

# A Recommender System for an IPTV Service Provider: a Real Large-Scale Production Environment

Riccardo Bambini, Paolo Cremonesi, and Roberto Turrin

WHITEPAPER *March 2010*

# A RECOMMENDER SYSTEM FOR AN IPTV SERVICE PROVIDER: A REAL LARGE-SCALE PRODUCTION ENVIRONMENT

Riccardo Bambini, Paolo Cremonesi, and Roberto Turrin

**Abstract** In this chapter<sup>1</sup> we describe the integration of a recommender system into the production environment of Fastweb, one of the largest European IP Television (IPTV) providers. The recommender system implements both collaborative and content-based techniques, suitable tailored to the specific requirements of an IPTV architecture, such as the limited screen definition, the reduced navigation capabilities, and the strict time constraints. The algorithms are extensively analyzed by means of off-line and on-line tests, showing the effectiveness of the recommender systems: up to 30% of the recommendations are followed by a purchase, with an estimated lift factor (increase in sales) of 15%.

## 1 INTRODUCTION

IP Television (IPTV) broadcasts multimedia content (e.g., movies, news programs, documentaries) in digital format via broadband Internet networks [11, 7]. IPTV services include scheduled television programs and video on demand (VoD) contents [12]. In the rest of the chapter we will generically refer to both scheduled television programs and video-on-demand contents as *items*.

In this chapter we present the integration of the Neptun's ContentWise recommender system in Fastweb, the first company in the world to have launched fully IP-based broadband TV services, in 2001. Fastweb serves hundreds of thousands of IPTV customers, with a catalog of thousands of multimedia contents. Since 2007 Fastweb is part of the Swisscom group. ContentWise recommender algorithms have been developed with the cooperation of the Computer Science Department at the Politecnico di Milano.

Riccardo Bambini  
Fastweb, via Francesco Caracciolo 51, Milano, Italy

Paolo Cremonesi  
Politecnico di Milano, p.zza Leonardo da Vinci 32, Milano, Italy

Roberto Turrin  
Neptun, via Durando 10, Milano, Italy

<sup>1</sup> This is a draft chapter from the book "Recommender Systems Handbook" edited by P.B.Kantor, F.Ricci, L.Rokach and B.Shapira, and published by Springer (ISBN: 978-0-387-85819-7). The book is available from Springer web site <http://www.springer.com/computer/ai/book/978-0-387-85819-7> or from Amazon <http://www.amazon.com/Recommender-Systems-Handbook-Paul-Kantor/dp/0387858199>

Differently from conventional television, IPTV allows an interactive navigation of the available content [5] and, in particular, IPTV allows to collect implicit usage data and explicit user preferences for providing a personalized user navigation. The user interacts with the IPTV system by means of a special electronic appliance, referred to as *set-top-box* (STB). There are substantial peculiarities of the STB that limit the user interaction: (i) users control the STB by means of a remote control, which is rather limited in the set of actions it allows to perform, (ii) the user interface is shown on a TV screen typically designed to be looked at from a distance larger than that between a PC and the user, and (iii) the system deals with multimedia content, whose navigation is not particularly fast, mainly because of the channel switching time. Differently from traditional e-commerce domains (e.g., Amazon, Netflix, iTunes, IMDB, Last.fm) where recommender systems have been exploited, IPTV recommender systems need to satisfy particular requirements:

- the list of proposed items has to be small because of the limited screen definition and the reduced navigation capabilities;
- the generation of the recommended items must respect very strict time constraints (few milliseconds) because TV's customers are used to a very responsive system;
- the system needs to scale up in a successful manner with both the number of customers and items in the catalog;
- part of the catalog is highly dynamic because of live broadcast channels.

The recommender system deployed in Fastweb generates recommendations by means of two collaborative algorithms (based on item-to-item similarities and dimensionality reduction techniques) and one content-based algorithm (based on latent semantic analysis). The recommender system selects the proper algorithm depending on the context. For instance, if the user is reading a movie synopsis, looking for movies with his preferred actors, the algorithm used is the content-based one. In order to respect the strict real-time requirements, the recommender system and the underlying algorithms follow a model-based approach and have been logically divided into two asynchronous stages, the batch stage and the real-time stage.

The input data of the whole architecture is composed by: (i) the item-content matrix and (ii) the user-rating matrix. The item-content matrix (ICM) describes the main attributes (metadata) of each item, such as the title of a movie, the set of actors and its genre(s). The user-rating matrix (URM) collects the ratings (i.e., preferences) of users about items. Ratings are mainly implicit, e.g., the system can detect if a user watched a program, without knowing explicitly the user's opinion about that program.

Before deploying the recommender system in production, extensive performance analysis has been performed by means of  $k$ -fold cross validation. The results suggest a 2.5% recall for the content-based algorithm, while the collaborative algorithms are able to reach a recall of more than 20%.

The recommender system has been released to production environment in October 2008 and is now available for one of the Fastweb VOD catalogs. The system is actually providing, on average, 30'000 recommendations per day, with peaks of almost 120 recommendations per minute during peak hours. On-line analysis shows that almost 20% of the recommendations are followed by a purchase from the users.

The estimated lift factor (i.e., increase in VOD sales) is 15%.

The rest of the chapter is organized as follows. Section 2 shows the typical architecture of an IPTV provider. Section 3 presents the architecture of the recommender system. Section 4 explains the recommender services implemented into the Fastweb IPTV architecture. Section 5 evaluates the quality of recommendations. Finally, Section 6 draws the conclusions.

## 2 IPTV ARCHITECTURE

IPTV, also called Internet Protocol Television, is a video service that delivers high quality traditional TV channels and on-demand video and audio contents over a private IP-based broadband network. From the end users perspective, IPTV looks and operates just like a standard TV service. The providers involved in deploying IPTV services range from cable and satellite TV carriers to large telephone companies and private network operators. IPTV has a number of unique features [5]:

**Support for interactive TV:** differently from conventional TV, where the communication is unidirectional, the two-way capabilities of IPTV systems allow the user to interact with the system.

**Time shifting:** IPTV permits the temporal navigation of programs (e.g., fast forward, pause and rewind) thanks to the Personal Video Recording (PVR), a mechanism for recording and storing IPTV content for later viewing.

**Personalization:** IPTV allows end users to personalize their TV viewing experience by letting them decide what they want to watch and when they want to watch it.

Figure 1 shows a generic IPTV system architecture that supports live broadcast TV channels (also called *linear channels*) and video on-demand (VOD). Broadcastm TV service consists in the simultaneous reception by the users of traditional TV channels either free-to-air or pay-per-view. Video on-demand service consists in viewing multimedia content made available by the service provider, upon request.

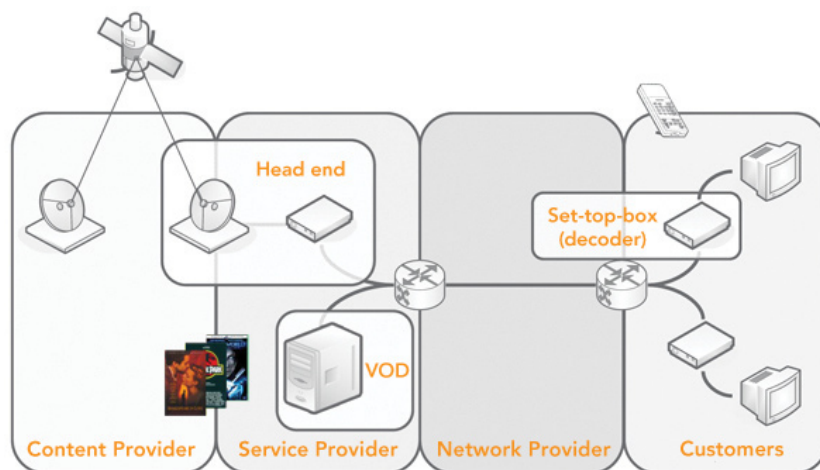


Fig. 1 Architecture of an IPTV system.

The IPTV data center (also known as the *headend*) receives linear channels from a variety of sources including terrestrial, satellite and cable carriers. Once received, a number of different hardware components, ranging from encoders and video servers, are used to prepare the video content for delivery over an IP based network. On-demand contents are stored in fast storage boxes (e.g., using solid-state disks).

The set-top-box (STB) is an electronic appliance that connects to both the network and the home television: it is responsible for processing the incoming packet stream and displaying the resulting video on the television. The user interacts with the STB by means of a hand-held remote control. The remote control gives the user access to additional features of the STB, such as the Electronic Program Guide (EPG), a listing of available channels and program for an extended time period (typically 36 hours or more).

## 2.1 IPTV search problems

To benefit from the rich set of IPTV channels and contents, users need to be able to rapidly and easily find what they are actually interested in, and do so effortlessly while relaxing on the couch in their living room, a location where they typically do not have easy access to the keyboard, mouse, and close-up screen display typical of desktop web browsing. However, searching for a live channel or a VOD content is a challenging problem for IPTV users [4].

When watching live television, users browse through a set of available channels until they find something interesting. Channel selection (zapping) involves two steps: (a) sampling the content to decide whether to continue or stop watching the channel, and (b) switching across multiple channels for repeated sampling, until a desired channel is found. The problem of quickly finding the right channel becomes harder as the number of channel offerings grows in modern IPTV systems. Moreover, IPTV channel switching time is not particularly responsive, compared to traditional TV, because of technological limitations [9].

When searching for VOD content, IPTV users generally have to either navigate a complex, pre-defined, and often deeply embedded menu structure or type in titles or other key phrases using an on-screen keyboard or triple tap input on the remote control keypad. These interfaces are cumbersome and do not scale well as the range of content available increases. Moreover, the television screens usually offer a limited resolution with respect to traditional personal computer screens, making traditional graphical user interfaces difficult to use.

This differs from traditional web-based domains (e.g., e-commerce web sites), where the content is textual, suited for information categorization and keyword-based seek and retrieval, and the input devices (keyboard and mouse) allow to point an arbitrary object on the screen and to easily enter text.

The integration of a recommender system into the IPTV infrastructure improves the user experience by providing a new and more effective way of browsing for interesting programs and movies. However, such integration has to deal with the following issues:

**User identification.** The STB is used indistinctly by all the components of a family, and the IPTV recommender system can not identify who is actually watching a certain program.

**Real-time requirements.** The IPTV recommender systems must generate recommendations within very strict real-time constraints (few milliseconds) in order to avoid a slow down of

the user navigation, already affected by the long channel switching time.

**Quality of content metadata.** Differently from web-based domains, content-based IPTV recommender algorithms makes use of low-quality metadata. This aspect is particularly evident with live channels, where new content is added every day at a very high rate, and the only available metadata that can be used to describe programs can be found in the EPG (electronic program guide).

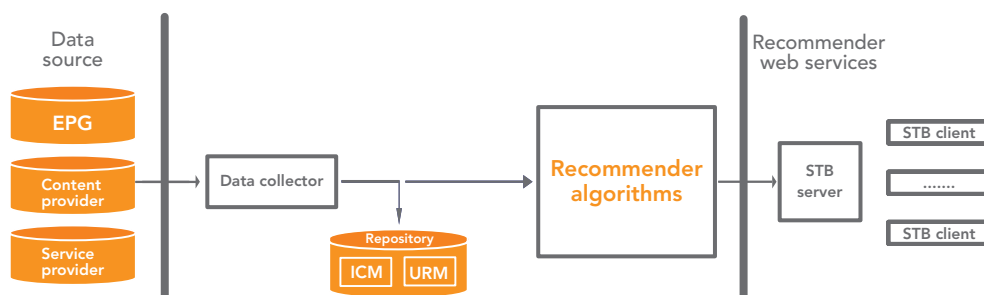
ecommmender algorithms makes use of low-quality metadata. This aspect is particularly evident with live channels, where new content is added every day at a very high rate, and the only available metadata that can be used to describe programs can be found in the EPG (electronic program guide).

### 3 RECOMMENDER SYSTEM ARCHITECTURE

The architecture of the Fastweb recommender system is shown in Figure 2.

#### 3.1 Data collection

The logical component in charge of pre-processing the data and generating the input of the recommender algorithm is referred to as *data collector*. The data collector gathers data from different sources, such as the EPG for information about the live programs, the content provider for information about the VOD catalog and the service provider for information about the users.



**Fig. 2** Architecture of the recommender system ContentWise.

The Fastweb recommender system does not rely on personal information from the users (e.g., age, gender, occupation). Recommendations are based on the past users' behavior (what they watched) and on any explicit preference they have expressed (e.g., preferred genres). If the users did not specify any explicit preferences, the system is able to infer them by analyzing the users' past activities.

An important question has been raised in Section 2: users interact with the IPTV system by means of the STB, but typically we can not identify who is actually in front of the TV. Consequently, the STB collects the behavior and the preferences of a set of users (e.g.,

the component of a family). This represents a considerable problem since we are limited to generate per-STB recommendations. In order to simplify the notation, in the rest of the paper we will refer to user and STB to identify the same entity. The user-disambiguation problem has been partially solved by separating the collected information according to the time slot they refer to. For instance, we can roughly assume the following pattern: housewives use to watch TV during the morning, children during the afternoon, the whole family at evening, while only adults watch TV during the night. By means of this simple time slot distinction we are able to distinguish among different potential users of the same STB.

Formally, the available information has been structured into two main matrices, practically stored into a relational database: the item-content matrix (ICM) and the user-rating matrix (URM).

The former describes the principal characteristics (metadata) of each item. In the following we will refer to the item-content matrix as  $\mathbf{W}$ , whose elements  $w_{ci}$  represent the relevance of characteristic (metadata)  $c$  for item  $i$ . The ICM is generated from the analysis of the set of information given by the content provider (i.e., the EPG). Such information concerns, for instance, the title of a movie, the actors, the director(s), the genre(s) and the plot. Note that in a real environment we can face with inaccurate information especially because of the rate new content is added every day. The information provided by the ICM is used to generate a content-based recommendation, after being filtered by means of techniques for PoS (Part-of-Speech) tagging, stop words removal, and latent semantic analysis [13].

Moreover, the ICM can be used to perform some kind of processing on the items (e.g., parental control).

The URM represents the ratings (i.e., preferences) of users about items. In the following we will refer to such matrix as  $\mathbf{R}$ , whose elements  $r_{pi}$  represent the rating of user  $p$  about item  $i$ . Such preferences constitute the basic information for any collaborative algorithm. The user rating can be either explicit or implicit, according to the fact that the ratings are explicitly expressed by users or are implicitly collected by the system, respectively.

Explicit ratings confidently represent the user opinion, even though they can be affected by biases [1] due to: user subjectivity, item popularity or global rating tendency. The first bias depends on arbitrary interpretations of the rating scale. For instance, in a rating scale between 1 and 5, some user could use the value 3 to indicate an interesting item, someone else could use 3 for a not much interesting item. Similarly, popular items tend to be overrated, while unpopular items are usually underrated. Finally, explicit ratings can be affected by global attitudes (e.g., users are more willing to rate movies they like).

On the other hand, implicit ratings are inferred by the system on the basis of the user-system interaction, which might not match the user opinion. For instance, the system is able to monitor whether a user has watched a live program on a certain channel or whether the user has uninterruptedly watched a movie. Despite explicit ratings are more reliable than implicit ratings in representing the actual user interest towards an item, their collection can be annoying from the user's perspective.

The current deployment of the Fastweb recommender system collects only implicit ratings, but the system is thought to work when implicit and explicit ratings coexist. The rating scale is between 1 and 5, where values less than 3 express negative ratings, values greater or equal to 3 express positive ratings. In absence of explicit information, the rating implicitly inferred by monitoring the user behavior is assumed to be positive (i.e., greater than 3).

In fact, whether a user starts watching a certain program, there must be some characteristic of this program appealing for the user (e.g., actors or genre). The fact that in well-know, explicit datasets, such as Netflix and Movielens, the average rating is higher than 3, motivates this assumption. The system treats differently live IPTV programs and VOD content:

**IPTV programs.** The rating is proportional to the percentage user play time (e.g., [8, 15]), i.e., the percentage of program the user has watched. Let us assume  $L$  is the program play time and  $t$  is the user play time. Play times less than 5 minutes are discarded. If a user watched the entire program the rating is 5, if the user watched 5 minutes the rating is 3, otherwise the rating is a value between 3 and 5 given by:

$$\hat{r} = 3 + 2 \frac{t-5}{L-5}, \quad 5 \leq t \leq L \quad (1)$$

where  $t$  and  $L$  are expressed in minutes.

At this early stage of the project, the main goal is not to define an accurate implicit rating mechanism, but rather, to filter out noisy information (e.g., TV channel zapping).

**VOD movies.** When watching a VOD movie, users explicitly request to buy and to pay for that movie. For that reason, independently from the user play time, when a user requests a VOD movie, the system assign an implicit ratings equals to 4.

As aforementioned, should Fastweb start collecting explicit ratings too, they will naturally coexist with implicit ratings in the URM.

The ratings stored in the URM, before being used by the recommender algorithms, are normalized by subtracting the constant value 2.5. This allows the algorithms to distinguish between positive and negative ratings, because values greater or equals to 2.5 (i.e., 3, 4, and 5) remain positive, while values less than 2.5 (i.e., 1 and 2) become negative. In the rest of the chapter we will assume that the recommender algorithms receive as input a normalized URM.

Finally, users can express *explicit preferences* about the content they would like to watch. For instance, by means of the graphical interface, a user can set his preferred actors. The content-based algorithm takes into consideration such information and biases the recommended movies toward the expressed preferences.

### 3.2 Batch and real-time stages

The recommender algorithms process the ICM and the URM described in Section 3.1 and they interface with the STB server by means of web services, as shown in Figure 3.

In order to respect the strict real-time requirements, the recommender system and the underlying algorithms follow a model-based approach [14, 3], i.e., they first develop a model of the user ratings and/or of the items, then they compute the recommendations. Consequently, the algorithms have been logically divided into two stages, *the batch* stage and the *real-time* stage:

- the batch stage creates a low dimensional representation (i.e., a model) of the input data. It is usually executed during the service off-peak hours, with a frequency which depends on the rate new items/users are added into the system (e.g., once a day);
- the real-time part uses the model in order to serve calls coming from the web services interface and satisfying the real-time constraints. The system output can be further constrained by post-processing, marketing rules (e.g., pushing up some movies, or filtering some channels).

The model repository makes the two stages asynchronous, e.g., while the real-time stage is recommending users by using a certain model, the batch stage can compute a new, updated model.

Despite such logical division of the recommending process, the model construction in real domains can still be challenging because of input data size and the related time and memory requirements. For this reason, we have implemented highperforming, parallel versions of the most demanding matrix operations, optimized for sparse and big matrices, such as: matrix-matrix and matrix-vector multiplication, matrix transposition, column/row normalization, and singular value decomposition (svd). In particular, svd has been used with two of the three recommender algorithms (one content-based and one collaborative), allowing to greatly reduce the space dimensionality, with benefits both in terms of memory and time complexity.

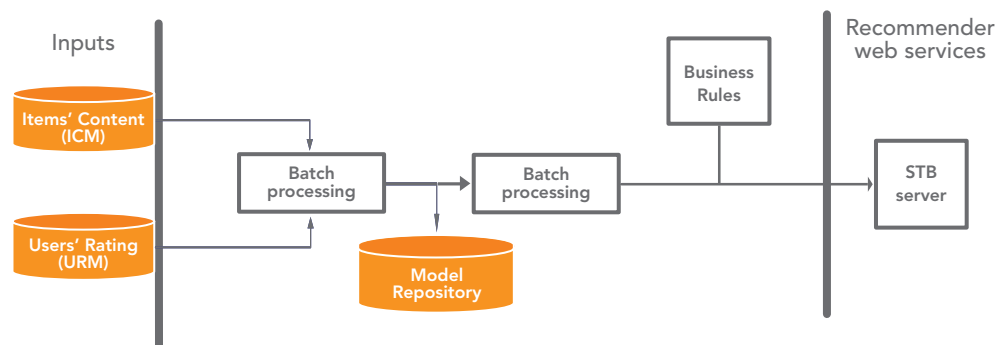


Fig. 3 Recommender system: batch and real-time stage.

As we will show in the following sections, by its own, svd defines a model of the data, cleaning up the data noise and strengthening the correlation among similar information.

Realistic datasets with millions of users and items can have in principle prohibitive memory requirements. Fortunately, matrices such as URM and ICM are typically very sparse. In fact, most of the users interact (e.g., rate or watch) with very few items compared with the size of the catalog (e.g., the average users have watched few dozens of movies in a catalog of thousands). Sparse matrices can be treated using very efficient representations. Note that, even though such matrices aren't sparse, we could have difficulties in maintaining the data in memory. For such reasons, we opted for a solution based on a sort of *memory virtualization*, similar to the swap capability of any operating systems. Differently from the operating system virtual memory, our virtualization policy is tailored ad-hoc for each matrix operation, in order to limit the data exchange between memory and storage.

## 4 RECOMMENDER SERVICES

This section presents the implemented recommender services and how they impact into the user interfaces and the IPTV architecture. The recommender system can generate both content-based and collaborative-based recommendations. As summarized in Figure 4, content-based algorithms are applied both to VOD and live TV domains, while collaborative algorithms are applied only to VOD. At the current stage of the integration, Fastweb is exposing the full set of recommender services to a selected set of beta test users before the effective release.





	VOD	Live TV
Item-based CF		
SVD-CF		
LSA-CB		

Fig. 4 Application of recommender algorithms to VOD and live TV.

The other customers have access to a reduced set of the recommender functionalities. An example of the user interface available by means of the STB is shown in Figure 5. The services released to the full customer base concern one of the catalog of VOD domain. Recommendations are provided by a content-based algorithm based on Latent Semantic Analysis. The content-based algorithm has been preferred for the first few months of activity because collaborative algorithms suffer from the cold-start problem. Moreover, collaborative recommenders need to record the behavior of users. This faces Fastweb with delicate legal questions that require, for instance, to obtain authorizations from customers for storing and managing their data, and to implement solutions to grant confidentiality and anonymity of such information.

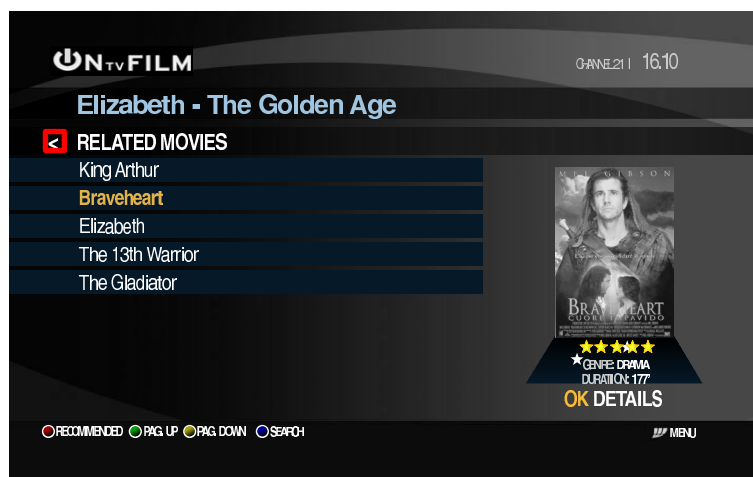


Fig. 5 Recommender system user interface

## 5 SYSTEM EVALUATION

In this section we first discuss the quality of the recommender system by means of accuracy metrics computed adopting an off-line testing. We later present some feedbacks from the on-line analysis of the recommender system.

The off-line tests are based on the views collected during 7 months of users' activity from one of the VOD catalogs. Figure 6 shows the evolution of the number of views. The average number of views per days is about 1600, with up to 3300 views during week-end days. Figure 7, 8, and 9 complete the analysis by showing the time evolution of, respectively, the number of active users, the number of active items, and the dataset density. Active users are users that have rated at least one item. Similarly, active items are items that have been rated by at least one user. The dataset density is the ratio between the number of ratings and the product of the number of active users and the number of active items. We can notice from Figure 9 that the trend is not monotone. In fact, when a new user watches her/his first item, we have one new active user, and the dataset decrease its density.

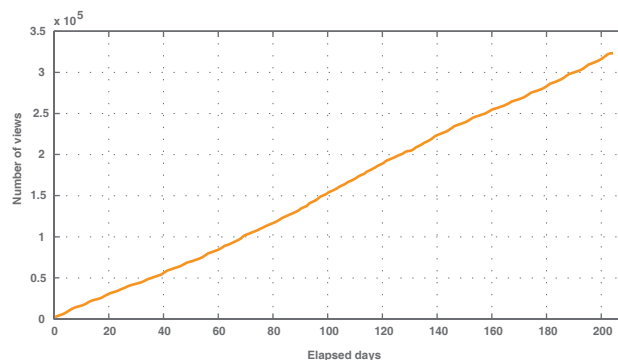


Fig. 6 Number of views collected during 7 months of users' activity from one of the VOD catalogs.

### 5.1 Off-line analysis

Typical approaches for recommender system evaluation are based either on error metrics (e.g., RMSE and MAE) [10] or classification accuracy metrics (e.g., recall, precision, and fall-out) [6, 2]. Since we only dispose of implicit ratings, expressing positive user interests, we are practically constrained in evaluating the quality of the system by means of accuracy metrics. To this end, Table 1 presents the recall of the three ContentWise algorithms<sup>2</sup>, respectively: LSA-CB, item-based-CF and SVD-CF.

Recall is often used in information retrieval, where it specifies the percentage of relevant items that have been retrieved by, for instance, a search engine. In our domain, recall indicates how many movies that users have effectively watched are recommended by the recommender algorithm. To this purpose, we follow a leaveone- out approach:

- for each user in the test set, we select one rated item

<sup>2</sup> To obtained a detailed report, please XXXX

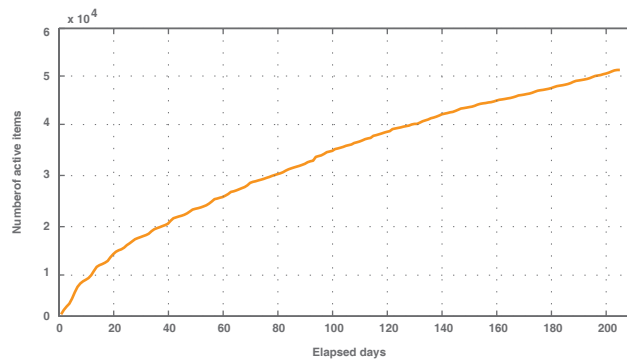


Fig. 7 Number of active users from the same VOD catalog.

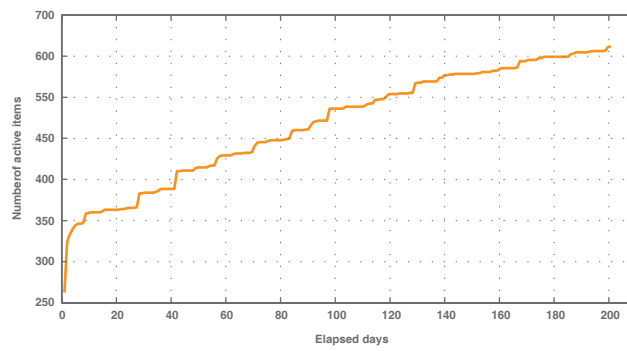


Fig. 8 Number of active items from the same VOD catalog.

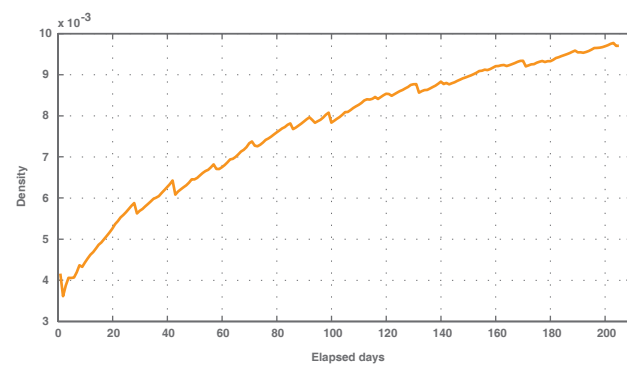


Fig. 9 Evolution of rating density in the same VOD catalog. Density is computed by considering the number of views (i.e., ratings) with respect to the number of active users and active items.

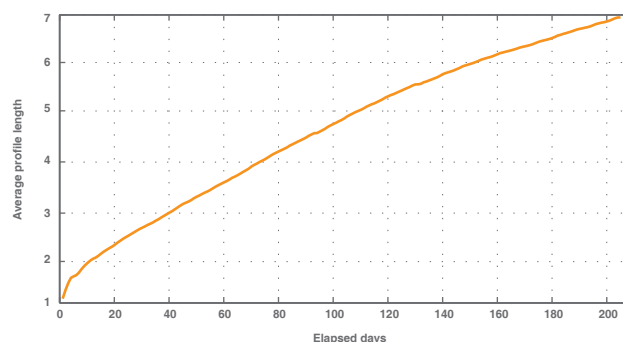
- the selected item is removed from the user profile, and we generate a recommendation for this modified user profile; items already rated by the user are filtered out from the recommendation list.
- if the removed item is recommended within the first 5 positions we have a *hit*, i.e., a movie that has been watched by a user has been recommended by the algorithm (accordingly to the Fastweb user interface, the recommended list is limited to 5 items);
- the process is repeated for each item and for each user.

The recall is the percentage of hits with respect to the number of tests.

The test set is selected differently according to the kind of algorithm. In fact, content-based algorithms build their model by using the ICM, and the test set can be the whole URM. On the other hand, the model of collaborative algorithms is based on the URM itself, so they have been evaluated with a 10-fold cross validation approach, i.e., we have randomly split the users into 10 folds and, in turn, one fold has been used as test set for computing the recall, while the remaining nine folds have been used to generate the model. Each test fold is analyzed by means of the leave-one-out approach. The reported results are the average recall among the 10 folds.

The table reports the recall of the recommender algorithms both after 3 months of activity and after 6 months of activity, showing the time evolution of the system. Furthermore, the quality of recommendation of the three algorithms are compared with a trivial algorithm, used only for comparison purposes: the *top-rated*. The *top-rated* algorithm is a basic collaborative algorithm that recommends to any user a fix list of items, ordered from the most popular to the less popular (discarding items already rated by the user).

Table 1 shows that, in some cases, the quality of recommendations after 6 months is lower than after 3 months. We would expect that, as the system collects ratings, it acquires more precise user profiles and the quality of recommendations should improve. However, after 3 months of activity there are 510 active items, while after 6 months we have 621 active items. In terms of probability, after 3 months an algorithm has to pick up 1 items among 510 candidates, while after 6 months the number of candidates is 621, as shown in Figure 8.



**Fig. 10** Time evolution of the average user profile length. Profile lengths are computed on users active in one of the VOD catalogs.

Algorithm	Recall	
	3 months	6 months
Collaborative Item-Based	14.0%	12.6%
Collaborative SVD	12.0%	12.0%
Content LSA	2.6%	2.6%
Top-rated	0.4%	1.0%

**Table 1** Recommendation quality in one of the VOD catalogs.

## 5.2 On-line analysis

In this section we integrate the previous results, obtained from an off-line analysis of the recommender algorithms, with an on-line analysis, i.e., we directly study the feedback on the running recommender system. As explained in Section 4, the reported data refer to the content-based algorithm applied on one of the VOD catalogs.

In order to empirically evaluate the recall, we assume that whether a user watches a movie after it has been recommended by the system, such movie is relevant for the user and this represents a *success* for the recommender system.

Let us define the *recommendation success*, which measure the number of movies that have been viewed within a certain time period after being recommended. Indicating with  $b(t)$  the recommendation success and with  $w(t)$  the number of movies watched by the same users within a time period  $t$  from a recommendation, we can compute an *empirical recall* as the percentage ratio between the recommendation success and the number of views:

$$\text{empirical recall}(t) = \frac{b(t)}{w(t)} \quad (2)$$

The empirical recall represents the percentage of views that have been triggered by the recommender algorithm. The specified indexes depend on the time period  $t$  that is taken into consideration after the recommendation has been provided to the user. Please note that a too long time period  $t$  could loose the dependency between the recommendation and the view. Table 2 shows the average quality of the system computed by monitoring the views within 2 hours, within 24 hours, and within 7 days from the recommendation. The reported results distinguish between popular and unpopular items.

From the table we can observe that the empiric recall is larger for unpopular movies with respect to popular movies. In fact, popular movies are already known by users, even without being suggested by the recommender system. For instance, either the user has already watched a popular movie (e.g., at cinema) or it is not interested in it.

2 hours	24 hours	7 days
17.0%	19.8%	24.70%

**Table 2** Average empiric recall on the considered VOD catalog. Results refer to three time periods after the recommendation (2 hours, 24 hours, and 7 days) and are separated between popular and unpopular movies.

## 6 CONCLUSIONS

The integration of the ContentWise recommender systems into the Fastweb architecture positively impacts both the customers and the service provider. Three major considerations derive from the on-line analysis, confirming the positive effects of the recommender system: (i) users prefer to browse the VOD catalog by means of the recommender interface, (ii) users tend to watch recommended movies within few hours, and (iii) users increase the number of watched movies.

Further experiments are currently running on the other catalogs of Fastweb, testing and tuning the quality of all the implemented recommender algorithms and monitoring the cold-start phase of the system in order to complete the release of recommender services. Other ongoing works are addressing the problem of accurately estimating implicit ratings from user behavior.

## REFERENCES

1. Bell, R.M., Koren, Y.: Scalable collaborative filtering with jointly derived neighborhood interpolation weights. 7th IEEE Int. Conf. on Data Mining pp. 43–52 (2007)
2. Cremonesi, P., Lentini, E., Matteucci, M., Turrin, R.: An evaluation methodology for recommender systems. 4th Int. Conf. on Automated Solutions for Cross Media Content and Multi-channel Distribution pp. 224–231 (2008)
3. Deshpande, M., Karypis, G.: Item-based top-n recommendation algorithms. ACM Transactions on Information Systems (TOIS) 22(1), 143–177 (2004). DOI <http://doi.acm.org/10.1145/963770.963776>
4. Geneve, U.D., Marchand-maillet, S.: Vision content-based video retrieval: An overview
5. Hand, S., Varan, D.: Interactive narratives: Exploring the links between empathy, interactivity and structure pp. 11–19 (2008)
6. Herlocker, J., Konstan, J., Terveen, L., Riedl, J.: Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems (TOIS) 22(1), 5–53 (2004)
7. Jensen, J.F.: Interactive television - a brief media history 5066, 1–10 (2008)
8. Kelly, D., Teevan, J.: Implicit feedback for inferring user preference: a bibliography. SIGIR Forum 37(2), 18–28 (2003). DOI <http://doi.acm.org/10.1145/959258.959260>
9. Lee, Y., Lee, J., Kim, I., Shin, H.: Reducing iptv channel switching time using h.264 scalable video coding. Consumer Electronics, IEEE Transactions on 54(2), 912–919 (2008). DOI 10.1109/TCE.2008.4560178

10. Powers, D.M.W.: Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness and Correlation (2000)
11. Rafey, R.A., Gibbs, S., Hoch, M., Gong, H.L.V., Wang, S.: Enabling custom enhancements in digital sports broadcasts pp. 101–107 (2001). DOI <http://doi.acm.org/10.1145/363361.363384>
12. Sun, J., Gao, S.: Iptv based on ip network and streaming media service station. MIPPR 2007: Remote Sensing and GIS Data Processing and Applications; and Innovative Multispectral Technology and Applications 6790(1), 67904Q (2007). DOI 10.1117/12.749611. URL <http://link.aip.org/link/?PSI/6790/67904Q/1>
13. Van Rijsbergen, C.J.: Information Retrieval, 2nd edition. Dept. of Computer Science, University of Glasgow (1979). URL [citeseer.ist.psu.edu/vanrijsbergen79information.html](http://citeseer.ist.psu.edu/vanrijsbergen79information.html)
14. Vozalis, E., Margaritis, K.: Analysis of recommender systems algorithms. Proc. of the 6th Hellenic European Conf. on Computer Mathematics and its Applications (2003)
15. Zhang, H., Zheng, S., Yuan, J.: A personalized tv guide system compliant with mhp. Consumer Electronics, IEEE Transactions on 51(2), 731–737 (2005). DOI 10.1109/TCE.2005.1468026