100, ratings below 85 are seldom used. Like expertise, this form of development is not tied to the user's changing preferences, nor to the changing tastes of the community. Would *BeerAdvocate* 'experts' be considered experts if their ratings were entered on *RateBeer*? Would expert movie reviewers on *Amazon* also be 'experts' gourmet food reviewers? The answers to such questions could help us to determine which forms of personal development are more salient in online communities.

Finally, we believe that our models of expertise may be used to facilitate expert discovery. We identified experts in review systems primarily for the sake of predicting users' ratings, but discovering experts may be useful in its own right. This topic is known as *expert recommendation* [1, 6, 7, 16, 17], and has applications to product ranking and review summarization, among others. For instance, a user may wish to read reviews written by the most expert members of a community, or by members most similar in expertise to themselves.

8. CONCLUSION

Users' tastes and preferences change and evolve over time. Shifting trends in the community, the arrival of new products, and even changes in users' social networks may influence their rating behavior. At the same time, users' tastes may change simply through the act of consuming additional products, as they gain knowledge and experience. Existing models consider temporal effects at the level of products and communities, but neglect the *personal development* of users: users who rate products at the same time may have less in common than users who rate products at *different* times, but who are at the same stage in their personal evolution. We developed models for such notions of user evolution, in order to assess which best captures the dynamics present in product rating data. We found that modeling users' personal evolution, or 'experience', not only helps us to discover 'acquired tastes' in product rating systems, but more importantly, it allows us to discover *when* users acquire them.

Acknowledgements

Thanks to Cristian Danescu-Niculescu-Mizil for help obtaining the data, and to Seth Myers and Dafna Shahaf for proofreading. This research has been supported in part by NSF IIS-1016909, CNS-1010921, CAREER IIS-1149837, IIS-1159679, ARO MURI, Docomo, Boeing, Allyes, Volkswagen, Intel, Okawa Foundation, Alfred P. Sloan Fellowship and the Microsoft Faculty Fellowship.

9. REFERENCES

[1] O. Alonso, P. Devanbu, and M. Gertz. Expertise identification and visualization from CVS. In *Working Conference on Mining Software Repositories*, 2008.

[2] R. Bell and Y. Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *ICDM*, 2007.

[3] J. Bennett and S. Lanning. The Netflix prize. In *KDD Cup and Workshop*, 2007.

[4] L. Bergroth, H. Hakonen, and T. Raita. A survey of longest common subsequence algorithms. In *SPIRE*, 2000.

[5] D. Berliner. *The Development of Expertise in Pedagogy*. AACTE Publications, 1988.

[6] J. Bian, Y. Liu, D. Zhou, E. Agichtein, and H. Zha. Learning to recognize reliable users and content in social media with coupled mutual reinforcement. In *WWW*, 2009.

[7] N. Craswell, D. Hawking, A. Vercoustre, and P. Wilkins. P@noptic expert: Searching for experts not just for documents. *Ausweb Poster Proceedings*, 2001.

[8] A. Cumming. Writing expertise and second-language proficiency. *Language learning*, 2006.

[9] C. Danescu-Niculescu-Mizil, R. West, D. Jurafsky, J. Leskovec, and C. Potts. No country for old members: User lifecycle and linguistic change in online communities. In *WWW*, 2013.

[10] Y. Ding and X. Li. Time weight collaborative filtering. In *CIKM*, 2005.

[11] G. Dror, D. Pelleg, O. Rokhlenko, and I. Szpektor. Churn prediction in new users of Yahoo! Answers. In *WWW CQA Workshop*, 2012.

[12] H. Einhorn. Expert judgment: Some necessary conditions and an example. *Journal of Applied Psychology*, 1974.

[13] D. Godes and J. Silva. Sequential and temporal dynamics of online opinion. *Marketing Science*, 2012.

[14] I. Hsieh-Yee. Effects of search experience and subject knowledge on the search tactics of novice and experienced searchers. *Journal of the American Society for Information Science*, 1994.

[15] N. Jindal and B. Liu. Opinion spam and analysis. In *WSDM*, 2008.

[16] P. Jurczyk and E. Agichtein. Discovering authorities in question answer communities by using link analysis. In *CIKM*, 2007.

[17] M. Karimzadehgan, R. White, and M. Richardson. Enhancing expert finding using organizational hierarchies. *Advances in Information Retrieval*, 2009.

[18] M. Karnstedt, T. Hennessy, J. Chan, P. Basuchowdhuri, C. Hayes, and T. Strufe. Churn in social networks. In *Handbook of Social Network Technologies*. Springer, 2010.

[19] J. Kolter and M. Maloof. Dynamic weighted majority: An ensemble method for drifting concepts. *JMLR*, 2007.

[20] Y. Koren. Collaborative filtering with temporal dynamics. *Commun. ACM*, 2010.

[21] Y. Koren and R. Bell. Advances in collaborative filtering. In *Recommender Systems Handbook*. Springer, 2011.

[22] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 2009.

[23] L. Kuncheva. Classifier ensembles for changing environments. In *Multiple Classifier Systems*. Springer, 2004.

[24] N. Lathia, S. Hailes, and L. Capra. Temporal collaborative filtering with adaptive neighbourhoods. In *SIGIR*, 2009.

[25] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 2003.

[26] H. Ma, D. Zhou, C. Liu, M. Lyu, and I. King. Recommender systems with social regularization. In *WSDM*, 2011.

[27] D. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambrdige University Press, 2003.

[28] J. McAuley, J. Leskovec, and D. Jurafsky. Learning attitudes and attributes from multi-aspect reviews. In *ICDM*, 2012.

[29] W. Moe and D. Schweidel. Online product opinions: Incidence, evaluation, and evolution. *Marketing Science*, 2012.

[30] D. Nguyen and C. Rosé. Language use as a reflection of socialization in online communities. In *ACL Workshop on Language in Social Media*, 2011.

[31] J. Nocedal. Updating quasi-newton matrices with limited storage. *Mathematics of Computation*, 1980.

[32] A. Pal, R. Farzan, J. Konstan, and R. Kraut. Early detection of potential experts in question answering communities. In *UMAP*, 2011.

[33] F. Ricci, L. Rokach, B. Shapira, and P. Kantor, editors. *Recommender Systems Handbook*. Springer, 2011.

[34] S. Romaine. *The language of children and adolescents: The acquisition of communicative competence*. Wiley, 1984.

[35] K. Sugiyama, K. Hatano, and M. Yoshikawa. Adaptive web search based on user profile constructed without any effort from users. In *WWW*, 2004.

[36] S. Thorne, R. Black, and J. Sykes. Second language use, socialization, and learning in internet interest communities and online gaming. *The Modern Language Journal*, 2009.

[37] A. Tsymbal. The problem of concept drift: Definitions and related work. Technical report, Trinity College Dublin, 2004.

[38] R. White, S. Dumais, and J. Teevan. Characterizing the influence of domain expertise on web search behavior. In *WSDM*, 2009.

[39] G. Widmer and M. Kubat. Learning in the presence of concept drift and hidden contexts. In *Machine Learning*, 1996.

[40] L. Xiang, Q. Yuan, S. Zhao, L. Chen, X. Zhang, Q. Yang, and J. Sun. Temporal recommendation on graphs via long- and short-term preference fusion. In *KDD*, 2010.

[41] L. Xiong, X. Chen, T. Huang, J. Schneider, and J. Carbonell. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *SDM*, 2010.