# Addressing the cold start with positive-only feedback through semantic-based recommendations

Paolo Tomeo[1], Ignacio Fernández-Tobías[2],
Iván Cantador[2] and Tommaso Di Noia[1]
[1]Politecnico di Bari
Via Orabona 4 - 70125 Bari, Italy
{paolo.tomeo,tommaso.dinoia}@poliba.it
[2]Universidad Autónoma de Madrid
Calle Francisco Tomás y Valiente 11 - 28049 Madrid, Spain
{ignacio.fernandezt,ivan.cantador}@uam.es

**Abstract**

Recommender systems aim to provide users with accurate item suggestions in a personalized fashion, but struggle in the case of cold start users, for whom there is a scarcity of preference data. User preferences can be either explicitly stated by the users —often by means of ratings—, or implicitly acquired by a system —for instance by mining text reviews, search queries, and purchase records. Recommendation methods have been mostly designed to deal with numerical ratings. However, real scenarios with user preferences expressed in the form of binary and unary (positive-only) feedback, e.g. the thumbs up/down in YouTube, and the likes in Facebook, are increasingly popular, and make the user cold start problem even more challenging. To address cold start with positive-only feedback situations, we propose to exploit data additional to user preferences by means of specialized hybrid recommendation methods. In particular, we investigate a number of graph-based and matrix factorization recommendation models that jointly exploit user preferences and item semantic metadata automatically extracted from the well known knowledge graph of DBpedia. Following a rigorous evaluation methodology for cold start, we empirically compare the above hybrid recommendation models on a Facebook dataset containing users likes for items in three different domains, namely books, movies and music. The achieved experimental results show that the semantics-aware hybrid approaches we propose, outperform content-based and collaborative filtering baselines. In addition to recommendation accuracy, in our evaluation we also consider individual and aggregate diversity of recommendations as key quality factors in users' satisfaction.

# 1 Introduction

Recommender systems (RSs) have become fundamental tools in helping users to find what is relevant for them in situations where information overload makes such task hard or even impossible. For such purpose, they capture, model and exploit user preferences, which can be obtained either explicitly by means of ratings, or implicitly e.g. by processing text reviews, and by mining item consuming and purchasing records. Two main families of recommendation approaches exist, namely content-based and collaborative filtering. The former commonly relies on content-based features to represent both user and item profiles, and provides personalized recommendations based on similarities between the user's and items profiles; the latter, in contrast, works with rating-based user/item profiles, and provides recommendations based on similarities between profiles of like-minded users.

The majority of the most effective collaborative filtering approaches have been designed to deal with numerical ratings, such as the 5-star ratings in Amazon[1] and Netflix[2], for both rating prediction and item ranking (a.k.a. top-N recommendation) tasks, and have been shown to generally outperform content-based approaches [1]. In many e-commerce and social network sites, however, user preferences are expressed in the form of binary and unary (positive-only) ratings, such as the thumbs up/down in YouTube[3] and the likes in Facebook[4], respectively. Moreover, in these cases, the well-known problem of cold-start in collaborative filtering [1], which refers to the scarcity of ratings at user level, is highly remarkable. In this context, the consideration of content-based features could improve the understanding of the users' preferences, as well as the finding of similar users and items. For instance, in the movie recommendation domain, a user may be suggested with movies based on her and others' preferences for particular genres, directors and actors. The usage of hybrid recommendation methods, which benefit from the advantages of both the aforementioned approaches, may be also explored to tackle the cold-start problem. In general, e-commerce and social network sites do not provide the content-based features that comprise item metadata. These features, nonetheless, can be extracted from text descriptions about the items, e.g., movie plots, song lyrics, and book synopses, or can be established by means of social tags manually assigned by users to the items. More recently, knowledge graphs have been exploited as a new source of metadata to enhance the description of catalog items. Via item linking techniques, it is possible to map a string representing an item, e.g., the name of a movie, to its corresponding entity in the knowledge graph thus providing access to its structured description.

While new recommendation approaches have been proposed over the years mainly devoted to maximizing accuracy, more recently it has been recognized that predictive accuracy of recommendations is not enough to judge the effec-

[1] Amazon online shopping, www.amazon.com
[2] Netflix movie and TV series streaming, www.netflix.com
[3] YouTube online video sharing, www.youtube.com
[4] Facebook online social network, www.facebook.com

tiveness of a recommender system [2]. The most accurate recommendations for a user are often too similar to each other, and attention has to be paid towards the goal of improving *diversity* in recommended items, known as *individual diversity* [2]. Indeed, a recent user study pointed out a strong correlation between perceived accuracy and user satisfaction for approaches able to better diversify the returned list of recommended items [3]. While this kind of dimension is list-wise, since it represents the diversity degree in the recommendations list, the importance of a system-wise diversity has been revealed from other studies [4]. In particular, *aggregate diversity* has been proposed to assess the ability of a system to cover the items catalog, and equally distribute the recommendations across all the items. Such quality factor results important for both a user and a business perspective: users may receive less obvious and more personalized recommendations, complying with the goal of helping on the discover of new contents [4], and businesses may increase their sales [5].

In this paper we focus on the cold-start problem in recommendations with positive-only feedback, through a number of graph-based and matrix factorization recommendation models that jointly exploit user ratings and item metadata. To obtain metadata for the available items, in this paper we present a method that automatically maps the items names to URIs of semantic entities in DBpedia[5], which is considered as the core repository of the Linked Open Data (LOD) cloud. In this context, the use of LOD does not merely allow describing items by means of content-based features, but also creating semantic networks that relate items and their attributes with each other, e.g., Kubrick's "The Full Metal Jacket" is a movie based on Hasford's "The Short-Timers" novel, and "Anti-War Films" is a subgenre of "Political Films."

We evaluate the above mentioned models with Facebook *likes* as source of user preferences. We propose to exploit semantic networks connecting users, liked items, and item attributes for recommendation purposes in two ways: first, by directly mining the networks via graph-based recommendation models; second, by extending content-based item profiles with related attributes, and incorporating the enriched profiles into matrix factorization recommendation models. We evaluate the two types of approaches, along with several baselines, on a Facebook dataset comprising three distinct domains, namely books, movies and music.

This paper considerably extends our previous work [6] where we showed preliminary results only in terms of recommendation accuracy. Here we also consider individual diversity and aggregate diversity as key quality factors for users' satisfaction, and conduct more in-depth analysis and discussion of the empirical results than in the previous paper. In particular, we also describe differences among the various models taking into account the trade-off between the different recommendation quality dimensions we evaluated. Our experiments show that the proposed hybrid recommendation models, which exploit rating and semantic metadata, outperform standard content-based and collaborative filtering baselines, particularly in the most extreme cold-start scenarios.

[5] wiki.dbpedia.org

## 2 Related Work

The user cold start problem arises when a new user registers into a system, and has not yet provided any preference feedback, either implicit or explicit, or when she had just a few interactions with the system, and the number of collected preferences is not enough to build an accurate user profile and then compute reliable recommendations. The cold start has been mainly addressed from two perspectives. The first perspective corresponds to active learning techniques [7], which attempt to collect feedback by directly asking the user to rate certain items before generating the recommendations. These techniques usually seek popular items that the user is likely to know, and whose rating would be useful to improve the overall system performance. The second perspective is based on the exploitation of additional side information about the users or the items in the recommendation process. For instance, Pazzani [8] used demographic data like gender, age, area code, education and employment information, to compute user-user similarities, while Braunhofer et al. [9] showed that information about the user's personality can be more effective in some applications. Recently, cross-domain recommendation methods have been proposed that exploit user preferences in different source domains to mitigate the lack of information in a target domain [10].

In this work we face the user cold-start problem with positive-only feedback and available item side information, by investigating recommendation models that can jointly exploit user feedback and additional item metadata. In particular, we propose the usage of semantic metadata extracted from DBpedia to build heterogeneous semantic networks that link users, items, and item attributes, and than the exploitation of such networks either directly through graph-based models, or by injecting their information into factorization-based models. Our first model is based on HeteRec [11], which uses meta-path based features to represent the connectivity between users and items along different types of paths. The second model is PathRank [12], an extension of the Personalized PageRank algorithm able to exploit different paths on a heterogeneous graph during the random walk process. As factorization-based models, we investigate Collective Matrix Factorization and Factorization Machines. The former is used to simultaneously factorize the user-item matrix and the item-item similarity matrix [13]. The latter represents a generalization of matrix factorization methods able to integrate different types of side information in the same model at a low computational cost [14]. In Section 4 we provide a detailed description of the above models.

The motivation behind the use of DBpedia for semantically building item profiles corresponds to the recent tendency of using external information from well-known Linked Open Data (LOD) resources in order to implement semantic-aware recommender systems. There are several advantages from the use of LOD in content-based and hybrid recommender systems. In particular, (i) great amount of multi-domain and ontological knowledge is freely available in the LOD cloud for feeding the systems, and (ii) content preprocessing for obtaining a structured representation of the item descriptions is not necessary [15]. In

[16] the authors show that the exploitation of DBpedia produce more diverse recommendation results compared to the usage of Freebase.

So far, many semantic-aware recommendation models have been proposed, in general aiming to improve recommendation accuracy. Authors in [17] proposed a recommender system fed by LOD, and reported experimental results in terms of precision and recall. In [18] DBpedia was used to enrich music playlists extracted from a Facebook profile with new related musicians while in [19] data coming from DBpedia have been used to enrich the Movielens dataset for film recommendation. An event recommendation system based on linked data and user diversity was proposed in [20]. Another use of LOD for content-based RSs was explored in [21], where the authors presented Contextual eVSM, a content-based context-aware recommendation framework that adopts a semantic representation based on distributional models, and entity linking techniques to detect entities in free text and map them to LOD. Graph-based distributional semantics is also at the base of the approach proposed in [22]. More recently, in [23] entity linking from textual information has been adopted as a preprocessing step to build a recommender systems in the sounds and music domains. Finally, a hybrid model presented in [24] exploits LOD sub-graphs for building a semantic network, and extracting relevant path-based features describing complex relationships between users and items, to eventually compute recommendations with a learning to rank algorithm.

Compared to most of the recommendation approaches proposed in the literature that have been specifically designed to exploit LOD, the graph-based and factorization-based models used in this paper represent a more generic and abstract approach to hybrid recommendation. Indeed, both graph-based models can applied to generic information graphs, Factorization Models can integrate any type of side information, and Collective Matrix Factorization relies on item-item similarities that can be computed with different metrics, not necessarily based on semantics.

# 3 Semantically Enriched Facebook Likes Dataset

The experimental evaluation presented in this paper is conducted on a Facebook dataset with user likes for book, movie and music items, we extended with item metadata extracted from DBpedia. In this section, we comprehensively describe the dataset and the developed process to automatically extract data from DBpedia to build semantic networks relating items and features.

**Original Positive-only Feedback Data.** Our dataset initially consisted of a large set of *likes* assigned by users to items in Facebook. Using the Facebook Graph API, a user's like is retrieved in the form of a 4-tuple with the following information: identifier, name and category of the liked item, and timestamp of the like creation. The name of an item is given by the user who created its Facebook page. As distinct names may exist for a particular item, users may express likes for different Facebook pages, which actually refer to the same item. Aiming at unifying and consolidating the items of the extracted Facebook likes,

we developed a method that automatically maps items to their corresponding DBpedia entities, e.g., `http://dbpedia.org/resource/The_Godfather` for the identified names of "The Godfather" movie. Details of the mapping process can be found in [6]. After filtering the items without a corresponding DBpedia resource, our final dataset contains 315870 likes provided by 1876 users on 4001 book pages, 1446017 likes provided by 26943 users on 3907 movie pages, and 1311974 likes provided by 49369 users on 5751 music pages. The data sparsity is more than 99% for all the three domains.

**Semantically Annotated Dataset.** For every linked entity, we accessed DBpedia to retrieve the its metadata, which afterwards has been used as input for the recommendation models. In particular, we posed SPARQL queries to the DBpedia endpoint asking for all the properties and objects in the triples having the target entity as subject. Since our ultimate goal is item recommendation, we only gathered metadata that may be relevant to relate common preferences of different users. Thus, we considered only meaningful subset of properties for each domain[6].

**Semantically Enriched Item Profiles.** Once we identify books, movies, and music artists and bands as the target items to be recommended, we can distinguish between three types of item metadata extracted from DBpedia. The first type of metadata is represented by the item attributes, e.g., the genre(s), director(s) and actors of a particular movie. Second, the item-item properties that directly relate items with each other, e.g., the novel that a movie is based on (`http://dbpedia.org/ontology/basedOn` property), the prequel/sequel of a movie (`http://dbpedia.org/ontology/previousWork` and `http://dbpedia.org/ontology/subsequentWork` properties), and the musicians of a band (`http://dbpedia.org/ontology/bandMemberproperty`). Finally, attribute-attribute properties generate **extended item attributes** that originally do not appear as metadata of the items, e.g., the subgenres of a particular music genre (`http://dbpedia.org/ontology/musicSubgenre` property). The above three types of item metadata constitute the semantically enriched item profiles that we adopted in the recommendation models we propose.

# 4 Evaluated recommendation models

In the following we present the proposed recommendation models, which jointly use user ratings and semantically enriched item profiles. The links between users, items, and item attributes extracted from DBpedia constitute a semantic network, that we propose to exploit i) directly by means of the graph-based models, and ii) indirectly by enriching item profiles that are incorporated into matrix factorization models.

**Graph-based Models.** The importance of graph-based approaches to recommendation has emerged concurrently with the increasing availability of additional user and item data. These approaches foster combining the user-item

[6]The complete list of DBpedia properties selected for each of the three domains in our dataset is shown in our previous work [6].

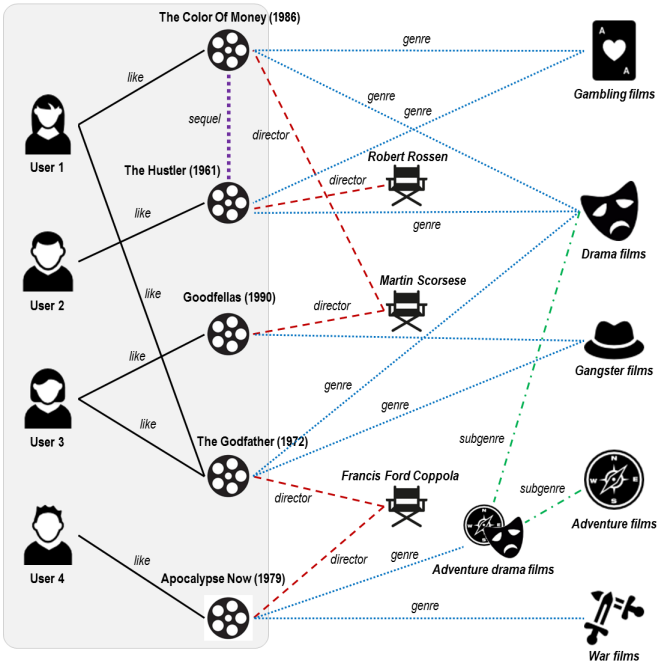


Figure 1: Example of a heterogeneous information network

rating matrix with side information into a graph, and then applying graph mining techniques. More specifically, as shown in Figure 1, a rating matrix is transformed into a bipartite graph component –which consists of user and item nodes linked with rating/like edges– extended to form a multipartite graph, including nodes representing additional entities, which are related to items. Within the graph it is also possible to include other edges, representing e.g. contextual information for the ratings, social connections between users, and semantic relations between entities [25]. The resulting graph can be thus defined as a heterogeneous information network consisting of a multi-typed and multi-relational directed graph, with nodes and edges of different types [26].

Structuring all the available data in the form of a graph leads to different advantages: (i) well-known graph-based algorithms can be used to develop hybrid recommender systems able to exploit the different types of information by surfing the graph [11]; (ii) both content and collaborative features are represented in a uniform setting, thus leveraging the multi-relational nature of the graph; (iii) the graph can be directly extended with information already available in the form of graphs, such as Linked Open Data [27]; (iv) exploring the graph may lead to relevant, but not obvious recommendations and also help on addressing the cold-start scenario. In fact, exploring longer paths in the network could overcome the lack of information connecting users and items.

Given a graph $G$, our aim is to produce personalized recommendations lever-

aging the knowledge it encodes. We propose to exploit the paths among users and items composed by different types of relations. For example, a user may be connected to an item $i$ by the relation $(like \circ director \circ director^{-1})$, which basically means that the user likes one or more items with same director of item $i$[7]. These sequences of relations are called meta-paths and their combination with nodes they cross by are called path instances [26].

**HeteRec** is a hybrid method based on matrix factorization that uses meta-path based latent features to represent the connectivity between users and items along different types of paths in a heterogeneous information network [11]. Briefly, for each meta-path, HeteRec computes the relative diffused user preferences matrix extending the similarity measure PathSim [26] in order to include the user feedback. More formally, the user preference diffusion score between user $u$ and item $j$, along a generic meta-path $P$, is defined as:

$$sim(u,j) = \sum_{i \in R(U)} \frac{2 \cdot r(u,i) \cdot |p_{i \to j} : p_{i \to j} :\in P|}{|p_{i \to i} : p_{i \to i} \in P| + |p_{j \to j} : p_{j \to j} \in P|} \quad (1)$$

where $p_{x \to y}$ is a path instance between the items $x$ and $y$.

Basically, the previous formula is a weighted sum of PathSim values among the items in the user profiles and the target item $j$, where the numerator measures the connectivity defined by the number of path instances between them following $P$, and the denominator represents the balance of their popularity in the graph. Once the matrices are computed, HeteRec factorizes them with a low-rank matrix factorization technique. Unfortunately, their application result infeasible with user preferences matrices which are usually dense. A truncation strategy can be used to keep the matrices sparse, reducing the amount of space and time consume [25], but could remove valuable information in the cold start scenario. Therefore, our model is directly based on the non-factorized diffused user preferences matrices. The estimated user-item preference matrix is finally computed as the weighted sum of the different meta-path matrices $R* = w_{P_1} \cdot R *_{P_1} + ... + w_{P_m} \cdot R*_{P_m}$, with $m$ being the number of meta-paths, while $w_{P_1}$ and $R*_{P_1}$ the weight and the diffused user preferences matrix of $i$-th meta-path, respectively. HeteRec splits the users into clusters, and then computes the importance of each meta-paths with a learning-to-rank approach. As we face the user cold-start situation, clustering the users is impracticable with a few ratings and without additional information. Therefore, we compute the meta-paths weights globally for all the cold-start users.

**PathRank** is an extension of the Personalized PageRank algorithm able to exploit different paths on a heterogeneous graph during the random walk process [12]. At each iteration, the random walker has three options: *transition*, move to one of adjacent nodes with probability $w_{trans}$ ; *restart*, restart the random walk from one of the query nodes with probability $w_{restart}$; path following, considering one of the meta-paths with probability $w_{path}$. Therefore, PathRank vector $\mathbf{r}$ is computed as:

$$\mathbf{r}^* = w_{trans} \cdot M_G^T \cdot \mathbf{r} + w_{restart} \cdot \mathbf{t} + w_{path} \cdot (w_{P_1} \cdot M_{P_1}^T + ... + w_{P_m} \cdot M_{P_m}^T) \cdot \mathbf{r} \quad (2)$$

[7]Given a relation $r$ going from $x$ to $y$ we denote with $r^{-1}$ the relation going from $y$ to $x$.

where $M_G$ is the item-item transition matrix of the full graph G, $M_{P_i}$ is the transition matrix of the $i$-th meta-path, $\vec{t}$ is the teleport vector representing the recommendation query (user profile) initialized with $1/|R(u)|$ for each item in $R(u)$, 0 otherwise.

**Factorization-based Models.** Matrix factorization (MF) models are considered as the state-of-the-art for collaborative filtering, and have been extensively studied in recent years [28]. These approaches gained popularity in the context of the Netflix prize, and since then they have been successfully used in many applications. Focusing on the rating prediction task, Funk [29] presented one of the first approaches that approximate the user-item rating matrix as the product of two low-rank matrices of user and item latent factors, respectively. Building on MF, Koren et al. [28] proposed the well-known SVD++ model where the user latent features are extended with additional parameters for each rated item. The motivation is that the information of whether a user chooses or not to rate an item is also an indicator of her preferences, and should be taken into account in the rating prediction. Despite their success, the previous models were designed to deal with numeric, explicit ratings. However, typical user feedback implicitly acquired by most real-world systems are positive-only, and require a different treatment. For such purpose, Hu et al. [30] presented a MF method that also models unobserved user-item interactions, as the lack of this information could indicate that the user dislikes the item or that she simply is unaware of it.

Basically, MF models learn low-rank representations of the user-item matrix deriving some latent factors from rating patterns, and then map both users and items to a joint latent factor space of dimensionality $k$, with $k$ usually much smaller then the dimensions of the original user-item matrix. Therefore, the interactions between users and items are modeled as inner products in the latent factor space. More formally, each item $i$ is associated with a vector $\mathbf{q_i} \in \mathbf{R}^k$, where each element in $q_i$ indicates the extent to which the item possesses the corresponding factor. While each user $u$ is associated with a vector $\mathbf{p_u} \in \mathbf{R}^k$, whose elements indicate the importance of the corresponding factor for the user. Finally, their scalar product captures the interaction between user $u$ and item $i$ and can be used for rating estimation as follows:

$$r^*(u,i) = \mathbf{q_i^T} \cdot \mathbf{p_u} \tag{3}$$

While the rating can be easily estimated once the model is computed, the major challenge remains to identify accurate mappings. Earliest implementations of matrix factorization models relied on imputation techniques to remove the user-item matrix sparsity filling in the void cells, for instance with the average ratings for a user or for an item [31]. However, imputation increases the amount of data, making the computation more expansive, and moreover can lead to less accurate recommendations [28]. Recently, a number of MF approaches that directly model only the observed ratings have been proposed. In the most common formulation of MF, to learn the model (i.e. $\mathbf{q_i}$ and $\mathbf{p_u}$ vectors) the systems aims at optimize a regularized squared error cost function, as follows

$$*min_{q^*,p^*} \sum_{(u,i)\in K} (r(u,i) - r^*(u,i) + \lambda \left( \sum_u ||\mathbf{q_i}||^2 + \sum_i ||\mathbf{p_u}||^2 \right) \quad (4)$$

where $K$ is the set of the $(u,i)$ pairs representing all the known interactions in the user-item matrix, and the term $\lambda$ controls the importance of the regularization term used to prevent overfitting. The choice of $\lambda$ depends on the data and can be determined by cross validation. Usually, *Stochastic Gradient Descent* is used for minimizing the optimization problem, and hence learning the factors[29].

A specific matrix factorization model has been proposed in [32] for better handling implicit feedback which takes into account both observed and unobserved feedback in the training process. The motivation is that the model should not only be able to predict high scores for relevant items, but also whether an item was rated or not. However, non-observed user-item interactions may not reflect the user does not like an item, but does not know it. Hence, the model includes a confidence hyperparameter $c_{(u,i)}$ in the loss function to penalize mistakes on observed and non-observed preference predictions differently:

$$*min_{q^*,p^*} \sum_{(u,i)} c_{(u,i)} \cdot (p(u,i) - r^*(u,i))^2 + \lambda \cdot \left( \sum_u ||\mathbf{q_i}||^2 + \sum_i ||\mathbf{p_u}||^2 \right) \quad (5)$$

where $p(u,i)$ represents a binarized derivation of $r(u,i)$, that returns 1 for $r(u,i) > 0$ and 0 otherwise.

The confidence parameter is set as $c_{(u,i)} = 1 + \alpha \cdot r(u,i)$ with $\alpha > 0$, so that mistakes predicting observed feedback are more penalized. It is important to note that Equation 5 considers all possible $(u,i)$ pairs, while Equation 4 uses only the known interactions in the user-item matrix. This is possible since the value of $p(u,i)$ is assumed as 0 for all the unknown $(u,i)$ pairs. The rationale of the method is that the system has minimal confidence in $p(u,i)$ for every user-item pair, but as it observes more evidence for positive preference, the confidence in $p(u,i)$ increases accordingly [32].

**Collective Matrix Factorization (CMF)** [13] is a representative matrix factorization method that originally showed significant improvements when item genres are taken into account for computing movie recommendations. The idea behind CMF is to simultaneously factorize the user-item matrix and the item-item similarity matrix. Predictions are still computed using Equation 3, but CMF includes an additional set of item latent vectors $\mathbf{s_j} \in \mathbf{R}^K$ feature to model the pairwise item interactions through the similarities. The loss function then becomes:

$$*min_{q^*,p^*,r^*}\gamma \cdot \sum_{(u,i)} c_{(u,i)} \cdot (p(u,i) - r^*(u,i))^2 + (1-\gamma) \cdot \sum_i \sum_j (s_{i,j} - \mathbf{q_i^T} \cdot \mathbf{s_j})^2 + \lambda \cdot \left( \sum_u ||\mathbf{q_i}||^2 + \sum_i ||\mathbf{p_u}||^2 + \sum_j ||\mathbf{s_j}||^2 \right) \tag{6}$$

where $s_{i,j}$ represents the content-based similarity between items $i$ and $j$; $\gamma \in (0, 1]$ weighs the importance of item similarities in the factorization. If $\gamma = 1$ the result is the same of IMF, whereas $\gamma$ close to 0 would ignore the preference predictions.

**Factorization Machines (FMs)** [14] are becoming increasingly popular, as they provide a principled and generic approach to integrate metadata into MF, showing promising results in the task of context-aware recommendation. A different set of approaches jointly factorizes the user-item preference and item-metadata matrices, sharing the item latent factors between both decompositions. Therefore, FMs provide a generic way to extend the standard MF model with different kinds of side information. The idea is to (one-hot) encode the user/item metadata information in a single feature vector $\mathbf{x} \in \mathbf{R}^{n=|U|+|I|+|F|}$ where $|U|, |I|, |F|$ are the number of users, items, and features, respectively. The model equation for a factorization machine of degree $d = 2$ is defined as:

$$r^*(u,i) = w_0 + \sum_{a=1}^{n} w_a \cdot x_a + \sum_{a=1}^{n} \sum_{b=a+1}^{n} \langle \mathbf{v_a}, \mathbf{v_b} \rangle \cdot x_a \cdot x_b \tag{7}$$

where the model parameters $w_0$, $\mathbf{w} \in \mathbf{R}^n$, $\mathbf{V} \in \mathbf{R}^{n \times k}$ have to be estimated, and $\langle \cdot, \cdot \rangle$ indicates the scalar product of two vectors. A row $\mathbf{v_i}$ in $V$ represents the $i$-th variable with $k$ factors. The $w_a$ parameters model the contribution of each component in the feature vector, whereas the weights for the pairwise interactions are factorized as the product of two latent feature vectors $v_a$ and $v_b$. FMs generalize all other MF models by taking in input any type of user/item metadata. Moreover, it has been demonstrated that the model in Equation 7 can be computed in linear time, hence parameters can be learned efficiently via stochastic gradient descent [14].

# 5 Experiments

In this section we detail the setting and results of the experiments performed to evaluate the recommendation quality in the cold-start situation. We adopted the graph-based and matrix factorization models presented in Section 4 on our Facebook dataset (see Section 3), for three distinct domains (books, movies and music). The goal is to evaluate the effectiveness of considering jointly user likes and item metadata to produce accurate recommendations for cold-start users, and to compare the different approaches in this setting.

**Evaluation Methodology.** The evaluation of the proposed techniques was based on the TestItems evaluation methodology proposed in [33], using a modified user-based 5-fold cross-validation strategy proposed in [34] for the cold-start user scenario. First, we selected the users with at least 20 likes, shuffled and split into five (roughly) equally sized subsets. In each cross-validation stage, we kept all the likes from four of the groups in the training set, whereas the likes from the users in the fifth group were randomly split into three subsets: training set (10 likes), validation set (5 likes), and testing (remaining likes, hence at least 5). In order to simulate different user profile sizes from one to ten likes, we repeated the training and the evaluation ten times, starting with the first like in the training set and incrementally increasing it one by one. This evaluation setting allowed us to evaluate each user profile size with the same test set thus avoiding potential biases in the evaluation, as some accuracy metrics can be sensitive to the test set size [34].

To evaluate the ranking accuracy of the recommendations, we used Mean Reciprocal Rank (MRR), which computes the average reciprocal rank of the first relevant recommended item, and hence results particularly meaningful when users are provided with few but valuable recommendations (i.e., Top-1 or Top-3) [1]. As recommendation accuracy has been proved to be not sufficient to guarantee a satisfying user experience, attention has been paid to other important quality factors such as individual and aggregate diversity. The former is a list-wise property that indicates the ability of a system to provide diverse recommendations to each user. Specifically, we use an intent-aware metric called BinomDiv [35] which measures the individual diversity also according to the user interests covered in the recommendation list. The latter is a system-wise property that measures the coverage of the items catalog and the distribution of the items across the users. We thus considered two metrics: catalog coverage (percentage of items recommended at least to once) and Entropy to analyze the items distribution [4].

**Baselines Models.** Besides the graph-based and factorization-based models presented in the previous section, we also evaluated a number of well-known content-based and collaborative filtering methods, and one hybrid method that integrates content similarity into user-based CF:

**Popularity-based (POP).** A non-personalized method that always recommends the most popular items not yet liked by the user.

**Content-based (CB).** A method that recommends the most similar items to those in the user's profile. We compute the similarity between items as the cosine between their TF-IDF feature vectors, obtained from the semantically-enriched item profiles.

**User-based Nearest Neighbors (UNN).** A method that estimates the score of candidate item i for target user $u$ by aggregating the preferences of other similar users: $r^*(u,i) = \sum_{v \in N(u) \cap U(i)} sim(u,v)$. Here $N(u)$ is the set of $u$'s $k$ most similar users, and $U(i)$ the set of users that liked item i. For the user-user similarity, we considered Jaccard's coefficient between the sets $I(u)$ and $I(v)$ of items liked by users $u$ and $v$, respectively.

**Item-based Nearest Neighbors (INN).** A method that works similarly to CB, with the difference that the item similarity is computed in a collaborative filtering fashion by exploiting the users' interactions rather than the items content. Specifically, we compute the score of item $i$ for user $u$ as $r^*(u,i) = \sum_{j \in I(u)} sim(i,j)$, where the item similarity is computed as the Jaccard coefficient between sets $U(i)$ and $U(j)$.

**Content-based Collaborative Filtering (HYB).** A method that integrates content information into the UNN approach by replacing the user similarity component. In particular, we generate the TF-IDF profile vector for each user by aggregating the content-based profile vectors of the user's liked items, computed as in the CB method. We compute the user-user similarities as the cosine between their corresponding profile vectors, and utilize the same formula as in UNN to compute the recommendation scores. Although the method relies on content-based similarities, it still follows the CF paradigm by exploiting the information from other users in the neighborhood.

**Sparse Linear Methods (SLIM).** This method is an implementation of the SLIM algorithm [36] available in MyMediaLite[8]. SSLIM refers to SLIM with side information [37].

## 5.1 Results

Table 1 shows the performance of the evaluated algorithms on the three domains in terms of MRR and BinomDiv, while Table 2 shows the results in terms of Catalog Coverage and Entropy, which together assess the aggregate diversity.

### 5.1.1 Books

**Accuracy.** PathRank achieves the best accuracy in the case of users with 1 and 2 likes; PathRank, UNN and HYB are the best methods with 3 likes; while HeteRec is best method from 4 to 10 likes. It is worth to note that POP baseline beats most of the methods except PathRank with 1 like, and also UNN and HYB with 2 likes. SLIM and SSLIM seem not able to face the cold-start problem, especially for users with less than 5 likes.

**Individual Diversity.** FMs provide the most diversified recommendations in all the configurations, except in the case of users with only 1 like, for whom UNN is better. Generally, POP, UNN and SLIM result close to FMs, while CB provides always the worst diversity.

**Aggregate Diversity.** In terms of catalog coverage, SSLIM is the best method with 1 like, INN with 2 likes, and CB for all the other sizes. Considering the entropy, SSLIM is always the best method, closely followed by SLIM. In general, CB, INN, SLIM and SSLIM provide the best values in term of both the two metrics, respect to all the other evaluated methods. POP is obviously the worst method, as it recommends only the most popular items.

**Overall Analysis.** We now discuss the results by considering accuracy, individual and aggregate diversity together. None of the analyzed methods is able

[8] http://www.mymedialite.net

Table 1: MRR and BinomDiv values for different sizes of target cold-start user profiles.

| | | MRR | | | | | | | | | | BinonDiv | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | #likes | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Books | CB | .086 | .120 | .144 | .160 | .168 | .178 | .199 | .198 | .205 | .214 | .491 | .489 | .487 | .487 | .482 | .487 | .492 | .492 | .493 | .485 |
| | POP | .244 | .246 | .248 | .251 | .252 | .255 | .255 | .261 | .263 | .266 | .674 | .690 | .702 | .703 | .710 | .724 | .732 | .738 | .740 | .746 |
| | INN | .145 | .177 | .216 | .241 | .262 | .277 | .303 | .318 | .331 | .35 | .655 | .674 | .654 | .665 | .672 | .657 | .673 | .674 | .672 | .670 |
| | UNN | .222 | .265 | **.286** | .289 | .29 | .306 | .314 | .323 | .329 | .337 | **.733** | .706 | .716 | .709 | .729 | .722 | .707 | .716 | .716 | .716 |
| | HYB | .247 | .253 | .283 | .286 | .292 | .308 | .322 | .333 | .339 | .349 | .640 | .647 | .661 | .659 | .656 | .671 | .657 | .666 | .669 | .671 |
| | SLIM | .13 | .111 | .194 | .215 | .242 | .286 | .323 | .323 | .335 | **.377** | .724 | .703 | .692 | .713 | .692 | .719 | .701 | .738 | .743 | .694 |
| | SSLIM | .115 | .119 | .199 | .212 | .247 | .271 | .289 | .303 | .315 | .326 | .706 | .722 | .685 | .615 | .695 | .703 | .710 | .665 | .649 | .653 |
| | IMF | .171 | .194 | .235 | .255 | .271 | .29 | .285 | .299 | .307 | .324 | .583 | .606 | .630 | .645 | .657 | .665 | .661 | .662 | .659 | .674 |
| | CMF | .175 | .186 | .249 | .258 | .251 | .285 | .302 | .317 | .327 | .358 | .601 | .640 | .664 | .680 | .683 | .703 | .690 | .708 | .706 | .710 |
| | FMs | .213 | .224 | .223 | .233 | .236 | .24 | .245 | .257 | .253 | .276 | .716 | **.726** | **.749** | **.741** | **.733** | **.740** | **.755** | **.757** | **.755** | **.764** |
| | HeteRec | .218 | .244 | .279 | **.297** | **.316** | **.331** | **.345** | **.353** | **.358** | .366 | .609 | .623 | .653 | .672 | .680 | .692 | .695 | .694 | .692 | .692 |
| | PathRank | **.251** | **.271** | .285 | .292 | .295 | .302 | .305 | .309 | .313 | .317 | .630 | .640 | .650 | .650 | .680 | .680 | .680 | .695 | .708 | .706 |
| Movies | CB | .082 | .107 | .119 | .135 | .144 | .154 | .162 | .169 | .175 | .182 | .298 | .293 | .289 | .281 | .274 | .263 | .256 | .249 | .241 | .236 |
| | POP | .287 | .289 | .292 | .294 | .297 | .299 | .302 | .305 | .308 | .311 | .304 | .336 | .354 | .368 | .378 | .386 | **.393** | **.400** | **.405** | **.410** |
| | INN | .233 | .301 | .336 | .359 | .377 | .39 | .405 | .415 | .423 | .433 | .289 | .308 | .315 | .321 | .323 | .327 | .329 | .332 | .333 | .337 |
| | UNN | **.332** | .320 | .318 | .330 | .348 | .378 | .397 | .405 | .416 | .426 | .360 | **.385** | **.404** | **.392** | **.396** | **.394** | **.393** | .393 | .396 | .395 |
| | HYB | .3 | .322 | .343 | **.366** | **.382** | **.398** | **.413** | **.426** | **.434** | **.446** | .348 | .381 | .389 | .373 | .368 | .367 | .362 | .362 | .362 | .361 |
| | SLIM | .157 | .173 | .236 | .280 | .306 | .333 | .349 | .372 | .390 | .413 | **.392** | .366 | .341 | .347 | .366 | .387 | .366 | .370 | .395 | .406 |
| | SSLIM | .159 | .192 | .249 | .290 | .311 | .338 | .361 | .382 | .394 | .417 | **.392** | .353 | .363 | .372 | .384 | .384 | .361 | .371 | .380 | .377 |
| | IMF | .256 | .291 | .314 | .334 | .348 | .364 | .376 | .389 | .400 | .413 | .299 | .320 | .335 | .344 | .347 | .355 | .358 | .363 | .366 | .368 |
| | CMF | .257 | .297 | .315 | .337 | .352 | .371 | .382 | .391 | .402 | .420 | .288 | .314 | .329 | .337 | .338 | .348 | .355 | .357 | .358 | .363 |
| | FMs | .29 | .321 | .334 | .35 | .358 | .368 | .375 | .386 | .391 | .399 | .357 | .338 | .343 | .348 | .352 | .357 | .353 | .354 | .361 | .362 |
| | HeteRec | .315 | **.346** | **.357** | **.366** | .374 | .382 | .388 | .395 | .401 | .408 | .311 | .328 | .334 | .337 | .341 | .343 | .346 | .348 | .350 | .354 |
| | PathRank | **.333** | .336 | .337 | .340 | .344 | .345 | .350 | .354 | .357 | .361 | .317 | .327 | .336 | .342 | .352 | .353 | .359 | .361 | .366 | .368 |
| Music | CB | .113 | .135 | .151 | .167 | .178 | .187 | .198 | .207 | .215 | .223 | .284 | .286 | .271 | .256 | .242 | .233 | .225 | .217 | .212 | .205 |
| | POP | .337 | .340 | .342 | .345 | .347 | .349 | .352 | .354 | .357 | .359 | .228 | .262 | .282 | .295 | .305 | .313 | .321 | .327 | .333 | .338 |
| | INN | .320 | .391 | .426 | .455 | .474 | .489 | .504 | .518 | .532 | .542 | .200 | .213 | .219 | .223 | .229 | .231 | .235 | .236 | .239 | .240 |
| | UNN | **.422** | .389 | .389 | .419 | .448 | .485 | .503 | .519 | .533 | .546 | .296 | .332 | .348 | .347 | .330 | .317 | .309 | .305 | .301 | .297 |
| | HYB | .356 | .383 | .413 | .443 | .469 | .491 | .505 | .522 | **.536** | .548 | .237 | .260 | .275 | .266 | .265 | .263 | .260 | .260 | .257 | .257 |
| | SLIM | .193 | .184 | .293 | .346 | .388 | .418 | .44 | .468 | .491 | .505 | **.355** | .347 | .322 | .309 | .358 | .290 | .368 | .361 | .350 | .351 |
| | SSLIM | .207 | .249 | .334 | .377 | .413 | .435 | .458 | .477 | .502 | .524 | .294 | .324 | .307 | .330 | .338 | .331 | .334 | .324 | .297 | .371 |
| | IMF | .347 | .396 | .427 | .451 | .471 | .488 | .503 | .516 | .532 | .544 | .196 | .217 | .232 | .241 | .249 | .253 | .256 | .260 | .261 | .265 |
| | CMF | .357 | .397 | .432 | .456 | .476 | **.493** | **.509** | **.525** | **.536** | **.550** | .187 | .210 | .225 | .234 | .243 | .245 | .250 | .250 | .255 | .258 |
| | FMs | .394 | **.427** | **.450** | **.467** | **.480** | **.493** | .504 | .514 | .524 | .533 | .344 | .331 | .315 | .306 | .309 | .305 | .310 | .300 | .306 | .303 |
| | HeteRec | .358 | .395 | .421 | .442 | .463 | .481 | .496 | .513 | .524 | .535 | .227 | .264 | .280 | .288 | .296 | .300 | .302 | .304 | .306 | .306 |
| | PathRank | .410 | .416 | .420 | .424 | .428 | .432 | .436 | .440 | .443 | .447 | .350 | **.380** | **.395** | **.404** | **.410** | **.413** | **.416** | **.419** | **.421** | **.422** |

Table 2: Catalog Coverage and Entropy values for different sizes of target cold-start user profiles.

| | | Catalog Coverage | | | | | | | | | | Entropy | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | #likes | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Books | CB | .346 | .359 | **.345** | **.328** | **.315** | **.307** | **.295** | **.285** | **.277** | **.273** | 9.78 | 10.05 | 9.93 | 9.80 | 9.73 | 9.63 | 9.55 | 9.48 | 9.42 | 9.38 |
| | POP | .003 | .003 | .003 | .003 | .003 | .003 | .004 | .004 | .004 | .004 | 3.37 | 3.40 | 3.43 | 3.46 | 3.48 | 3.51 | 3.54 | 3.56 | 3.58 | 3.60 |
| | INN | .362 | **.374** | .341 | .314 | .291 | .272 | .257 | .237 | .223 | .211 | 9.59 | 9.56 | 9.17 | 8.90 | 8.70 | 8.51 | 8.35 | 8.14 | 8.00 | 7.87 |
| | UNN | .117 | .081 | .073 | .071 | .068 | .065 | .062 | .060 | .058 | .059 | 6.68 | 6.08 | 6.07 | 6.09 | 6.10 | 6.01 | 5.98 | 5.98 | 5.98 | 6.01 |
| | HYB | .102 | .091 | .085 | .080 | .078 | .076 | .073 | .072 | .067 | .069 | 6.76 | 6.59 | 6.56 | 6.51 | 6.51 | 6.51 | 6.45 | 6.44 | 6.39 | 6.41 |
| | SLIM | .398 | .310 | .243 | .186 | .171 | .148 | .144 | .133 | .126 | .115 | 11.24 | 10.89 | 10.57 | 10.20 | 10.05 | 9.79 | 9.69 | 9.54 | 9.42 | 9.30 |
| | SSLIM | **.426** | .357 | .284 | .236 | .207 | .171 | .159 | .142 | .134 | .116 | **11.44** | **11.17** | **10.78** | **10.46** | **10.36** | **10.07** | **9.92** | **9.75** | **9.65** | **9.46** |
| | IMF | .057 | .054 | .051 | .049 | .048 | .048 | .047 | .045 | .046 | .045 | 7.18 | 6.97 | 6.82 | 6.74 | 6.68 | 6.64 | 6.60 | 6.58 | 6.58 | 6.59 |
| | CMF | .098 | .087 | .078 | .075 | .071 | .072 | .070 | .065 | .069 | .065 | 7.81 | 7.51 | 7.28 | 7.18 | 7.13 | 7.05 | 7.06 | 7.02 | 7.07 | 7.02 |
| | FMs | .077 | .086 | .111 | .109 | .109 | .111 | .127 | .118 | .120 | .120 | 6.46 | 6.82 | 7.29 | 7.35 | 7.40 | 7.39 | 7.66 | 7.55 | 7.57 | 7.60 |
| | HeteRec | .245 | .241 | .223 | .199 | .175 | .162 | .141 | .120 | .102 | .092 | 7.99 | 7.81 | 7.50 | 7.17 | 6.86 | 6.67 | 6.44 | 6.21 | 6.03 | 5.92 |
| | PathRank | .182 | .137 | .113 | .093 | .082 | .077 | .069 | .065 | .058 | .055 | 6.98 | 6.32 | 5.94 | 5.63 | 5.46 | 5.37 | 5.25 | 5.19 | 5.11 | 5.06 |
| Movies | CB | .861 | **.822** | **.771** | **.734** | **.698** | **.673** | **.651** | **.631** | **.612** | **.598** | 5.48 | 5.38 | 5.26 | 5.17 | 5.09 | 5.03 | 4.97 | 4.93 | 4.89 | 4.85 |
| | POP | .003 | .003 | .003 | .004 | .004 | .004 | .004 | .004 | .005 | .005 | 1.83 | 1.85 | 1.86 | 1.88 | 1.89 | 1.91 | 1.92 | 1.93 | 1.95 | 1.96 |
| | INN | .856 | .650 | .495 | .394 | .332 | .292 | .260 | .239 | .221 | .206 | 4.84 | 4.30 | 4.00 | 3.81 | 3.69 | 3.59 | 3.52 | 3.47 | 3.42 | 3.38 |
| | UNN | .185 | .173 | .180 | .138 | .122 | .107 | .101 | .097 | .095 | .092 | 3.09 | 3.28 | 3.27 | 3.16 | 3.08 | 3.02 | 3.02 | 3.02 | 3.03 | 3.04 |
| | HYB | .207 | .196 | .173 | .158 | .146 | .135 | .129 | .123 | .120 | .117 | 3.53 | 3.53 | 3.45 | 3.40 | 3.37 | 3.35 | 3.34 | 3.33 | 3.33 | 3.33 |
| | SLIM | .855 | .680 | .591 | .526 | .490 | .462 | .439 | .427 | .419 | .398 | **11.35** | **10.93** | **10.70** | **10.51** | **10.38** | **10.26** | **10.18** | **10.09** | **10.03** | **9.99** |
| | SSLIM | **.891** | .708 | .572 | .504 | .460 | .430 | .410 | .397 | .381 | .369 | **11.35** | **10.93** | 10.62 | 10.41 | 10.27 | 10.12 | 10.04 | 9.96 | 9.90 | 9.84 |
| | IMF | .162 | .156 | .149 | .141 | .137 | .134 | .131 | .130 | .129 | .128 | 4.29 | 4.15 | 4.05 | 3.99 | 3.95 | 3.93 | 3.92 | 3.91 | 3.91 | 3.91 |
| | CMF | .142 | .137 | .132 | .126 | .122 | .120 | .114 | .116 | .111 | .111 | 4.22 | 4.06 | 3.97 | 3.93 | 3.87 | 3.85 | 3.84 | 3.83 | 3.82 | 3.83 |
| | FMs | .036 | .195 | .216 | .224 | .227 | .235 | .237 | .247 | .240 | .244 | 2.28 | 3.52 | 3.65 | 3.74 | 3.78 | 3.82 | 3.86 | 3.89 | 3.91 | 3.93 |
| | HeteRec | .450 | .292 | .216 | .166 | .134 | .115 | .098 | .087 | .077 | .070 | 3.62 | 3.23 | 3.03 | 2.90 | 2.81 | 2.75 | 2.71 | 2.67 | 2.65 | 2.63 |
| | PathRank | .098 | .044 | .025 | .020 | .017 | .015 | .014 | .013 | .013 | .012 | 2.47 | 2.30 | 2.24 | 2.22 | 2.21 | 2.20 | 2.20 | 2.20 | 2.20 | 2.21 |
| Music | CB | .867 | **.835** | **.799** | **.762** | **.732** | **.698** | **.673** | **.649** | **.628** | **.608** | 11.19 | 10.98 | 10.76 | 10.55 | 10.37 | 10.20 | 10.04 | 9.91 | 9.78 | 9.67 |
| | POP | .002 | .002 | .002 | .002 | .003 | .003 | .003 | .003 | .003 | .003 | 3.36 | 3.39 | 3.42 | 3.45 | 3.47 | 3.50 | 3.52 | 3.54 | 3.56 | 3.59 |
| | INN | .841 | .604 | .469 | .398 | .351 | .316 | .294 | .280 | .266 | .256 | 9.84 | 8.79 | 8.28 | 7.97 | 7.77 | 7.62 | 7.51 | 7.43 | 7.36 | 7.31 |
| | UNN | .223 | .207 | .222 | .171 | .159 | .146 | .138 | .137 | .136 | .135 | 7.03 | 7.30 | 7.09 | 6.75 | 6.61 | 6.56 | 6.59 | 6.61 | 6.62 | 6.65 |
| | HYB | .290 | .244 | .209 | .191 | .176 | .166 | .158 | .152 | .147 | .145 | 8.45 | 8.12 | 7.77 | 7.55 | 7.40 | 7.31 | 7.24 | 7.19 | 7.16 | 7.13 |
| | SLIM | **.929** | .746 | .658 | .605 | .550 | .526 | .520 | .504 | .495 | .484 | 11.88 | 11.35 | **11.14** | **10.94** | **10.78** | **10.70** | **10.62** | **10.57** | **10.51** | **10.46** |
| | SSLIM | .702 | .645 | .481 | .428 | .396 | .376 | .352 | .340 | .336 | .322 | **12.23** | **11.48** | 10.99 | 10.73 | 10.56 | 10.45 | 10.36 | 10.29 | 10.24 | 10.19 |
| | IMF | .146 | .139 | .131 | .129 | .125 | .123 | .121 | .119 | .119 | .119 | 8.67 | 8.23 | 7.99 | 7.84 | 7.77 | 7.71 | 7.67 | 7.65 | 7.64 | 7.63 |
| | CMF | .161 | .142 | .135 | .128 | .124 | .124 | .120 | .120 | .119 | .119 | 8.72 | 8.24 | 7.96 | 7.84 | 7.77 | 7.72 | 7.68 | 7.66 | 7.66 | 7.65 |
| | FMs | .051 | .077 | .093 | .103 | .112 | .116 | .122 | .124 | .130 | .133 | 5.32 | 5.94 | 6.30 | 6.48 | 6.65 | 6.74 | 6.85 | 6.92 | 6.99 | 7.05 |
| | HeteRec | .519 | .426 | .362 | .316 | .279 | .253 | .231 | .214 | .201 | .189 | 8.33 | 7.69 | 7.25 | 6.95 | 6.74 | 6.58 | 6.45 | 6.36 | 6.28 | 6.21 |
| | PathRank | .092 | .041 | .025 | .021 | .018 | .016 | .015 | .014 | .014 | .014 | 4.81 | 4.41 | 4.29 | 4.25 | 4.22 | 4.22 | 4.22 | 4.22 | 4.22 | 4.23 |

to overcome all the others in terms of all the metrics. Analyzing MRR and BinonDiv together, INN seems to achieve the best trade-off with almost all the user profile size, followed by HeteRec from 5 to 10 likes. Analyzing MRR and catalog coverage, INN and HeteRec provide the best trade-offs for all the profile sizes, followed by SLIM and SSLIM. Considering the trade-off between MRR and entropy, SLIM and SSLIM provide the best results, followed by INN and HeteRec. Even though SLIM and SSLIM provide good catalog coverage and the best entropy values, it is important to note that their recommendation accuracy is very low for the majority of cold users (i.e., with less than 5 likes).

Summing up, graph-based methods obtain the best results in the books domain with best accuracy and good coverage, but in different situations: PathRank is better with less likes (strong cold-start), and HeteRec overcomes it where more likes are available (weak cold-start); they, however, provide less diversified list of recommendations (individual diversity) with respect to most of the other methods. The three factorization-based models (IMF, CMF, FMs) are not particularly effective in this domain. We also notice that using metadata information leads to better recommendations. In particular, we can see that HYB beats UNN in seven out of ten cases. Moreover, CMF gives the same importance to user preferences and item metadata, obtaining the better accuracy with the trade-off parameter $\gamma = 0.5$.

#### 5.1.2 Movies

**Accuracy.** PathRank and UNN provide the most accurate recommendations to users with 1 like, closely followed by HeteRec. Until 4 likes are available, HeteRec yields the best performance. As more likes are observed, the HYB method consistently outperforms the rest, closely followed by INN and UNN. Regarding CMF and SLIM, we observe a similar behavior to the books domain. CB is always the worst method, while SLIM and SSLIM are able to overcome POP only in the case of users with more than 4 likes. On a side note, as FMs beat IMF with few likes, item metadata results also valuable in MF-based models, especially in the most cold-start situations (likes less than 5).
**Individual Diversity.** UNN provides the most diversified list of recommendations in almost all the configurations, followed by HYB, SLIM and SSLIM. CB always results with the worst diversity, except for users with 1 like.
**Aggregate Diversity.** In terms of catalog coverage, SSLIM is again the best method with 1 like, while in the other configurations CB overcomes all the other methods, followed by SLIM, SLIM, INN. Recalling that POP should be always the worst method in terms of coverage, we note that PathRank provides results not significantly greater than POP, and is hence unable to cover the catalog.
**Overall Analysis.** Again, none of the analyzed methods arise as the best solution for all the recommendation quality dimensions. Analyzing MRR and BinonDiv together, UNN seems to obtain the best trade-off, followed by HYB; while CB is always the worst choice. Among the factorization-based models, IMF and FMs show very similar trends, while CMF provide slightly less diversified recommendation lists. The two graph-based methods, in contrast, are

the most accurate methods for users with less than 5 likes, providing individual diversity results on the average. Analyzing MRR and catalog coverage, INN provides one of the best trade-offs between the two dimensions considering almost all the profile sizes. However, SLIM and SSLIM result the best choice for users with more than 7 likes. Again, SLIM and SSLIM are the best methods when MRR and entropy are considered together.

Once again we note that content information is especially beneficial in terms of accuracy, while the other quality dimensions do not receive a particular improvement. We also observe that graph-based methods are better than the MF-based approaches in terms of accuracy and individual diversity when very few likes are observed, but they strongly penalize the aggregate diversity.

#### 5.1.3 Music

**Accuracy.** As for movies domain, UNN provides the most accurate suggestions for users with only 1 like. Then, FMs and CMF overcome almost all the other methods in the other user profile size configurations. In particular, FMs result the best approach from 2 to 6 likes, and CMF from 6 to 10. PathRank seems to be still competitive in the extreme cold-start scenario (1 and 2 likes). Therefore, we can confirm again that hybrid recommender systems are effective for generating accurate recommendations. In particular, we see that CMF and FMs always overcome IMF, which only uses collaborative information, and SLIM with side information (SSLIM) is better that SLIM.
**Individual Diversity.** In contrast to the other two domains, PathRank is able to overcome all the other methods in terms of BinomDiv. FMs is the best choice among the factorization-based methods, while IMF and CMF provide the less diversified list of recommendations compared to all the other methods.
**Aggregate Diversity.** Again CB overcomes all the other methods in terms of catalog coverage, except for the case of 1 like, where SLIM is still the best approach. SLIM follows CB from 2 to 10, while SSLIM and INN place themselves down. Conversely, all the three factorization-based methods and PathRank provide the lowest values of catalog coverage.
**Overall Analysis.** PathRank, UNN and FMs show the best trade-off between MRR and BinonDiv from 1 to 5 likes; while SSLIM, UNN, and FMs seem generally to be the best methods from 5 to 10 likes, except from 8 to 10 likes, where SLIM provides the best trade-off. Analyzing MRR and catalog coverage together, INN is again one of the best methods; when users have from 7 to 10 likes, SLIM results a good alternative to INN, with less accuracy but better catalog coverage. Considering the trade-off between MRR and entropy, INN, CMF, and IMF generally provide a good balance, in particular with the most extreme cold-start scenarios. With more than 7 likes, SLIM and SSLIM deserve to be taken into account, since they obtain very high entropy with slight loss of accuracy.

We can conclude that matrix factorization models perform better in this domain and are also able to exploit item metadata, since CMF and FMs beat IMF in each configuration. In particular, CMF is able to adequately combine

likes and item metadata. As the optimal trade-off parameter for CMF is 0.1 in this domain, this method gives more importance to the content information as opposed to user preferences, demonstrating that metadata is valuable in the cold-start scenario.

# 6 Conclusions

Providing relevant suggestions of items for cold-start users is a well-known problem in recommender systems. In this paper we conducted a comparison of different hybrid recommendation approaches that jointly exploit user ratings and semantic item metadata extracted from DBpedia. Specifically, we evaluated a number of graph-based and matrix factorization algorithms in the top-N recommendation task with positive-only feedback, using a Facebook likes dataset covering three distinct domains —namely books, movies and music— in terms of not just accuracy, but also regarding individual and aggregate diversity. The achieved results shown that by exploiting semantic information about the items, the models are able to provide relevant recommendations even for users with very few likes, hence addressing the cold-start problem, while diversity does not always benefit from the integration of semantic information into the hybrid approaches. In the books domain, the analyzed graph-based models were generally more effective compared to the factorization-based models and baselines. In the movies domain, graph-based models provided better results than factorization-based models particularly in extreme cold-start situations, while no relevant differences emerged for moderate cold-start users. Finally, factorization-based models were the best approaches in the music domain in almost all the cold-start scenarios.

# References

# References


[1] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, N. Oliver, and A. Hanjalic, "CLiMF: Learning to maximize reciprocal rank with collaborative less-is-more filtering," in *Proceedings of the Sixth ACM Conference on RecSys*, pp. 139–146, 2012.

[2] P. Castells, N. J. Hurley, and S. Vargas, "Novelty and diversity in recommender systems," in *Recommender Systems Handbook*, pp. 881–918, Boston, MA: Springer US, 2015.

[3] M. D. Ekstrand, F. M. Harper, M. C. Willemsen, and J. A. Konstan, "User perception of differences in recommender algorithms," in *Proceedings of the 8th ACM Conference on RecSys*, pp. 161–168, ACM, 2014.

[4] G. Adomavicius and Y. Kwon, "Improving aggregate recommendation diversity using ranking-based techniques," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 5, pp. 896–911, 2012.

[5] D. Fleder and K. Hosanagar, "Blockbuster culture's next rise or fall: The impact of recommender systems on sales diversity," *Management science*, vol. 55, no. 5, pp. 697–712, 2009.

[6] P. Tomeo, I. Fernández-Tobías, T. Di Noia, and I. Cantador, "Exploiting linked open data in cold-start recommendations with positive-only feedback," in *CERI '16*, 2016.

[7] N. Rubens, M. Elahi, M. Sugiyama, and D. Kaplan, "Active learning in recommender systems," in *Recommender Systems Handbook*, pp. 809–846, Springer, 2015.

[8] M. J. Pazzani, "A framework for collaborative, content-based and demographic filtering," *Artificial Intelligence Review*, vol. 13, no. 5, pp. 393–408, 1999.

[9] M. Braunhofer, M. Elahi, and F. Ricci, *User Personality and the New User Problem in a Context-Aware Point of Interest Recommender System.* Springer International Publishing, 2015.

[10] I. Cantador, I. Fernández-Tobías, S. Berkovsky, and P. Cremonesi, "Cross-domain recommender systems," in *Recommender Systems Handbook*, pp. 919–959, Springer, 2015.

[11] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norick, and J. Han, "Personalized entity recommendation: A heterogeneous information network approach," in *Proceedings of the 7th ACM WSDM*, pp. 283–292, ACM, 2014.

[12] S. Lee, S. Park, M. Kahng, and S.-g. Lee, "PathRank: A novel node ranking measure on a heterogeneous graph for recommender systems," in *Proceedings of the 21st ACM CIKM '12*, pp. 1637–1641, 2012.

[13] A. P. Singh and G. J. Gordon, "Relational learning via collective matrix factorization," in *Proceedings of the 14th ACM SIGK KDD*, pp. 650–658, 2008.

[14] S. Rendle, "Factorization machines," in *Proceedings of the 2010 IEEE ICDM '10*, pp. 995–1000, 2010.

[15] M. de Gemmis, P. Lops, C. Musto, F. Narducci, and G. Semeraro, *Semantics-Aware Content-Based Recommender Systems*, pp. 119–159. Springer US, 2015.

[16] P. T. Nguyen, P. Tomeo, T. Di Noia, and E. Di Sciascio, "Content-based recommendations via dbpedia and freebase: A case study in the music domain," in *The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part I*, pp. 605–621, 2015.

[17] B. Heitmann and C. Hayes, "Using linked data to build open, collaborative recommender systems," in *In: AAAI Spring Symposium: Linked Data Meets Artificial Intelligence.*, 2010.

[18] C. Musto, G. Semeraro, P. Lops, M. de Gemmis, and F. Narducci, *Leveraging Social Media Sources to Generate Personalized Music Playlists*, pp. 112–123. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012.

[19] T. Di Noia, R. Mirizzi, V. C. Ostuni, D. Romito, and M. Zanker, "Linked open data to support content-based recommender systems," in *I-SEMANTICS 2012 - 8th International Conference on Semantic Systems, I-SEMANTICS '12, Graz, Austria, September 5-7, 2012*, pp. 1–8, 2012.

[20] H. Khrouf and R. Troncy, "Hybrid event recommendation using linked data and user diversity," in *Proceedings of the 7th ACM Conference on Recommender Systems*, RecSys '13, pp. 185–192, ACM, 2013.

[21] C. Musto, G. Semeraro, P. Lops, and M. de Gemmis, "Combining distributional semantics and entity linking for context-aware content-based recommendation," in *User Modeling, Adaptation, and Personalization - 22nd International Conference, UMAP 2014, Aalborg, Denmark, July 7-11, 2014. Proceedings*, pp. 381–392, 2014.

[22] J. Rosati, P. Ristoski, T. Di Noia, R. D. Leone, and H. Paulheim, "RDF graph embeddings for content-based recommender systems," in *Proceedings of the 3rd Workshop on New Trends in Content-Based Recommender Systems co-located with ACM Conference on Recommender Systems*, pp. 23–30, 2016.

[23] S. Oramas, V. C. Ostuni, T. Di Noia, X. Serra, and E. D. Sciascio, "Sound and music recommendation with knowledge graphs," *ACM TIST*, vol. 8, no. 2, pp. 21:1–21:21, 2017.

[24] T. D. Noia, V. C. Ostuni, P. Tomeo, and E. D. Sciascio, "SPrank: Semantic path-based ranking for top-n recommendations using linked open data," *ACM TIST*, vol. 8, no. 1, pp. 9:1–9:34, 2016.

[25] C. Shi, X. Kong, Y. Huang, P. S. Yu, and B. Wu, "Hetesim: A general framework for relevance measure in heterogeneous networks.," *CoRR*, vol. abs/1309.7393, 2013.

[26] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, "Pathsim: Meta path-based top-k similarity search in heterogeneous information networks," in *In VLDB 11*, 2011.

[27] C. Bizer, T. Heath, and T. Berners-Lee, "Linked data - the story so far," *International Journal on Semantic Web and Information Systems*, vol. 5, no. 3, pp. 1–22, 2009.

[28] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, pp. 30–37, Aug. 2009.

[29] S. Funk, "Netflix update: Try this at home," in *http://sifter.org/ simon/journal/20061211.html*, 2006.

[30] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *ICDM '08*, pp. 263–272, 2008.

[31] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl, "Application of dimensionality reduction in recommender system - a case study," in *In ACM WebKDD Workshop*, 2000.

[32] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, pp. 263–272, IEEE Computer Society, 2008.

[33] A. Bellogin, P. Castells, and I. Cantador, "Precision-oriented evaluation of recommender systems: An algorithmic comparison," in *RecSys '11*, pp. 333–336, 2011.

[34] D. Kluver and J. A. Konstan, "Evaluating recommender behavior for new users," in *RecSys '14*, pp. 121–128, 2014.

[35] S. Vargas, L. Baltrunas, A. Karatzoglou, and P. Castells, "Coverage, redundancy and size-awareness in genre diversity for recommender systems," in *RecSys '14*, pp. 209–216, 2014.

[36] X. Ning and G. Karypis, "SLIM: Sparse linear methods for top-n recommender systems," in *Proceedings of the IEEE 11th ICDM*, pp. 497–506, 2011.

[37] X. Ning and G. Karypis, "Sparse linear methods with side information for top-n recommendations," in *RecSys '12*, pp. 155–162, 2012.