# Incorporating Impressions to Graph-Based Recommenders

Fernando B. Pérez Maurera[1,2,*], Maurizio Ferrari Dacrema[1], Pablo Castells[3] and Paolo Cremonesi[1]

[1]Politecnico di Milano, Milan, Italy
[2]ContentWise, Milan, Italy
[3]Universidad Autónoma de Madrid, Madrid, Spain

**Abstract**

Graph-based approaches have become an effective strategy to model the users' preferences in recommender systems accurately; however, despite their excellent recommendation quality, the literature still needs to incorporate impressions (past recommendations) into existing approaches. By their definition, impressions contain the selection of the most relevant items for the user; enriching the users' profiles with those items may lead to higher-quality recommendations. In this work, we propose and empirically explore the effectiveness of two approaches that include impressions into graph-based recommenders. Both approaches are simple yet extensible as they do not change the definitions of the recommenders; but transform their main data structure: the graph's adjacency matrix. The results of our experiments suggest that our approaches may improve the recommendation quality of graph-based recommenders that do not use impressions; however, we also find that beyond-accuracy metrics may become negatively affected.

**Keywords**

Recommender Systems, Impression, Slate, Exposure, Taxonomy

## 1. Introduction

Graph-based recommenders model the relationship between users and items of a recommender system using graphs [1]. In this work, we focus graph-based recommenders that use **bipartite undirected graphs**, such as the ones used by Cooper et al. [2] and Christoffel et al. [3]. This type of graph represents users and items as its nodes, and the edges indicate a pair-wise relation between the user and the item. The graph is bipartite between the user and item nodes, meaning edges only connect a user node to an item node.

Impressions are a data source that contains the items shown to the users. In other words, impressions are the recommendations generated by the recommender. The community's awareness and interest in impressions has increased in recent years. The increased interest is supported by papers incorporating impressions into existing recommenders or developing new recommenders that use impressions. For instance, Aharon et al. [4] incorporated a latent factor that estimates the preference of users to repeated impressions with the same item to a traditional matrix factorization recommender. Recently, Pérez Maurera et al. [5, 6] performed a systematic literature review about impressions in recommender systems. In their review, they show how previous works have incorporated impressions into recommendation approaches. For instance, the literature incorporates impressions into a wide range of techniques, some papers [7, 8] include them to heuristics, while others include them to machine learning [9, 10], deep learning [11, 12], or reinforcement learning [13, 14] recommenders.

The literature has yet to include impressions on existing graph-based approaches or to develop new ones using impressions. In this work, we address this gap and propose two methods to incorporate impressions into this type of recommenders. We propose adding impressions to the primary data structure used in graph-based recommenders: the adjacency matrix. To test the effectiveness of our approaches, we perform an evaluation study on three public datasets with impressions and compare the recommendation quality of two strong graph-based recommenders when building the adjacency matrix with the traditional formulation and our approaches.

## 2. Graph-Based Recommenders

Before defining the main mathematical structures needed by graph-based recommenders, we present the mathematical notation for this section and the following ones. $\mathcal{U}$ is the set of all users, and $\mathcal{I}$ is the set of all items. We use the superscript $\intercal$ to indicate a transpose of a matrix. At the same time, we use the subscripts $u$ and $i$ to obtain the row $u$ and column $i$ of a matrix, respectively. When

*Workshop on Learning and Evaluating Recommendations with Impressions (LERI) @ RecSys 2023, September 18-22 2023, Singapore*
*Corresponding author.
fernandobenjamin.perez@polimi.it (F. B. Pérez Maurera); maurizio.ferrari@polimi.it (M. Ferrari Dacrema); pablo.castells@uam.es (P. Castells); paolo.cremonesi@polimi.it (P. Cremonesi)
0000-0001-6578-7404 (F. B. Pérez Maurera); 0000-0001-7103-2788 (M. Ferrari Dacrema); 0000-0003-0668-6317 (P. Castells); 0000-0002-1253-8081 (P. Cremonesi)

those subscripts are combined, we obtain the value on the cell indicated by the $u$ row and $i$ column.

Graph-based recommenders construct the graph of user-item relations by first inspecting using the user-rating matrix $URM \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|}$. This matrix holds the interactions users performed with the system, where an interaction is any action the users performed on an item. When the recommender system uses *explicit interactions*, $URM$ holds the ratings the users performed, e.g., numbers between 1-5. When the recommender system uses *implicit interactions*, $URM$ holds binary indicators that tell whether the user interacted with an item or not:

$$URM_{u,i} = \begin{cases} 1 & \text{if } u \text{ interacted with } i \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

Graph-based recommenders model user-item interactions by building an undirected bipartite graph $G = (V, E, X, W)$, where $V$ is the set of nodes, $E$ is the set of edges, $X$ are the attributes of nodes, and $W$ are the attributes of the edges. For graph-based recommenders, nodes are users and items, and the edges are the interactions (either explicit or not) between a user and an item. The content of the $URM$ gives the weight of the edges. As the graph is bipartite, the edges only connect user nodes to item nodes, i.e., no edge connects two item nodes or two user nodes. This graph is represented mathematically using an adjacency matrix $A \in \mathbb{R}^{(|\mathcal{U}|+|\mathcal{I}|)\times(|\mathcal{U}|+|\mathcal{I}|)}$ which encodes the information of the graph:

$$A = \begin{pmatrix} 0 & URM \\ URM^\intercal & 0 \end{pmatrix} \quad (2)$$

# 3. Adding Impressions to Graph-based Recommenders

This section presents our two methods to incorporate impressions into graph-based recommenders. First, the definition of an impression is the recommendation list shown to the user by the recommender system. Second, we define the user-impressions matrix $UIM \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|}$ as a matrix that indicates whether a user has been impressed with an item:

$$UIM_{u,i} = \begin{cases} 1 & \text{if } u \text{ was impressed with } i \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

As the impressions also contain the interacted items, the contents of the traditional $URM$ are also in $UIM$. In other words, $UIM$ contain interacted and impressed but non-interacted user-item pairs.

## 3.1. Users Profiles with Impressions

We propose to use impressions as the users' profiles to build the adjacency matrix. Such an approach contrasts with the traditional formulation that uses the interactions of users, as shown in Equation 2. The reasoning behind this method is that impressions include additional information to the users' profiles, i.e., by definition of impressions, we can expand the users' profiles to have the items the recommender system presented to the user. Graph-based methods may leverage the additional information in users' profiles to produce higher-quality recommendations.

Mathematically, we construct the adjacency matrix of the graph using the $UIM$ instead of the $URM$:

$$A = \begin{pmatrix} 0 & UIM \\ UIM^\intercal & 0 \end{pmatrix} \quad (4)$$

## 3.2. Directed Graphs with Impressions

We propose to change the definition of the graph; mainly, we define it as a *directed graph*, where the edges go in two directions. First, when a user interacts with an item, we create an edge from the user node to the item node. Second, when a user is impressed with an item, we create an edge from the item node to the user node. The traditional characteristics of the graph remain unchanged, i.e., only user and item nodes are permitted, and the graph is bipartite.

As we change the definition of the graph, we also change the definition of the adjacency matrix $A$, which now reflects the directed edges:

$$A = \begin{pmatrix} 0 & URM \\ UIM^\intercal & 0 \end{pmatrix} \quad (5)$$

# 4. Experimental Methodology

In our experiments, we follow the same evaluation methodology as Pérez Maurera et al. [15, 16] as they already proposed a framework to evaluate recommenders that use impressions.

**Datasets** We train and evaluate our recommenders in two recently published datasets with impressions: MIND [17] and ContentWise Impressions [18].

**Datasets Processing** As we construct binary versions of the $URM$ and $UIM$, we de-duplicate the user's interactions with the same item by keeping the one with the latest recorded date and time. As interactions are paired with impressions, the de-duplication process also removes the impressions associated with the removed interactions. Lastly, we remove users with less than three interactions from the datasets.

**Datasets Splits** We partition those datasets into three splits *train*, *validation*, and *test* using a user-wise leave-last-out strategy: for each user, we sort their interactions and impressions by their recorded date and time. Then, we place the last interaction and impression in the *test* split, the second last interaction and impression in the *validation* split, and the remaining interactions and impressions in the *train* split.

**Recommendation Task** For a given user, the recommenders are tasked to generate a recommendation list of $N$ items containing relevant items to the users. In our experiments, an item is considered relevant if the user interacted with such an item in the test split. We evaluate the recommendations using traditional accuracy and beyond-accuracy metrics. Regarding accuracy metrics, we evaluate the recommenders using the normalized Discounted Cumulative Gain (**nDCG**) [19], **Precision** [19], and **Recall** [19]. Regarding beyond-accuracy metrics, we evaluate the recommenders using the **Coverage** [20], the **Diversity Gini** [20], and the **Novelty** [20].

**Baseline Methods** In our experiments, we train two well-known graph-based recommenders: $P^3_\alpha$ and $RP^3_\beta$. Those recommenders have shown excellent competitive recommendation quality in previous works [21, 22] against other recommenders and represent the strongest graph-based baselines. $P^3_\alpha$ was proposed by Cooper et al. [2] and performs random walks through the adjacency matrix $A$ to model the users' preference to items in the catalog. $RP^3_\beta$ was proposed by He et al. [23] and applies the same concept as $P^3_\alpha$; however, with the difference that the edges of the graph are weighted to account for the items' popularity. Despite the simplicity and extension capabilities of our approaches, we do not evaluate graph-based recommenders using deep learning, e.g., LightGCN [23] due to their long training time.

**Proposed Methods** As presented in Section 3, our approaches incorporate impressions to graph-based recommenders by modifying their adjacency matrix. To try the effectiveness of our methods, we train $P^3_\alpha$ and $RP^3_\beta$ using the adjacency matrices defined in Equation 4 and Equation 5. We denote those approaches with the suffixes "UP" and "DG". In total, we perform our experiments with three variants of both $P^3_\alpha$ and $RP^3_\beta$: one is the traditional (see Equation 2), one uses our UP approach, and the other uses our DG approach.

**Hyper-parameter Tuning** We perform Bayesian optimization [24] to search the hyper-parameter space of the recommenders and select the combination of hyper-parameters that yield the highest **nDCG**. Mainly, the optimizer trains each recommender 50 times, out of which the first 16 times it selects random hyper-parameters; in the remaining cases, the optimizer exploits those hyper-parameters with the highest likelihood to become the optimal set of hyper-parameters. When tuning the hyper-parameters, the recommender is trained using the *train* split and is evaluated against the *validation* split. At the end of the Bayesian search, the optimizer trains the recommender on the union of the *train* and *validation* splits, it selects the hyper-parameters with the highest recommendation quality found, and evaluates the recommender on the test set.

# 5. Results & Discussion

Table 1 reports the recommendation quality of the recommenders evaluated in our study for a recommendation list with 20 items. Mainly, those results correspond to those obtained when evaluating the recommender against the test split and with the best hyper-parameters found in validation. Due to the number of hyper-parameter cases (50), the number of datasets (2), and the number of recommenders (6), the experiments involved the training and evaluation of 600 models.

## 5.1. Accuracy

From Table 1, we notice two patterns in the results dependent on the dataset we test against. On the MIND dataset, our proposed methods are effective and produce recommenders that achieve a higher recommendation accuracy when compared to the baseline method, i.e., $P^3_\alpha$ and $RP^3_\beta$ benefit from the inclusion of impressions in the adjacency matrix, either by considering only impressions (denoted as UP, see Equation 4) or by building a directed graph (denoted as DG, see Equation 5). Overall, the UP approach is the one that obtains the highest accuracy in five out of the six comparisons; in the remaining one, the DG approach achieves the highest **nDCG**. The results suggest that the UP approach can model user preferences with finer granularity than the DG and traditional approaches. Moreover, for the UP approach, the relative improvements in accuracy on this dataset are higher when training a $P^3_\alpha$ recommender than a $RP^3_\beta$ recommender. The opposite occurs for the DG approach, where pairing it with a $RP^3_\beta$ recommender yields a higher relative improvement than pairing it with a $P^3_\alpha$ recommender. Lastly, for $P^3_\alpha$ recommender, we see minimum relative improvements in the accuracy of 50 %. In contrast, for the $RP^3_\beta$ recommender, we see minimum relative improvements in the accuracy of 100 %.

On the ContentWise Impressions dataset, the results are not favorable; indeed, none of our proposed methods achieve a higher recommendation accuracy when compared to the baseline method, i.e., $P^3_\alpha$ and $RP^3_\beta$ ob-

**Table 1**
Top-20 accuracy and beyond-accuracy metrics of graph-based recommenders when evaluated on the MIND and ContentWise Impressions datasets. The recommender without suffix (i.e., $P^3_\alpha$ and $RP^3_\beta$) use the adjacency matrix shown in Equation 2. The suffix "-UP" indicates the recommender uses our proposed adjacency matrix shown in Equation 4. The suffix "-DG" indicates the recommender uses our proposed adjacency matrix shown in Equation 5. For each metric, we indicate in **boldface** the methods using impressions with higher accuracy than the recommender without impressions.

| Dataset | Method | nDCG | Precision | Recall | Coverage | Diversity Gini | Novelty |
|---|---|---|---|---|---|---|---|
| MIND | $P^3_\alpha$ | 0.0249 | 0.0027 | 0.0533 | 0.4288 | 0.0515 | 0.0039 |
| | $P^3_\alpha$-UP | **0.0564** | **0.0065** | **0.1292** | 0.0821 | 0.0071 | **0.0040** |
| | $P^3_\alpha$-DG | **0.0423** | **0.0041** | **0.0821** | 0.1129 | 0.0063 | 0.0034 |
| | $RP^3_\beta$ | 0.0276 | 0.0030 | 0.0597 | 0.5904 | 0.0670 | 0.0039 |
| | $RP^3_\beta$-UP | **0.0561** | **0.0064** | **0.1285** | 0.1004 | 0.0071 | **0.0040** |
| | $RP^3_\beta$-DG | **0.0594** | **0.0063** | **0.1264** | 0.0698 | 0.0048 | 0.0037 |
| ContentWise Impressions | $P^3_\alpha$ | 0.0948 | 0.0104 | 0.2077 | 0.2718 | 0.0255 | 0.0073 |
| | $P^3_\alpha$-UP | 0.0491 | 0.0057 | 0.1145 | 0.1207 | 0.0112 | 0.0070 |
| | $P^3_\alpha$-DG | 0.0685 | 0.0079 | 0.1578 | 0.0999 | 0.0061 | 0.0064 |
| | $RP^3_\beta$ | 0.0976 | 0.0107 | 0.2145 | 0.4122 | 0.0247 | 0.0069 |
| | $RP^3_\beta$-UP | 0.0492 | 0.0057 | 0.1144 | 0.1673 | 0.0182 | **0.0073** |
| | $RP^3_\beta$-DG | 0.0720 | 0.0082 | 0.1636 | 0.1236 | 0.0090 | 0.0067 |

tain the highest **nDCG**, **Precision**, and **Recall** when the adjacency matrix is built using interactions alone (see Equation 2). Contrary to the results on the other dataset, the recommendation quality between the DG and UP approaches is interchanged, i.e., on this dataset, the DG approach obtains higher accuracy than the UP approach. Furthermore, both approaches have negative relative improvements in recommendation quality; specifically, between $-20\,\%$ and $-50\,\%$ less accuracy than the traditional recommender.

## 5.2. Beyond-Accuracy

When inspecting the beyond-accuracy metrics, we notice that our approaches generally yield lower values on all datasets and both $P^3_\alpha$ and $RP^3_\beta$ recommenders. Notably, the coverage and the diversity Gini metrics are the most affected, with negative relative improvements ranging from $-55\,\%$ to $-95\,\%$.

When inspecting the **Coverage** metric, the results indicate that although our approaches yield higher accuracy, they recommend fewer unique items from the catalog. At the same time, when inspecting the **Diversity Gini** metric, the results indicate that our approaches have less diversity. Some exceptions to the behavior on the previous metrics occur when inspecting the **Novelty** metric; particularly, the metric measure the ability of recommenders to recommend *unpopular* items. In three cases, the novelty of the UP approach was higher than the DG and the traditional approaches; despite that result, the relative improvement ranges from $1\,\%$ to $5.5\,\%$.

## 5.3. Future Works

This work proposes two approaches to incorporate impressions into existing graph-based recommenders. We empirically validate the effectiveness of our approaches by conducting experiments on two datasets with impressions. These experiments show positive and negative outcomes: our approaches consistently obtain more accurate recommendations on one dataset; however, it comes with less diverse recommendations. Future works may analyze this effect and understand whether such results are due to the recommendation domain or characteristics of the datasets.

Our approaches do not currently model the nuances of users and their preference toward impressions. Notably, we assign the same *importance* to impressions and interactions on the definition of the adjacency matrix of our approaches (see Section 3). This is more evident in the DG approach, as we construct a directed graph where each edge is 0 or 1. A future direction is to tune the importance of impressions on the edges of the graph when constructing the adjacency matrix; this would be similar to the ideas behind the $RP^3_\beta$ recommender.

Another direction to pursue is to explore our approaches on other graph-based recommenders, particularly those that use deep learning and message-passing [25, 26] to learn users' preferences, e.g., NGCF [27] and LightGCN [23].

## References


[1] A. N. Nikolakopoulos, X. Ning, C. Desrosiers, G. Karypis, Trust your neighbors: A comprehensive survey of neighborhood-based methods for recommender systems, in: F. Ricci, L. Rokach, B. Shapira (Eds.), Recommender Systems Handbook, Springer US, 2022, pp. 39–89. doi:10.1007/978-1-0716-2197-4_2.

[2] C. Cooper, S. Lee, T. Radzik, Y. Siantos, Random walks in recommender systems: exact computation and simulations, in: C. Chung, A. Z. Broder, K. Shim, T. Suel (Eds.), 23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014, Companion Volume, ACM, 2014, pp. 811–816. doi:10.1145/2567948.2579244.

[3] F. Christoffel, B. Paudel, C. Newell, A. Bernstein, Blockbusters and wallflowers: Accurate, diverse, and scalable recommendations with random walks, in: H. Werthner, M. Zanker, J. Golbeck, G. Semeraro (Eds.), Proceedings of the 9th ACM Conference on Recommender Systems, RecSys 2015, Vienna, Austria, September 16-20, 2015, ACM, 2015, pp. 163–170. doi:10.1145/2792838.2800180.

[4] M. Aharon, Y. Kaplan, R. Levy, O. Somekh, A. Blanc, N. Eshel, A. Shahar, A. Singer, A. Zlotnik, Soft frequency capping for improved ad click prediction in yahoo gemini native, in: W. Zhu, D. Tao, X. Cheng, P. Cui, E. A. Rundensteiner, D. Carmel, Q. He, J. X. Yu (Eds.), Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019, ACM, 2019, pp. 2793–2801. doi:10.1145/3357384.3357801.

[5] F. B. Pérez Maurera, M. Ferrari Dacrema, P. Castells, P. Cremonesi, Impression-aware recommender systems, CoRR abs/2308.07857 (2023). doi:10.48550/arXiv.2308.07857.

[6] F. B. Pérez Maurera, M. Ferrari Dacrema, P. Castells, P. Cremonesi, Characterizing impression-aware recommender systems, in: LERI 2023: Proceedings of the 1st Workshop on Learning and Evaluating Recommendations with Impressions, Singapore, September 19, 2023, CEUR Workshop Proceedings, CEUR-WS.org, 2023.

[7] Q. Zhao, G. Adomavicius, F. M. Harper, M. C. Willemsen, J. A. Konstan, Toward better interactions in recommender systems: Cycling and serpentining approaches for top-n item lists, in: C. P. Lee, S. E. Poltrock, L. Barkhuus, M. Borges, W. A. Kellogg (Eds.), Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing, CSCW 2017, Portland, OR, USA, February 25 - March 1, 2017, ACM, 2017, pp. 1444–1453. doi:10.1145/2998181.2998211.

[8] P. Lee, L. V. S. Lakshmanan, M. Tiwari, S. Shah, Modeling impression discounting in large-scale recommender systems, in: S. A. Macskassy, C. Perlich, J. Leskovec, W. Wang, R. Ghani (Eds.), The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014, ACM, 2014, pp. 1837–1846. doi:10.1145/2623330.2623356.

[9] F. Borisyuk, L. Zhang, K. Kenthapadi, Lijar: A system for job application redistribution towards efficient career marketplace, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017, ACM, 2017, pp. 1397–1406. doi:10.1145/3097983.3098028.

[10] D. C. Liu, S. K. Rogers, R. Shiau, D. Kislyuk, K. C. Ma, Z. Zhong, J. Liu, Y. Jing, Related pins at pinterest: The evolution of a real-world recommender system, in: R. Barrett, R. Cummings, E. Agichtein, E. Gabrilovich (Eds.), Proceedings of the 26th International Conference on World Wide Web Companion, Perth, Australia, April 3-7, 2017, ACM, 2017, pp. 583–592. doi:10.1145/3041021.3054202.

[11] Y. Zhang, Z. Chan, S. Xu, W. Bian, S. Han, H. Deng, B. Zheng, KEEP: an industrial pre-training framework for online recommendation via knowledge extraction and plugging, in: M. A. Hasan, L. Xiong (Eds.), Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, October 17-21, 2022, ACM, 2022, pp. 3684–3693. doi:10.1145/3511808.3557106.

[12] P. Covington, J. Adams, E. Sargin, Deep neural networks for youtube recommendations, in: S. Sen, W. Geyer, J. Freyne, P. Castells (Eds.), Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016, ACM, 2016, pp. 191–198. doi:10.1145/2959100.2959190.

[13] S. Li, A. Karatzoglou, C. Gentile, Collaborative filtering bandits, in: R. Perego, F. Sebastiani, J. A. Aslam, I. Ruthven, J. Zobel (Eds.), Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17-21, 2016, ACM, 2016, pp. 539–548. doi:10.1145/2911451.2911548.

[14] J. McInerney, B. Lacker, S. Hansen, K. Higley, H. Bouchard, A. Gruson, R. Mehrotra, Explore, exploit, and explain: personalizing explainable recommendations with bandits, in: S. Pera, M. D. Ekstrand, X. Amatriain, J. O'Donovan (Eds.), Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2-7, 2018, ACM, 2018, pp. 31–39. doi:10.1145/3240323.3240354.

[15] F. B. Pérez Maurera, M. Ferrari Dacrema, P. Cremonesi, Replication of recommender systems with impressions, in: G. Pasi, P. Cremonesi, S. Orlando, M. Zanker, D. Massimo, G. Turati (Eds.), Proceedings of the 12th Italian Information Retrieval Workshop 2022, Milan, Italy, June 29-30, 2022, volume 3177 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2022. URL: http://ceur-ws.org/Vol-3177/paper20.pdf.

[16] F. B. Pérez Maurera, M. Ferrari Dacrema, P. Cremonesi, Towards the evaluation of recommender systems with impressions, in: J. Golbeck, F. M. Harper, V. Murdock, M. D. Ekstrand, B. Shapira, J. Basilico, K. T. Lundgaard, E. Oldridge (Eds.), RecSys '22: Sixteenth ACM Conference on Recommender Systems, Seattle, WA, USA, September 18 - 23, 2022, ACM, 2022, pp. 610–615. doi:10.1145/3523227.3551483.

[17] F. Wu, Y. Qiao, J. Chen, C. Wu, T. Qi, J. Lian, D. Liu, X. Xie, J. Gao, W. Wu, M. Zhou, MIND: A large-scale dataset for news recommendation, in: D. Jurafsky, J. Chai, N. Schluter, J. R. Tetreault (Eds.), Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020, Association for Computational Linguistics, 2020, pp. 3597–3606. doi:10.18653/v1/2020.acl-main.331.

[18] F. B. Pérez Maurera, M. Ferrari Dacrema, L. Saule, M. Scriminaci, P. Cremonesi, Contentwise impressions: An industrial dataset with impressions included, in: M. d'Aquin, S. Dietze, C. Hauff, E. Curry, P. Cudré-Mauroux (Eds.), CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020, ACM, 2020, pp. 3093–3100. doi:10.1145/3340531.3412774.

[19] S. Rendle, Item recommendation from implicit feedback, in: F. Ricci, L. Rokach, B. Shapira (Eds.), Recommender Systems Handbook, Springer US, 2022, pp. 143–171. doi:10.1007/978-1-0716-2197-4_4.

[20] P. Castells, N. Hurley, S. Vargas, Novelty and diversity in recommender systems, in: F. Ricci, L. Rokach, B. Shapira (Eds.), Recommender Systems Handbook, Springer US, 2022, pp. 603–646. doi:10.1007/978-1-0716-2197-4_16.

[21] M. Ferrari Dacrema, S. Boglio, P. Cremonesi, D. Jannach, A troubling analysis of reproducibility and progress in recommender systems research, ACM Trans. Inf. Syst. 39 (2021) 20:1–20:49. doi:10.1145/3434185.

[22] V. W. Anelli, D. Malitesta, C. Pomo, A. Bellogín, E. D. Sciascio, T. D. Noia, Challenging the myth of graph collaborative filtering: a reasoned and reproducibility-driven analysis, in: RecSys '23: Seventeenth ACM Conference on Recommender Systems, Singapore, September 18 - 23, 2023, ACM, 2023. URL: https://arxiv.org/abs/2308.00404.

[23] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, M. Wang, Lightgcn: Simplifying and powering graph convolution network for recommendation, in: J. X. Huang, Y. Chang, X. Cheng, J. Kamps, V. Murdock, J. Wen, Y. Liu (Eds.), Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020, ACM, 2020, pp. 639–648. doi:10.1145/3397271.3401063.

[24] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, N. de Freitas, Taking the human out of the loop: A review of bayesian optimization, Proc. IEEE 104 (2016) 148–175. doi:10.1109/JPROC.2015.2494218.

[25] M. M. Bronstein, J. Bruna, T. Cohen, P. Velickovic, Geometric deep learning: Grids, groups, graphs, geodesics, and gauges, CoRR abs/2104.13478 (2021). URL: https://arxiv.org/abs/2104.13478.

[26] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, G. E. Dahl, Neural message passing for quantum chemistry, in: D. Precup, Y. W. Teh (Eds.), Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017, volume 70 of *Proceedings of Machine Learning Research*, PMLR, 2017, pp. 1263–1272. URL: http://proceedings.mlr.press/v70/gilmer17a.html.

[27] X. Wang, X. He, M. Wang, F. Feng, T. Chua, Neural graph collaborative filtering, in: B. Piwowarski, M. Chevalier, É. Gaussier, Y. Maarek, J. Nie, F. Scholer (Eds.), Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019, ACM, 2019, pp. 165–174. doi:10.1145/3331184.3331267.