# Metadata in movies recommendation: a comparison among different approaches

Lucas Tobal Percevali, Marcelo Garcia Manzato
Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo
Av. Trabalhador Sancarlense, 400, PO Box 668 – 13560-970
São Carlos, SP – Brazil
lucastobal@grad.icmc.usp.br, mmanzato@icmc.usp.br

## ABSTRACT

This paper proposes a study and comparison among a variety of metadata types in order to identify the most relevant pieces of information in movie recommendation. We used three algorithms available in the literature to analyze the descriptions, and compared each other using the metadata extracted from two datasets, namely MovieLens and IMDB. As a result of our evaluation, we found out that the movies' genres are the kind of description that generates better predictions for the considered content-based recommenders.

## Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Indexing methods*

## General Terms

Design, Algorithms

## Keywords

recommender systems, metadata, matrix factorization, latent factors.

## 1. INTRODUCTION

Due to the large amount of information present in the World Wide Web, we observe a difficulty for users to deal with this huge quantity of content available. This problem is known as information overload, and a tool that helps individuals to manage such content is recommender systems. There are a number of ways to build recommender systems; basically they are classified as content-based filtering, collaborative filtering and the combination of both of them [1, 2].

Content-based filtering recommends multimedia content to the user based on a profile containing information regarding the content, such as genre, keywords, subject, etc. These metadata are weighted according to past ratings, in order to characterize the user's main interests. A problem with this approach is over-specialization, which happens when the system recommends only items that are too similar to the items already rated [1].

An alternative to this problem is the collaborative filtering, which is based on clusters of users or items. In the first case, items that are appreciated by a group of users with the same interests are recommended to a particular user of that group. In the second case, if two items have the same evaluation by different users, then these items are similar, so it is expected that the users have likely tastes for similar items [2].

One disadvantage of collaborative filtering is to calculate similarity between users and/or items in a vectorial space composed of user ratings in a user-item matrix. Similarity metrics (Pearson correlation, cosine similarity, etc.) must be applied to this matrix in order to infer clusters of similar users or items. However, such vectorial space makes this a large dimensionality matrix. Besides, the vectors are redundant because there will be users with similar ratings for the same items [2].

Such limitations have inspired researchers to use dimensionality reduction techniques, such as Singular Value Decomposition (SVD), in order to extract latent semantic relationships between users and items, transforming the vectorial space into a feature space containing topics of interest [10, 5, 6, 9]. Nevertheless, other challenges have to be dealt with, such as sparsity, overfitting and data distortion caused by imputation methods [5].

Considering the limitations and challenges depicted above, hybrid recommenders play an important role because they group together the benefits of content based and collaborative filtering. It is known that limitations of both approaches, such as the cold start problem, overspecialization and limited content analysis, can be reduced when combining both strategies into a unified model [1]. However, most recent systems which exploit latent factor models do not consider the matadata associated to the content, which could provide significant and meaningful information about the user's interests.

In related work [8, 7, 1, 3], we verify a set of recommender algorithms which exploit latent factors, collaborative filtering, metadata awareness and implicit feedback. However, there is a lack of study about which metadata type generates the best results in the domain of movies. In this way, the present paper aims to compare a variety of movie metadata with three recommendation algorithms in order to identify those pieces of information that are more important in the process of recommending movies to the user.

This work is structured as follows: in Section 2 we describe the models considered in this evaluation; in Section 3 we depict how the metadata is extracted; Section 4 presents the evaluation of different metatadata applied to the three considered algorithms; and finally, in Sections 5 and 6 we discuss the final remarks, future work and acknowledgements.

## 2. CONSIDERED MODELS

In this section we describe in more details the models used to study and compare the different types of metadata considered in this paper.

### 2.1 Notation

Following the same notation as [5, 8] we use special indexing to distinguish users, items and metadata: a user is indicated as $u$, an item as $i$ and $j$, and the item's metadata as $g$. A rating by a user to an item is indicated by $r_{ui}$, and the predicted one is $\hat{r}_{ui}$. The $(u, i)$ pairs for which $r_{ui}$ is known are represented by the set $K = \{(u, i)|r_{ui} \text{ is known}\}$.

The described methods will use baseline estimators in order to overcome overfitting. Similar to [5, 8] we denote $\lambda_1, \lambda_2, ...$ the constants used for regularization. Other notations will be explained throughout the paper.

### 2.2 Baseline Estimate

Baseline estimates are used to overcome tendencies that are presented from the data according to users' and items' intrinsic characteristics. For instance, a user may rate a very good film with the value 4, whereas another user may rate the same movie with the value 5 but indicating the same degree of interest.

In this way, to surpass these differences, baseline estimates are used to adjust the data taking into consideration these effects. A baseline for an unknown rating $r_{ui}$ is denoted $b_{ui}$ and is defined as:

$$b_{ui} = \mu + b_u + b_i \tag{1}$$

where $\mu$ is the overall average rating and $b_u$ and $b_i$ are the deviations observed to the user and item. To estimate theses parameters, one can solve the associated least square problem:

$$\min_{b*} \sum_{(u,i)\in K} (r_{ui} - \mu - b_u - b_i)^2 + \lambda(\sum_u b_u^2 + \sum_i b_i^2) \tag{2}$$

The first term aims to find the user and item biases that fit the given ratings, whereas the second avoids overfitting by penalizing the magnitudes of the parameters.

### 2.3 gSVD++

This method is an extension of Koren's SVD++ model [5], which incorporates metadata into a latent factor approach that considers implicit feedback. It can relate descriptions such as genre of movies, list of actors, keywords, language, country of production, among others.

To predict a rating $\hat{r}_{ui}$, the method uses the following equation:

$$\hat{r}_{ui} = b_u + \left(q_i + |G(i)|^{-\alpha} \sum_{g\in G(i)} x_g\right)^T \left(p_u + |N(i)|^{-0.5} \sum_{j\in N(u)} y_j\right) \tag{3}$$

In order to incorporate metadata into the model, the set $G(i)$ contains the description of some kind about the item. The implicit feedback is represented by $N(u)$, which contains the set of items that user $u$ provided an implicit preference. Each user $u$ is associated to a user factors vector $p_u \in \mathbb{R}^f$ and each item $i$ is associated to an item factors vector $q_i \in \mathbb{R}^f$. It also considers the indirect user information represented by a factors vector $y_i \in \mathbb{R}^f$ and a factors vector that contains the item's metadata $x_g \in \mathbb{R}^f$.

It was adopted a gradient descent scheme to solve the system depicted in Equation 3, using the following regularized squared error function:

$$\min_{p_*,q_*,x_*,y_*} \sum_{(u,i)\in K} \Bigg( b_{ui} = \mu + b_u + b_i - \left(q_i + |G(i)|^{-\alpha} \sum_{g\in G(i)} x_g\right)^T \left(p_u + |N(i)|^{-0.5} \sum_{j\in N(u)} y_j\right)\Bigg)^2 + \lambda\left(b_u^2 + b_i^2 + ||p_u^2|| + ||q_i^2|| + \sum_{j\in N(u)} y_j^2 + \sum_{g\in G(i)} x_g^2\right). \tag{4}$$

### 2.4 PrefGenreKNN

The PrefGenreKNN algorithm [7] is based on a factorization procedure that uses user preferences associated to the movies genres to infer latent factors about the considered descriptions. Instead of the user versus item matrix, it is adopted an alternative approach of constructing a user versus genres matrix, whose each cell contains a weight that describes how much a user has interest to that particular description. Although the model was originally proposed to work with genres, it can easily be adapted to analyze other kinds of metadata, such as keywords, list of actors, etc.

To calculate those weights, first of all the set of genres are paired into a genre cloud defined as $cloud_g(u, r)$. The cloud contains the pair $(g, n_{g,u,r})$ where $g$ represents the genre and $n_{g,u,r}$ represents the frequency of occurrence of genre $g$ that user $u$ associated with rating $r$.

Then, in addition to the genre cloud, the tf-idf measure is used to calculate how important a genre is to a particular rating in the set of all ratings:

$$\textit{tf-idf}(g, u, r) = n_{g,u,r} \log\left(\frac{|K|}{1 + |K(g, u)|}\right) \tag{5}$$

where $K(g, u)$ is the set of different ratings assigned to genre $g$ by user $u$.

Finally, the weight is computed:

$$w(g, u) = \frac{\sum_{r\in K(g,u)} \textit{tf-idf}(g, u, r) r}{\sum_{r\in K(g,u)} r} \tag{6}$$

After the weighting procedure, Singular Value Decomposition is used to factorize the user-genre matrix $M$ composed of associated $w(g, u)$. The matrix $M$ is factorized into three matrices $M = V\Sigma T^T$. Before the user-genre matrix factorization, an imputation procedure is executed in order to substitute the weights $w(g, u)$ which are zero with the user average rating offset defined as:

$$\bar{r}_u = \mu + \frac{1}{|K(u)|} \sum_{j \in K(u)} (r_{uj} - \mu) \tag{7}$$

where $K(u)$ is the set of items rated by user $u$.

In advance, the following equation is used to enrich the data, that is, to incorporate latent factor information into the weights which are missing in the matrix:

$$w'(g,u) = \begin{cases} V_k.\Sigma_k.T_k^T(g) \text{ if } w(g,u) = 0 \\ (1-\gamma)w(g,u) + \gamma(V_k.\Sigma_k.T_k^T(g)), \\ \text{otherwise} \end{cases} \tag{8}$$

where the vector $V_{k(u)}$ is the row of the user $u$ in the matrix $V$, $\gamma$ is the weighting parameter and $T_k^T(g)$ is the row of the genre $g$ in the matrix $T^T$.

Finally, the user similarity is computed using the Pearson correlation:

$$sim(u,v) = \frac{\sum_{g \in G}(w'(g,u) - \bar{w}'(u))(w'(g,v) - \bar{w}'(v))}{\sqrt{\sum_{g \in G}(w'(g,u) - \bar{w}'(u))^2 \sum_{g \in G}(w'(g,v) - \bar{w}'(v))^2}} \tag{9}$$

where $w(u)$ and $w(v)$ are the user $u$ and $v$ averages for the weights associated to the available genre.

## 2.5 ItemAttributeKNN

The ItemAttributeKNN is also based on collaborative filtering [1]; nevertheless, instead of using the ratings from users to the items, it uses the metadata from the items to generate the clusters of similar items. Initially, a baseline prediction is applied as described in Subsection 2.2. Then, it uses a matrix with the items descriptions to calculate the similarity between items:

$$w_{ij} = \frac{n_{ij}}{n_{ij} + \lambda_4} p_{ij} \tag{10}$$

where $i$ and $j$ are items, $n_{ij}$ denotes the number of users that rated both $i$ and $j$, and $p_{ij}$ is the Pearson correlation coefficient, which measures the tendency of users to rate items $i$ and $j$ similarly. A typical value for the parameter $\lambda_4$ is 100.

And finally, to predict the recommendation, it uses the following equation:

$$\hat{r}_{ui} = b_{ui} + \frac{\sum_{w \in K(i,j)} w_{ij}(r_{ui} - b_{ui})}{\sum_{w \in K(i,j)} w_{ij}} \tag{11}$$

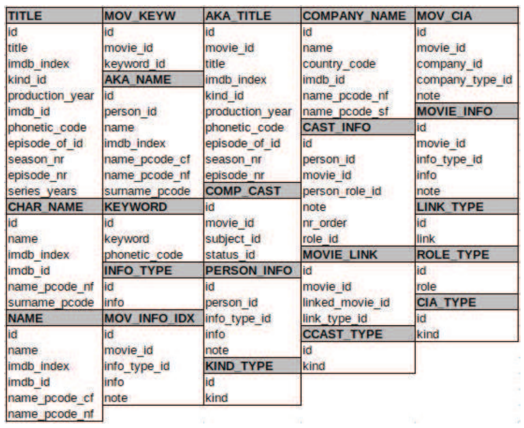
## 3. METADATA EXTRACTION

The MovieLens 100k[1] and the Internet Movie Database (IMDB)[2] datasets were used in this paper in order to discover meaningful latent factors in movies recommendation. Since the MovieLens database has little information about the movie, we relate each of its movies to the IMDB database, which has a more complete description about the movies.

[1]http://www.grouplens.org/node/73

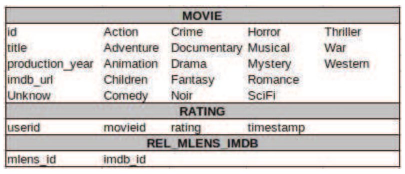
[2]http://www.imdb.com/interfaces

Figures 1 and 2 show the metadata contained in each database. The most relevant data are the indexes to the movies from MovieLens and IMDB because it is through these indexes that we can align the information in these two databases. Also, Figure 2 shows the table *rel_mlens_imdb* is the one that contains the indexes from MovieLens and their respective index from IMDB.



| TITLE | MOV_KEYW | AKA_TITLE | COMPANY_NAME | MOV_CIA |
|---|---|---|---|---|
| id | id | id | id | id |
| title | movie_id | movie_id | name | movie_id |
| imdb_index | keyword_id | title | country_code | company_id |
| kind_id | **AKA_NAME** | imdb_index | imdb_id | company_type_id |
| production_year | id | kind_id | name_pcode_nf | note |
| imdb_id | person_id | production_year | name_pcode_sf | **MOVIE_INFO** |
| phonetic_code | name | phonetic_code | **CAST_INFO** | id |
| episode_of_id | imdb_index | episode_of_id | id | movie_id |
| season_nr | name_pcode_cf | season_nr | person_id | info_type_id |
| episode_nr | name_pcode_nf | episode_nr | movie_id | info |
| series_years | surname_pcode | **COMP_CAST** | person_role_id | note |
| **CHAR_NAME** | **KEYWORD** | id | note | **LINK_TYPE** |
| id | id | movie_id | nr_order | id |
| name | keyword | subject_id | role_id | link |
| imdb_index | phonetic_code | status_id | **MOVIE_LINK** | **ROLE_TYPE** |
| imdb_id | **INFO_TYPE** | **PERSON_INFO** | id | id |
| name_pcode_nf | id | id | movie_id | role |
| surname_pcode | info | person_id | linked_movie_id | **CIA_TYPE** |
| **NAME** | **MOV_INFO_IDX** | info_type_id | link_type_id | id |
| id | id | info | **CCAST_TYPE** | kind |
| name | movie_id | note | id | |
| imdb_index | info_type_id | **KIND_TYPE** | kind | |
| imdb_id | info | id | | |
| name_pcode_cf | note | kind | | |
| name_pcode_nf | | | | |

**Figure 1: IMDB database**



| MOVIE | | | | |
|---|---|---|---|---|
| id | Action | Crime | Horror | Thriller |
| title | Adventure | Documentary | Musical | War |
| production_year | Animation | Drama | Mystery | Western |
| imdb_url | Children | Fantasy | Romance | |
| Unknow | Comedy | Noir | SciFi | |
| **RATING** | | | | |
| userid | movieid | rating | timestamp | |
| **REL_MLENS_IMDB** | | | | |
| mlens_id | imdb_id | | | |

**Figure 2: MovieLens database**

Since the movies present in the MovieLens database have already indexes, we kept these indexes and we related the movie title from MovieLens to the IMBD. It was necessary to manipulate the data in MovieLens because the movie titles were written in English form (e.g. Godfather, The). So, we fixed these names to the form used in IMDB (e.g. The Godfather). After fixing the names, we created a table that has the indexes from both MovieLens and IMDB movies. Therefore, having the table $T$ that has indexes from both MovieLens and IMDB, it was possible to use the information presented in the IMDB to explore more types of metadata in order to find those most relevant descriptions in movies recommendation. The types of metadata considered in this paper are: list of actors, directors and cast, country of production, language spoken, keywords and genre list.

## 4. EVALUATION

In the evaluation presented in this paper we compared seven different types of metadata: genres, country, language, list of actors, cast, list of keywords and director. All these pieces of information were inputted into each of the three methods previously described. These methods were implemented using MyMediaLite [4] library and were measured using RMSE (Root Mean Squared Error) and MAE (Mean Average Error) metrics. For each considered number of factors and method, we used 5-fold cross-validation in order to give it more confidence.

The tests were executed with the MovieLens dataset which contains 100k ratings given by 943 users on 1682 movies; each user has rated at least 20 movies. Once we related the MovieLens dataset with the IMDB dataset we found a amount of 1671 movies that were on the both datasets. Also, to a more accurate result, we used only the ratings that had movies on both databases which leaves us with the total of 99968 ratings. Therefore, 11 movies are not related in the databases and this corresponds to 32 ratings in the 100k ratings from MovieLens.

After executing the algorithms with each metadata, we obtained the results illustrated in Tables 1 and 2. Some scores could not be found because the algorithm could not deal with the amount of data presented in these files.

**Table 1: RMSE**

| | gSVD++ | ItemAttributeKNN | PrefGenreKNN |
|---|---|---|---|
| Genre | 0,9023 | 0,9298 | 0,9465 |
| Country | 0,9146 | 0,9294 | 0,9456 |
| Language | 0,9122 | 0,9298 | 0,9456 |
| Actor | 0,9068 | 0,9298 | - |
| Cast | 0,9077 | 0,9298 | - |
| Keyword | 0,9074 | 0,9298 | - |
| Director | 0,9066 | 0,9298 | - |

**Table 2: MAE**

| | gSVD++ | ItemAttributeKNN | PrefGenreKNN |
|---|---|---|---|
| Genre | 0,7086 | 0,7303 | 0,7453 |
| Country | 0,7194 | 0,7303 | 0,7438 |
| Language | 0,7170 | 0,7310 | 0,7438 |
| Actor | 0,7136 | 0,7310 | - |
| Cast | 0,7141 | 0,7310 | - |
| Keyword | 0,7141 | 0,7310 | - |
| Director | 0,7121 | 0,7310 | - |

The gSVD++ algorithm presented the best results with the genre metadata having an RMSE of 0,9023 and MAE of 0,7086. In addition, it is also possible to affirm that directors, with a RMSE of 0,9066 and MAE of 0,7121 on gSVD++, are another important piece of information in movies recommendation. The country information, in turn, was the type of metadata which scored the worst results when using this algorithm.

On the other hand, analyzing the ItemAttributeKNN algorithm, we can see that regardless of the information type, all runs produced very similar results. This means that the clusters of items are formed similarly for each metadata considered, and consequently, the prediction rule is applied in the same way for all cases.

With respect to PrefGenreKNN, we note that country and language were able to produce better accuracy than genres. This implies that according to the procedure adopted by the algorithm to compute the users' preferences, pieces of information related to nationality represent better the interests of a user than the movies' genres.

It is also worth to consider that the gSVD++ algorithm generated the best RMSE and MAE results regardless of the type of metadata. This is due to the fact that it also incorporates implicit feedback, which is an important piece of information that contributes to characterize the user's interests more accurately.

## 5. FINAL REMARKS

This paper presented a set of state-of-art algorithms that uses metadata to predict ratings in movie recommendation systems. Different types of metadata were adopted in order to find which ones generate the best results. After comparing the three algorithms described previously, we found the genres as the piece of information that most contributes positively to the performance of the gSVD++ algorithm, and country/language as the ones which contribute to the accuracy of the ItemAttributeKNN and PrefGenreKNN algorithms.

In future work, we plan to evaluate the algorithms with a combination of two or more types of metadata in order to verify whether multimodal information can generate better prediction. In order to do so, however, it will be necessary to extend the algorithms to exploit the descriptions in an effective fashion.

## 6. ACKNOWLEDGEMENTS

The author would like to thank the financial support from RUSP Institucional and FAPESP, process number 2011/17366-2.

## 7. 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] M. D. Ekstrand, J. Riedl, and J. A. Konstan. Collaborative filtering recommender systems. *Foundations and Trends in Human-Computer Interaction*, 4(2):175–243, 2011.

[3] Z. Gantner, L. Drumond, C. Freudenthaler, S. Rendle, and L. Schmidt-Thieme. Learning attribute-to-feature mappings for cold-start recommendations. In *2010 IEEE 10th International Conference on Data Mining (ICDM)*, pages 176–185, dec. 2010.

[4] Z. Gantner, S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. MyMediaLite: A free recommender system library. In *Proceedings of the 5th ACM Conference on Recommender Systems*, RecSys '11, pages 305–308, New York, NY, USA, 2011.

[5] Y. Koren. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Trans. Knowl. Discov. Data*, 4(1):1:1–1:24, Jan. 2010.

[6] M. Kurucz, A. A. Benczúr, and B. Torma. Methods for large scale svd with missing values. In *KDD Cup Workshop 2007*, 2007.

[7] M. G. Manzato. Discovering Latent Factors from Movies Genres for Enhanced Recommendation. In *Proceedings of the 6th ACM Conferece on Recommender Systems*, RecSys '12, pages 249–252, New York, NY, USA, 2012.

[8] M. G. Manzato. gSVD++: supporting implicit feedback on recommender systems with metadata awareness. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, SAC '13, pages 908–913, New York, NY, USA, 2013. ACM.

[9] A. Paterek. Improving regularized singular value decomposition for collaborative filtering. In *Proc. KDD Cup Workshop at SIGKDD'07, 13th ACM Int. Conf. on Knowledge Discovery and Data Mining*, pages 39–42, 2007.

[10] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl. Application of Dimensionality Reduction in Recommender System – A Case Study. In *Proceedings of ACM SIGKDD Conference on Knowledge Discovery in Databases*, Boston, MA, USA, 2000.