A Market-based Approach to Address the New Item Problem

Sarabjot Singh Anand
Department of Computer Science
University of Warwick
Coventry, United Kingdom
s.s.anand@warwick.ac.uk

Nathan Griffiths
Department of Computer Science
University of Warwick
Coventry, United Kingdom
n.e.griffiths@warwick.ac.uk

ABSTRACT

In this paper we propose a market-based approach for seeding recommendations for new items in which publishers bid to have items presented to the most influential users for each item. Users are able to select (or not) items for rating on an earn-per-rating basis, with payment given for providing a rating regardless of whether the rating is positive or negative. This approach reduces the time taken to obtain ratings for new items, while encouraging users to give honest ratings (to increase their influence) which in turn supports the quality of recommendations. To support this approach we propose techniques for inferring the social influence network from users' rating vectors and recommendation lists. Using this influence network we apply existing heuristics for estimating a user's influence, adapting them to account for the new items already presented to a user. We also propose an extension to Chen *et al.*'s Degree Discount heuristic [Chen et al. 2009], to enable it to be used in this context. We evaluate our approach on the MovieLens dataset and show that we are able to reduce the time taken to achieve coverage, while supporting the quality of recommendations.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: Information Search; I.2.6 [**Artificial Intelligence**]: Learning

General Terms

Algorithms, Experimentation

Keywords

New item problem, influence spread, social networks

1. INTRODUCTION

The key issue with new items being introduced into a collaborative filtering based recommender system is that these items must get rated by a few users before the item can get recommended. The time elapsed between release of the new item into the market and it first appearing within a recommendation list is referred to as *item latency*. Reducing this item latency benefits: (i) the producer of the item, since the item will reach the audience quicker, resulting in increased revenue through sales, (ii) the consumer of the item (referred to as the user) who gains access to an item of interest sooner, and (iii) the owner of the recommender system (the retailer) who gains by increasing customer retention, loyalty and sales revenue by recommending the item to customers before their competitors.

To date, two methods have been studied in literature and allegedly practised by the publisher, or producer, that aim to influence the recommender system and/or influence the customer. Namely, getting reviewers to write positive reviews for their product, and attacking the recommender system by injecting profiles (shilling attacks) [Mobasher et al. 2007]. Both of these methods aim to create positive sentiment in the market and may or may not be targeted to particular segments. The focus is on increasing sales without necessarily taking customer satisfaction into account.

With the proliferation of social networks on the Internet, marketeers have turned to viral marketing. The key here is to pick a small number of "influential" people within a social network that become champions for products through (mis)informing others about the value of the product. On the other hand, retailers have considered various ways, typically based on item descriptors (content) to predict user ratings for new items and hence recommend them sooner than the organic rise of the item in recommendation lists.

In this paper, motivated by the effectiveness of online auctions such as Google Adwords[1], we propose a novel approach to introducing new items into collaborative recommender systems, adopting ideas from auctions and social network analysis. Our proposal is to produce a ranked earn-per-rating (EPR) list of new items for each user from which the user can choose and provide a rating for a payment. In contrast to producers competing to get a high ranking in the adspace on the search results page for particular search terms, producers here bid to gain high rankings on the earn-per-rating lists of the most influential users of the recommender system for their item. Achieving higher rankings should lead to greater coverage and improved visibility.

The key contributions of the paper are: (i) we present a market-based approach for seeding recommendations for new items to reduce item latency, in which customer satisfaction is an integral part of the approach, unlike other

[1] http://adwords.google.com

RecSys'11, October 23–27, 2011, Chicago, Illinois, USA.

methods for publishers to influence the recommender system, such as injecting profiles or paid positive reviews; (ii) using the concept of 'influence' in social networks we provide a mechanism for identifying the best set of users to which we present new items on an earn-per-rating basis; (iii) we present a selection of mechanisms for inferring the social influence network from users' rating vectors and recommendation lists; (iv) we apply and evaluate existing heuristics for estimating the influence of a user in a social network in the context of recommender systems, adapting them to take into account the new items already presented to a user; and (v) we propose and evaluate an extension to Chen *et al.*'s Degree Discount heuristic [Chen et al. 2009], to enable it to be used in directed weighted graphs (since the original heuristic only applies to undirected unweighted graphs).

The remainder of this paper is organised as follows. Section 2 discusses related work. Our approach is introduced in Section 3, and our experimental setup described in Section 4. We discuss our results in Section 5, and describe our conclusions and directions for future work in Section 6.

2. RELATED WORK

2.1 Addressing Cold Start

Cold Start is the general term used to refer to three different situations faced by a collaborative filtering based recommender: the system cold start where the rating matrix is empty, the user cold start in that new users need to rate items before they receive any recommendations, and the item cold start in that a new item must be rated by a "sufficient" number of users before it can be recommended. In this paper we focus on the item cold start problem (also referred to as the new item problem) [Huang et al. 2004, Lam et al. 2008, Park and Chu 2009]

Park et al. use global user bots that either aggregate ratings in the rating matrix or use item content features [Park et al. 2006]. The bots are treated like any other user by the collaborative filtering algorithm. They show that using as few as 7 bots can improve the accuracy of the recommender system in new user, new item and new system (sparse matrix) situations.

Park and Tuzhilin propose the clustering of long tail items (i.e. items with below a user defined number of ratings) [Park and Tuzhilin 2008]. The clustering is based on a set of derived user and item attributes such as the average rating, popularity of items rated and likability of items. Their results suggest improved accuracy is achievable through clustering long tail items. Note that the minimum frequency of an item was set to 10. In this paper we propose a methodology for dealing with items with no ratings.

Schein et al. propose incorporating item content into the recommendation process by building a person/actor aspect model [Schein et al. 2002]. Using the MovieLens dataset they map all movies onto the top k actors acting in the movie and use Hofmann's aspect model to model the joint probability of users and actors using a set of latent classes z. To recommend a movie, m, they map m onto its set of actors and estimate the probability $p(z|m)$ which is in turn used to estimate $p(u|m)$ for user u.

Amazon's Vine Program[2] invites selected reviewers to review pre-release and new items based on feedback on their past reviews by other customers. While the goal here is to provide potential customers of the products with third party reviews to support their purchase decision, it appears to be divorced from Amazon's recommender system. In this paper we look specifically at how a market based approach can be incorporated into the recommender system so that potential customers gain awareness of new items through the recommender system.

[2]http://www.amazon.com/gp/vine/help

2.2 Influence Propagation in Social Networks

An important goal of marketers is to leverage influential nodes (or the *network value* of nodes [Domingos and Richardson 2001]) within a social network to market products to the whole network. This is referred to as the *influence maximization problem* where we aim to pick a (minimal) set of users that would maximize the spread of information through the social network. Several influence propagation models have been proposed in social network analysis literature [Domingos and Richardson 2001, Kempe et al. 2003]. The target set of nodes are set as active at the start of influence propagation. In subsequent cycles, neighbours of active nodes are made active according to some model of influence propagation. The models can be classified into two types: those that use node-specific thresholds and those based on interacting particle systems [Kempe et al. 2003].

In the *Linear Threshold Model*, a node is influenced by each of its neighbours to varying degrees, as defined by the edge weights. Each node v chooses a threshold θ_v. When the sum of the weights of the active neighbours of v exceeds θ_v, v becomes active.

In the *Independent Cascade Model* (ICM) [Goldenberg et al. 2001], each time a node v becomes active it gets one chance to activate each of its inactive neighbours w, with some probability p_{vw}. Kempe et al. showed that a simple hill-climbing algorithm can be guaranteed to find a set of k target nodes that has a performance slightly better than 63% of the optimal set [Kempe et al. 2003]. They also proposed a weighted variant of the ICM where the probability of a node v influencing node w is dependent on the degree of w. One of the key issues with the greedy approach is the need to estimate the quality of the target set. Numerous heuristics have been proposed to improve the speed of estimating the influence spread of a node [Anagnostopoulos et al. 2008, Chen et al. 2009], but it remains a problem for large networks.

Domingos and Richardson propose modelling users using a Markov random field [Domingos and Richardson 2001]. The network value of a user is then computed as the difference between the profit when marketing to a user or not doing so. They use the EachMovie rating matrix along with movie genre as input to their estimation of the probability of a user purchasing a movie. Of particular interest in the context of this paper is the fact that when evaluating their proposal, all movies rated by less than 1% of the customer base were removed from the data. It is these "new" movies specifically that are of interest to us in this paper.

3. A MARKET-BASED APPROACH

Our proposed solution to introducing a new item into the recommender system is to choose a set of users and to persuade them to rate the item. As we assume a cost associated with getting users to rate the item, we want to minimize the cost while maximising the reach of the item within the sub-

U	the set of users
I	the set of items
I_t	the set of new items at time t
L_a^t	the ranked list of new items (EPR list) for user u_a at time t
L^t	the set of EPR lists for all users at time t
t_k	the time when item i_k was introduced into I
S_k^t	the set of users u_a for whom item $i_k \in L_a^t$
$\hat{U}_k^t$	$\hat{U}_k^t \subset U = \cup_{t_k \leq t' \leq t} S_k^{t'}$ is the subset of users u_a for whom i_k has appeared in L_a on at least one occasion prior to $t \geq t_k$
U_k^t	$U_k^t \subset \hat{U}_k^t$ is the set of users that have chosen to rate item i_k by choosing it from their list $L_a^{t'}$ at a time $t' < t$.
B_k^t	the total budget for item $i_k \in I_t$
b_k^t	the bid for item $i_k \in I_t$, i.e. the amount paid to the most influential user for a rating
R_k^t	the total revenue made by users in time t from rating item i_k
$\mathcal{I}(u_a\|G^t)$	the network value (influence spread) of user u_a on the social network G^t
$pu(l_{ak}^t)$	the probability of an item i_k in L_a^t ranked l_{ak}^t being chosen by user u_a for rating
m_k^t	price paid to rater of item i_k who choses to rate it at time t

Table 1: Concepts and notation used.

set of users who have an interest in the item. We assume that at any time t there is a set of new items I_t competing for ratings from users. A producer of items assigns an overall budget and a bid amount for each item introduced into the market at time t. The budget determines the maximum overall spend for a new item and the bid is the maximum price paid to a user for a rating. The actual price paid is dependent on the influence of the user with respect to the item. In the following description we use the concepts and notation defined in Table 1.

The recommender system places a new item in a user's EPR list according to the item rank for the user. The rank l_{ak}^t of an item i_k within a user u_a's EPR list L_a^t is a function of the bid price for the item b_k^t and the *rate of uptake* of the item ru_k^t at time t. The rate of uptake is defined as:

$$ru_k^t = \begin{cases} 1 & \text{if } t = t_k \\ \dfrac{\sum_{u_a \in U_k^t} pu(l_{ak}^t)}{\sum_{u_a \in \hat{U}_k^t} pu(l_{ak}^t)} & \text{otherwise} \end{cases} \quad (1)$$

where $t \geq t_k$. The *rate of uptake* is a measure of the average appeal of the item computed from the uptake of the item for rating by users from their respective EPR lists since time t_k.

The number of users with i_k on their EPR list is determined by the budget and bid for the item, such that $|S_k^t| = B_k^t / b_k^t$. The price paid by the producer of i_k to each user in S_k^t that decides to rate the item is $m_k^t = b_k^t \times (\mathcal{I}(S_k^t)/\mathcal{I}(\hat{S}_k^t))$, where $\hat{S}_k$ is the optimal seed set of size $|S_k^t|$ for item i_k.

The expected value of influence spread for S_k^t is then,

$$E[\mathcal{I}(S_k^t|G^t)] = \sum_{u_a \in S_k^t} pu(l_{ak}^t)\mathcal{I}(u_a|G^t)$$

Similarly the expected value of the revenue for the paid raters is $E[R_k^t] = m_k^t|S_k^t|ru_k^t$. Hence the objective function of our combinatorial opimization problem is:

$$\underset{\{S_k^t : i_k \in I_t\}}{\arg\max} \sum_{i_k \in I_t} E[\mathcal{I}(S_k^t|G^t)]E[R_k^t] \quad (2)$$

Thus, we want to allocate items to each user's EPR list so as to maximize the influence spread for the new items in I_t while maximizing revenue for the users. In order to perform this allocation we must address three fundamental questions. First, how do you transform a rating matrix into an influence network G_t of users (noting that this graph itself is dynamic since the influence of user u_i on user u_j will change as ratings are added)? Second, how do you compute the influence of a user over another? Third, how does influence spread within the influence network? In the remainder of this section we describe our solutions to these questions.

3.1 Inferring the Influence Network

We define an influence network as a weighted graph $G_t = \langle U, E_t \rangle$. The weight of an edge represents the influence of one node on another. The graph may be directed or undirected depending on whether the influence metric is symmetric or not. In the context of recommendation systems the medium of influence is the recommendation lists, and the list for a user u_j is dependent on the various users in u_j's neighbourhood. Influence could be computed based on the two methods for generating a user's neighbourhood, either using a threshold on similarity or fixing the number of neighbours for each user. Hence E_t can be defined in one of two ways, leading to two separate influence networks:

G1: $(u_i, u_j) \in E_t$ if the similarity between u_i and u_j is greater than some threshold τ. This is a symmetric measure and so the resulting graph is undirected.

G2: $(u_i, u_j) \in E_t$ if u_i is one of the k most similar users to u_j given a fixed number of neighbours k from which recommendations are generated. This is not a symmetric measure of influence, since u_j may not be one of the k neighbours of u_i despite the inverse being true. Hence this leads to a directed graph.

For both of these graphs, there are two alternatives for determining influence weights, in addition to simply using the similarity between users. These methods are based on the following principle. User u_j's influence on u_i can result in certain items (those liked by u_j) appearing in u_i's recommendation list. Equally, the influence of u_j could result in items not liked by u_j not appearing in u_i's recommendation list, despite u_i's other neighbours liking them. To model this dual role of item promotion and demotion within the candidate list of recommendations for u_i, we propose the following two approaches. (Note that p below can either be 1 or 2, depending on which of the underlying graphs, G1 or G2, are used)

G3|$\mathbf{G_p}$: the similarity between the rating vector of u_j and the unpruned recommendation list (candidate items and their ratings) for u_i can be used as the edge weight.

G4|$\mathbf{G_p}$: the similarity between the recommendation list for u_i with and without u_j in the neighbourhood could be used as a measure of influence. The lower the similarity, the greater the influence of u_j on u_i.

This results in six alternative ways of defining the influence network. In this paper we limit our experiments to G2, G3|G2 and G4|G2 as previous research has shown that a fixed neighbourhood size results in better recommendations than using a similarity threshold [Herlocker et al. 1999].

3.2 Computing Influence and Influence Spread

There are four ways in which the influence spread of a user can be modelled within the influence graph:

- $\mathcal{I}(u_j|G_t)$, where influence spread is independent of the item,
- $\mathcal{I}(u_j|c(i_k), G_t)$, where $c : I \rightarrow D$ and D is the content description of items, meaning that influence spread is dependent on the content of the item,
- $\mathcal{I}(u_j|U_k^t, G_t)$, where the influence spread is dependent on the users that have already rated the item, and
- $\mathcal{I}(u_j|c(i_k), U_k^t, G_t)$, where the influence spread is dependent on the users that have already rated the item and the content of the item.

Now suppose that we have $|I_t|$ new items that need to be introduced into the recommender system at time t, and so we must find $|I_t|$ sets of users. Clearly there is limited number of items an individual can rate at any time and so there is a constraint on the solution that maximizes the objective.

A user may want to maximize their revenue or enjoyment in consuming the new items. Clearly a user can maximise revenue by rating lots of items randomly. So a question that arises is: why should a rater of a new item rate it accurately? The incentive for providing accurate ratings for items is that it ensures that the user's influence on others remains high. This in turn leads to improved offers to rate new items in the future and to revenue. Also, since the ratings are added to the user's profile, poor ratings will result in poor recommendations in the future. We however acknowledge that the effect on future earnings may not necessarily create incentives of the right magnitude as suggested in previous literature [Lambert and Shoham 2008] and future research will look to quantify the effect of inaccurate ratings or indeed methods for ensuring more truthful ratings are input.

The producer wants to maximize return on their investment and the retailer wants to maximize customer loyalty (and hence revenue) for both paid and unpaid raters. We assume that if the recommender system is accurate over the long term, it will increase customer loyalty. Hence, from the perspective of the retailer, the objective of any intervention is to make the probability of an item being recommended converge to the expected value, or for the average rating for an item to converge to its expected value in a shorter time.

As stated in Equation 2 the solution to the combinatorial optimisation problem is a set of seed users S_k^t. The ranking of new items presented to the corresponding users is affected by the potential of the user to influence others, the bid by the producer and the likelihood of a user choosing the item. To compute the influence of a set of users we now adapt the following algorithms of influence spread in social network literature to our domain.

1. Degree Centrality: This method produces a ranking of users in descending order of their outdegree in G_t. The seed set S_k^t is simply the set containing the top $|S_k^t|$ elements of the list. We multiply the degree for each user u_a with the uptake probability $pu(l_{ak}^t)$ to account for the position that the item i_k would be in the user u_a's EPR list. The drawback of using Degree Centrality for seed selection is that it does not take into account the overlap in the set of nodes in the network influenced by the individually chosen seeds nodes. Hence, the influence spread computed for each potential seed u_j, $\mathcal{I}(u_j|G_t)$, is independent of U_k^t.
2. Single Discount: In the Single Discount model [Chen et al. 2009], when computing the degree of a node v, the edges from v to any nodes that are already within the seed set S_k^t are ignored. Given that the original Single Discount method was based on an undirected social network, we only consider the outdegree of the node when computing its influence spread.
3. Degree Discount: Also proposed by Chen et al. and originally for undirected graphs, this model discounts the degree of the node v by $2t_v + (d_v - t_v)t_v p$, where d_v is the degree of v and t_v is the number of nodes in v's neighbourhood that are already in the seed set [Chen et al. 2009]. Given that we have a directed graph, for each node v we distinguish between the outdegree do_v and indegree di_v, and their associated sets of neighbours $D_o(v)$ and influencers $D_i(v)$, and calculate the discounted degree of v as the expected number additional nodes directly influenced by v as $do_v - so_v - si_v - (do_v - so_v)p$, where so_v is the number of seeds in $D_o(v)$, si_v is the number of seeds in $D_i(v)$ and p is the probability of a node influencing a node in its neighbourhood. Where the probability p_{vw} of a node v influencing a node w in its neighbourhood is dependent on the edge weight (v, w), the discounted degree is computed as:

$$\prod_{u_i \in D_i(v) \cap S} (1 - p_{vu_i}) \cdot (1 + \sum_{u_i \in D_o(v) \setminus S} p_{vu_i})$$

While the Single and Degree Discount heuristics only account for the direct influence of nodes in the seed set, Greedy Selection [Kempe et al. 2003] also accounts for the overlap in indirect influence. Greedy Selection, whether based on the Independent Cascade Model (ICM) or the Weighted Cascade Model (WCM), is computationally prohibitive. In both cases the expected value of influence spread is computed using Monte Carlo methods. Chen et al. [Chen et al. 2009] showed that Degree Discount actually provides a good approximation of the results obtained by Greedy Selection. To measure the influence spread of the seed set chosen by the degree based algorithms above, we execute the Random Cascade algorithm (either Weighted or Independent) to measure the influence spread of the seed set.

Unlike the standard usage of influence spread algorithms in social network analysis, in our case we are interested in choosing a set of seeds for each of the items in I_t. Given our desire to maximise the expected value of influence spread, rather than ranking nodes within the influence network solely based on their network value, we rank nodes by the product of their probability of uptake and network value.

3.3 The Algorithm

Our market-based approach for seeding recommendations is embodied by the MarketItem algorithm (Algorithm 1).

Algorithm 1 The MarketItem algorithm.

```
pu: uptake probability distribution
while I^t ≠ ∅ do
  Î^t: sorted list of i_k ∈ I^t in descending order of b_k^t × ru_k^t
  for k = 1 to |Î^t| do
    n_k^t ← B_k^t / b_k^t
    Ŝ_k^t ← SelectSeeds(G^t, i_k)
    S_k^t ← SelectSeeds(G^t, i_k, pu)
    m_k^t ← I(S_k^i) / I(Ŝ_k^t) × b_k^t
  end for
  for each u_a ∈ U do
    C_k^t ← selectItemsToRate(L_a^t, pu)
    for each i_k ∈ C_k^t do
      r_ak ← rateItem(i_k, u_a)
      B_k^t ← B_k^t − m_k^t
    end for
  end for
  for each i_k ∈ I^t do
    ru_k^t ← updateRateOfUptake(L^t, C_k^t)
    if B_k^t < b_k^t or Û_k^t = U_k^t then
      remove i_k from I^t
    end if
  end for
  t ← t + 1
end while
```

The approach allocates items to users' EPR lists while there are new items for which the budget exceeds the bid (i.e. $B_k^t \geq b_k^t$).

The first step is to sort the items into descending order according to the product of bid b_k^t and rate of uptake ru_k^t. Then, for each item in turn we select a set of users S_k^t to whose EPR lists the item will be added. The number of users is determined by the bid and the remaining budget, such that $|S_k^t| = B_k^t/b_k^t$. As described above, the heuristics for determining this set of users are based on the influence of each user for the item, along with the likely uptake by that user ($pu(l_{ak}^t)$), which in turn is determined by the position the item would take on the user's EPR list. The *SelectSeeds* function populates the EPR list L_a^t of each user u_a in the seed set S_k^t. (Note that we do not consider users that have already rated the item since its introduction into the recommendation system, and this filtering is incorporated into *SelectSeeds*.) In this way, since items are sorted by bid and rate of uptake, the items with a high bid and rate of uptake will be considered earlier by the algorithm, and so have an earlier opportunity to be allocated to the EPR lists of influential users. We also calculate the optimal seed set $\hat{S}_k^t$ that would be chosen if the EPR lists were empty, i.e. the new item was allocated to the most influential users. The payment m_k^t given to users who rate the item is the product of the bid and the proportion of the expected influence achieved by the actual set compared to that of the optimal set, i.e. $m_k^t = \mathcal{I}(S_k^i)/\mathcal{I}(\hat{S}_k^t) \times b_k^t$.

Once all new items have been allocated to the users' EPR lists, each user then selects items to rate, and for each selected item provides a rating. When an item is rated by a user they receive a payment m_k^t, and this amount is deducted from the remaining budget for the item. Finally, based on the ratings that have occurred in this round, the rate of uptake for each item is re-calculated and any items whose budget is spent or that have been rated by all users are removed from the new item set.

4. EXPERIMENTAL SETUP

4.1 Simulation

To evaluate the MarketItem algorithm we ran simulations using the MovieLens 100K data set [Herlocker et al. 1999]. Twenty six items (the items with more than 350 ratings) were picked as surrogates for new items, examples of which are shown in Table 2. Item bids were assigned at random to the new item surrogates and a fixed budget was assigned to all these items. The budget was selected to ensure that there are sufficient ratings in the original MovieLens data to provide ratings for the surrogate new items. In this case, the budget was set to £78 since the lowest bid item had a bid of 0.2 and 390 ratings ($78 = 390 \times 0.2$). The graph G_t was learnt from the training ratings within the rating matrix, which consisted of all ratings within the rating matrix other than those belonging to the new item surrogates.

The simulation ran over a number of cycles (until no further paid ratings were obtained), with each cycle being considered to represent a notional "day". All new items are added into the simulation at the start, i.e. on "day 0". In each cycle t, each new item k with remaining budget was assigned to the EPR lists of a set of users S_k^t, with $|S_k^t| = B_k^t/b_k^t$. Each new item was limited to be assigned only to users that have rated it in the MovieLens data (but withheld from the training data) but have not rated it within a previous simulation cycle[3]. Each user u_a picked m items from their EPR list L_a^t. We modelled m as a random variable that follows the binomial distribution $B(p, k)$, where p is the probability of an item on the list being picked, estimated from the data as the number of ratings per day by the user, and k is the maximum number of items that a user may pick in one cycle. The probability of a user picking an item off their EPR list reduced exponentially with the rank of the item on the list[4], i.e., $pu(l_{ak}^t) = \alpha\lambda_p e^{-\lambda_p l_{ak}^t}$, where α is the normalisation constant defined as $\alpha = 1/\sum_{i_k \in L_a^t} \lambda_p e^{-\lambda_p l_{ak}^t}$ and λ_p is a constant that determines how quickly the probability decays with rank position in the EPR list. Ratings assigned to the new items picked by the user were the rating of that item by the user in the full MovieLens rating matrix.

To conduct our simulations, we need a mechanism for simulating different market conditions. We do this by defining a probability distribution on the simulation cycles that determines the probability of a cycle occurring. We used an exponential distribution where the probability of a cycle occurring is $p(sc_i) = \beta\lambda_r e^{-\lambda_r sc_i}$, where sc_i is the ith simulation cycle, λ_r is a constant that defines how steeply the probability reduces, and β is the normalisation constant. A

[3]Note that collaborative filtering could have been used to predict the ratings for any user chosen to rate an item, however we chose not to do so as it would add noise to the evaluation and it would not be easy to infer whether any difference in accuracy of the methods evaluated in the next section was a result of errors in rating prediction or an artefact of the method itself.

[4]This is in line with previous work in recommender systems where metrics such as half-life utility score have been proposed on a similar assumption [Breese et al. 1998].

Item Id	Title	Num. Ratings	Avg. Rating	Std. Dev.	Bid
1	Toy Story	452	3.88	0.93	0.9
7	Twelve Monkeys	392	3.80	0.98	0.5
50	Star Wars	583	4.36	0.88	1.5
...					

Table 2: Example surrogate new items.

large value for λ_r simulates a high rate of new items being introduced into the recommender system, resulting in a lower likelihood of later simulation cycles materialising.

Using this method we model the case where our original set of 26 new items are "replaced" by other new items, meaning that the original items appear sufficiently low on users' EPR lists that there is little likelihood of them being selected. Thus a high value of λ_r causes the system to be "flooded" by new items early on in the simulation, in the sense that although those of our original new items with high bids will appear high on users' EPR lists on "day 0", they will rapidly drop down the lists. Items are placed in the EPR lists according to a combination of the user's influence, the bid, and the rate of uptake. When introduced, new items receive a rate of uptake of 1, and in subsequent cycles the rate of uptake for an item is determined by the number of users that choose to rate that item (as defined in Equation 1). Consequently, previously introduced high bid items with remaining budget will appear below new items with equal bids in a given user's EPR list (assuming equal influence for the user). Conversely, lower values of λ_r mean that items are introduced slowly, and there is an increased probability of later simulation cycles occurring (i.e. our original new items still featuring in prominent positions on users' EPR lists), and in the extreme case this reduces the effect of the bid value, since all items will eventually appear in high positions on influential users' EPR lists.

4.2 Evaluation

As a baseline algorithm, we assigned new items within the simulation to a random set of users, rather than using any influence based selection. This algorithm is referred to in the Results section as Random. One of the objectives of evaluation is to show that the use of influence based selection mechanisms outperform random selection. The other objectives of the evaluation include: (i) evaluating the different methods for inferring an influence network from a rating matrix, (ii) evaluating the effectiveness of various influence spread algorithms in selecting users for rating new items, (iii) evaluating the effect of neighbourhood size on the influence spread algorithms, and (iv) evaluating the effectiveness of the MarketItem algorithm to seed new items into a recommender system in the presence of different "market conditions", as defined by the rate at which new items are introduced into the recommender system.

To measure the effectiveness of the the MarketItem algorithm, we use the following metrics:

User Coverage (UC): The average percentage of users for whom a rating can be predicted for the new items. This value will be 0 at the start of the simulation as the rating matrix contains no ratings for the new items. With each cycle of the simulation, as ratings are inserted into the rating matrix, the coverage would be expected to increase. We would expect the MarketItem algorithm to result in the coverage of items to be positively correlated to the bid values of the items within the first few cycles, and for the correlation to drop as more cycles of the simulation occur.

Graph	λ_r	Num Ratings		Coverage		MAE	
		E[]	Corr	E[]	Corr	E[]	Corr
G2	0.2	72.7	−0.8	0.61	0.37	0.53	0.12
	0.4	60.79	−0.83	0.58	0.53	0.56	−0.15
	0.6	52.72	−0.85	0.55	0.61	0.59	−0.28
	0.8	47.35	−0.84	0.52	0.65	0.6	−0.33
	1	43.6	−0.82	0.5	0.67	0.61	−0.35
G3|G2	0.2	72.29	−0.80	0.62	0.42	0.52	0.11
	0.4	60.57	−0.83	0.58	0.58	0.56	−0.16
	0.6	52.62	−0.83	0.55	0.66	0.58	−0.27
	0.8	47.29	−0.81	0.52	0.7	0.6	−0.31
	1	43.58	−0.78	0.5	0.72	0.61	−0.33
G4|G2	0.2	73.32	−0.78	0.61	0.41	0.52	0.15
	0.4	60.68	−0.8	0.57	0.57	0.56	−0.08
	0.6	52.57	−0.8	0.54	0.64	0.58	−0.2
	0.8	47.19	−0.78	0.52	0.66	0.59	−0.26
	1	43.42	−0.74	0.5	0.68	0.6	−0.29

Table 3: Effect of the Influence Network Inference methods and λ_r on Expected Number of Ratings, UC and MAE with SingleDiscount and 20 neighbours

Recommendation List Coverage (RLC): The percentage of recommendation lists (assuming some size of neighbourhood and recommendation list, we used the values of 5 and 50, respectively, in our experiments) within which the item appears. Once again we would expect RLC to behave in a similar fashion to UC.

Mean Absolute Error (MAE): We treat the complete rating matrix provided by the MovieLens data as the "gold standard" and compare the item ratings predicted for users with a null rating in the matrix for the new items using the full rating matrix with those predicted using the ratings that have been "seeded" by the MarketItem algorithm. MAE can be defined as:

$$MAE = \frac{1}{|R|} \sum_{r_{ak} \in R} |\hat{r}_{ak} - r'_{ak}|$$

where R is the set of ratings predicted by the full rating matrix for the new items, $\hat{r}_{ak}$ is the rating predicted by the full rating matrix and r'_{ak} is the rating predicted by the MarketItem based rating matrix. If the MarketItem algorithm rating matrix is unable to predict a rating r_{ak} it uses the average rating for u_a. We would expect the MAE to reduce as the simulation cycles progress and for it to be lower when using influence based seed selection approach as compared to the random approach.

To evaluate the MarketItem algorithm we have simulated a variety of market conditions, by varying λ_p and λ_r, for the alternative algorithms for seed set selection, and have calculated the expected value for each of the three metrics defined above.

5. RESULTS

Table 3 shows the results of running MarketItem on the influence network resulting from each of the three methods

Neighbrhd Size	λ_r	Num Ratings		Coverage		MAE	
		E[]	Corr	E[]	Corr	E[]	Corr
5	0.2	71.72	−0.79	0.36	0.68	0.64	0.02
	0.4	59.39	−0.81	0.31	0.8	0.67	−0.06
	0.6	51.21	−0.8	0.27	0.83	0.68	−0.1
	0.8	45.8	−0.76	0.25	0.85	0.696	−0.11
	1	42.05	−0.7	0.23	0.85	0.7	−0.11
20	0.2	72.7	−0.8	0.61	0.37	0.53	0.12
	0.4	60.79	−0.83	0.58	0.53	0.56	−0.15
	0.6	52.72	−0.85	0.55	0.61	0.59	−0.28
	0.8	47.35	−0.84	0.52	0.65	0.6	−0.33
	1	43.6	−0.82	0.5	0.67	0.61	−0.35
40	0.2	73.47	−0.8	0.72	0.19	0.44	0.25
	0.4	61.11	−0.82	0.69	0.27	0.49	−0.02
	0.6	53.06	−0.82	0.67	0.34	0.51	−0.18
	0.8	47.72	−0.79	0.66	0.39	0.53	−0.26
	1	43.98	−0.75	0.64	0.43	0.55	−0.31

Table 4: Effect of Neighbourhood Size using Single Discount (λ_p = 0.9 and Influence Network, G2)

Algorithm	λ_r	RLC		Coverage		MAE	
		E[]	Corr	E[]	Corr	E[]	Corr
Random	0.2	0.198	0.16	0.27	−0.76	0.69	0.29
	0.4	0.191	0.15	0.25	−0.69	0.7	0.21
	0.6	0.184	0.15	0.23	−0.58	0.71	0.17
	0.8	0.178	0.15	0.21	−0.47	0.72	0.14
	1	0.174	0.16	0.2	−0.38	0.72	0.12
Degree	0.2	0.209	0.22	0.41	0.57	0.59	0.03
	0.4	0.206	0.22	0.36	0.71	0.63	−0.12
	0.6	0.203	0.23	0.32	0.76	0.65	−0.18
	0.8	0.2	0.24	0.29	0.77	0.66	−0.2
	1	0.197	0.25	0.27	0.78	0.67	−0.21
Single Discount	0.2	0.2	0.24	0.36	0.68	0.64	0.02
	0.4	0.194	0.29	0.31	0.8	0.67	−0.06
	0.6	0.188	0.33	0.27	0.83	0.68	−0.1
	0.8	0.184	0.37	0.25	0.85	0.696	−0.11
	1	0.18	0.39	0.23	0.85	0.7	−0.11
Degree Discount	0.2	0.231	0.16	0.43	0.55	0.6	0.19
	0.4	0.219	0.17	0.37	0.7	0.63	0.04
	0.6	0.209	0.18	0.32	0.74	0.65	−0.01
	0.8	0.201	0.19	0.3	0.75	0.67	−0.04
	1	0.195	0.20	0.27	0.76	0.67	−0.04

Table 5: Effect of the Influence Spread algorithms and λ_r on RLC, UC and MAE using a neighbourhood size of 5

in Section 3.1 for a fixed neighbourhood size. For each metric, the expected value (E[]) and its correlation with the new item bid values is shown. As can been seen from the results, the difference is very small, suggesting that the effectiveness of MarketItem is independent of the method used for inferring the influence network.

Table 4 shows the effect of neighbourhood size on the coverage and MAE when using the Single Discount algorithm. The results show that coverage increases and MAE reduces as neighbourhood size increases. It is also interesting to note that while the number of ratings is unaffected by neighbourhood size (which is expected since the number of ratings is only a function of item bid and budget and not the influence network topology), the correlation between the bid and coverage reduces as the neighbourhood size increases. This is because each rating that is inserted into the rating matrix by MarketItem, has a wider effect on user neighbourhoods. Similar patterns are observed for the other influence spread algorithms, but are omitted here due to limited space.

Table 5 shows the efficacy of the various influence spread algorithms for a neighbourhood size of 5. As can be seen, all three influence spread algorithms show an improved coverage over a Random assignment of seeds. For all three algorithms the improvement over Random is between 33% and 35% depending on the value of λ_r. MAE also improves by between 7% and 14.5%. Of the three influence spread algorithms Degree and Degree Discount appear to be more effective for the given topology and Degree seems to be more faithful to the bid values in that there is a negative correlation between the bid value and the MAE. Interestingly, when the neighbourhood size increases, DegreeDiscount shows better coverage than Degree (an improvement of 13.3% for neighbourhood size 20) and SingleDiscount (an improvement of 8.8% for neighbourhood size 20).

Table 6 shows the effect of the λ_r and λ_p parameters on user coverage, number of ratings and MAE metrics when using SingleDiscount for assigning items to influential users. As λ_p increases there is a strong bias, within the simulation, towards the user picking items at the top of their EPR list. As these items are by design items with high bid values, the correlation of the coverage with item bid is strongly positive and the strength of this correlation increases with λ_p. At the same time the coverage itself drops as λ_p increases. This is also to be expected as the lower bid items will ultimately get more ratings (since the budget goes further) and so are likely to get high coverage, assuming that all cycles of the simulation are completed and the items are not pushed down the EPR lists by new items with high bids. The data supports this, and the correlation of coverage with bid falls substantially for low λ_r values, i.e. the bid is less relevant if there are few new items per cycle, since items persist on the EPR lists and so are likely to be picked eventually.

When λ_p is low, then low bid items are more likely to get picked in early cycles of the simulation and so coverage will be higher, even for large λ_r values. The difference in coverage also widens as λ_r reduces as more influential users select more low bid items. MAE increases by a very small amount as λ_p increases but the effect of λ_r is more pronounced, where MAE reduces as λ_r reduces, i.e. when the simulation runs for longer with items introduced gradually. Once again this is expected as in later cycles of the simulation more users rate the items with lower bids and the error converges to that observed when using the complete rating matrix to predict null ratings for the new item surrogates. Note that as λ_r drops, the correlation of MAE with item bid goes from being slightly negatively correlated to being weakly positively correlated suggesting that the error is more a function of the number of ratings than of the coverage. Finally, the number of ratings increases as λ_r reduces which would be expected as more cycles of the simulation are completed for low λ_r values.

6. CONCLUSIONS AND FUTURE WORK

In this paper we have presented a new approach to solving the new item problem. The proposed approach is imple-

λ_p	λ_r	Num Ratings		Coverage		MAE	
		E	Corr	E	Corr	E	Corr
1.2	0.2	72.4	−0.801	0.357	0.662	0.651	0.078
	0.4	60.22	−0.799	0.305	0.806	0.674	−0.017
	0.6	52.07	−0.73	0.269	0.845	0.689	−0.05
	0.8	46.66	−0.601	0.244	0.861	0.698	−0.057
	1	42.9	−0.451	0.226	0.869	0.704	−0.055
0.9	0.2	71.72	−0.789	0.365	0.683	0.644	0.025
	0.4	59.39	−0.809	0.311	0.801	0.67	−0.066
	0.6	51.21	−0.8	0.275	0.835	0.686	−0.101
	0.8	45.8	−0.762	0.25	0.849	0.696	−0.112
	1	42.05	−0.703	0.232	0.856	0.702	−0.113
0.6	0.2	71.45	−0.752	0.381	0.611	0.634	0.03
	0.4	58.79	−0.756	0.323	0.772	0.665	−0.073
	0.6	50.58	−0.748	0.284	0.821	0.684	−0.107
	0.8	45.17	−0.734	0.259	0.841	0.695	−0.116
	1	41.4	−0.716	0.24	0.851	0.703	−0.115
0.3	0.2	67.67	−0.697	0.409	0.376	0.618	0.105
	0.4	54.98	−0.673	0.353	0.623	0.655	−0.004
	0.6	47.02	−0.65	0.315	0.714	0.677	−0.043
	0.8	41.85	−0.632	0.289	0.748	0.69	−0.056
	1	38.29	−0.616	0.27	0.762	0.698	−0.06

Table 6: Effect of λ_r and λ_p on Expected Number of Ratings, UC and MAE using SingleDiscount with neighbourhood size 5

mented as the MarketItem algorithm and evaluated using the MovieLens 100K data set. MarketItem is based on the principle that it is in the interest of the producers, consumers and retailer to reduce the item latency, and that the producer of the item would provide money to users to rate new items. MarketItem uses influence spread algorithms from social network analysis and adapts them to the goal of picking the optimal lists of items to present to users that optimise the coverage of new items within the recommender system along with the expected earnings of users by rating these new items. The MarketItem algorithm was evaluated using a simulation that clearly shows the effectiveness of our adaptation of the DegreeDiscount heuristic in choosing users that would be best approached to rate new items.

In the future we aim to apply the MarketItem algorithm to additional recommender data sets. We further aim to incorporate item content within the MarketItem algorithm when selecting a set of users as paid raters. We also aim to analyse the effect of rating new items on the users influence on the recommender system. We will aim to quantify the harm that can be caused by users that rate new items randomly, or through other strategies to increase their revenue through new item rating.

7. REFERENCES

[Anagnostopoulos et al. 2008] Anagnostopoulos, A., Kumar, R., and Mahdian, M. 2008. Influence and correlation in social networks. In *Proc. of the 14th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*. 7–15.

[Breese et al. 1998] Breese, J. S., Heckerman, D., and Kadie, C. 1998. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence*. 43–52.

[Chen et al. 2009] Chen, W., Wang, Y., and Yang, S. 2009. Efficient influence maximization in social networks. In *Proc. of the 15th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*. 199–208.

[Domingos and Richardson 2001] Domingos, P. and Richardson, M. 2001. Mining the network value of customers. In *Proc. of the 7th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*. 57–66.

[Goldenberg et al. 2001] Goldenberg, J., Libai, B., and Muller, E. 2001. Using complex systems analysis to advance marketing theory development. *Academy of Marketing Science Review 2001,* 9, 1–20.

[Herlocker et al. 1999] Herlocker, J., Konstan, J., Borchers, A., and Riedl, J. 1999. An algorithmic framework for performing collaborative filtering. In *Proc. of the 1999 Conf. on Research and Development in Information Retrieval*.

[Huang et al. 2004] Huang, Z., Chen, H., and Zeng, D. 2004. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Trans. Inf. Syst. 22,* 1, 116–142.

[Kempe et al. 2003] Kempe, D., Kleinberg, J., and Tardos, E. 2003. Maximizing the spread of influence through a social network. In *Proc. of the 9th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*. 137–146.

[Lam et al. 2008] Lam, X. N., Vu, T., Le, T. D., and Duong, A. D. 2008. Addressing cold-start problem in recommendation systems. In *Proc. of the 2nd Int. Conf. on Ubiquitous information management and communication*. 208–211.

[Lambert and Shoham 2008] Lambert, N. and Shoham, Y. 2008. Truthful surveys. In *Proceedings of the 4th International Workshop on Internet and Network Economics*. WINE '08. Springer-Verlag, Berlin, Heidelberg, 154–165.

[Mobasher et al. 2007] Mobasher, B., Burke, R., Bhaumik, R., and Williams, C. 2007. Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Trans. Internet Technol. 7*.

[Park and Chu 2009] Park, S.-T. and Chu, W. 2009. Pairwise preference regression for cold-start recommendation. In *Proc. of the 3rd ACM Conf. on Recommender systems*. 21–28.

[Park et al. 2006] Park, S.-T., Pennock, D., Madani, O., Good, N., and DeCoste, D. 2006. Naïve filterbots for robust cold-start recommendations. In *Proc. of the 12th ACM SIGKDD Int. Conf. on Knowledge discovery and data mining*. 699–705.

[Park and Tuzhilin 2008] Park, Y.-J. and Tuzhilin, A. 2008. The long tail of recommender systems and how to leverage it. In *Proc. of the 2008 ACM Conf. on Recommender systems*. 11–18.

[Schein et al. 2002] Schein, A. I., Popescul, A., Ungar, L. H., and Pennock, D. M. 2002. Methods and metrics for cold-start recommendations. In *Proc. of the 25th annual Int. ACM SIGIR Conf. on Research and development in information retrieval*. 253–260.