Avoiding Monotony: Improving the Diversity of Recommendation Lists

Mi Zhang
Information Hiding Laboratory,
School of Computer Science and Informatics,
University College Dublin, Ireland
& Department of Computer Science and Engineering, Fudan University, China
mi.zhang@ucd.ie

Neil Hurley
Information Hiding Laboratory,
School of Computer Science and Informatics,
University College Dublin, Ireland
neil.hurley@ucd.ie

ABSTRACT

The primary premise upon which *top-N* recommender systems operate is that *similar* users are likely to have *similar* tastes with regard to their product choices. For this reason, recommender algorithms depend deeply on similarity metrics to build the recommendation lists for end-users. However, it has been noted that the products offered on recommendation lists are often *too similar* to each other and attention has been paid towards the goal of improving *diversity* to avoid monotonous recommendations.

Noting that the retrieval of a set of items matching a user query is a common problem across many applications of information retrieval, we model the competing goals of maximizing the diversity of the retrieved list while maintaining adequate similarity to the user query as a binary optimization problem. We explore a solution strategy to this optimization problem by relaxing it to a trust-region problem. This leads to a parameterized eigenvalue problem whose solution is finally quantized to the required binary solution. We apply this approach to the *top-N* prediction problem, evaluate the system performance on the Movielens dataset and compare it with a standard item-based *top-N* algorithm. A new evaluation metric $ItemNovelty$ is proposed in this work. Improvements on both diversity and accuracy are obtained compared to the benchmark algorithm.

Categories and Subject Descriptors

h.3 [**INFORMATION STORAGE AND RETRIEVAL**]: Information Search and Retrieval; h.m [**MISCELLANEOUS**]

General Terms

Theory, Algorithms, Performance

Keywords

diversity, accuracy, novelty, recommender system, metrics

RecSys'08, October 23–25, 2008, Lausanne, Switzerland.

1. INTRODUCTION

Recommender systems have been very effective to help users find interesting and relevant items from a large information space. Most research has been driven by the goal of improving the accuracy of recommender systems and relatively little has focused on improving the utility of the recommendation list, considering other desirable system characteristics. The accuracy metrics typically used to measure system performance are largely directed at average performance over the entire user set, such as MAE [2], or precision and recall [4]. Predictions are deemed successful so long as the predicted set overlaps with the user's real preferences. However, these metrics do not account for other qualities of the prediction, in particular, they do not consider the *difficulty* of the prediction. It is reasonable to assume that a user will be more satisfied with a system that offers recommendations from a less obvious set of items that the user likes, than one that always offers items from a core set that the user has frequently used or purchased in the past. Users may become frustrated that there is little variety in the products offered by the system.

Take an academic paper recommender system as an example. Suppose all of the recommendations made were for papers written by a single author that the user has read in the past. Even if the user is strongly interested in these papers and the system is very good at ranking them in order of preference, it is a poor recommender system because it is easy for the user to find these papers by other means. In fact, the academic paper recommender system may be regarded as underperforming when evaluated by the standard metrics, if it offers less common topics that the user likes, instead of topics the user has already read and knows well.

We tackle this lack of variety issue head-on, by proposing methods for maximizing the *diversity* of a recommendation list, while maintaining an acceptable level of matching quality. In this paper, we show how these competing concerns can be presented as constrained binary optimization problems and propose solution strategies for these problems.

The **contributions** we make in this paper are as follows:

- **Diversity problem definition.** This paper formulates the diversity problem as the optimization of an objective function under certain constraints, in which the feasible solutions are binary vectors. The objective and constraints are either quadratic or linear expressions in the binary vector. This formulation is useful in that it opens the possibility for the application of

many solution strategies that have been developed for solving problems of this type in other contexts. In particular, we propose one approach based on solving a *trust region* relaxation of the binary problem. Note that our formulation can be applied not just to the *top-N* recommendation problem, but more generally to many problems in information retrieval, in which the goal is to match a sub-set of items to a user query. It is also noteworthy that our method seeks to find the best *sub-set* of items over all possible sub-sets. This is in contrast to other work on this problem, such as [17, 15], in which ad-hoc strategies have been applied to rank items for inclusion in the recommendation list.

- ***Top-N* recommendation with diversity.** We demonstrate how to apply our technique in the context of the *top-N* recommendation problem, using a general framework for *top-N* recommendation which encapsulates the well-known item-based algorithm implemented in the SUGGEST system [3]. We show that our technique succeeds in generating recommendation lists with different levels of diversity and matching quality, depending on the input parameters. Moreover, we show that the extra diversity introduced as a consequence of our method does have the potential to move the *novel* items that users like higher in the recommendation list.

- **New evaluation metrics.** We propose an objective measure of diversity which only requires the existence of a function to determine the *distance* or *dissimilarity* between two items. Moreover, we propose an objective measure of the *difficulty* of a recommendation task by proposing a *novelty* metric for items in a user profile. The novelty value of an item depends on the other items in the user profile, and the difficulty of the recommendation task increases with increasing item novelty. The novelty value is therefore used to generate test sets of different levels of difficulty upon which to measure system performance, using the standard precision metric.

- **Recommendation list utility.** Our objective is to improve the utility of user's recommendation list by applying diversification. From the results of experiments operating on the above metrics, a significant diversity improvement is obtained while maintaining satisfactory accuracy.

Notation

Sets are represented by uppercase letters such as C and R. Column vectors are represented by bold lowercase letters such as $\mathbf{x}$. The transpose of a column vector to a row vector is written as $\mathbf{x}^T$ and the dot product of two vectors $\mathbf{x}$ and $\mathbf{y}$ is $\mathbf{y}^T\mathbf{x}$. $\|\mathbf{x}\| \triangleq \sqrt{\mathbf{x}^T\mathbf{x}}$ is the norm of the vector $\mathbf{x}$. The column vector whose entries are all 1 is represented as $\mathbf{1}$.

2. RELATED WORK

The drawbacks of developing recommender systems with accuracy as the single goal have been articulated in recent research. For example, in [14], informal arguments that the recommender community should move beyond the conventional accuracy metrics and their associated experimental methodologies are made and new user-centric directions for evaluating recommender systems are proposed. One direction that has drawn recent interest is the evaluation of different system characteristics, such as the diversity of the system output. In fact, in the context of case-based reasoning systems, [15] argues that diversity can sometimes be as important as similarity to the target query. In this work, a definition of set diversity as *average dissimilarity*— a definition that we adopt in this paper —is introduced. The authors propose three heuristic algorithms for selecting retrieval sets that combine similarity and diversity. The best of these methods— *greedy selection* —adds cases one at a time to the retrieval set, according to a heuristic measure combining diversity and similarity. This is in contrast to our approach where the *entire* retrieval set is considered as a whole and an optimal set according to a well-defined objective function is found. Furthermore, [15] does not examine the impact that additional diversity has on retrieval performance. In Section 7.5, we examine this issue by explicitly evaluating the system's ability to recommend novel items.

In [17] the issue of diversification of recommendation lists is tackled. The authors propose a similarity metric using a taxonomy-based classification and use it to compute an *intra-list similarity metric* to determine the overall diversity of the recommended list. Intra-list similarity is analogous to Smyth's [15] notion of subset diversity, except that it is a decreasing rather than increasing function of diversity. The authors provide a heuristic algorithm to increase the diversity of the recommendation list. Their results show that lists ordered for greater diversity perform worse on accuracy measures than unaltered lists, but nevertheless users preferred the altered lists. In fact, depending on the user's intentions, the makeup of items appearing on the list affects the user's satisfaction with the recommender more than changes in the accuracy of the items on the list.

Another paper [6] examined the conditions in which diversity can be increased without loss of similarity, and presented an approach to deliver such similarity-preserving increases in diversity when possible. The role of diversity in conversational recommender systems is clarified in [5], highlighting the pitfalls of naively incorporation current diversity-enhancing techniques into existing recommender systems. Finally, [1] examines the effect of recommender systems on the diversity of sales. It uses a measure of statistical dispersion called the Gini coefficient to measure sales diversity.

3. DIVERSITY MEASURES

The general problem of finding a sub-set of items that best match a query or request issued by a system user is common across many applications of information retrieval. The items in question may be products offered for purchase in a recommender system, or documents returned by a search engine for instance.

Let $\mathcal{Q}$ be a set of possible queries which may be issued to the system and $\mathcal{I}$ the item-set. Denote a query issued by user u as $q_u \in \mathcal{Q}$ and let $R \subset \mathcal{I}$ be the sub-set returned by the system. We will refer to R as the *recommended list*. We can model the system algorithm as the application of a *matching function* $f_m : \mathcal{Q} \times 2^{\mathcal{I}} \rightarrow \mathbb{R}$, such that $f_m(q_u, R)$ is a real-valued matching value associated with sub-set R, where, without loss of generality, we assume that higher matching values correspond to better matches. The system may then return a sub-set R of some fixed size which maximizes

$f_m(q_u, R)$, or the smallest sub-set R whose matching value is above a given threshold.

In this paper, we consider matching functions where the matching value of a sub-set is formed as the *aggregate* of matching values of items contained in the sub-set. That is, there exists a function $g_m : \mathcal{Q} \times \mathcal{I} \rightarrow \mathbb{R}$, such that

$$f_m(q_u, R) = \sum_{i \in R} g_m(q_u, i)\,.$$

Generally, the query q_u is an inaccurate or coarse representation of the user's true requirements (e.g. a set of keywords to describe a particular type of document that the user wishes to retrieve). The matching function is a heuristic to capture the user's subjective opinion on how well an item agrees with the requirements and is unlikely to be fully accurate. So the best matched items according to g_m do not correspond necessarily to the most desirable items from the user's point of view. For this reason, many practical systems are interactive, allowing the user to scroll through a set of choices to make a decision. However, in recent research, it has been observed that, because all items in R are well matched to the same query q_u, it is likely that they are also very similar *to each other*. This can lead to an unsatisfactory system from the user's point of view, since the choice offered by the system has limited variety and motivates the goal of adding greater choice or *diversity* to the set R. The argument that suggests that the standard approach leads to recommended lists with low diversity also suggests that lists with greater diversity will also have greater variance in the matching values of the items contained in them. Clearly a system that offers poorly matched items, with low $g_m(q_u, i)$ value, is unlikely to produce satisfactory results. So, the joint goals of offering a set R of high diversity and offering a set R of high matching value, stand in opposition to each other. In the following, we will present objective functions that capture the trade-offs between these goals and show how the maximization of these objective functions can be represented as a binary quadratic programming problem.

3.1 Diversity of a Set

A number of different definitions of set diversity have been proposed in the literature. One approach is to model diversity as the average *rarity* of the elements in the set [10]. Another, which has been discussed in other application contexts, (e.g. [8, 11]) and which has been used in a similar context to ours in [15], is to model diversity as the *aggregate* or equivalently *average dissimilarity* of all pairs of items in the set. Specifically, given a *distance* function, $d : \mathcal{I} \times \mathcal{I} \rightarrow \mathbb{R}$, such that $d(i, j)$ is the distance or dissimilarity between elements $i, j \in \mathcal{I}$, the diversity $f_D(R)$ is given as the average dissimilarity of all pairs of elements contained in R. That is,

$$f_D(R) = \frac{2}{p(p-1)} \sum_{i \in R} \sum_{j \neq i \in R} d(i, j)\,,$$

where $p = |R|$. Here we assume that this distance function is symmetric ($d(i, j) = d(j, i)$).

The distance function is application-dependent and may correspond for example to a distance between feature vectors under a mapping of $\mathcal{I}$ into a feature space. Often, the matching function $g_m(q_u, i)$ is a decreasing function of some distance metric and, if the same metric is used to measure dissimilarity, then the opposition of the competing requirements of diversity and high matching are immediately clear. However, in the following, all that is required is that that item-item dissimilarities and query matching values can be calculated, without restriction on the feature space or method used to calculate these values.

4. QUADRATIC OBJECTIVE FUNCTIONS

Let $C \subseteq \mathcal{I}$ be a set of size $|C| = M$. Let the elements of C be listed as $C = \{c_1, \ldots, c_M\}$. It is useful to represent subsets $R \subseteq C$ using an M-dimensional indicator vector $\mathbf{y}$, such that $y(i) = 1$ if $c_i \in R$ and $y(i) = 0$ otherwise. Moreover, let D be the $M \times M$ distance matrix with (i, j)-element $d(c_i, c_j)$. Then, the diversity of R may be written as the *quadratic form*[1]

$$f_D(\mathbf{y}) = \frac{1}{p(p-1)} \mathbf{y}^T \mathrm{D} \mathbf{y}\,. \tag{1}$$

Moreover the matching function may be written as the linear expression

$$f_m(\mathbf{y}) = \mathbf{m}_u^T \mathbf{y}\,,$$

where $\mathbf{m}_u$ is the M-dimensional vector with elements $m(i) = g_m(q_u, c_i)$.

With these expressions, it is possible to express the trade-offs between diversity and matching as constrained optimization problems.

Problem 1

Find the vector $\mathbf{y}^*$ defined as

$$\begin{aligned} \mathbf{y}^* &= \arg\max \mathbf{y}^T \mathrm{D} \mathbf{y} && (2)\\ s.t. \quad & \mathbf{m}_u^T \mathbf{y} \geq m_{\mathrm{tol}}\,, && (3)\\ & \mathbf{1}^T \mathbf{y} = p\,, && (4)\\ & y(i) \in \{0, 1\} \forall i = 1, \ldots, M\,. && (5) \end{aligned}$$

In this optimization problem, we seek to maximize the diversity of the set R, under the constraint that the matching value of R be greater than some *matching tolerance* m_{tol}. The final two constraints specify that $\mathbf{y}$ is a binary vector with p non-zero values i.e. $|R| = p$.

Problem 2

Find the vector $\mathbf{y}^*$ defined as

$$\begin{aligned} \mathbf{y}^* &= \arg\max \mathbf{m}_u^T \mathbf{y} && (6)\\ s.t. \quad & \mathbf{y}^T \mathrm{D} \mathbf{y} \geq d_{\mathrm{tol}}\,, \\ & \mathbf{1}^T \mathbf{y} = p\,, \\ & y(i) \in \{0, 1\} \forall i = 1, \ldots, M\,. \end{aligned}$$

In this optimization problem, we seek to maximize the matching value of the set R, under the constraint that the diversity of R be greater than some *diversity tolerance* d_{tol}. Again, the final two constraints specify that $\mathbf{y}$ is a binary vector with p non-zero values i.e. $|R| = p$.

[1]Note that this equivalence depends on $d(c_i, c_i) = 0\,\forall i$. In fact nothing changes in the analysis, as long as $d(c_i, c_i)$ is constant $\forall i$.

Problem 3

Given a fixed parameter $\theta \in [0,1]$, find the vector $\mathbf{y}^*$ defined as

$$\begin{aligned} \mathbf{y}^* &= \arg\max (1-\theta)\alpha \mathbf{y}^T \mathrm{D}\mathbf{y} + \theta\beta \mathbf{m}_u^T \mathbf{y} \qquad (7) \\ s.t. \quad & \mathbf{1}^T\mathbf{y} = p\,, \\ & y(i) \in \{0,1\}\ \forall i = 1,\ldots,M\,. \end{aligned}$$

Here, the trade-off between the two objectives is explicitly expressed using the parameter θ that represents the importance given to the matching value in comparison to that given to diversity. α and β are normalization parameters to ensure that diversity and matching measures are normalized to the same scale, for example, that both lie in the range $[0,1]$.

5. SOLUTION STRATEGIES

Problems 1 and 3 above correspond to binary quadratic programming problems with linear constraints. Problem 2 is a binary linear programming problem with quadratic and linear constraints. There is a vast literature on solution strategies for such problems [16]. A general strategy is to search through a space of solutions, using for example branch and bound search, where the solutions are generated by solving *relaxations* of the binary problem to problems with real-valued solutions. Writing $\mathbf{x} \in \mathbb{R}^M$ as the solution to the real-valued problem, a quantization strategy must then be applied to map this solution to a binary vector $\mathbf{y}$. In the following, we will focus mainly on Problem 3, where the objective represents an explicit trade-off between our two concerns. Additionally, we will pursue a solution to Problem 1, through a two-stage process of initially identifying a candidate set C in which all sub-sets of size p satisfy (3) and then pursuing solutions to the quadratic program without constraint (3), which is equivalent to solving Problem 3 with $\theta = 0.0$.

5.1 Quadratic Relaxations

A key step in the solution of a binary quadratic programming problem is the choice of relaxation to a real-valued problem. Indeed, the real-valued problem may itself be a hard problem. While polynomial time solutions exist when the Hessian matrix, D, is positive semi-definite (i.e. has no negative eigenvalues), otherwise the real-valued quadratic program is NP-complete. A common relaxation of (2) is *spectral relaxation*, where the relaxed problem is

$$\begin{aligned} \mathbf{x}^* &= \arg\sup \mathbf{x}^T \mathrm{D}\mathbf{x} \qquad (8) \\ s.t. \quad & \|\mathbf{x}\|^2 = p \quad \mathbf{x} \in \mathbb{R}^M \qquad (9) \end{aligned}$$

and is solved by finding the eigenvector corresponding to the maximum eigenvalue of D. It is clear that any feasible solution $\mathbf{y}$ of (2) is a feasible solution of (8), so that $\mathbf{x}^*\mathrm{D}\mathbf{x}^*$ is an upper bound on the maximum value of (2). Given that fast eigensolvers for large-scale systems exist, this approach has the advantage of speed. However, in many cases (see [9], for example) the solution $\mathbf{x}^*$ is far from the best binary solution $\mathbf{y}^*$ and quantization yields a poor binary solution.

Spectral relaxation can be equally applied to Problem 3, through a standard modification of the problem. Extending $\mathbf{y}$ to an $M+1$ dimensional vector, we can write the objective (7) as the quadratic form

$$(\mathbf{y}^T, y(M+1)) \begin{pmatrix} (1-\theta)\alpha\mathrm{D} & \frac{\theta\beta}{2}\mathbf{m}_u \\ \frac{\theta\beta}{2}\mathbf{m}_u^T & 0 \end{pmatrix} (\mathbf{y}^T, y(M+1))^T\,, \qquad (10)$$

with the additional constraint $y(M+1) = 1$. However, the corresponding spectrally relaxed problem imposes no constraint on $x(M+1)$. Hence, if D is a negative definite matrix (e.g. $\mathrm{D} \triangleq -\mathrm{S}$ for some positive definite similiarity matrix, S), then, in the limit as $\theta \to 0$, the best relaxed solution is $\mathbf{x}^* = (0,\ldots,0,\sqrt{p})^T$, which clearly cannot be converted to a good binary solution of (10).

It is noteworthy that spectral relaxation is essentially the basis underlying the method in [7], where spam users are detected through a principal components analysis (PCA) of the similarity matrix. Here, a subset of *maximally similar* users is extracted using a user-user similarity matrix, while in our context, we are locating *maximally dissimilar* items using an item-item dissimilarity matrix.

At the other end of the scale from spectral relaxation, *semi-definite programming* relaxations can yield higher quality solutions but can be inefficient. An alternative relaxation to (7) and the one which we use for our application is

$$\begin{aligned} \mathbf{x}^* &= \arg\sup (1-\theta)\alpha \mathbf{x}^T \mathrm{D}\mathbf{x} + \theta\beta \mathbf{m}_u^T \mathbf{x} \qquad (11) \\ s.t. \quad & \|\mathbf{x}\|^2 = p \quad \mathbf{x} \in \mathbb{R}^M\,, \end{aligned}$$

for which a fast solution is presented in [12]. Note that in [12], the constraint is actually $\|\mathbf{x}\|^2 \leq p$, but adding $k(p - \mathbf{x}^T\mathbf{x})$ to (11) for a sufficiently large penalty k, will yield a solution with $\|\mathbf{x}\|^2 = p$. A code for solving this problem, called LSTRS, is described in [13]. The method is based on a reformulation of a *trust-region* sub-problem as a parameterized eigenvalue problem:

$$B_\delta = \begin{pmatrix} (1-\theta)\alpha\mathrm{D} & \frac{\theta\beta}{2}\mathbf{m}_u \\ \frac{\theta\beta}{2}\mathbf{m}_u^T & \delta \end{pmatrix} \qquad (12)$$

with δ a scalar parameter. LSTRS relies on matrix-vector products only and has low and fixed storage requirements, features that make it suitable for large-scale computations.

5.2 Quantization Strategies

Once the real-valued solution $\mathbf{x}^*$ to the relaxed problem is found, it is necessary to quantize the values to a candidate binary solution $\mathbf{y}$. One approach is to consider the dot product $\mathbf{y}^T\mathbf{x}^*$. Indeed, writing $\lambda_{\max}$ (resp. $\lambda_{\min}$) for the maximum (resp. minimum) eigenvalue of D, if $\mathbf{x}^*$ is a solution to the spectrally relaxed problem and we write $\mathbf{e} = \mathbf{y} - \mathbf{x}^*$, it is easy to check that

$$\mathbf{y}^T\mathrm{D}\mathbf{y} = -p\lambda_{\max} + 2\lambda_{\max}\mathbf{y}^T\mathbf{x} + \mathbf{e}^T\mathrm{D}\mathbf{e}\,.$$

Next, using the fact that $\mathbf{e}^T\mathrm{D}\mathbf{e} \geq \lambda_{\min}\mathbf{e}^T\mathbf{e}$ for all vectors $\mathbf{e}$, we have that

$$\mathbf{y}^T\mathrm{D}\mathbf{y} \geq p(2\lambda_{\min} - \lambda_{\max}) + 2\mathbf{y}^T\mathbf{x}(\lambda_{\max} - \lambda_{\min})\,.$$

Thus maximizing $\mathbf{y}^T\mathbf{x}^*$ maximizes a lower bound on the quadratic form. Since $\mathbf{y}$ is binary and noting that $-\mathbf{x}^*$ is also a valid solution, the dot product is maximized by setting $y(i) = 1$, wherever $x(i)$ is one of the p highest elements of $+\mathbf{x}$ or $-\mathbf{x}$, depending on which gives the larger value. While relaxation and quantization can form just one step in a larger branch-and-bound search, for efficiency reasons in the results that follow, we do no further searching once a feasible binary solution is generated.

1. A *candidate set* $C \subseteq \mathcal{I}$ of potential items to recommend to u is generated such that $C \cap P_u = \emptyset$. Let $l = |C|$ and list the items in C as $C = \{c_1, \ldots, c_l\}$.
2. The *candidate set* C is sorted according to a *sorting function* which results in a permutation π such that $\pi(j) = j^{\text{th}}$ most highly ranked item in C. A general strategy that may be used to sort C is as follows:
   (a) An *intermediate set* $I \subseteq C$ of "*best*" items in C is formed using some criterion.
   (b) The ranking is chosen such that all items in I are ranked more highly than items in $C - I$.
3. The *recommendation set*, R, is formed from the N most highly ranked items, $R = \{c_{\pi(1)} \ldots c_{\pi(N)}\}$

Figure 1: Framework for *Top-N* Prediction

6. TOP-N RECOMMENDATION

We apply our diversity analysis to top-N recommendation. Given a past history of purchases made by a set of users, the problem of top-N recommendation is to find a subset $\mathrm{R} \subseteq \mathcal{I}$ of N products to recommend for purchase to a given user u. Let $\mathcal{U} = \{u_1, \ldots, u_P\}$ be the set of possible system users and $\mathcal{I} = \{i_1, \ldots, i_Q\}$ the set of items or products available for purchase, with $P = |\mathcal{U}|$ the total population of system users and $Q = |\mathcal{I}|$, the total number of products. For each user u, the *user profile* $P_u \subseteq \mathcal{I}$ is the set of items purchased by u.

The framework we use for computing a top-N recommendation list for a user u is given in Figure 1. These steps encapsulate the SUGGEST recommendation engine [3]. In the case of SUGGEST an *item-item* similarity matrix is learned during a training phase. Using an input parameter K, C is formed as the intersection of the sets of K most similar items to each item in the user profile. The sorting function then computes the similarity of each candidate item to *all* items in the profile and the N best items are recommended. The general framework allows the introduction of diversity as an additional criterion for selecting and ranking C.

6.1 Diverse Recommendation Lists

To produce recommendation lists that are both similar to the user profile *and* diverse, we solve the optimization problems of Section 4. To do so, we note two steps within the framework at which diversity criteria can be applied. The first is the selection of the candidate set C. Solving Problem 3, for a given θ and a given size $p = l$, allows the selection of l items from $\mathcal{I} - P_u$, which are diverse as well as similar to the user profile. Alternatively, or additionally, from an intermediate set I of items in C that are most similar to the user profile, we can solve Problem 3, to extract the most diverse subset of size $p = N$ from I. The intermediate set is introduced for efficiency reasons only. Diversity criteria can be applied to the entire set C (i.e. $I = C$) and this is in fact the approach used in our evaluation in Section 7.5.

To apply our method, it is necessary to compute the dissimilarity matrix D and determine the matching function $g_m(q_u, i)$. Depending on what application-level information is available, there are many potential ways to do this. For example, in the movie recommendation domain, the dissimilarity function could be based on genre information associated with each movie. For our analysis however, we assume that only purchase transaction information is available. In particular, given the item-item cosine correlation matrix Φ learned from the data using a method such as that given in [3], define the item-item similarity matrix as $\mathrm{S} \triangleq \Phi$. In keeping with [15], the distance matrix D may be chosen as $\mathrm{D} \triangleq 1 - \Phi$. Alternatively, without signficantly changing the optimisation problem, we may simply take $\mathrm{D} \triangleq -\Phi$.

Note that, for the *top-N* problem that we are considering, the transaction matrix consists of user-item entries that are 1 if the corresponding user has purchased the item and 0 otherwise. As a consequence, $\Phi_{ij} \in [0, 1]$. Each of the above matrices is $M \times M$, where M is the size of the super-set from which items are selected. Writing the user profile as the M-dimensional vector $\mathbf{u}$ such that $u(j) = 1$ if user u has purchased the j^{th} item in the super-set and zero otherwise, the matching vector $\mathbf{m}_u$, giving the similarity of each item to the user profile $\mathbf{u}$, is $\mathbf{m}_u \triangleq \mathrm{S}\mathbf{u}$. Hence, the matching value of a sub-set specified by the indicator vector $\mathbf{y}$ is $\beta \mathbf{u}^T \mathrm{S} \mathbf{y}$, where the normalization parameter $\beta \triangleq 1/(p\, \mathbf{1}^T \mathbf{u})$ ensures that the matching value of a sub-set is contained in the same absolute range ($[0, 1]$) as the diversity value (1).

7. EVALUATION

When solving the optimisation problems in this section, we have taken $\mathrm{D} \triangleq -\Phi$. However, so that direct comparison can be made with other work in this area, we have plotted diversity using the more usual definition, $\mathrm{D} \triangleq 1 - \Phi$. Since $\mathbf{y}^T(1 - \Phi)\mathbf{y} = p^2 - \mathbf{y}^T \Phi \mathbf{y}$ for any binary vector $\mathbf{y}$ of norm p, the binary solution using either definition is the same. However, the optimal for the real-valued relaxed problem differs depending on the definition of D so that each definition could lead to different solutions, although we would not expect a significant difference in results.

7.1 Item Novelty

The purpose of introducing diversity into a recommendation engine is to increase the probability of recommending products that the user likes but which are not too alike other products the user has purchased. It is useful to introduce a measure to capture this notion. For any sub-set $R \subseteq \mathcal{I}$, we define the *novelty* of an item $i \in R$ as

$$n_R(i) \triangleq p(f_D(R) - f_D(R - \{i\})) = \frac{1}{p-1} \sum_{j \in R} d(i, j)\,, \quad (13)$$

that is, the amount of additional diversity that i brings to the set R. Highly novel items are also difficult items for a collaborative filtering algorithm to recommend.

7.2 Evaluation Dataset

Our diversity method is evaluated using the Movielens[2] dataset. This is a standard evaluation dataset consisting of $1,000,000$ ratings that $6,040$ users gave for $3,900$ movies. The dataset is highly sparse, containing only 4% of all possible ratings. As we are interested in the *top-N* prediction

[2] `http://www.movielens.org`

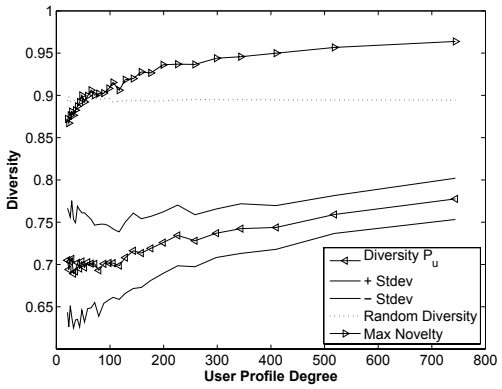

Figure 2: Moving average plot of window size 100 of the diversity of the user profiles P_u in the Movielens dataset against the user profile degree. The variance of the diversity is indicated by two lines that are one standard deviation above and below the mean diversity plot. Also shown is the diversity of a randomly selected item-set of the same size and the maximum novelty of items in the user profile.

problem, we follow [3] and do not consider the rating value when calculating similarities, instead, any rating is accepted as evidence that the movie was watched (and hence "liked" or "purchased").

The collaborative filtering algorithm has been shown to work well on this dataset, under the usual metrics. Thus, we expect that there is strong similarity between items in user profiles i.e. between movies that a user has rated. Nevertheless, highly novel items do exist in the dataset and the probability that standard algorithms will recommend such items is very small.

It is interesting to examine the diversity of user profiles in the Movielens dataset. In Figure 2, user profile diversity is plotted against profile size in a moving average plot, alongside the diversity of randomly chosen item sub-sets of the same size. It is clear that there is a high degree of inter-profile similarity exhibited in the Movielens dataset, indicated by the fact that the diversity is well below that of a randomly selected item-set. However, plotting the maximum novelty of all items in the user profile in this plot also shows that profiles do contain items which are significantly more difficult to recommend than the average items in the dataset.

7.3 Experimental Methodology

Following [3], we split the dataset into a training dataset Y_T and a test dataset Y_P by randomly assigning 50% of the user profiles to each set. The training dataset is used to learn an item-item correlation matrix Φ, by applying the cosine similarity method. The values in Φ lie in the range $[0, 1]$, and Φ is symmetric positive definite.

To evaluate the method, a large number of user profiles P_u are selected at random from Y_P. For each profile, $\gamma|P_u|$ items are removed from the profile and remainder of Y_P is used to make a *top-N* prediction for u. Here $\gamma \in [0, 1]$ represents the percentage of the user profile that is removed. We refer to the removed items as the **profile test-set**, written as $T_u(\mu)$ where $\mu \in [0, 1]$ is a parameter representing the difficulty of the prediction problem. In particular, $T_u(\mu)$ is selected as follows:

- **Novel Test Set Selection**: Using (13) to sort the items in P_u, the items in $T_u(\mu)$ are selected with equal probability from the top $(100 \times \mu)\%$ of the *most novel* items in P_u. Clearly, the smaller μ, the harder the prediction problem. As $\mu \to 1$, test set selection tends to random selection with equal probability from the whole user profile.

Our primary performance metric is *precision*, defined as

$$\rho(\mu) \triangleq \sum_{u \in U} \frac{|T_u(\mu) \cap R|}{N|U|}, \tag{14}$$

where U is the set of all tested users.

7.4 Set Diversity and Matching

We firstly observe the diversity and matching quality of sets produced by solving the optimization problems of Section 4. In particular, (7) is solved to select the candidate set C, from the set of all possible items $\mathcal{I} - P_u$. The candidate set size is fixed to $l = 500$ items. Figure 3 shows the resulting diversity of the candidate set, averaged over all tested users, plotted against the parameter θ. For comparison purposes, also shown in Figure 3 is the average diversity of the candidate set when chosen according to the SUGGEST algorithm. We set $K = 20$ in the SUGGEST algorithm, so that the candidate set is constructed through the intersection of the 20 most similar items to each item in the user profile. To allow direct comparison, we terminate the SUGGEST algorithm as soon as the candidate set contains $l = 500$ items. Lastly, we show the diversity of a candidate set where $l = 500$ items are chosen with equal probability from $\mathcal{I} - P_u$. In Figure 4, the average similarity of the candidate sets to the user profile is plotted. From these plots, it is clear that we can use our method to control diversity and matching quality of the generated sets. As $\theta \to 0$, our method produces sets of greater diversity than random sets, while as $\theta \to 1$, our method produces sets of higher similarity than the SUGGEST algorithm. The interesting range for θ is $\theta \in [0.5, 0.8]$, in which sets of greater diversity can be selected at a cost of reduced matching quality.

7.5 Top-N Prediction Using Diversity

We now examine whether using diversity can produce better system performance, particularly for recommendation of novel items. In this experiment, we use the standard SUGGEST algorithm to select a candidate set C of $l = 500$ items. Again, the SUGGEST parameter $K = 20$ is chosen. We take the intermediate set $I = C$, set the size of R to $N = 20$, and thus select R directly from C, by one of three methods:

1. **Random**:The items in R are selected with equal probability from C;

2. **SUGGEST**: The candidate set is sorted according to the SUGGEST algorithm (i.e. in order of descending similarity to items in the user profile) and the most similar items are placed in R;

3. **Theta**: The optimization problem (7) is solved with $p = N$ and a fixed value of θ to select the set R.

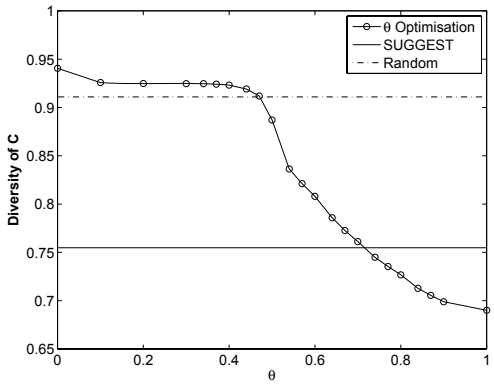

Figure 3: Mean diversity of a candidate set of size $l = 500$, plotted against θ. Also shown is the mean diversity of a randomly chosen candidate set and of a candidate set chosen using the SUGGEST algorithm.

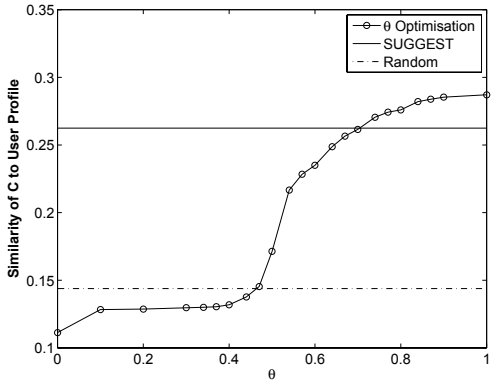

Figure 4: Mean similarity of a candidate set of size $l = 500$, plotted against θ. Also shown is the mean similarity of a randomly chosen candidate set and of a candidate set chosen using the SUGGEST algorithm.

Because novel items have low similarity to the other items on user profile, the SUGGEST algorithm is highly unlikely to recommend them, even if they are in the candidate set. The Random algorithm is likely to perform better for novel items, since once the item is in the candidate set, it has the same chance as all other items of being recommended. We take $\gamma = 0.1$, (i.e. the profile test set size $|T_u|$ is 10% of the total user profile size $|P_u|$) and examine how the algorithms perform for different values of the novelty parameter μ. So, for example, when $\mu = 0.3$, if the user profile is made up of 100 items, then the profile test set contains 10 items with these items drawn from the 30 most novel items in the profile, according to the item novelty value $n_{P_u}(.)$, calculated from (13).

The diversity and similarity of the recommendation list R is shown in Figure 5. The plots follow the same shape as for the candidate set, with the interesting values of θ in the range $[0.5, 0.9]$. Figure 6 plots the precision $\rho(\mu)$ against μ with μ varied in the range $[\gamma, 1.0]$, for the SUGGEST al-

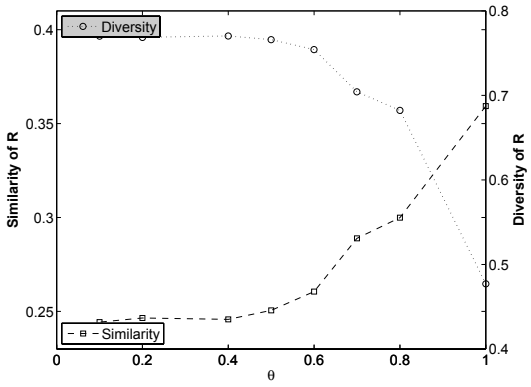

Figure 5: Mean similarity and diversity of the recommended set R, plotted against θ.

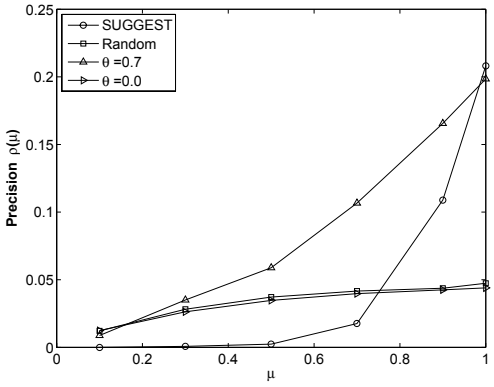

Figure 6: Precision of the recommendation algorithms, plotted against the novelty value of the test set.

gorithm, the Random algorithm and the Theta algorithm using two different values, $\theta = 0.0$ and $\theta = 0.7$. Firstly, it is interesting to observe the behavior of the SUGGEST algorithm. For $\mu \leq 0.3$, $\rho(\mu) = 0$, that is, the algorithm fails completely when the test set is drawn from nearly 1/3 of the user profile containing items of highest novelty. Up to $\mu = 0.5$ the value of $\rho(\mu)$ is still near to 0. This illustrates how standard performance measures can hide important detail about the quality of the recommendation. The standard performance result usually reported corresponds to $\mu = 1.0$, but the recommendation quality falls off quickly once the novelty of the recommendation problem is increased. The Theta algorithm, with $\theta = 0.0$ corresponds closely to the random algorithm, since in this case, the most diverse set R is selected by the optimization algorithm, without any consideration of matching quality. For very novel test sets (i.e. low μ), both algorithms perform better than SUGGEST, as expected.

When $\theta = 0.7$, a strong improvement over the SUGGEST algorithm can be observed. In particular, it may be observed that the performance drops less sharply as $\mu \to 0$ than it does for the SUGGEST algorithm. In fact, when $\mu = 0.3$, the performance has dropped by 76% from that achieved at

$\mu = 1.0$ and while this is a significant drop, it compares very favorably with SUGGEST where the performance has been completely lost at this point.

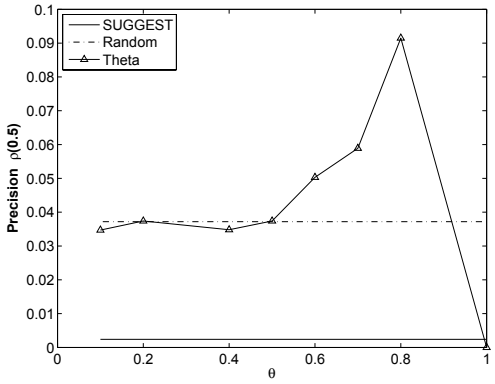

Figure 7: Precision of the recommendation algorithms, plotted against θ with novelty value $\mu = 0.5$.

In Figure 7 we plot precision against θ for a fixed novelty value $\mu = 0.5$, showing the full range of θ in which a clear performance improvement is achieved over both the Random and SUGGEST algorithms.

8. CONCLUSION

In this paper we have shown how to express the competing concerns of maximizing the diversity of a retrieved list while also maximizing its similarity to the target query, as a binary optimization problem. Furthermore, we have given an approach for solving this optimization problem. Applying this theory to the *top-N* recommendation problem, we have shown how to control the diversity and matching quality of recommendation lists. Our evaluation on the Movielens dataset has shown that our method can increase the likelihood of the system recommending novel items, while maintaining good performance on the core items. It is notable however that, even with diversity, the probability of recommending novel items is very low and this will be the case whenever similarity is used as a primary selection criterion. Future work will apply the methodology presented here to other problems in information retrieval. We will also consider other methods for enabling novel but relevant recommendations.

Acknowledgements

This work is supported by Science Foundation Ireland, grant number 07/RFP/CMSF219.

9. REFERENCES

[1] D. Fleder and K. Hosanagar. Recommender systems and their impact on sales diversity. In *Proceedings of the 8th ACM conference on Electronic commerce*, pages 192–199, 2007.

[2] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22(1):5–53, Jan. 2004.

[3] G. Karypis. Evaluation of item-based top-N recommendation algorithms. In *Proceedings of the Tenth International Conference on Information and Knowledge Management*, pages 247–254. ACM SIGMIS, 2001.

[4] C. D. Manning, P. Raghavan, and H. Schutze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[5] L. McGinty and B. Smyth. On the role of diversity in conversational recommender systems. In *Workshop Proceedings of the Fifth International Conference on Case-Based Reasoning(ICCBR 2003)*, page 1065, 2003.

[6] D. McSherry. Diversity-conscious retrieval. In *Proceedings of the 3rd European Conference on Case-Based Reasoning*, pages 219–233, 2002.

[7] B. Mehta, T. Hofmann, and P. Fankhauser. Lies and propaganda: Detecting spam users in collaborative filtering. In *Proceedings of the 12th international conference on Intelligent user interfaces*, pages 14–21, 2007.

[8] K. Nehring and C. Puppe. A theory of diversity. *Econometrica*, 70(3):1155–1198, May 2002.

[9] C. Olsson, A. P. Eriksson, and F. Kahl. Solving large scale binary quadratic problems: Spectral methods vs. semidefinite programming. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition(CVPR '07)*, pages 1–8, 2007.

[10] G. P. Patil and C. Taillie. Diversity as a concept and its measurement. *Journal of the American Statistical Association*, 77(379):548–561, Sep. 1982.

[11] D. Reynolds. Comparison of background normalization methods for text-independent speaker veri cation. In *Fifth European Conference on Speech Communication and Technology*, 1997.

[12] M. Rojas, S. A.Santos, and D. C.Sorensen. A new matrix-free algorithm for the large-scale trust-region subproblem. *SIAM Journal on Optimization archive*, 11(3):611–646, 2000.

[13] M. Rojas, S. Santos, and D. Sorensen. LSTRS: MATLAB software for large-scale trust-region subproblems and regularization. *ACM Trans. Math. Software*, 34(2):11, 2008.

[14] J. R. Sean M. McNee and J. A. Konstan. Accurate is not always good: How accuracy metrics have hurt recommender systems. In *extended abstracts on Human factors in computing systems(CHI'06)*, pages 1097–1101, 2006.

[15] B. Smyth and P. McClave. Similarity vs. diversity. In D. W. Aha and I. Watson, editors, *ICCBR*, volume 2080 of *Lecture Notes in Computer Science*, pages 347–361. Springer, 2001.

[16] L. N. Vicente, J. J. Judice, and P. M. Pardalos. Parametric linear programming techniques for the indefinite quadratic programming problem. *IMA Journal of Management Mathematics*, 4(4):343–349, 1992.

[17] C. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen. Improving recommendation lists through topic diversification. In *Proceedings of the 14th International Conference on World Wide Web*, pages 22–32, 2005.