Pareto-Efficient Hybridization for Multi-Objective Recommender Systems

Marco Tulio Ribeiro[1,2] Anisio Lacerda[1,2] Adriano Veloso[1] Nivio Ziviani[1,2]

[1]Universidade Federal de Minas Gerais
Dept. of Computer Science
Belo Horizonte, Brazil
{marcotcr, anisio, adrianov, nivio}@dcc.ufmg.br

[2]Zunnit Technologies
Belo Horizonte, Brazil
{marcotcr, anisio, nivio}@zunnit.com

ABSTRACT

Performing accurate suggestions is an objective of paramount importance for effective recommender systems. Other important and increasingly evident objectives are novelty and diversity, which are achieved by recommender systems that are able to suggest diversified items not easily discovered by the users. Different recommendation algorithms have particular strengths and weaknesses when it comes to each of these objectives, motivating the construction of hybrid approaches. However, most of these approaches only focus on optimizing accuracy, with no regard for novelty and diversity. The problem of combining recommendation algorithms grows significantly harder when multiple objectives are considered simultaneously. For instance, devising multi-objective recommender systems that suggest items that are simultaneously accurate, novel and diversified may lead to a conflicting-objective problem, where the attempt to improve an objective further may result in worsening other competing objectives. In this paper we propose a hybrid recommendation approach that combines existing algorithms which differ in their level of accuracy, novelty and diversity. We employ an evolutionary search for hybrids following the Strength Pareto approach, which isolates hybrids that are not dominated by others (i.e., the so called Pareto frontier). Experimental results on two recommendation scenarios show that: (i) we can combine recommendation algorithms in order to improve an objective without significantly hurting other objectives, and (ii) we allow for adjusting the compromise between accuracy, diversity and novelty, so that the recommendation emphasis can be adjusted dynamically according to the needs of different users.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Filtering

Keywords

Hybridization, Pareto-Optimality, Diversity, Novelty

1. INTRODUCTION

Recommender systems are increasingly emerging as enabling mechanisms devoted to overcoming problems that are inherent to information overload, providing intelligent information access and delivery, and thus potentially improving browsing and consumption experience. Historically, the typical goal of a recommender system is to maximize accuracy as much as possible in predicting and matching user information needs, often by considering individual delivered items in isolation [12]. More recently, however, it has become a consensus that the success of a recommender system depends on other dimensions of information utility, notably the diversity and novelty of the suggestions performed by the system [9, 19, 25, 33]. More specifically, even being accurate, obvious and monotonous recommendations are generally of little use, since they do not expose users to unprecedent experiences.

Increasing novelty and diversity by completely giving up on accuracy is straightforward - and meaningless, since the system will not meet the users needs anymore. In fact, there is an apparent trade-off between these dimensions, which becomes evident by inspecting the performance of existing top-N recommendation algorithms. An easy conclusion is that different algorithms may perform distinctly depending on the dimension of interest (i.e., the best performer in terms of accuracy is not the best one in terms of novelty and diversity), and thus it is hard to point to a best performer if all the dimensions are considered simultaneously. A conclusion which is harder to reach is whether these algorithms are indeed complementary, so that the strengths of an algorithm may compensate the weaknesses of others. The potential synergy between different recommendation algorithms is of great importance to multi-objective recommender systems, since they must achieve a proper level of each dimension (i.e., objective).

In this paper we hypothesize that it is possible to properly aggregate different recommendation algorithms, so that the resulting hybrids balances the level of accuracy, diversity and novelty in its suggestions. In this case, each potential hybrid is given as a weighted combination of well-established recommendation algorithms (e.g., simple algorithms as well as representative of the state-of-the-art). Our proposed hybridization approach consists in finding appropriate weights for the constituent algorithms. By considering each dimension (i.e., accuracy, novelty and diversity) as a separate objective, we reduce the hybridization task to a multi-objective optimization problem, in which we search for the optimal combination of weights that maximizes accuracy, diversity and novelty.

Since the considered objectives are potentially conflicting, we employ an evolutionary search for optimal hybrids. Evolutionary algorithms denote a class of optimization methods that are characterized by a set of candidate solutions (aka individuals) called a population, which is maintained during the entire optimization process. The population of individuals evolves towards better (and potentially optimal) solutions by employing genetic operators, such as reproduction, mutation and crossover. In our context, each individual represents a possible combination of weights (i.e., a possible hybrid). Optimal hybrids lie in the so-called Pareto frontier [37], and are optimal in the sense that no hybrid in the frontier can be improved upon without hurting at least one of its objectives. Therefore, the evolutionary algorithm evolves the population towards producing hybrids that are located closer to the Pareto frontier, and then a linear search returns the most dominant hybrid [37], which is likely to balance accuracy, novelty and diversity. Alternatively, hybrids in the Pareto frontier can be selected according to a certain need, allowing the recommender system to adjust the compromise between accuracy, novelty and diversity, so that the recommendation emphasis can be adapted dynamically according to the needs of each user (i.e., new users may benefit more from more accurate suggestions, whereas older users may require more novel and diversified suggestions).

We conducted a systematic evaluation involving different recommendation scenarios, with explicit user feedback (i.e., movies from the MovieLens dataset), as well as implicit user feedback (i.e., artists from the LastFM dataset). The experiments showed that it is possible to (i) combine different algorithms in order to produce better recommendations and (ii) control the desired balance between accuracy, novelty and diversity. In order to evaluate the baseline algorithms and our hybrids, we used the methodology for top-N evaluation proposed in [12] and measured novelty and diversity using the framework proposed in [33].

2. PRELIMINARIES

In this section we review the main concepts about evolutionary algorithms and multi-objective optimization. Finally, we discuss related work on hybrid and multi-objective recommender systems.

2.1 Evolutionary Algorithms

Evolutionary algorithms are meta-heuristic optimization techniques that follow processes such as inheritance and evolution as key components in the design and implementation of computer-based problem solving systems [15, 20]. In evolutionary algorithms, a solution to a problem is represented as an individual in a population pool. The individuals may be represented as different data structures, such as vectors, threes, or stacks [26]. If the individual is represented as a vector, for example, each position in the vector is called a gene.

Typically, evolutionary algorithms employ a training and a validation set, as described in Algorithm 1. Initially, the population starts with individuals created randomly (line 6). The evolutionary process is composed of a sequence of solution generations. The process evolves generation by generation through genetic operations (lines 7-12). The goal of this process is to obtain better solutions after some generations. A fitness function is used to assign a fitness value to each individual (line 9), which represents its performance on the training set or in a cross validation set. To produce a new generation, genetic operators are applied to individuals with the aim of creating more diverse and better individuals (line 12). Typical operators include reproduction, mutation, and crossover.

2.2 Multi-Objective Optimization

Since we are interested in maximizing three different objectives for the sake of recommender systems (i.e. accuracy, novelty, and diversity), we use a multi-objective evolutionary algorithm. In multi-objective optimization problems there is a set of solutions that are superior to the remainder when all the objectives are considered together. In general, traditional approaches to multi-objective optimization problems are very limited because they become too expensive as the size of the problem grows [8]. Multi-objective evolutionary algorithms are a suitable option to overcome such an issue.

Algorithm 1 Evolutionary Algorithm.

```
1  Let $\mathcal{M}$ be a training set
2  Let $\mathcal{V}$ be a validation set
3  Let $\mathcal{N}_g$ be the number of generations
4  Let $\mathcal{N}_I$ be the number of individuals
5  $\mathcal{S} \leftarrow \emptyset$
6  $\mathcal{P} \leftarrow$ initial random population of individuals
7  For each generation $g$ of $\mathcal{N}_g$ do
8      For each individual $i \in \mathcal{P}$ do
9          $fitness \leftarrow fitness(i, \mathcal{M}, \mathcal{V})$
10     $\mathcal{S}_g \leftarrow \mathcal{N}_I$ top-ranked individuals of generation $g$
       according to their fitness
11     $\mathcal{S} \leftarrow \mathcal{S} \cup \mathcal{S}_g$
12     $\mathcal{P} \leftarrow$ New population created by applying genetic
       operators to individuals in $\mathcal{S}_g$
13 BestIndividual$\leftarrow$ SelectionMethod($\mathcal{S}$)
```

Typically, multi-objective evolutionary algorithms are classified as Pareto or non-Pareto [37]. In the non-Pareto optimization case, the objectives are combined into a single evaluation value that is used as fitness value (i.e., average of the objectives). In Pareto algorithms, on the other hand, a vector of objective values is used (i.e., the individual is given as an objective vector). The evaluation of Pareto approaches follows the *Pareto dominance* concept. An individual dominates another if it performs better in at least one of the objectives considered. Given two arbitrary individuals, the result of the dominance operation has two possibilities: (i) one individual dominates another, or (ii) the two individuals do not dominate each other. An individual is denoted as *non-dominated* if it is not dominated by any other individual in the population, and the set of all non-dominated individuals compose the *Pareto frontier*.

In this work we use a second version of the strength Pareto evolutionary algorithm (SPEA-2) [36, 37]. The aim is to find or approximate the Pareto-optimal set for multi-objective problems. The main features of this algorithm are: (i) the fitness assignment scheme takes into account how many individuals each individual dominates or is dominated by, (ii) it uses a nearest neighbour density estimation technique to break ties in solutions with the same fitness, (iii) the size of the population of non-dominated solutions is a fixed value η. Thus, we have two situations. First, when the actual number of non-dominated solutions is lower than η, the population is filled with dominated solutions; second, when the actual number of non-dominated solutions exceeds η, some of them are discarded by a truncation operator which preserves boundary conditions,even though we always keep the current Pareto Frontier in a list separate from the population, so we can later retrieve the individuals in it.

2.3 Related Work

Traditionally, hybrid recommender strategies are the combination of two different families of algorithms - namely, content-based and collaborative filtering [1]. In this work, we combine many (up to 8) recommendation algorithms - different content-based and collaborative filtering algorithms that deal with explicit and implicit feedback, etc. We treat each recommendation algorithm as a black-box, so adding or removing recommendation algorithms is easy. Different hybridization strategies have been proposed to combine recommender methods, such as weighted approaches [10], voting mechanisms [30], switching between different recommenders [6, 24], and re-ranking the results of one recommender with another [7].

A prominent use of hybridization in recommender systems is the Belkor system that won the Netflix competition [4, 5]. Their method is a statically weighted linear combination of 107 collaborative filtering engines. There are important differences between their work and ours: (i) their solution is single-objective (accuracy), (ii) they combine only collaborative filtering information, and (iii) the recommendation task is rating prediction, focused on RMSE - which makes the aggregation simpler, since all of the ratings are on the same scale and consist of the same items.

There has been an increasing consensus in the recommender systems community about the importance of proposing algorithms and methods to enhance novelty and diversity [17, 33]. As showed in [35], user satisfaction does not always correlate with high recommender accuracy. Thus, different multi-objective algorithms have been proposed to improve user experience considering either diversity or novelty. For instance, in [35], the authors define a greedy reranking algorithm that diversifies baseline recommendations. Another approach to improve diversity is presented in [34], where they suggest an optimization method to improve two objective functions reflecting preference similarity and item diversity.

On the other hand, novelty has been understood as recommending long-tail items, i.e., those items which few users have accessed. In [33], the authors present hybrid strategies that combine collaborative filtering with graph spreading techniques to improve novelty. The authors in [9] take an alternative approach: instead of assessing novelty in terms of the long-tail items that are recommended, they follow the paths leading from recommendations to the long tail using similarity links. As far as we know, this is the first work that proposes a hybrid method that is multi-objective in terms of the three metrics, i.e., accuracy, diversity and novelty.

Extensive research has also been performed exploiting the robust characteristics of genetic algorithms in recommender systems. For instance, in [28] the authors build a content-based recommender system and use genetic algorithms to assign proper weights to the words. Such weights are combined using the traditional IR *vector space model* [2] to produce recommendations. In [23] the authors use a genetic algorithm to build a recommender method that considers the browsing history of users in real-time. In contrast to our approach (which uses a GA to combine multiple recommender methods), they use GA to build a single-method.

In [22], the authors present an implementation of GA for optimal feature weighting in the multi-criteria scenario. Their application of GA consists in selecting features that represent users' interest in a collaborative filtering context, in contrast to our method, which focuses on assigning weights to different recommendation algorithms in order to improve the overall performance in terms of accuracy, novelty and diversity.

3. PARETO-EFFICIENT HYBRIDIZATION

In this section we introduce our search approach for Pareto-Optimal hybrids. We start by discussing how different recommendation algorithms are combined, so that potential hybrids are created. Then we describe the evolutionary search for Pareto-Optimal hybrids. Finally, we discuss an approach to deal with the compromise between accuracy, novelty and diversity, so that the system is able to adjust itsef for different user perspectives.

3.1 Weighted Hybridization

Our hybridization approach is based on assigning weights to each constituent algorithm. We denote the set of constituent algorithms as A and the score given by algorithm A_j for an item i is represented by $A_j(i)$. As the constituent algorithms may output scores in drastically different scales, a simple normalization procedure is necessary to ensure that all algorithms in A operate in the same scale. The aggregated score for each item i is calculated as:

$$S(i) = \sum_{j=1}^{|A|} A_j(i) * W_j \qquad (1)$$

where W is a vector that represents the weight assigned to each constituent algorithm. The assignment of weights to each algorithm is formulated as a search problem which we discuss next.

3.2 Searching for Pareto-Optimal Hybrids

Finding a suitable vector of weights W can be viewed as a search problem in which possible solutions are given as a combination of weights $\{w_1, w_2, \ldots, w_{|A|}\}$, such that each w_i is selected in a way that optimizes a established criterion. We consider the application of evolutionary algorithms for searching optimal solutions. These algorithms iteratively evolve a population of individuals towards optimal solutions by performing operations based on reproduction, mutation, recombination, and selection [18]. This approach is interesting because we have no knowledge of the search space, since any number of different algorithms may be used, in different domains. Next, we precisely define an individual.

Definition 1: *An individual is a candidate solution, which is encoded as a sequence of* $|A|$ *values* $[w_1, w_2, \ldots, w_{|A|}]$*, where each* w_i *indicates the weight associated with algorithm* $A_i \in A$.

Each algorithm A_i assigns scores to items using a cross-validation set. Finally, weights are assigned to each algorithm and their scores are aggregated according to Equation 1, producing an individual. A fitness function is computed for each individual in order to make them directly comparable, so that the population can evolve towards optimal solutions.

Definition 2: *An optimal solution is a sequence of weights* $W = \{w_1, w_2, \ldots, w_{|A|}\}$*, satisfying:*

$$\text{maximize } \phi(o_i) \; \forall \; o_i \in \{\text{accuracy, novelty, diversity}\} \qquad (2)$$

where $\phi(o_i)$ is a metric used to measure an objective, which can be either accuracy, novelty or diversity. These metrics are better discussed in Section 4. For now it suffices to notice that the performance of each individual is given by a 3-dimensional objective vector, containing the average accuracy, novelty and diversity over the users in the cross validation set (since different metrics may operate in different scales, we normalize each $\phi(o_i)$ to the 0-1 interval). Searching for optimal solutions, therefore, is a multi-objective optimization problem, in which the value of $\phi(o_i)$ must be maximized for each of the 3 objectives that compose an optimal solution. Therefore, multiple optimal individuals are possible. It is worth noticing that different datasets and combinations of algorithms and A will generate different optimal individuals.

A general strategy for solving a multi-objective optimization problem is to exploit the concept of Pareto dominance, which may be used to find solutions that are not dominated by others. These non-dominated solutions lie in the so-called Pareto frontier, and are optimal in the sense that no solution in the frontier can be improved upon without hurting at least one of its objectives. Therefore, the evolutionary algorithm evolves the population towards producing individuals that are located closer to the Pareto frontier, and then a linear search returns the individual which simply maximizes the average (or some other combination, as we see on the next section) of the three objectives. Under this strategy, we follow the well-known Strength Pareto Evolutionary Algorithm approach [36], which has shown to be highly effective and also because it provides more diverse results when compared to existing approaches [11, 13, 32] for many problems of interest. The Strength Pareto approach isolates individuals that achieve a compromise between maximizing the competing objectives by evolving individuals that are likely to be non-dominated by other individuals in the population.

It is worth noticing that our approach does not depend on which recommendation algorithms are being aggregated, nor does it depend on the data domain. This makes adding or removing algorithms trivial, and allows the data to determine how each algorithm contributes to each of the objectives - an algorithm may be the most accurate when ratings are available, but not so accurate when only implicit feedback is used.

The Pareto-Optimal search is computationally expensive. However, it can be performed in an off-line manner, and with low frequency. After the Pareto-Optimal weights are discovered, there is no need to perform the search repeatedly, unless a recommendation algorithm is added or removed, or a lot of new feedback data enters the system. Therefore, using this approach would not hinder the system's online performance.

3.3 Adjusting the System Priority

It is well recognized that the role that a recommender system plays may vary depending on the target user. For instance, according to [19], the suggestions performed by a recommender system may fail to appear trustworthy to a new user because it does not recommend items the user is sure to enjoy but probably already knows about. Based on this, a recommender system might prioritize accuracy instead of novelty or diversity for new users, while prioritizing novelty for users that have already used the system for a while. This is made possible by our hybridization approach, by searching which individual in the *Pareto frontier* better solves the user's current needs.

The choice of which individual in the Pareto frontier is accomplished by performing a linear search on all of the individuals, in order to find which one maximizes a simple weighted mean on each of the three objectives in the objective vector, where the weights in the weighted mean represent the priority given to each objective. It is worth noting that fitness values are always calculated using the cross-validation set. Therefore, considering a 3-dimensional priority vector Q, which represents the importance of each objective j, the individual in the Pareto frontier P is chosen as:

$$\underset{i \in P}{arg\ max} \sum_{j=1}^{|O|} Q_j O_{ij} \quad (3)$$

4. EVALUATION METHODOLOGY

The testing methodology we adopted in this paper is similar to the one described in [12], which is appropriate for the top-N recommendation task. For each dataset, ratings are split into two subsets: the training set $\mathcal{M}$ and the test set $\mathcal{T}$. The training set $\mathcal{M}$ can (if necessary) be split into two subsets: the cross-validation training set $\mathcal{C}$ and the cross-validation test set $\mathcal{V}$, which is used in order to tune parameters or adjust models. The test set $\mathcal{T}$ and the cross-validation test set $\mathcal{V}$ only contain items that are considered relevant to the users in the set. For explicit feedback (i.e., MovieLens), this means that the sets $\mathcal{T}$ and $\mathcal{V}$ only contain 5-star ratings.

In the case of implicit feedback (i.e., Last.fm), we normalized the observed item access frequencies of each user to a common rating scale [0,5], as used in [33]. Namely, $r(u,i) = n * F(\mathit{frec}_{u,i})$, where $\mathit{frec}_{u,i}$ is the number of times u has accessed i and $F(\mathit{frec}_{u,i}) = |j \in \mathbf{u}|f_{u,j} < f_{u,i}|/|\mathbf{u}|$ is the cumulative distribution function of $\mathit{frec}_{u,i}$ over the set of items accessed by the user u, denoted as $\mathbf{u}$. In this case, the test set and the cross validation test set only contain ratings such that $r(u,i) >= 4$, since the number of 5 star ratings is very small using this mapping of implicit feedback into ratings. It is worth noting that all the sets have a corresponding implicit feedback set, used by the recommendation algorithms that can deal with implicit feedback.

The detailed procedure to create $\mathcal{M}$ and $\mathcal{T}$ is the same used in [12], in order to maintain compatibility with their results. Namely, for each dataset we randomly sub-sampled 1.4% of the ratings from the dataset in order to create a probe set. The training set $\mathcal{M}$ contains the remaining ratings, while the test set $\mathcal{T}$ contains all the 5-star ratings in the probe set (in the case of explicit feedback) or 4+ star ratings (in the case of implicit feedback mapped into explicit feedback). We further divided the training set in the same fashion, in order to create the cross-validation training and test sets $\mathcal{C}$ and $\mathcal{V}$. The ratings in the probe sets were not used for training.

In order to evaluate the algorithms, we first train the models using $\mathcal{M}$. Then, for each item in $\mathcal{T}$ that is relevant to user u:

- We randomly select 1,000 additional items unrated by user u. The assumption is that most of them will not be interesting to u.
- The algorithm in question forms a ranked list by ordering all of the 1,001 items. The most accurate result corresponds to the case where the test item i is in the first position.

Since the task is top-N recommendation, we form a top-N list by picking the N items out of the 1,001 that have the highest rank. If the test item i is among the top-N items, we have a *hit*. Otherwise, we have a *miss*. Recall and precision are calculated as follows:

$$\mathit{recall}(N) = \frac{\#\mathit{hits}}{|T|} \quad (4)$$

$$\mathit{precision}(N) = \frac{\#\mathit{hits}}{N * |T|} = \frac{\mathit{recall}(N)}{N} \quad (5)$$

In order to measure the novelty of the recommendations, we used a popularity-based item novelty model proposed in [33], so that the probability of an item i being seen is estimated as:

$$P(\mathit{seen}|i_k) = \frac{|u \in U|r(u,i) \neq \emptyset|}{|U|} \quad (6)$$

where U denotes the set of users. Since the testing methodology supposes that most of the 1,000 additional unrated items are not relevant, we used the metrics in the framework proposed in [33] without relevance awareness. The novelty of a top-N recommendation list from R presented to user u is therefore given by:

$$\mathit{nov}(R(N)) = \mathit{EPC}(N) = C \sum_{i_k \in R}^{i_N} \mathit{rd}(k)(1 - p(\mathit{seen}|i_k)) \quad (7)$$

where $\mathit{rd}(k)$ is a rank discount given by $\mathit{rd}(k) = 0.85^{k-1}$[33] and C is a normalizing constant given by $1/\sum_{i_k \in R}^{i_N} \mathit{rd}(k)$. Therefore, this metric is rank-sensitive (i.e. the novelty of the top-rated items counts more than the novelty of other items). As is the case with precision and recall, we average the EPC@N value of the top-N recommendation lists over the test set.

We used a distance based model in order to measure the diversity of the recommendation lists. Once again, we used the metrics from [33] without relevance-awareness. The recommendation diversity, therefore, is given by:

$$\mathit{div}(R(N)) = \mathit{EILD}(N) = \sum_{\substack{i_k \in R \\ i_l \in R \\ l \neq k}}^{i_N, l_N} C_k \mathit{rd}(k) \mathit{rd}(l|k) d(i_k, i_l) \quad (8)$$

where $rd(l|k) = rd(max(1, l-k))$ reflects a relative rank discount between l and k, and $d(i_k, i_l)$ is the cosine similarity between two items, given by:

$$d(i,j) = \frac{|\mathbf{U_i} \cap \mathbf{U_j}|}{\sqrt{|\mathbf{U_i}|}\sqrt{|\mathbf{U_j}|}} \quad (9)$$

such that $\mathbf{U_i}$ denotes the users that liked item i, and $\mathbf{U_j}$ denotes the users that liked item j.

5. EXPERIMENTAL EVALUATION

We apply the methodology presented in Section 4 to two different scenarios, in order to evaluate our hybrid approach: movie and music recommendation. For movie recommendation, we used the MovieLens dataset [27]. This dataset contains 1,000,209 ratings from 6,040 users on 3,883 movies. For music recommendation, we used an implicit preference dataset from [9], which consists of 19,150,868 user accesses to music tracks on the website Last.fm[1]. This dataset involves 176,948 artists and 992 users, and we considered the task of recommending artists to users. Mapping the implicit feedback into user-artist ratings yielded a total of 889,558 ratings, which were used by the algorithms that cannot deal with implicit feedback, and to separate the dataset into the training and test sets $\mathcal{M}$ and $\mathcal{T}$.

5.1 Recommendation Algorithms

We selected eight recommendation algorithms to provide the base for our hybrids. To represent latent factor models, we selected PureSVD with 50 and 150 factors (**PureSVD50** and **PureSVD150**), described in [12]. These were the only algorithms we used that are based on explicit feedback. To compute the scores for the items in the Last.fm dataset, we used the mappings of implicit feedback into ratings explained in Section 5.3.

As for recommendation algorithms that use implicit feedback, we used algorithms available in the MyMediaLite package [16]. We used **WeightedItemKNN** (WIKNN) and **WeightedUserKNN** (WUKNN) as representative of neighbourhood models based on collaborative data [14] (we only used **WeightedItemKNN** on the MovieLens dataset, as MyMediaLite's implementation cannot yet handle datasets where the number of items is very large, which is the case in the Last.fm dataset). As a baseline, and to allow for comparison with [12], we used MyMediaLite's **MostPopular** implementation, which is the same as TopPop in [12]. We also used **WRMF** – a weighted matrix factorization method based on [21, 29], which is very effective for data with implicit feedback. In order to represent content-based algorithms, we used **ItemAttributeKNN**(IAKNN), a K-nearest neighbor item-based collaborative filtering using cosine-similarity over the movie genres for MovieLens (we could not use this method in the Last.fm dataset, because it does not contain content data). Finally, we used **UserAttributeKNN**(UAKNN), a K-nearest neighbor user-based collaborative filtering using cosine-similarity over the user attributes, such as sex, age, etc. (which both datasets provide).

5.2 Hybrid Approaches

As a baseline, we used a voting-based hybrid based on Borda-Count (BC) which is similar to [30], where each constituent algorithm gives n points to each item i such that $n = |R| - p_i$, where $|R|$ is the size of the recommendation list and p_i is the position of i in R. We also used STREAM as baseline, a stacking-based approach with additional meta-features [3]. We used the same additional meta-features as [3], namely, the number of items that a certain user has rated and the number of users that has rated a certain item (denoted as RM_1 and RM_2 in [3]). We tried the learning algorithms proposed in [3], and Linear Regression yielded the best results, so the results presented for STREAM are generated using Linear Regression as the meta-learning algorithm. Our last baseline is the weighted hybrid we proposed in Section 3.1, using equal weights for each constituent algorithm. We called this baseline Equal Weights (EW).

As for our genetic approach, we combined all of the the recommendation algorithms cited in the last subsection. We used an open-source implementation of SPEA2 [36, 37] from DEAP [31].We used a two points crossover operator [20], and a uniform random mutation operator with probability 0.05. SPEA-2 was configured with the following parameters:

	Datasets	
Parameters	Movielens	Last.fm
Population Size	100	100
Gene length	8 algorithms	6 algorithms
# of Objectives	3	3
# of Generations	300	800
Mutation Rate	0.2	0.2
Crossover Rate	0.5	0.5

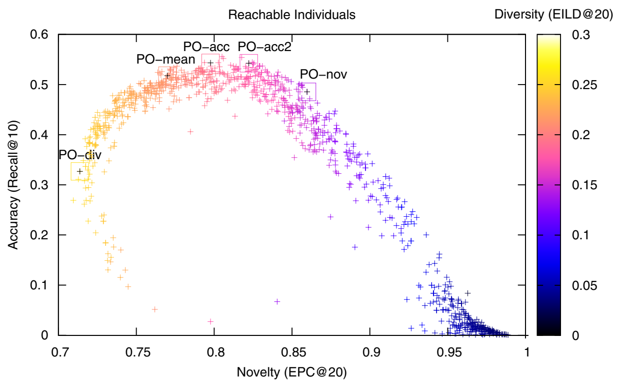
5.3 Results and Discussion

The results achieved by each of the constituent recommendation algorithms can be seen in Tables 1 and 2. We show the accuracy results (recall and precision) over different values of N. Since both EPC(novelty) and EILD(diversity) are rank-sensitive metrics, we only presented their values for $N = 20$. There is a clear compromise between accuracy, novelty and diversity of these algorithms. For the MovieLens dataset (Table 1), the constituent algorithm that provides the most accurate recommendations is PureSVD50. The constituent algorithm that provides the most novel recommendation with an acceptable degree of accuracy is PureSVD150, but its accuracy is much worse than the accuracy obtained by PureSVD50, and its diversity is much worse than the other algorithms. TopPop provided the most diverse recommendations, although it performs significantly worse in accuracy and novelty. It is worth noting that ItemAttributeKNN is based only on genres, which explains its poor accuracy results.

On the Last.fm dataset, the constituent algorithm that provides the most accurate recommendations is WRMF. This is expected, as Last.fm is originally an implicit feedback dataset, to which WRMF is more suitable. Once again, PureSVD150 proved its capacity to suggest novel items, being the algorithm with the most novel recommendations. WeightedUserKNN proved to be the algorithm that provided the most diverse recommendations, while maintaining a reasonable accuracy degree. In this dataset the compromise between the three objectives is once again illustrated by the fact that there is no algorithm that dominates the others in every objective.

Regarding the performance of the baselines in the MovieLens dataset, STREAM performs worse then PureSVD50 on accuracy, maintaining the same level of novelty and performing better in terms of diversity. Borda Count performed poorly on accuracy and reasonably well in terms of novelty and diversity. Equal Weights performed poorly on accuracy and novelty and well on diversity. On the Last.fm dataset, STREAM performed slightly worse than WRMF in accuracy, while maintaining the same level of diversity and improving slightly on novelty. Once again, Borda Count performed poorly on accuracy and reasonably well on novelty and di-

[1]www.Last.fm



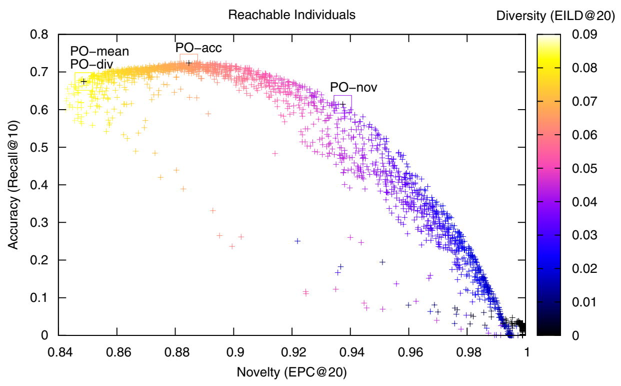


Figure 1: Individuals found in the Movielens (left) and Last.fm (right) datasets.

versity. Finally, Equal Weights performed poorly on accuracy and novelty, while performing well on diversity.

Now, with our evolutionary approach, we could reach any of the individuals in Figure 1, which represent the accuracy (in this case, Recall@10) and novelty (EPC@20) of the recommendations in x and y axes, and diversity (EILD@20) with a color scale. These graphics show the results in the test set for the individuals that represented the Pareto frontier in the cross-validation. It is clear that there is a compromise between the three objectives: the individuals with the most novel recommendations provide less accurate and diverse lists, and so on. This compromise can be adjusted dynamically with little extra cost, since the cost of reaching these individuals is as low as a linear search (for the individual that maximizes a weighted mean, as described on Section 3.2) over the Pareto frontier individuals' scores on the cross validation set. The Pareto frontier consists of 1,418 individuals in the MovieLens dataset and of 1,995 individuals in the Last.fm dataset, so a linear search can be done very quickly. We chose to demonstrate a few of these individuals in Tables 1 and 2. First, Pareto-Optimal-mean (PO-mean) represents the individual that optimizes the mean of the three normalized objectives, assuming each of them are equally important. This would be an option if personalization was not desired, or if the designers of the recommender systems did not know which combination of the three objectives would result in higher user satisfaction. However, in a more realistic situation, the recommender system would most likely want to select different individuals for different users. We selected as examples the following individuals, which were found by the process explained in Section 3.2 with the represented associated weighted vectors:

- PO-acc: [Accuracy:0.85, Novelty:0.1, Diversity:0.05]
- PO-acc2: [Accuracy:0.7, Novelty:0.3, Diversity:0]
- PO-nov: [Accuracy:0.15, Novelty:0.85, Diversity:0]
- PO-div: [Accuracy:0.15, Novelty:0.15, Diversity:0.7]

We compared PO-acc and PO-acc2 with PureSVD50, which is the stand-alone algorithm with the most accurate recommendations. Both perform as well as PureSVD50 on accuracy, but PO-acc performs much better on diversity (and equally well on novelty), and PO-acc2 performs better on novelty while maintaining the diversity level. We compared PO-nov with Pure-SVD150, which presented the most novel recommendations to the users, with reasonable accuracy. PO-nov performs slightly better on novelty than PureSVD150, but performs much better in terms of accuracy, and slightly on diversity. Finally, we compared PO-div with MostPopular, the algorithm with the most diverse recommendations. PO-div loses very slightly on diversity, while improving on accuracy and novelty. We were able, therefore, to find individuals in the Pareto frontier that performed close or better than the best algorithms in each individual objective, but better on the other objectives. Once again, we could have chosen to compromise more accuracy and diversity if we desired more novelty, as is shown by Figure 1 (left).

As for the Last.fm dataset, we selected the following individuals:

- PO-acc: [Accuracy:0.7, Novelty:0.3, Diversity:0]
- PO-nov: [Accuracy:0.15, Novelty:0.85, Diversity:0]
- PO-div: [Accuracy:0.45, Novelty:0.05, Diversity:0.5]

For the Last.fm dataset, we compared PO-acc with WRMF, which is the most accurate stand-alone algorithm on this dataset. PO-acc is much more accurate than WRMF, while also improving on novelty and maintaining the diversity level. The individual PO-nov was compared with PureSVD150, and it performed equally well on accuracy, while delivering a much higher novelty, and only a slightly worse diversity. PO-div was compared against WeightedUserKNN, and it faired equally well on diversity and novelty, while slightly improving on accuracy. It is worth noticing that the individual represented by PO-div is the same individual that maximizes the mean with equal weight (PO-mean). Once again, we were able to find interesting individuals in the Pareto frontier, but we could have reached any of the individuals in Figure 1 (right) by tweaking the weight value for each objective.

6. CONCLUSIONS

In this paper, we propose a hybridization technique for combining different recommendation algorithms, following the Strength Pareto approach. We show that different recommendation algorithms do not perform uniformly well when evaluated in accuracy, novelty and diversity, but our technique allows for the dynamic adjustment of the compromise between these three aspects of user satisfaction. This can be very useful in different scenarios, one example being the personalization of recommendations according to the users. According to [25], "New users have different needs from experienced users in a recommender. New users may benefit from an algorithm which generates highly ratable items, as they need

		Accuracy								Novelty	Diversity
	Algorithm	R@1	R@5	R@10	R@20	P@1	P@5	P@10	P@20	EPC@20	EILD@20
Constituent Algorithms	PSVD50 †	**0.1900**	**0.4155**	**0.5402**	**0.6643**	**0.1900**	**0.0831**	**0.0540**	**0.0332**	0.8070	0.1667
	PSVD150 ●	0.1237	0.3203	0.4450	0.5658	0.1237	0.0641	0.0445	0.0283	0.8519	0.1375
	TopPop ◇	0.0722	0.2061	0.2895	0.3994	0.0722	0.0412	0.0289	0.0200	0.7079	**0.2598**
	WRMF	0.1513	0.3453	0.4545	0.5674	0.1513	0.0691	0.0455	0.0284	0.7847	0.1993
	WIKNN	0.1529	0.3564	0.4624	0.5806	0.1529	0.0713	0.0462	0.0290	0.7744	0.2160
	WUKNN	0.1510	0.3364	0.4437	0.5707	0.1510	0.0673	0.0444	0.0285	0.7560	0.2215
	UAKNN	0.0614	0.1762	0.2504	0.3387	0.0614	0.0352	0.0250	0.0169	0.7386	0.2270
	IAKNN	0.0000	0.0072	0.0105	0.0144	0.0000	0.0014	0.0011	0.0007	**0.9723**	0.0106
Baselines	STREAM	**0.1792**	**0.3961**	**0.5169**	**0.6426**	**0.1792**	**0.0792**	**0.0517**	**0.0321**	0.8078	0.1914
	BC	0.0473	0.1657	0.2639	0.4352	0.0473	0.0331	0.0264	0.0218	**0.8210**	0.1609
	EW	0.1562	0.3574	0.4752	0.5980	0.1562	0.0715	0.0475	0.0299	0.7441	**0.2284**
Our Hybrids	PO-mean †	0.1894	0.3991	0.5176	0.6400	0.1894	0.0798	0.0518	0.0320	0.7700	0.2096
	PO-acc †	**0.1999**	**0.4227**	**0.5432**	**0.6705**	**0.1999**	**0.0845**	**0.0543**	**0.0335**	0.7977	0.1897
	PO-acc2 †	0.1946	0.4188	0.5425	0.6659	0.1946	0.0838	0.0543	0.0333	0.8223	0.1685
	PO-nov ●	0.1513	0.3725	0.4854	0.6111	0.1513	0.0745	0.0485	0.0306	**0.8597**	0.1411
	PO-div ◇	0.0820	0.2271	0.3269	0.4470	0.0820	0.0454	0.0327	0.0223	0.7138	**0.2563**

Table 1: Results for Recommendation Algorithms on the MovieLens dataset, with the three objectives (i.e., accuracy, novelty, and diversity). The recommender methods variants are grouped into: (i) constituent algorithms, (ii) hybrid baselines, and (iii) our proposed hybrids. We used the symbols: †, •, ◇ to point out our method and the respective baseline. For instance, PSVD150 is the baseline with respect to the selected PO-nov individual. For each group, the best results for each metric are in bold. Underlined values means that the selected individual and the respective baseline are statistically different (95%).

to establish trust and rapport with a recommender before taking advantage of the recommendations it offers." Therefore, our approach could be used to provide new users with the most accurate recommendations as possible, even if the recommendations are not novel at all - so the users would have items to rate, and build trust in the system. The costly part of our technique (the evolutionary algorithm) is performed off-line, and the online cost of choosing an individual in the pareto frontier and weighting the results for different algorithms is very small, since the pareto frontier is comprised of few individuals.

We performed highly reproducible experiments on public datasets of implicit and explicit feedback, using open-source implementations. In our experiments, we demonstrated our technique's ability to balance each of the objectives according to the desired compromise, and we showed some examples of reached solutions that are competitive with the best algorithms according to each objective and almost always better on the other objectives.

7. ACKNOWLEDGMENTS

This work was partially sponsored the Brazilian National Institute of Science and Technology for the Web (grant MCT/CNPq 573871/2008-6), and by the authors' individual grants from CNPq.

8. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005.

[2] R. Baeza-Yates and B. Ribeiro-Neto. *Modern information retrieval*. Addison-Wesley, 2011 (second edition).

[3] X. Bao, L. Bergman, and R. Thompson. Stacking recommendation engines with additional meta-features. In *ACM RecSys*, pages 109–116, 2009.

[4] R. Bell, Y. Koren, and C. Volinsky. Chasing $1,000,000: How we won the netflix progress prize. *ASA Statistical and Computing Graphics Newsletter*, 18(2):4–12, 2007.

[5] J. Bennett, S. Lanning, and N. Netflix. The netflix prize. In *KDD Cup and Workshop in conjunction with KDD*, 2007.

[6] D. Billsus and M. Pazzani. User modeling for adaptive news access. *User modeling and user-adapted interaction*, 10(2):147–180, 2000.

[7] R. Burke. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12(4):331–370, 2002.

[8] R. Cecchini, C. Lorenzetti, A. Maguitman, and N. Brignole. Multiobjective evolutionary algorithms for context-based search. *JASIST*, 61(6):1258–1274, 2010.

[9] O. Celma and P. Herrera. A new approach to evaluating novel recommendations. In *ACM RecSys*, pages 179–186, 2008.

[10] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin. Combining content-based and collaborative filters in an online newspaper. In *ACM SIGIR Workshop on Recommender Systems*, pages 40–48, 1999.

[11] D. Corne, J. Knowles, and M. Oates. The pareto envelope-based selection algorithm for multi-objective optimisation. In *Parallel Problem Solving from Nature*, pages 839–848, 2000.

[12] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *ACM RecSys*, pages 39–46, 2010.

[13] K. Deb. Multi-objective genetic algorithms: Problem difficulties and construction of test problems. *Evolutionary Computation*, 7(3):205–230, 1999.

[14] C. Desrosiers and G. Karypis. A comprehensive survey of neighborhood-based recommendation methods. In F. Ricci,

		Accuracy								Novelty	Diversity
	Algorithm	R@1	R@5	R@10	R@20	P@1	P@5	P@10	P@20	EPC@20	EILD@20
Constituent Algorithms	PSVD50	**0.3859**	0.5997	0.6649	0.7178	**0.3859**	0.1199	0.0665	0.0359	0.8878	0.0557
	PSVD150 ●	0.3265	0.5241	0.6055	0.6667	0.3265	0.1048	0.0605	0.0333	**0.8998**	0.0484
	TopPop	0.1879	0.4114	0.5198	0.6224	0.1879	0.0823	0.0520	0.0311	0.8508	0.0755
	WRMF †	0.3834	**0.6148**	**0.7073**	**0.7858**	0.3834	**0.1230**	**0.0707**	**0.0393**	0.8735	0.0669
	WUKNN ◇	0.3272	0.5662	0.6562	0.7340	0.3272	0.1132	0.0656	0.0367	0.8481	**0.0821**
	UAKNN	0.1922	0.3790	0.4712	0.5328	0.1922	0.0758	0.0471	0.0266	0.8605	0.0726
Baselines	STREAM	**0.3898**	**0.6022**	**0.6685**	**0.7185**	**0.3898**	**0.1204**	**0.0668**	**0.0359**	**0.8882**	0.0555
	BC	0.2973	0.5346	0.6026	0.6692	0.2973	0.1069	0.0603	0.0335	0.8606	0.0742
	EW	0.3017	0.5850	0.6785	0.7595	0.3017	0.1170	0.0679	0.0380	0.8473	**0.0807**
Our Hybrids	PO-mean †	0.3387	0.5857	0.6749	0.7487	0.3387	0.1171	0.0675	0.0374	0.8487	**0.0831**
	PO-acc †	**0.4269**	**0.6501**	**0.7239**	**0.7822**	**0.4269**	**0.1300**	**0.0724**	**0.0391**	0.8839	0.0643
	PO-nov ●	0.3143	0.5320	0.6141	0.6800	0.3143	0.1064	0.0614	0.0340	**0.9374**	0.0392
	PO-div ◇	0.3387	0.5857	0.6749	0.7487	0.3387	0.1171	0.0675	0.0374	0.8487	**0.0831**

Table 2: Results for Recommendation Algorithms on the Last.fm dataset, with the three objectives (i.e., accuracy, novelty, and diversity). The recommender methods variants are grouped into: (i) constituent algorithms, (ii) hybrid baselines, and (iii) our proposed hybrids. We used the symbols: †, ●, ◇ to point out our method and the respective baseline. For each group, the best results for each metric are in bold. Underlined values means that the selected individual and the respective baseline are statistically different (95%).

L. Rokach, B. Shapira, and P. Kantor, editors, *Recommender Systems Handbook*, pages 107–144. Springer, 2011.

[15] A. Eiben and J. Smith. *Introduction to evolutionary computing*. Springer Verlag, 2003.

[16] Z. Gantner, S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. MyMediaLite: A free recommender system library. In *ACM RecSys*, 2011.

[17] M. Ge, C. Delgado-Battenfeld, and D. Jannach. Beyond accuracy: evaluating recommender systems by coverage and serendipity. In *ACM RecSys*, pages 257–260, 2010.

[18] E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.

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

[20] J. Holland. *Adaptation in natural and artificial systems*. Number 53. University of Michigan press, 1975.

[21] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *ICDM*, pages 263–272, 2008.

[22] C. Hwang. Genetic algorithms for feature weighting in multi-criteria recommender systems. *JCIT*, 5(8), 2010.

[23] M. Jung, J. Oh, and E. Lee. Genetic recommend generating method with real-time fitness function adaption. *International Journal of u-and e-Service, Science and Technology*, 1(1):9–16, 2008.

[24] G. Lekakos and P. Caravelas. A hybrid approach for movie recommendation. *Multimedia tools and applications*, 36(1):55–70, 2008.

[25] S. McNee, J. Riedl, and J. Konstan. Being accurate is not enough: how accuracy metrics have hurt recommender systems. In *CHI Extended Abstract*, pages 1097–1101. ACM, 2006.

[26] Z. Michalewicz. *Genetic algorithms+ data structures*. Springer Verlag, 1996.

[27] B. N. Miller, I. Albert, S. K. Lam, J. A. Konstan, and J. Riedl. Movielens unplugged: experiences with an occasionally connected recommender system. In *IUI*, pages 263–266, 2003.

[28] J. Pagonis and A. Clark. Engene: A genetic algorithm classifier for content-based recommender systems that does not require continuous user feedback. In *UKCI*, pages 1–6, 2010.

[29] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang. One-class collaborative filtering. In *ICDM*, pages 502–511, 2008.

[30] M. J. Pazzani. A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review*, 13(5-6):393–408, 1999.

[31] F. Rainville, F. Fortin, M. Gardner, M. Parizeau, and C. Gagne. Deap: A python framework for evolutionary algorithms. In *EvoSoft Workshop (GECCO 2012)*, 2012.

[32] N. Srinivas and K. Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248, 1994.

[33] S. Vargas and P. Castells. Rank and relevance in novelty and diversity metrics for recommender systems. In *ACM RecSys*, pages 109–116, 2011.

[34] M. Zhang and N. Hurley. Avoiding monotony: improving the diversity of recommendation lists. In *ACM RecSys*, pages 123–130, 2008.

[35] C. Ziegler, S. McNee, J. Konstan, and G. Lausen. Improving recommendation lists through topic diversification. In *WWW*, pages 22–32, 2005.

[36] E. Zitzler, M. Laumanns, and L. Thiele. Spea2: Improving the strength pareto evolutionary algorithm. Technical Report 103, ETH, Zurich, 2001.

[37] E. Zitzler and L. Thiele. Multi-objective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, 1999.