Hybrid algorithms for recommending new items in personal TV

Fabio Airoldi
Politecnico di Milano - DEI
P.zza Leonardo da Vinci, 32
Milano, Italy
fabio.airoldi@gmail.com

Paolo Cremonesi
Politecnico di Milano - DEI
P.zza Leonardo da Vinci, 32
Milano, Italy
paolo.cremonesi@polimi.it

Roberto Turrin
Moviri - R&D
Via Schiaffino, 11
Milano, Italy
roberto.turrin@moviri.com

ABSTRACT

Recommending TV programs in the interactive TV domain is a difficult task since the catalog of available items is very dynamic, i.e., items are continuously added and removed.

Despite recommender systems based on collaborative filtering typically outperform content-based systems in terms of recommendation quality, they suffer from the *new item* problem, i.e., they are not able to recommend items that have few or no ratings.

On the contrary, content-based recommender systems are able to recommend both old and new items but the general quality of the recommendations in terms of relevance to the users is low.

In this article we present two different approaches for building hybrid collaborative+content recommender systems, whose purpose is to produce relevant recommendations, while overcoming the new item issue. The approaches are tested on the implicit ratings collected from 15'000 IPTV users over a period of six months.

1. INTRODUCTION

Interactive television allows providers to deliver to their costumers a huge amount of digital content. Since discovering interesting items can be difficult in such wide collections, recommender systems are used to provide the users with personalized lists of items that they may like.

Recommendations are based on the user preferences, referred to as ratings, that the system has gathered either explicitly (typically in the $1 \ldots 5$ scale) or implicitly (typically in binary format: 1 if the user likes an item, 0 otherwise). In the interactive TV (iTV) domain ratings are often implicit [2, 13, 12], inferred by tracking the users' activity (e.g., the purchased movies or the watched TV programs).

Item catalogs in the settings of iTV are intrinsically subject to frequent modifications: new programs are usually added as soon as they are available and old content becomes no longer available. We can identify:

(i) A set of items that are repeated over time, such as television seasons, weekly talk shows, or reruns of movies. We can assume that a number of users have watched these items, thus providing implicit ratings.

(ii) A set of items that are shown for the first time, such as the first showing of a movie. We can assume that no ratings have been collected about these items.

Most recommender systems are based on collaborative filtering algorithms, i.e., they recommend items on the basis of the preferences of users similar to the target user. However, since no ratings have been collected for new programs, they cannot be recommended by collaborative recommender systems. This issue is known as the *new-item* problem.

Alternatively to collaborative filtering, recommender systems can implement content-based filtering algorithms. Since content-based approaches base their predictions upon the description of TV programs in terms of features - such as genre and actors - they are not influenced by the lack of ratings. Unfortunately, collaborative algorithms have been systematically proven to outperform content-based algorithms in terms of recommendation quality, measured by standard accuracy (e.g., recall and precision) and error metrics (e.g., RMSE and MAE). As an example, if we consider the state-of-the-art recommender algorithms implemented in [2], in our experiments the collaborative approach reaches a recall equals to 19%, while the content-based approach does not go over 3%.

In the settings of iTV, we would like to build a recommender system not affected by the new-item problem, but with a quality comparable to collaborative filtering. Several *hybrid algorithms* have been proposed in the literature merging into a unique algorithm both content-based and collaborative filtering. Some of them have been even proven to outperform base collaborative recommender systems in therms of quality. However, the proposed solutions are rarely used in production environments mainly because of scalability issues. Furthermore, most approaches are designed to work in the case of explicit, non-binary ratings, without any particular focus on the new-item problem.

In this work we present two families of hybrid collaborative+content algorithms which are specifically designed to respect the requirements of commercial real-time recommender systems. We take into account state-of-the art collaborative and content-based recommender algorithms typically used in the presence of implicit, binary ratings. Each hybrid solutions is composed by one collaborative and one content-based algorithm.

The main idea behind the first family of hybrid algorithms is to augment the existing ratings used for training the collaborative algorithm with additional ratings estimated with the content-based algorithm. Diversely, the second family of hybrid algorithms merges together the item-to-item similarities computed by the collaborative and the content-based algorithms.

As a comparison, we also implemented a state-of-the-art and a trivial hybrid algorithm. The latter simply mixes in a unique recommendation list the items recommended with

the collaborative and those recommended with the content-based algorithm.

The recommendation quality of the hybrid algorithms has been evaluated in terms of recall against the quality of baseline algorithms on the dataset implicitly collected by an IPTV provider over a period of six months. A specific testing methodology has been designed in order to evaluate the quality of recommender algorithm in presence of new items.

In Section 2 we present an overview of the existing panorama regarding hybrid recommender systems, with particular focus on the algorithms that are most promising in the dynamic iTV scenario. In Section 3 we describe the reference state-of-the-art algorithms we included in our experiments. In Section 4 we illustrate the new algorithms that we developed. Section 5 details the dataset we used for the evaluation and on the testing methodologies we adopted. Section 6 shows and discusses the results we obtained. Finally, Section 7 draws the conclusions and leads on some possible future work.

2. RELATED WORK

Recently, several television operators have considered the integration of a recommender system into their architectures in order to provide personalized content (e.g., [2]).

The need for recommender systems in TV applications is motivated by the fact that users generally appreciate to receive personal suggestions generated according to their dynamically updated profiles, as shown in [26]. The same study reported also that customers prefer to be recommended with items similar to the ones that they already rated, but also with items that their friends have enjoyed.

Recommender systems used to deliver personalized TV experiences are typically content-based [36], collaborative [35] or hybrid solutions [34].

In the next paragraphs we summarize the limitations of non-hybrid recommender systems and we present an overview of the most interesting existing algorithms suitable personalized TV purposes.

2.1 Drawbacks of standard algorithms

The two main families of recommender algorithms are the content-based and the collaborative filtering.

The former are based on the analysis of the content of items (e.g., genre and actors). On the contrary, the latter suggest items on the basis of the preferences of users similar to the target user.

Collaborative algorithms have recently received much more attention than content-based solutions. The main reason is that they usually reach a higher recommendation quality than content-based systems [17, 8]. Furthermore, while collaborative algorithms can be easily implemented in any domain, content-based system are much more complex because they require to analyze the items' content (e.g., parsing textual data) [23].

However, since collaborative algorithms are based on the user ratings they have the following major drawbacks [1]:

- New-item. Collaborative filtering is particularly affected by the new-item problem, being not able to recommend items that have received few or no ratings because the system does not have enough information.
- Popularity bias. Collaborative filtering is biased towards the most popular items, i.e., items which have been rated by many users are more likely to be recommended than items that have few ratings.

2.2 Hybrid algorithms

During the last years several approaches have tried to overcome to the drawbacks of single recommender approaches by combining them into new *hybrid* recommender algorithms, from simple implementations (e.g., [34, 25, 10, 6]) up to very complex algorithms, such as the BellKor solution winning the Netflix prize [3], which combines predictions from 107 different baseline recommender systems [4]. The idea of merging multiple predictors has been often used in the settings of machine learning to improve the classification accuracy (e.g., [21]).

Burke performed an extensive survey and classification of hybrid recommender systems [6, 7].

Mixed algorithms present simultaneously the items recommended by two different recommender algorithms. As an example, Smyth and Cotter [34] show in the same interface recommendations generated by a content-based and a collaborative algorithm.

In case the confidence of a prediction is measurable, it is possible to implement a *switched* algorithm, that selects the best algorithm to use according to some confidence estimates. For instance, the system 'NewsDude' [5] combines a content-based nearest-neighbor recommender, a collaborative recommender, and naïve Bayes classifier. Ratings are predicted by the algorithm which shows the highest confidence and then the user is recommended with the items that have the highest predictions.

Weighted algorithms compute a linear combination of the ratings predicted by two (or more) recommender algorithms. A similar approach has been proposed by Mobasher et. al. [25], that linearly combines item-to-item similarities generated by different recommender algorithms. Differently from other hybrid solutions, this method can be used on implicit datasets (see Section 3.3 for further details).

Meta-level recommender systems use the model produced by an auxiliary algorithm for training the primary algorithm. As an example, the restaurant recommender proposed by Pazzani in [28] builds a feature-based model of the users using content-based information. This model is then used by a collaborative algorithm to compute recommendations.

Melville et al. propose a *feature-augmentation* algorithm denoted "Content boosted collaborative filtering" (CBCF) [24], which Burke's survey [6, 7] reports as one of the best algorithms. Their approach basically consists in creating 'augmented user profiles' by adding 'pseudo-ratings' to original user profiles prior to generating recommendations. Pseudo-ratings are generated using a content-based naïve Bayes classifier and can be interpreted as the ratings that users would give to unrated items, given the items' features. Rating prediction is computed with a variant of the user-based collaborative approach, where user-to-user similarities are computed as the Pearson correlation coefficient between the original user profiles and the augmented user profiles and the weights assigned to pseudo-ratings depend on the number of rated and co-rated items for each user. Melville et. al. reported that their algorithm performed better than the content-based naïve Bayes and the user-based collaborative algorithms in terms of MAE (Mean Absolute Error). They also showed that their algorithm is less susceptible to problems induced by sparsity: at 99.9% sparsity their algorithm

performed exactly like the content-based baseline.

Regardless several hybrid algorithms have been brought to the scientific attention during the last few years, there is no exhaustive literature on hybrid recommender algorithms studied to alleviate the new-item problem, with the only notable exception being the work of Schein et. al. [33, 32]. They describe a *generative* algorithm with the specific purpose to have high performance in cold-start situations. In their work they present a multi-aspect probabilistic model which is used to compute the probability that an item is liked or not by a user. The aspects include collaborative data as well as content-based data and one or more latent aspect. The hidden aspects are used to model the hypothesis that, as an example, a user liked a specific movie because of some particular, latent motivation. This kind of algorithms, along with association rule mining, neural networks and Boltzman machines, even if promising, are however difficult to integrate in existing systems, as they use a completely different approach from traditional collaborative filtering and do not properly scale with the number of users and items.

Also, hybrid algorithms are subject to a series of problematics. First, many of them (e.g., weighted hybrids) need to use underlying algorithms that are able to perform rating predictions. This makes them useless on implicit domains like iTV applications: algorithms which rely on predicting ratings strictly need explicit ratings to work properly as they are usually optimized with respect to error metrics (such as MAE or RMSE - e.g., [22]). However, the presence of only implicit, binary ratings (as in our datasets) does not allow to use this set of algorithms since we do not have proper ratings.

Additionally, some of the algorithms have scalability issues. As an example, CBCF [24] requires augmented user profiles in order to produce recommendations. Since it is impossible to store the augmented URM (it would be a full matrix), they need to be generated at real-time. However, this requires to compute the pseudo ratings for all the user in the neighborhood using a naive bayes classifier. Using the neighborhood size suggested by Melville et. al., this would mean generating 30 recommendation lists using the bayesian classifier just to produce one hybrid recommendation list.

Other simpler hybrid algorithms belonging to the weighted, switching, and mixed categories, need to generate at real-time a number of recommendation lists which is equal to the number of the underlying baseline recommenders. The consequence of this is that even the simplest algorithm belonging to those categories (e.g., the interleaved approach described in Section 3.3) would have a halved throughput with respect to a non-hybrid solution.

The only state-of-the-art hybrid approach with a complexity comparable to non-hybrid algorithms is SimComb [25]; however, we did not find any reference of this solution applied to implicit datasets.

3. STATE-OF-THE-ART ALGORITHMS

Recommender algorithms usually collect user ratings in a user-by-item matrix, from here on referred to as User Rating Matrix (URM) and denoted by $\mathbf{R}$, where r_{ui} is the rating given by user u to item i. The u-th row of this matrix is denoted by $\mathbf{r}_u$ and represents the profile of user u.

The URM is typically very sparse (users rate, on average, only a few items). In the settings of iTV we assume to have only implicit feedbacks, so that the URM contains binary ratings, where the value 1 and 0 indicate, respectively, whether the TV program has been watched or not.

Collaborative filtering algorithms analyze the URM to identify similarity between either users (user-based) or items (item-based), or hidden relations between users and items (latent factors). Item-based collaborative algorithms have been proved to outperform user-based approaches in terms of scalability and recommendation quality [1]. They discover item-to-item similarities using metrics such as the cosine or the adjusted cosine similarity [30]. Rating prediction for a specific item is computed by considering the ratings given by the target user to items similar to it (denoted *neighbors*). Finally, latent factors collaborative algorithms are typically based on singular value decomposition (SVD) to extrapolate underlying relations between users and items [31, 22].

Content-based recommender systems base recommendation on the features extracted from the items' content, such as: the actors, the genre, and the summary of a movie. The typical bag-of-word approach represents items as vectors of features stored into a feature-by-item matrix, which is referred to as Item Content Matrix (ICM) and denoted by $\mathbf{W}$. Items' content requires to be processed in order to identify terms (*tokenizing*), to remove useless words (*stop words*), to normalize words (*stemming*), and to weight each feature (*weighting*). As for the latter, weights are usually computed using the TF-IDF metric [29]. Users can be represented as feature vectors, composed by the features of the movies they have watched. Rating prediction can be computed as the similarity between user and item feature vectors.

In the following section we describe the state-of-the-art recommender algorithms we have taken into consideration. They comprise two collaborative filtering algorithms, one content-based algorithm, and two hybrid algorithms.

The algorithms we have considered are designed to be used with implicit, binary datasets. Furthermore, the algorithms share the property of not requiring any user-specific parameters to be learned in advance. As a consequence, these algorithms are able to recommend any user at real-time on the basis of his/her most recent ratings.

The state-of-the-art collaborative and content-based algorithms will be used in Section 4 for defining two families of hybrid recommender algorithms, each one combining one content-based and one collaborative filtering: *filtered feature augmentation* and *similarity injection*.

3.1 Collaborative

In the following we present one item-based (NNCosNgbr) and one latent factor (PureSVD) state-of-the-art collaborative algorithms suitable for predicting top-N recommendations in the case of binary ratings.

Non-normalized cosine KNN (NNCosNgbr).

This is a state-of-the-art item-based collaborative filtering algorithm described in [20]. Note that in the case of binary ratings we cannot compute similarity metrics such as the Pearson coefficient and the Adjusted Cosine, thus the item-item similarity has been measured as the cosine similarity.

Rating prediction for item i has been computed by summing up the similarities between item i and its neighbors, where the set of neighbors - denoted by $\mathcal{D}^k(u;i)$ - has been limited to the k most similar items. This approach is usually known as knn (k nearest neighborhood). Rating prediction

is computed as:

$$\hat{r}_{ui} = \sum_{j \in \mathcal{D}^k(u;i)} r_{uj} \cdot s_{ij}^{\text{CF}} \quad (1)$$

where s_{ij}^{CF} represents the cosine similarity between item i and j, computed as[1]:

$$s_{ij}^{\text{CF}} = \frac{\mathbf{r}_i \cdot \mathbf{r}_j^\intercal}{\|\mathbf{r}_i\|_2 \cdot \|\mathbf{r}_j\|_2} \quad (2)$$

This algorithm has been proved to have a good quality, as shown in [11] and [15].

PureSVD.

PureSVD is a latent factor collaborative algorithm, proved to grant high recommendation quality in terms of recall [11].

Let f denote the number of latent factors used for representing users and items. Typical values of f are in the range $[50, 300]$. SVD allows to factorize the URM into three f-dimensional matrices - $\mathbf{U}$, $\mathbf{\Sigma}$, and $\mathbf{Q}$ - and to compute a f-rank approximation of the original URM:

$$\mathbf{R}_f = \mathbf{U} \cdot \mathbf{\Sigma} \cdot \mathbf{Q}^\intercal \quad (3)$$

where $\mathbf{U}$ and $\mathbf{Q}$ are orthonormal matrices and $\mathbf{\Sigma}$ is diagonal. User u and item i are represented by f-dimensional vectors, respectively: $\mathbf{p}_u$ and $\mathbf{q}_i$. Rating prediction is computed using the inner product:

$$\hat{r}_{ui} = \mathbf{p}_u \cdot \mathbf{q}_i^\intercal \quad (4)$$

If we define $\mathbf{P} = \mathbf{\Sigma} \cdot \mathbf{U}$, we can observe that, since $\mathbf{U}$ and $\mathbf{Q}$ are orthonormal, $\mathbf{P} = \mathbf{R} \cdot \mathbf{Q}$ and (4) can be rewritten as:

$$\hat{r}_{ui} = \mathbf{r}_u \cdot \mathbf{Q} \cdot \mathbf{q}_i^\intercal \quad (5)$$

Finally, observe that the inner product $\mathbf{q}_i \cdot \mathbf{q}_j^\intercal$ represents the similarity between items i and j.

3.2 Content-based

Differently to collaborative filtering, content-based recommender systems rely only on content features. In the following we describe LSA [2], a well-known approach based on SVD.

Latent Semantic Analysis (LSA). Latent Semantic Analysis [16, 19, 2] uses SVD to reduce the dimensionality of the ICM and to capture underlying relations - like synonymy - between items. The weights in the ICM are computed using the TF-IDF metric[29]. Let l be the number of factors - typically referred to as *latent size* - to be used for representing items' features. The ICM can be factorized and approximated as:

$$\mathbf{W}_l = \mathbf{Z} \cdot \mathbf{\Lambda} \cdot \mathbf{Y}^\intercal \quad (6)$$

Let us define $\mathbf{B} = \mathbf{Y}\mathbf{\Lambda}$. Similarity between items i and j is denoted by s_{ij}^{CBF} and can be computed as the cosine between vectors $\mathbf{b}_i$ and $\mathbf{b}_j$:

$$s_{ij}^{\text{CBF}} = \frac{\mathbf{b}_i \cdot \mathbf{b}_j^\intercal}{\|\mathbf{b}_i\|_2 \cdot \|\mathbf{b}_j\|_2} \quad (7)$$

and rating prediction is computed as:

$$\hat{\mathbf{r}}_{ui} = \mathbf{r}_u \cdot \mathbf{B} \cdot \mathbf{b}_i^\intercal \quad (8)$$

[1] $\mathbf{x} \cdot \mathbf{y}$ denotes the inner product between vectors $\mathbf{x}$ and $\mathbf{y}$, and $\|\mathbf{x}\|_2$ denotes the Euclidean norm of vector $\mathbf{x}$.

3.3 Hybrid algorithms

In the following sections we present two families of hybrid recommender systems based on one collaborative filtering and one content-based filtering. The requirement for an algorithm to be used in the second solution is to allow to derive a similarity measure between items. Thus, all state-of-the art collaborative and content-based algorithms presented in Sections 3.1 and 3.2 can be used in the following hybrid approaches.

Interleaved.

This algorithm is a trivial hybrid implementation that forms the recommendation list to suggest to the target user by alternating, in turn, one item predicted by the collaborative algorithm and one predicted by the content-based algorithm.

SimComb.

Mobasher et. al. [25] developed a weighted hybrid algorithm in which item-to-item similarity values are computed as the linear combination between content-based and collaborative similarities:

$$c_{ij} = \alpha \cdot s_{ij}^{\text{CBF}} + (1 - \alpha) \cdot s_{ij}^{\text{CF}} \quad (9)$$

where s_{ij}^{CBF} and s_{ij}^{CF} are computed, respectively, using (2) and (7). Finally, rating prediction can be computed using (1), where s_{ij}^{CF} is to be replaced with c_{ij}. We refer to this algorithm as simComb.

4. PROPOSED ALGORITHMS

The algorithms we present in the following sections are designed to overcome some of the limitations of existing algorithms. Our first priority was to design hybrid recommender systems able to work on implicit, binary ratings, and to grant good recommendation quality even when only content information is available (new items). In addition, we also focus on scalability in order to design algorithms to be used in real iTV scenarios.

In the next two sections we present two families of hybrid solutions: *Filtered Feature Augmentation* and *Similarity Injection Knn.*

4.1 Filtered Feature Augmentation (FFA)

Filtered Feature Augmentation is a feature augmentation algorithm mainly inspired by CBCF (Content Boosted Collaborative Filtering) [24]. Differently from Mellville's solution, our approach

1. does not need any user-specific parameter to be learned;
2. allows to use different content-based and collaborative filtering;
3. computes item-item similarities on the basis of the ratings augmented with pseudo-ratings derived from the content-based filtering, but use the original user profiles for predicting ratings (as opposed to computing item-item similarities using the original URM and augmenting user profiles for predicting ratings).

Figure 1 shows the learning process: the CBF trains the content-based algorithm and computes the pseudo-ratings for unknown rating values to be added to the original URM.

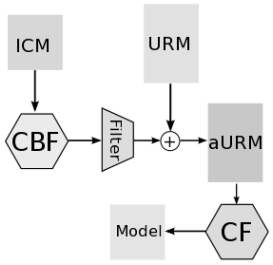


Figure 1: Model generation for FFA. The CBF stage comprises content-based model generation and recommendations for every item.

The augmented URM (namely, aURM) is used as input for the collaborative filtering.

Since adding all pseudo-ratings would lead to a dense, very large augmented URM, the filter selects only the most relevant pseudo ratings. In our experiments we used two different filters: a simple one which excludes all the pseudo-ratings which are lower than a fixed threshold (*FFAt*) and a more sophisticated one which use the Gini impurity measure [14] in order to add both high and low pseudo ratings to increase the intrinsic information to the item profiles (*FFAg*). The Gini coefficient is defined as:

$$\text{Gini}(\mathbf{v}) = 1 - \sum_{x \in \mathbf{v}} p_x^2 \tag{10}$$

where p_x is the probability of x in $\mathbf{v}$. In our specific case $\mathbf{v}$ is an item profile (i.e., a column of the URM) and $p_x = \frac{n_x}{n}$, where n_x is the number of ratings equal to x in $\mathbf{v}$ and n is the number of ratings in $\mathbf{v}$. When $\text{Gini}(\mathbf{v}) = 0$, $\mathbf{v}$ is pure (and brings almost no information). As we want to add informative pseudo-ratings, the filter let only pass the pseudo-ratings that increment the most the Gini index for each item. This is done until at least g pseudo ratings are added to each item-profile. The value of g depends on the number of original ratings for each user profile (denoted by n):

$$g = \begin{cases} n_{\min} - n + (h \cdot n) & \text{if } n \leq n_{\min} \\ \frac{(h \cdot n_{\min}^2)}{n} & \text{otherwise} \end{cases} \tag{11}$$

where $n_{\min}$ and h are parameters. In our experiments we used the average number of ratings as the value for $n_{\min}$ and 0.3 for h.

Rating prediction has been computed by using (1), where r_{ui} are the the original user ratings, and s_{ij}^{CF} is the similarity between items i and j computed using (2) on the augmented URM.

4.2 Similarity Injection Knn (simInjKnn)

Similarity Injection Knn builds a model using item-item similarities obtained by one collaborative and one content-based.

We first compute item-item similarities s_{ij}^{CF} using collaborative filtering, retaining only the k most similar items. Similarly, we compute item-item similarities s_{ij}^{CBF} using content-based filtering.

The similarities are later merged into a unique item-item similarity matrix $\mathbf{S}$, by adopting a two-step process:

Table 1: Dataset characteristics: number of different values for each type of feature (a) and number of items for each language (b).

(a) Types of features

Type	#values
actors	693
categories	8
directors	88
genres	16
languages	3

(b) Languages

Language	#items
German	703
French	84
Italian	7

- all the elements s_{ij}^{CF}'s are copied into $\mathbf{S}$. Thus, each items is filled with k similarity values deriving from the collaborative filtering.
- the elements s_{ij}^{CBF}'s are later inserted into the corresponding empty (e.g., zeros) elements of S in such a way to have, for each item, a set of k additional similarity measures deriving from the content-based filtering.

At the end of the construction, each item will have a total of $2k$ similar items:

- the k-most collaborative-based similar items
- the k-most content-based similar items not appearing in the previous set

.

Exactly half the similarities come from the content-based technique and half from the collaborative technique.

The framework allows different combinations of item-based collaborative and content-based filtering. In this work we present the results based on LSA as content-based algorithm and NNCosNgbr as collaborative filtering, since they reached the highest quality in terms of recall. Thus, s_{ij}^{CF} and s_{ij}^{CBF} have been computed using (2) and (7), respectively. Finally, rating prediction is estimated with (1), where s_{ij}^{CF} is replaced by s_{ij}.

5. DATASET AND EVALUATION METHODOLOGY

We evaluated our algorithms using a variant of the methodology described in [22, 11], designed to evaluate the performance of recommender algorithms with focus on the *new-item* problem.

We used the dataset provided by an IPTV provider[2]. It contains 25765 binary ratings implicitly collected over a period of six months given by 15563 users on 794 items (0.0021 sparsity). Content-based features comprise: actors, directors, category (e.g., movie, series, documentary...), title, genres, summary and language, for a total of 11291 features and an average of 18 features per item. The main characteristics of the available features are summarized in Table 1.

The evaluation of recommender systems is typically performed by using either error metrics or accuracy metrics

[2]Ratings refers to the dataset TV2 available at http://home.dei.polimi.it/cremones/memo/

[18]. Since error metrics - such as MAE (mean absolute error) and RMSE (root mean square error) - rely on computing the error between actual and predicted ratings, they cannot be measured on implicit, binary datasets where this information is not available [20, 15].

For such reason we focus on accuracy metrics, that estimate the fraction of relevant items which are actually recommended (recall) or the fraction of the recommended items that are actually relevant (precision). In addition, recent works [11, 27] consider accuracy metrics as more suitable, with respect to error metrics, for evaluating the top-N recommendation task [18], i.e., the capability of the recommender system to suggest very limited lists of items that are likely to be of interest for the users.

The standard definition of recall - which is typically used in the settings of information retrieval - is:

$$\text{recall} = \frac{|\text{relevant} \wedge \text{retrieved}|}{|\text{relevant}|} \tag{12}$$

Usually, in the settings of recommender systems, the set of relevant items is composed by items positively rated (e.g., if the rating is 5 out of 5). However, since we are facing with an implicit, binary dataset, in our evaluation we have considered - analogously to other works such as [15, 12, 13] - all rated items to be relevant, as we do not have any further information about the degree of user satisfaction.

5.1 Performance on new items

In order to specifically evaluate the impact of new items on the quality of the different algorithms we developed a testing methodology, that we refer to as *sliding window*, which is an extension of the evaluation methodology presented in [22, 11].

The original approach evaluates the quality of recommender algorithms by measuring the recall as a function of the number of items displayed to the user (N). The test consists in excluding a certain amount of ratings (*test set*) and using the remaining ratings to train the algorithms (*training set*). All available content-based features are used for training the content-based and hybrid algorithms. Once an algorithm has been trained with the ratings in the training set, each rating r_{ui} in the test set is tested as follows:

- we predict the score for all items unrated by user u
- we select the top-N items according to the estimated score
- if item i appears in the top-N recommendation list, we have a '*hit*', i.e., the system has correctly suggested a relevant item.

With respect to (12), the set of relevant items corresponds to the test set, while the set of relevant, retrieved items corresponds to the number of hits. Thus, recall can be rewritten as a function of N, i.e., the number of items displayed to users:

$$\text{recall}(N) = \frac{\#hits}{|\text{test-set}|} \tag{13}$$

Because of the high dataset sparsity, we have formed the test set by randomly selecting the 20% of ratings in order to have a significant number of samples.

This evaluation methodology is not able to measure the quality of the recommendations on new item problem. Therefore we have implemented some modifications.

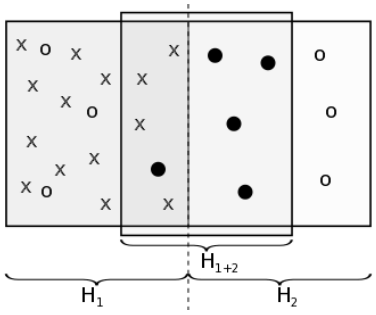


Figure 2: Sliding window. Blank circles represent ratings in the set $\mathcal{T}$, solid circles represent ratings in the test set, and x's represent ratings in the training set.

Let M denote the number of items in the dataset. We randomly divide items in two sets, $\mathcal{H}_1$ and $\mathcal{H}_2$, each one composed by $M/2$ items. Test set and training set are defined as a function of the percentage parameters β as follows:

- we define a set $\mathcal{T}$ by randomly selecting 20% of ratings in the URM.
- the *training set* is defined as the set of ratings related to items in $\mathcal{H}_1$, excluding all ratings in $\mathcal{T}$.
- we form a set $\mathcal{H}_{1+2}$ composed by $M/2$ items, $100 - \beta\%$ randomly extracted from set $\mathcal{H}_1$ and $\beta\%$ randomly extracted from set $\mathcal{H}_2$.
- *test set* is composed by the ratings in $\mathcal{T}$ related to items in $\mathcal{H}_{1+2}$.

Figure 2 schematically shows how the different sets are formed. Blank circles, solid circles, and x's refer to the ratings, respectively, in the set $\mathcal{T}$, in the test set, and in the training set.

For each value of β we have composed a training and a test set and we have computed the quality of the recommender algorithm in terms of recall, as defined in (13), where N has been set equal to 20. The test is the same as the one described for the original evaluation methdology. For each rating r_{ui} in the test set: (i) we predict the score for all items unrated by user u, (ii) we select the top-20 items according to the estimated score, and (iii) we verify if there has been a hit, i.e., if item i appears in the top-20.

Note that only ratings related to half the items are available for training the algorithms. The parameter β specifies the percentage of *new-items*, i.e., the percentage of items that do not have ratings in the training set and so that cannot be recommended by standard collaborative filtering. In our experiments we varied β from 0% to 100%.

Collaborative filtering is expected to be able to recommend only items in $\mathcal{H}_2$, so the higher β the lower the quality of the algorithm. When $\beta = 100\%$ we expect the quality of any collaborative filtering to be 0.

On the other hand, content-based algorithms are trained exclusively with content-based features, thus resulting totally independent from ratings included into the training set. We expect their quality not to be influenced by β. Finally, hybrid approaches can be training by the ratings related to half the items and with all available content-based features.

6. RESULTS AND DISCUSSION

In our test we considered the state-of-the-art recommender algorithms described in Section 3 and the hybrid approaches proposed in Section 4. As for the former, we included two item-based collaborative algorithms – PureSVD and non-normalized cosine (NNCosNgbr) – a content-based algorithm - Latent Semantic Analysis (LSA) – and two hybrid algorithms – interleaved and simComb. As for the latter, we included simInjKnn and two variants of filtered feature augmentation, referred to as FFAg and FFAt. FFAg uses a filter based on Gini's impurity measure, while FFAt uses a filter based on a fixed threshold, that has been set to 1, thus excluding all the pseudo-ratings lower than 1 (the number of pseudo-ratings added to the URM was the 27.5% of the original number of ratings).

The latent size l of LSA has been set to 300, while the number of features f in PureSVD is equal to 50. The neighborhood size k has been set equals to 200 for NNCosNgbr. As for SimComb, the coefficient α has been set to 0.3 as it empirically provides the better results. Finally, the neighborhood size k used for simInjKnn is 100.

Figure 3 shows the sliding window results, plotting the recall of the state-of-the-art (a) and the proposed hybrid (b) algorithms as a function of β, i.e., the percentage of new items. The recall has been computed assuming $N = 20$.

The most perceptible result is the very poor quality of the content-based algorithm, whose recall does not go over 3%, much lower than the best collaborative approach - NNCosNgbr - whose recall is about 25% when recommending only old items. In addition, we can observe that the three hybrid algorithms - simComb, FFAt, and simInjKnn - have a recall higher than collaborative and content-based state-of-the-art solutions. As for the other two hybrid algorithms: the interleaved approach confirms to have a poor quality as based on a very trivial technique, while the lower quality of FFAg is motivated by the fact that binary ratings do not bring enough information for the filter to work properly. However, the results of the sliding window test show that FFAt and SimInjKnn outperform the state-of-the-art algorithm SimComb when more than 40% of items is new. In addition, we can notice that recall drastically falls down when $\beta = 100\%$, with exception for the content-based algorithm that is not influenced by the presence of unrated items.

As a final observation, let us consider the performance implications of the proposed hybrid solutions. The Similarity Injection Knn (SimInjKnn) requires to build two item-to-item similarity matrices, which is the same effort as required for the interleaved algorithm. Filtered Feature Augmentation (FFAg and FFAt), on the other hand, requires to compute pseudo-ratings for each item. According to the number of items and users, this can be computationally costly. The main benefit of Filtered Feature Augmentation is that it is completely independent from the collaborative and content-based algorithms used, as opposed to Similarity Injection Knn, which strictly requires algorithms able to express item-item similarities.

7. CONCLUSIONS AND FUTURE WORKS

The new hybrid algorithms we proposed have been proved to improve the overall performance of the state-of-the-art algorithms in suggesting new items, though their performance is limited by the poor quality of content-based algorithms.

In future work we plan to implement between-subjects controlled experiments for a subjective evaluation of the proposed solutions. In fact, recent works have pointed out that objective evaluation of recommender systems (i.e., based on error and accuracy metrics) is not always aligned with the actual user perception of the recommendation quality, as measured via controlled experiments (e.g., [9, 27]).

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. *Knowledge and Data Engineering, IEEE Transactions on*, 17(6):734–749, 2005.

[2] R. Bambini, P. Cremonesi, and R. Turrin. *Recommender Systems Handbook*, chapter A Recommender System for an IPTV Service Provider: a Real Large-Scale Production Environment. to appear, Springer, 2010.

[3] R. Bell, Y. Koren, and C. Volinsky. The BellKor solution to the Netflix Prize.

[4] R. M. Bell and Y. Koren. Scalable Collaborative Filtering with Jointly Derived Neighborhood Interpolation Weights. *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 43–52, Oct. 2007.

[5] D. Billsus and M. J. Pazzani. User modeling for adaptive news access. *User Modeling and User-Adapted Interaction*, 10:147–180, February 2000.

[6] R. Burke. Hybrid recommender systems: Survey and experiments. volume 12, pages 331–370, Hingham, MA, USA, November 2002. Kluwer Academic Publishers.

[7] R. Burke. The adaptive web. chapter Hybrid web recommender systems, pages 377–408. Springer-Verlag, Berlin, Heidelberg, 2007.

[8] L. Candillier, K. Jack, F. Fessant, and F. Meyer. State-of-the-Art Recommender Systems. *IGI Global*, pages 1–22, 2009.

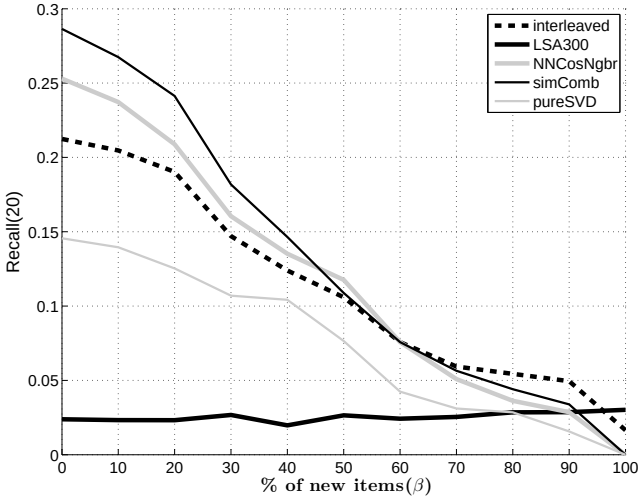
[9] L. Chen and P. Pu. A user-centric evaluation framework of recommender systems. In *In ACM Conference on Recommender Systems (RecSysŠ10), Workshop on User-Centric Evaluation of Recommender Systems and Their Interfaces (UCERSTIŠ10).*, pages 14–21, Barcelona, Spain, 2010. Sept. 26-30, 2010.

[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 *Proceedings of ACM SIGIR Workshop on Recommender Systems*, August 1999.

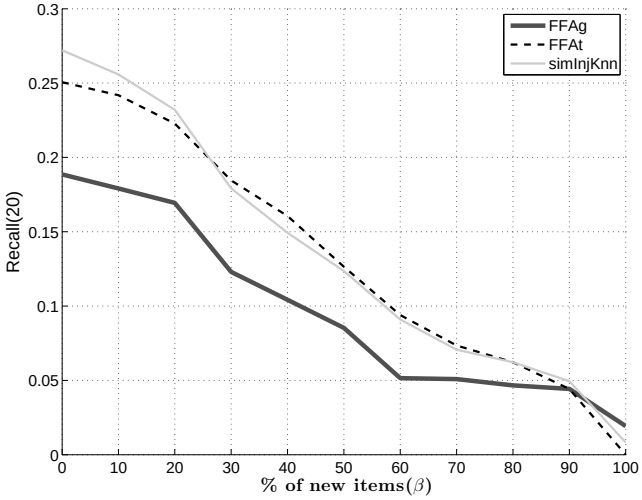
[11] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *RecSys*, pages 39–46, 2010.

[12] P. Cremonesi and R. Turrin. Analysis of cold-start recommendations in iptv systems. In *RecSys '09: Proceedings of the 2009 ACM conference on RecommenderSystems*, pages 1–4. ACM, 2009.

[13] P. Cremonesi and R. Turrin. Time-evolution of iptv recommender systems. In *Proc. of the 8th European Conference on Interactive TV and Video*, Tempere, Finland, June 2010. ACM.



(a) state-of-the-art algorithms



(b) new algorithms

Figure 3: Results of the sliding window test. Figure (a) refers to state-of-the-art approaches, while Figure (b) to the proposed hybrid solutions.

[14] B. de Ville. *Decision trees for business intelligence and data mining: using SAS enterprise miner*. SAS Publishing, 2006.

[15] M. Deshpande and G. Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, 22(1):143–177, 2004.

[16] G. W. Furnas, S. Deerwester, S. T. Dumais, T. K. Landauer, R. A. Harshman, L. A. Streeter, and K. E. Lochbaum. Information retrieval using a singular value decomposition model of latent semantic structure. pages 465–480, New York, NY, USA, 1988. ACM Press.

[17] M. A. Ghazanfar and A. Prugel-Bennett. A scalable, accurate hybrid recommender system. In *Proceedings of the 2010 Third International Conference on Knowledge Discovery and Data Mining*, WKDD '10, pages 94–98, Washington, DC, USA, 2010. IEEE Computer Society.

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

[19] P. Husbands, H. Simon, and C. Ding. On the use of singular value decomposition for text retrieval. Oct. 2000.

[20] G. Karypis. Evaluation of item-based top-n recommendation algorithms. In *Proceedings of the tenth international conference on Information and knowledge management*, CIKM '01, pages 247–254, New York, NY, USA, 2001. ACM.

[21] J. Kittler, M. Hatef, R. Duin, and J. Matas. On combining classifiers. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(3):226 –239, mar 1998.

[22] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434, New York, NY, USA, 2008. ACM.

[23] P. Lops, M. Gemmis, and G. Semeraro. Content-based Recommender Systems: State of the Art and Trends. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, chapter 3, pages 73–105. Springer US, Boston, MA, 2011.

[24] P. Melville, R. J. Mooney, and R. Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Eighteenth national conference on Artificial intelligence*, pages 187–192, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

[25] B. Mobasher, X. Jin, and Y. Zhou. Semantically Enhanced Collaborative Filtering on the Web. In *Proceedings of the 1st European Web Mining Forum (EWMF2003)*, pages 57–76, Sept. 2003.

[26] R. Navarro-Prieto, P. Rebaque-Rivas, and J. Hernández-Pablo. Recommending content for itv: what the users really want? In *Proceedings of the 8th international interactive conference on Interactive TV&Video*, EuroITV '10, pages 123–126, New York, NY, USA, 2010. ACM.

[27] S. N. A. P. R. T. Paolo Cremonesi, Franca Garzotto. Comparative evaluation of recommender system quality. In *CHI extended abstract on Human factors in computing systems*. ACM - to appear, 2011.

[28] M. J. Pazzani. A framework for collaborative, content-based and demographic filtering. *Artif. Intell. Rev.*, 13:393–408, December 1999.

[29] G. Salton, editor. *Automatic text processing*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1988.

[30] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. *10th Int. Conf. on World Wide Web*, pages 285–295, 2001.

[31] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Incremental singular value decomposition algorithms

for highly scalable recommender systems. *5th Int. Conf. on Computer and Information Technology (ICCIT 2002)*, pages 399–404, 2002.

[32] A. Schein, A. Popescul, L. Ungar, and D. Pennock. Generate models for cold-start recommendations. In *ACMSIGIR Workshop on RecommenderSystems.*, 2001.

[33] A. Schein, A. Popescul, L. Ungar, and D. Pennock. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2002)*, pages 253–260, 2002.

[34] B. Smyth and P. Cotter. A personalised tv listings service for the digital tv age. *Knowl.-Based Syst.*, 13(2-3):53–59, 2000.

[35] H. Zhang, S. Zheng, and J. Yuan. A personalized tv guide system compliant with mhp. *Consumer Electronics, IEEE Transactions on*, 51(2):731–737, May 2005.

[36] J. Zimmerman, K. Kurapati, A. L. Buczak, D. Schaffer, S. Gutta, and J. Martino. Chapter 5 tv personalization system design of a tv show recommender engine and interface.