Inferring User Decision-Making Processes in Recommender Systems with Knowledge Graphs

Discussion Paper

Vito Walter Anelli[1], Tommaso Di Noia[1],
Eugenio Di Sciascio[1], Antonio Ferrara[1] and Alberto Carlo Maria Mancino[1]

[1]*Polytechnic University of Bari, via Orabona, 4, 70125 Bari, Italy*

Abstract

This paper proposes a sparse factorization approach, KGFlex, that represents each item feature as an embedding. With KGFlex, the user-item interactions are a factorized combination of the item features relevant to the user. An entropy-driven module drives the training considering only the feature involved in the user's decision-making process. Extensive experiments confirm the approach's effectiveness, considering the ranking accuracy, diversity, and induced bias. The public implementation of KGFlex is available at https://split.to/kgflex.

Keywords

Recommender systems, Information Retrieval, feature factorization, entropy, knowledge graphs

1. Introduction

At least once, everyone asked a friend to suggest a movie to watch. This natural and yet effective behavior of obtaining recommendations from similar people is the foundation of Collaborative Filtering (CF) recommendation techniques. In fact, it is their remarkable accuracy that helped Recommender Systems getting famous. Popular examples of CF are Matrix Factorization [1], Nearest Neighbors, and the recent Deep Learning models [2]. Despite the notable performance, these techniques have some severe limitations: they need a considerable number of transactions to work, and they fail to suggest fresh items. On the other side of the coin, there are Content-Based Filtering (CBF) methods. At the cost of lower accuracy, not only are they interpretable, but they also succeed in suggesting fresh items. However, these algorithms are affected by the overspecialization problem, and they struggle to suggest items different from the user history. To benefit from both approaches and alleviate the drawbacks, researchers integrated into Collaborative Filtering the side information used in content-based approaches such as tags [3],

SEBD 2022: The 30th Italian Symposium on Advanced Database Systems, June 19-22, 2022, Tirrenia (PI), Italy
vitowalter.anelli@poliba.it (V. W. Anelli); tommaso.dinoia@poliba.it (T. Di Noia); eugenio.disciascio@poliba.it (E. Di Sciascio); antonio.ferrara@poliba.it (A. Ferrara); alberto.mancino@poliba.it (A. C. M. Mancino)
https://sisinflab.poliba.it/people/vito-walter-anelli/ (V. W. Anelli);
https://sisinflab.poliba.it/people/tommaso-di-noia/ (T. Di Noia);
https://sisinflab.poliba.it/people/eugenio-di-sciascio/ (E. Di Sciascio);
https://sisinflab.poliba.it/people/antonio-ferrara/ (A. Ferrara);
https://sisinflab.poliba.it/people/alberto-carlo-maria-mancino/ (A. C. M. Mancino)

demographic data [4], structured knowledge [5]. However, rarely do these models express their full potential since they combine large and dense models with hundreds or thousands of features, and they are computationally expensive. Among the various information sources, Knowledge Graphs ($\mathcal{KG}s$) are gaining momentum. This work discusses KGFlex [6], a sparse embedding model that extracts facts and knowledge from publicly available knowledge graphs to describe the catalog items. The underlying factorization model shapes the user interactions with item features through low-dimensionality embeddings. KGFlex examines the user-specific decision-making process of consuming or not consuming an item to weight feature embeddings employing an entropy-based strategy. Consequently, the user profile only comprises an individual representation of each relevant feature. To evaluate the efficacy of KGFlex, we conduct extensive experiments on two different publicly available datasets extracting content information from DBpedia[1][2]. The results show that KGFlex has competitive accuracy performance, and at the same time, generates highly diversified recommendations with a low induced bias.

2. Knowledge Graph Sparse Embeddings

KGFlex exploits the knowledge encoded in a knowledge graph as side information to characterize both items and users. One of the main assumptions is that users decide to enjoy an item based on a subset of its characteristics, implying that not all the item features are equally important. In the following, we show how KGFlex describes each user and item with a set of features. Taking a cue from information theory KGFlex exploits the notion of information gain to measure the relevance of a feature for a user in deciding to consume or not an item.

2.1. From Knowledge Graphs to Decision-Making

In KGFlex, items and users are characterized by sets of features extracted from a $\mathcal{KG}$, a knowledge base of semantically linked machine-understandable data [7]. During the years, $\mathcal{KG}s$ have gained more and more success thanks to the Linked Data initiative [8]. Today we can benefit from $1{,}527$ different $\mathcal{KG}s$ connected in the so-called Linked Open Data Cloud[3].

A knowledge graph $\mathcal{KG}$ can be represented as pairs of entities linked to each other by binary relations. Each connection in $\mathcal{KG}$ is denoted by $\sigma \xrightarrow{\rho} \omega$, where σ is a subject entity, ρ is a directed relation (predicate), and ω is an object entity. Hereinafter, we generalize the previous notion to multi-hop predicates (i.e., considering chains of predicates that connect two entities at a higher depth). Let n-hop predicate be defined as $\rho = \langle \rho_1, ..., \rho_n \rangle$ if $\sigma \xrightarrow{\rho_1} \omega_1 \xrightarrow{\rho_2} ... \xrightarrow{\rho_n} \omega_n \in \mathcal{KG}$. For convenience, $h(\rho) = n$ for $\rho : \sigma \xrightarrow{\rho} \omega_n \in \mathcal{KG}$ denotes the depth of the predicate chain, and when no confusion arises we will use $\sigma \xrightarrow{\rho} \omega$ to denote a generic chain with $h(\rho) \geq 1$.

2.1.1. Extraction of Item and User Features from Knowledge Graph

Given a collection of items $\mathcal{I}$ and a knowledge graph $\mathcal{KG}$, we assume each element in $i \in \mathcal{I}$ has a mapping to a corresponding entity in $\mathcal{KG}$. Under this assumption, an item i can be explored,

[1] http://dbpedia.org
[2] https://github.com/sisinflab/LinkedDatasets
[3] https://lod-cloud.net/datasets

at depth n, to identify the set $\mathcal{F}_i^{(n)}$ of the semantic features describing it:

$$\mathcal{F}_i^{(n)} = \{\langle \rho, \omega \rangle \mid i \xrightarrow{\rho} \omega \in \mathcal{KG}, h(\rho) \in \{1, \ldots, n\}\}. \tag{1}$$

At this stage, feature filtering, graph pruning, and semantic feature selection techniques [9] can be applied to control the computational and memory load and improve system performance.

We describe each user $u \in \mathcal{U}$ with the set $\mathcal{F}_u$ of the features representing the items $\mathcal{I}_u \subseteq \mathcal{I}$ enjoyed by u:

$$\mathcal{F}_u^{(n)} = \bigcup_{i \in \mathcal{I}_u} \mathcal{F}_i^{(n)}. \tag{2}$$

Finally, we define the overall set $\mathcal{F}^{(n)}$ of features in the system:

$$\mathcal{F}^{(n)} = \bigcup_{i \in \mathcal{I}} \mathcal{F}_i^{(n)}. \tag{3}$$

In the following, for convenience, the (n) superscript is omitted whenever it is not relevant in the context.

2.1.2. Information Gain of User Features

In information theory, entropy is used to measure the uncertainty of a random variable.

In particular, the entropy $H(V)$ of a random variable V with k possible values in $\{v_1, \ldots, v_k\}$ is defined as:

$$H(V) = -\sum_{i=1}^{k} P(V = v_i) \log_2 P(V = v_i). \tag{4}$$

It is straightforward to check that a coin that always comes up heads has *zero* entropy, while a fair coin equally likely to come up heads or tails when flipped has entropy 1. Notably, if B_q is a binary random variable that is true with probability q, we have $H(B_q) = -q \log_2 q - (1-q) \log_2 (1-q)$. For instance, given a dataset $\mathcal{D}$ of training samples in the form $(\mathbf{x}, y)$, with $\mathbf{x} \in \mathbb{R}^F$ and $y \in \{0,1\}$ is a binary variable that is true with probability q, its entropy is $H(\mathcal{D}) = H(y_q)$. As a consequence, a dataset with a balanced number of positive and negative samples has entropy 1.

In this context, the information gain $IG(\mathcal{D}, x_d)$ measures the expected reduction in information entropy obtained from the observation of one of the attributes x_d in $\mathbf{x}$:

$$IG(\mathcal{D}, x_d) = H(\mathcal{D}) - H(\mathcal{D}|x_d), \tag{5}$$

where $H(\mathcal{D}|x_d)$ is the expected entropy of $\mathcal{D}$ conditioned on x_d. If x_d can assume k distinct values $x_{d,1}, \ldots x_{d,k}$ with a categorical probability distribution,the dataset $\mathcal{D}$ is partitioned into k mutually exclusive subsets and the following conditioned entropy is determined:

$$H(\mathcal{D}|x_d) = \sum_{i=1}^{k} P(x_d = x_{d,i}) H(\mathcal{D}|x_d = x_{d,i}). \tag{6}$$

Since the information gain defined in Eq. (5) returns a measure of the importance of a single attribute in distinguishing positive from negative examples in a dataset, we build, for each user u, a balanced dataset $\mathcal{D}_u$ with all the consumed items from $\mathcal{I}_u$ and the same amount of negative items randomly picked up from $\bigcup_{v\in\mathcal{U},v\neq u}\mathcal{I}_v \setminus \mathcal{I}_u$. Each sample in $\mathcal{D}_u$ is provided with a set of binary variables corresponding to the features in $\mathcal{F}_u$. Each variable indicates, for each item i in $\mathcal{D}_u$, the presence ($f=1$) or the absence ($f=0$) of the corresponding feature in the set $\mathcal{F}_i$. According to the definition, $H(\mathcal{D}_u)=1$. Therefore, the information gain for each feature $f\in\mathcal{F}_u$ can be computed using the dataset $\mathcal{D}_u$. Let p_{uf} be the number of positive samples in $\mathcal{D}_u$ for which $f=1$, n_{uf} the number of negative samples for which the same feature is present, and $t_{uf}=p_{uf}+n_{uf}$. Analogously, $p_{u\neg f}=|\mathcal{I}_u|-p_{uf}$ is the number of positive samples with $f=0$, $n_{u\neg f}=|\mathcal{I}_u|-n_{uf}$ is the number of negative samples with $f=0$, and $t_{u\neg f}=p_{u\neg f}+n_{u\neg f}$. Following Eqq. (5) and (6):

$$IG(\mathcal{D}_u,f)=1-H(\mathcal{D}_u|f=1)-H(\mathcal{D}_u|f=0), \tag{7}$$

$$H(\mathcal{D}_u|f=1)=\frac{t_{uf}}{|\mathcal{D}_u|}\left(-\frac{p_{uf}}{t_{uf}}\log_2\frac{p_{uf}}{t_{uf}}-\frac{n_{uf}}{t_{uf}}\log_2\frac{n_{uf}}{t_{uf}}\right), \tag{8}$$

$$H(\mathcal{D}_u|f=0)=\frac{t_{u\neg f}}{|\mathcal{D}_u|}\left(-\frac{p_{u\neg f}}{t_{u\neg f}}\log_2\frac{p_{u\neg f}}{t_{u\neg f}}-\frac{n_{u\neg f}}{t_{u\neg f}}\log_2\frac{n_{u\neg f}}{t_{u\neg f}}\right). \tag{9}$$

We finally associate a weight $k_{uf}=IG(\mathcal{D}_u,f)$ to each pair of user u and feature f to represent the influence of a feature —in the view of the user— in the prediction of user-item interactions.

2.2. Sparse Interaction of Feature Embeddings for Prediction

KGFlex models the features in $\mathcal{F}$ as collaboratively learned embeddings in a latent space. Since KGFlex promotes the idea of having user fine-tuned versions of the same model, we have both a global representation of the features in $\mathcal{F}$ and a personal view, for each user u, of the features in $\mathcal{F}_u\subseteq\mathcal{F}$. Notably, the model is structured into two distinct parts. On the one hand, KGFlex keeps a set $\mathcal{G}$ of global trainable embeddings and biases shared among all the users, with $\mathcal{G}=\{(\mathbf{g}_f\in\mathbb{R}^E, b_f\in\mathbb{R}),\ \forall f\in\mathcal{F}\}$. On the other hand, each user in KGFlex also has his/her personal representation of the features he/she interacted with, i.e., the features in $\mathcal{F}_u$. These embeddings are collected within the set $\mathcal{P}^u$, defined as $\mathcal{P}^u=\{\mathbf{p}^u_f\in\mathbb{R}^E,\ \forall f\in\mathcal{F}_u\}$. Then, the inner product between the personal representation $\mathbf{p}^f_u$ and the global representation $\mathbf{g}_f$, plus a bias value b_f, estimates the affinity of user u to feature f. The sum of such affinities for all the features in $\mathcal{F}_{ui}=\mathcal{F}_u\cap\mathcal{F}_i$, weighted according to the pre-computed entropy-based coefficients, estimates the interaction $\hat{x}_{ui}$ between user u and item i:

$$\hat{x}_{ui}=\sum_{f\in\mathcal{F}_{ui}}k_{uf}(\mathbf{p}^u_f\mathbf{g}_f+b_f). \tag{10}$$

Eq. (10) encodes the strategy KGFlex exploits to handle thousands of model features. In fact, it takes advantage of user profile to involve only a small subset of them in the estimate of the user-item affinity.

To learn the model parameters, KGFlex adopts Bayesian Personalized Ranking (BPR), the most common pair-wise Learning to Rank strategy, based on a maximum posterior estimator. Given a training set $\mathcal{T}=\{(u,i^+,i^-)\mid i^+\in\mathcal{I}_u\wedge i^-\in\mathcal{I}\setminus\mathcal{I}_u,\forall u\in\mathcal{U}\}$, BPR optimizes the loss $L=\sum_{(u,i^+,i^-)\in\mathcal{T}}\ln\sigma(\hat{x}_{ui^+}-\hat{x}_{ui^-})$, with the assumption that a user u prefers a consumed item i^+

over a non-consumed item i^{-}. The model parameters are learnt with an optimization algorithm like SGD, as described in Rendle et al. [10]. To that aim, we derive:

$$\frac{\partial}{\partial\theta}\hat{x}_{ui} = \begin{cases} k_{uf}\mathbf{g}_f & \text{if } \theta = \mathbf{p}_f^u, \\ k_{uf}\mathbf{p}_f^u & \text{if } \theta = \mathbf{g}_f, \\ k_{uf} & \text{if } \theta = b_f, \\ 0 & \text{else.} \end{cases} \tag{11}$$

3. Experimental Setup

In the following, we present a discussion about datasets, preprocessing, baselines, and evaluation protocol that guided the experimentation of KGFlex.

3.1. Datasets and Filtering

The evaluation of the performance of KGFlex is conducted on two well-known datasets: *Yahoo! Movies* and *Facebook Books*. The datasets have been binarized, retaining ratings of 3 or higher. To ensure a fair comparison with the baselines, an iterative 10-core, and 5-core preprocessing procedure is performed on the first and second datasets. The items have been described with a set of semantic features retrieved through a KG exploration at depth 2 of the DBpedia $\mathcal{KG}$ using a public DBpedia URI mapping. Some features (based on their 1-hop predicate) have not been considered since they do not provide useful information [9]: *dbo:wikiPageWikiLink*, *owl:sameAs*, *rdf:type*, *gold:hypernym*, *rdfs:seeAlso*, *dbp:wordnet_type*, *dbo:wikiPageExternalLink*, *dbo:thumbnail*, *prov:wasDerivedFrom*, and *dbp:wikiPageUsesTemplate*. Therefore, we removed the features associated with less than ten items. Finally, we kept the user's 100 most informative features from the 1- and 2- hop exploration to reduce the computational costs.

3.2. Baselines, Evaluation Protocol and Metrics

To assess the effectiveness of KGFlex, we compare it with BPR-MF [10], a latent factor model based on the same pair-wise optimization criterion used in KGFlex, a batch version of Rendle et al. [11] MF, NeuMF [12], and kaHFM [5], a factorization-based model making use of knowledge graphs. We have chosen the all unrated items protocol using the hold-out 80-20 splitting strategy, which considers as candidate items all the items not rated by the user. All the models have been tested in 10 different configurations of hyperparameters with the Bayesian hyperparameter optimization search. For the sake of reproducibility, we provide our code and a working configuration file for the Elliot framework [13, 14]. Our goal is to assess the **accuracy** and **beyond-accuracy** performance of KGFlex, along with its **fairness** properties with respect to the popularity of the items. In detail, we have measured the **recommendation accuracy** with nDCG [11], also used as the validation metric. Then, we have evaluated the **diversity**, adopting Item Coverage (IC) [15] and Gini Index (Gini) [16]. Finally, three **bias metrics** have been used to evaluate how the algorithms consider the items from the long-tail: ACLT [17], PopREO and PopRSP, specific applications of RSP, and REO [18]. PopREO estimates the equal opportunity of items, encouraging the True Positive Rate of popular and unpopular items to be the same. PopRSP measures statistical parity, assessing whether the ranking probability distributions for popular and unpopular items are the same in the recommendation.

Table 1
Comparison of KGFlex with baselines (in boldface) and other reference algorithms. Among the baselines, the best result is in boldface, the second-best result is underlined. For all the metrics, the cutoff is 10.

a) Yahoo! Movies

	nDCG	IC	Gini	ACLT	PopREO	PopRSP
Random	0.00960	1050	0.84956	5.52026	0.09847	0.00980
Most Popular	0.15850	49	0.01263	0.00000	1.00000	1.00000
VSM	0.04777	370	0.05245	3.11768	0.49588	0.45751
Item-kNN	0.30739	745	0.15826	0.98842	0.70585	0.83466
MultiVAE	0.23696	399	0.09136	0.23473	0.85433	0.96127
BPR-MF	0.18571	151	0.02191	0.00064	0.99543	0.99989
MF	<u>0.28971</u>	455	0.09024	0.08232	0.87345	0.98645
NeuMF	0.09184	50	0.01134	0.00064	1.00000	0.99989
kaHFM	**0.30055**	<u>757</u>	<u>0.16591</u>	<u>0.46238</u>	<u>0.76103</u>	<u>0.92339</u>
KGFlex	0.24640	**851**	**0.28015**	**2.14469**	**0.44768**	**0.63355**

b) Facebook Books

	nDCG	IC	Gini	ACLT	PopREO	PopRSP
Random	0.00690	782	0.86167	5.26045	0.09794	0.00749
Most Popular	0.09393	16	0.01265	0.00000	1.00000	1.00000
VSM	0.03617	523	0.18874	3.80558	0.22616	0.29958
Item-kNN	0.12903	769	0.37520	2.22524	0.48852	0.59861
MultiVAE	0.11914	620	0.18279	0.46368	0.77695	0.91818
BPR-MF	0.09473	17	0.01318	0.00000	1.00000	1.00000
MF	<u>0.09557</u>	87	0.02376	0.00000	1.00000	1.00000
NeuMF	0.07142	17	0.01245	0.00000	1.00000	1.00000
kaHFM	**0.12667**	<u>540</u>	<u>0.13866</u>	<u>0.32942</u>	<u>0.87663</u>	<u>0.94197</u>
KGFlex	0.08526	**606**	**0.30703**	**3.02641**	**0.15210**	**0.44852**

3.3. Results

3.3.1. Evaluating the Overall Performance

Table 1a depicts the evaluation outcome for the aforementioned metrics with a cutoff of 10 on the *Yahoo! Movies* dataset. KGFlex shows satisfactory accuracy results, being outperformed only by kaHFM and MF. It is noteworthy that KGFlex significantly outperforms BPR-MF, although both are learned with a pair-wise BPR optimization, thus underlining the beneficial role of the extracted knowledge. Moreover, examining the item coverage and Gini values, we note the high degree of personalization provided by KGFlex. We link this result to the personalized view of the knowledge granted by the framework. Moreover, in KGFlex the collaborative signal on explicit user interests ensures to recommend diverse items among the ones sharing characteristics of interest for the user. The aforementioned behavior is not confirmed in *Facebook Books* (see Table 1b). Indeed, the accuracy results seem to remain below the performance of other factorization-based approaches. However, the diversity results show how BPR-MF, MF, and NeuMF may have been flooded by popularity signal, which led them to perform poorly regarding the item coverage and Gini metrics. Instead, KGFlex does not suffer from this problem and approaches the superior performance of Item-kNN in terms of diversity.

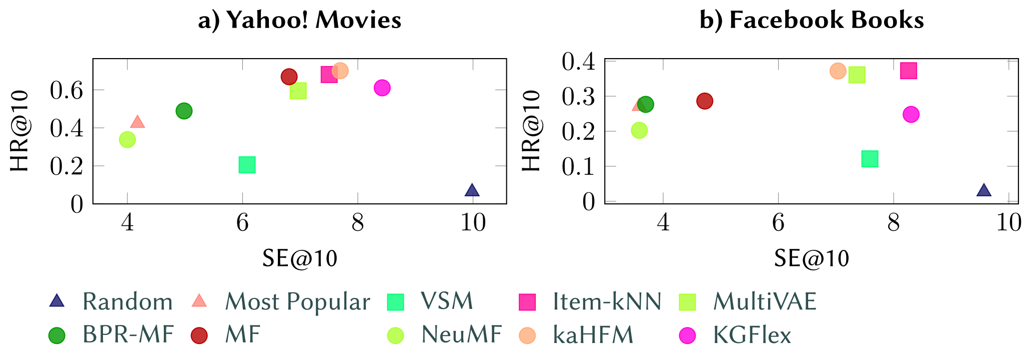

Figure 1: Accuracy vs. distributional diversity. The plots show the value of HR@10 against SE@10: the closer to the top-right corner the better.

3.3.2. Investigating the Accuracy/Personalization Trade-Off

What we have analytically observed is substantially confirmed in Figure 1. These graphs show the joint behavior of KGFlex on accuracy and distributional diversity by analyzing the value of Hit Ratio (HR) with respect to the Shannon Entropy (SE) statistics, which measures how diversely distributed are the recommended items. Among factorization-based approaches, KGFlex approaches the right-top margin to a greater extent, remarking its capability of providing highly personalized recommendations, probably due to the joint operation of the global and the personal views of the same features. The kaHFM model usually is the second-best model, while the other approaches seem to perform very poorly in at least one dimension or do not have a stable position when varying the dataset.

3.3.3. Analyzing the Algorithmic and Popularity Bias

Oftentimes, unpopular items are not recommended and remain underrepresented [19], thus causing a fairness issue for items and an inappropriate recommendation for users who do not prefer very popular items. While there is a wide range of approaches in literature aiming to reduce the recommendation biases [20, 21, 22, 23, 24, 25, 26] , we study whether KGFlex is inherently resilient to algorithmic bias. From Tables 1a and 1b, it is noteworthy that KGFlex always outperforms all the other factorization-based approaches and generally outperforms the other approaches. The value of ACLT (the higher the better) is comparable with the value obtained by VSM. This result is further supported by the values of PopREO and PopRSP (the smaller the better). Concerning those metrics, KGFlex and VSM continue to grant the less biased recommendations. Interestingly, while both exploit the same optimization criterion, we notice how KGFlex consistently improves BPR-MF, which is known to be vulnerable to imbalanced data and to produce biased recommendations [18].

4. Conclusion

We introduced KGFlex, a promising knowledge-aware approach to generate recommendations from implicit feedback. KGFlex has demonstrated its ability to take the best from content-based and factorization-based recommendation approaches. Moreover, thanks to the high degree of expressiveness provided by the personalized representation of content information, KGFlex guarantees satisfactory and diverse recommendations and resilience to algorithmic bias.

References

[1] Y. Koren, R. M. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, IEEE Computer 42 (2009) 30–37.

[2] S. Zhang, L. Yao, A. Sun, Y. Tay, Deep learning based recommender system: A survey and new perspectives, ACM Comput. Surv. 52 (2019) 5:1–5:38. URL: https://doi.org/10.1145/3285029. doi:10.1145/3285029.

[3] Y. Zhu, Z. Guan, S. Tan, H. Liu, D. Cai, X. He, Heterogeneous hypergraph embedding for document recommendation, Neurocomputing 216 (2016) 150–162.

[4] W. X. Zhao, S. Li, Y. He, L. Wang, J. Wen, X. Li, Exploring demographic information in social media for product recommendation, Knowl. Inf. Syst. 49 (2016) 61–89.

[5] V. W. Anelli, T. D. Noia, E. D. Sciascio, A. Ragone, J. Trotta, How to make latent factors interpretable by feeding factorization machines with knowledge graphs, in: ISWC (1), volume 11778 of *Lecture Notes in Computer Science*, Springer, 2019, pp. 38–56.

[6] V. W. Anelli, T. D. Noia, E. D. Sciascio, A. Ferrara, A. C. M. Mancino, Sparse feature factorization for recommender systems with knowledge graphs, in: RecSys, ACM, 2021, pp. 154–165.

[7] T. Berners-Lee, J. Hendler, O. Lassila, The semantic web, Scientific American 284 (2001) 34–43. URL: http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21.

[8] T. Heath, C. Bizer, Linked Data: Evolving the Web into a Global Data Space, Synthesis Lectures on the Semantic Web, Morgan & Claypool Publishers, 2011. URL: https://doi.org/10.2200/S00334ED1V01Y201102WBE001. doi:10.2200/S00334ED1V01Y201102WBE001.

[9] T. D. Noia, C. Magarelli, A. Maurino, M. Palmonari, A. Rula, Using ontology-based data summarization to develop semantics-aware recommender systems, in: ESWC, volume 10843 of *Lecture Notes in Computer Science*, Springer, 2018, pp. 128–144.

[10] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, BPR: bayesian personalized ranking from implicit feedback, in: UAI, AUAI Press, 2009, pp. 452–461.

[11] S. Rendle, W. Krichene, L. Zhang, J. R. Anderson, Neural collaborative filtering vs. matrix factorization revisited, in: RecSys, ACM, 2020, pp. 240–248.

[12] X. He, T. Chua, Neural factorization machines for sparse predictive analytics, in: SIGIR, ACM, 2017, pp. 355–364.

[13] V. W. Anelli, A. Bellogín, A. Ferrara, D. Malitesta, F. A. Merra, C. Pomo, F. M. Donini, T. D. Noia, Elliot: A comprehensive and rigorous framework for reproducible recommender systems evaluation, in: SIGIR, ACM, 2021, pp. 2405–2414.

[14] V. W. Anelli, A. Bellogín, A. Ferrara, D. Malitesta, F. A. Merra, C. Pomo, F. M. Donini, T. D. Noia, V-elliot: Design, evaluate and tune visual recommender systems, in: RecSys, ACM, 2021, pp. 768–771.

[15] G. Adomavicius, Y. Kwon, Improving aggregate recommendation diversity using ranking-based techniques, IEEE Trans. Knowl. Data Eng. 24 (2012) 896–911.

[16] P. Castells, N. J. Hurley, S. Vargas, Novelty and diversity in recommender systems, in: F. Ricci, L. Rokach, B. Shapira (Eds.), Recommender Systems Handbook, Springer, 2015, pp. 881–918. URL: https://doi.org/10.1007/978-1-4899-7637-6_26.

[17] H. Abdollahpouri, R. Burke, B. Mobasher, Managing popularity bias in recommender systems with personalized re-ranking, in: FLAIRS Conference, AAAI Press, 2019, pp. 413–418.

[18] Z. Zhu, J. Wang, J. Caverlee, Measuring and mitigating item under-recommendation bias in personalized ranking systems, in: SIGIR, ACM, 2020, pp. 449–458.

[19] H. Abdollahpouri, M. Mansoury, R. Burke, B. Mobasher, The unfairness of popularity bias in recommendation, in: RMSE@RecSys, volume 2440 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2019.

[20] C. Zhou, J. Ma, J. Zhang, J. Zhou, H. Yang, Contrastive learning for debiased candidate generation in large-scale recommender systems, in: KDD, ACM, 2021, pp. 3985–3995.

[21] R. Borges, K. Stefanidis, On mitigating popularity bias in recommendations via variational autoencoders, in: SAC, ACM, 2021, pp. 1383–1389.

[22] T. Schnabel, A. Swaminathan, A. Singh, N. Chandak, T. Joachims, Recommendations as treatments: Debiasing learning and evaluation, in: ICML, volume 48 of *JMLR Workshop and Conference Proceedings*, JMLR.org, 2016, pp. 1670–1679.

[23] D. Jannach, L. Lerche, I. Kamehkhosh, M. Jugovac, What recommenders recommend: an analysis of recommendation biases and possible countermeasures, User Model. User Adapt. Interact. 25 (2015) 427–491.

[24] M. D. Ekstrand, M. Tian, I. M. Azpiazu, J. D. Ekstrand, O. Anuyah, D. McNeill, M. S. Pera, All the cool kids, how do they fit in?: Popularity and demographic biases in recommender evaluation and effectiveness, in: FAT, volume 81 of *Proceedings of Machine Learning Research*, PMLR, 2018, pp. 172–186.

[25] L. Boratto, G. Fenu, M. Marras, Connecting user and item perspectives in popularity debiasing for collaborative recommendation, Inf. Process. Manag. 58 (2021) 102387.

[26] Y. Zhang, F. Feng, X. He, T. Wei, C. Song, G. Ling, Y. Zhang, Causal intervention for leveraging popularity bias in recommendation, in: SIGIR, ACM, 2021, pp. 11–20.