IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 14 June 2022, accepted 24 July 2022, date of publication 28 July 2022, date of current version 23 August 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3194536

SURVEY

Recommendation Systems: An Insight Into Current Development and Future Research Challenges

MATTEO MARCUZZO[1], ALESSANDRO ZANGARI[2], ANDREA ALBARELLI[3], AND ANDREA GASPARETTO[2]

[1]Digital Strategy Innovation, 30175 Venice, Italy
[2]Department of Management, Ca' Foscari University of Venice, 30123 Venice, Italy
[3]Department of Environmental Sciences, Informatics and Statistics, Ca' Foscari University of Venice, 30123 Venice, Italy

Corresponding author: Andrea Gasparetto (andrea.gasparetto@unive.it)

ABSTRACT Research on recommendation systems is swiftly producing an abundance of novel methods, constantly challenging the current state-of-the-art. Inspired by advancements in many related fields, like Natural Language Processing and Computer Vision, many hybrid approaches based on deep learning are being proposed, making solid improvements over traditional methods. On the downside, this flurry of research activity, often focused on improving over a small number of baselines, makes it hard to identify reference methods and standardized evaluation protocols. Furthermore, the traditional categorization of recommendation systems into content-based, collaborative filtering and hybrid systems lacks the informativeness it once had. With this work, we provide a gentle introduction to recommendation systems, describing the task they are designed to solve and the challenges faced in research. Building on previous work, an extension to the standard taxonomy is presented, to better reflect the latest research trends, including the diverse use of content and temporal information. To ease the approach toward the technical methodologies recently proposed in this field, we review several representative methods selected primarily from top conferences and systematically describe their goals and novelty. We formalize the main evaluation metrics adopted by researchers and identify the most commonly used benchmarks. Lastly, we discuss issues in current research practices by analyzing experimental results reported on three popular datasets.

INDEX TERMS Recommendation systems, survey, collaborative filtering, content-based, hybrid methods, learning-to-rank, taxonomy, evaluation protocols.

I. INTRODUCTION

The volume of digital information has been increasing at an exponential rate within the last few decades. This has led to what is commonly defined as the *information overload* problem, which describes those situations in which users find themselves dealing with excessive amounts of information, and are actually hindered in their ability to navigate it and make decisions in its regard. Whenever content providers offer goods or services in numbers that are intractably large for individual customers, an automated method able to guide them towards a custom selection of content becomes a necessity. Recommendation Systems (RSs) [1] are such methods, functioning as an indispensable tool to users, as well as increasing sales and views for providers. RSs have an incredibly wide range of applications, such as e-commerce, social media, video hosting platforms, online news platforms, music libraries and much more. With this review, we aim to provide a strong foundational overview of this research area, describe its latest advancements and precisely frame the most important issues and challenges that should be addressed.

A. RECOMMENDATION TASK

We begin by providing a brief overview of the generic task tackled by recommenders. A RS can be generally described

The associate editor coordinating the review of this manuscript and approving it for publication was Amir Masoud Rahmani.

as a framework that suggests *items* to *users* utilizing any type of data that regards either or both of them, as well as historical *interactions* between them. These are the three main actors of RSs — users, items and interactions — and are used as generic terms regardless of what they concretely represent in different scenarios. Interactions are considered to be a user's feedback, and are either *explicit* (user reviews of an item, e.g., a score in the range 1–5) or *implicit* (user acts on an item without indication of preference) [2]. Some approaches split interactions into additional subcategories based on the more concrete action type that describes them (e.g., click, buy, view, etc.). RSs based on implicit feedback face additional difficulties, as all interactions are weak signals: items selected in the past give a weak indication of what a user may want to see in the future, and there are no explicit negative interactions [3]. In either case, all of the remaining items (not interacted with) are weak negatives, in the sense that it is unknown how the user would react to them, a fact that poses its own challenges (most notably, how to handle the large number of negatives).

As far as learning objectives are concerned, in the case of explicit feedback the task is frequently framed as a prediction of how a user will rate an item. Instead, in the case of implicit feedback the task can be defined as "maximization of the rate of consumption". Because the signals are weak, the problem is not what the user will like or not, but what the user is likely to interact with. The meaning of "consumption" is domain-dependent. The watch time for a video or the dwell time on a website page can both be considered consumption signals for a video sharing platform and a news agency. On the other hand, advertisement platforms are more likely to be interested in the maximization of Click-Through Rate (CTR), that is, the fraction of clicks on an item over the number of times it has been seen.

B. PROBLEM DEFINITION

Item recommendation, also known as top-*n* recommendation, is the task of selecting the best items from a large catalog for a user in a given context. In this section, we give a short formal introduction to common notational conventions.

Formally, we define a user u and an item i as belonging to corresponding sets, i.e., $u \in U$ and $i \in I$. Again, these are generic terms that abstract from what they are concretely, which is instead described by their representation. User and item representations are very flexible and depend on the data utilized by the system itself — Section III will explore these different representations. The simplest representation for these actors is based on user and item identifiers (ids), supplied with no further information, meaning the system works solely on user–item interactions. Some authors prefer to incorporate users in a "context" [3], which encapsulates both the user and additional contextual information such as time, location, and previous interactions of that user. We keep these concepts separate, though the resulting methods are the same. Interactions between users and items are most commonly organized in a matrix R, where $r \in R$ can be an explicit rating (e.g., 1 to 5) or an implicit signal (1 if the interaction has occurred, 0 otherwise). Therefore, r_{ui} represents the interaction between user u and item i. In general, most recommenders systems can be seen as having to design a *scoring* or *utility* function:

$$f(u, i) = f(i|u), \quad f : U \times I \to \mathbb{R} \tag{1}$$

This utility indicates the degree of preference towards the item of the user. The choice and design of such function are core aspects of the modeling process of a RS [3], [4].

C. RELATED WORK

There are a number of recently published surveys and articles on the area of RSs, though the vast majority addresses a particular sub-field without attempting to capture it as a whole. Here we briefly mention the most relevant to our work, highlighting their merits and how they differ from this survey.

The authors of [5] organize a survey from the perspective of modeling recommenders with the accuracy goal, and limited to neural approaches. Collaborative filtering approaches are reviewed in [6], which also showcases hybrid approaches that integrate information derived from social networks. In [7], neural recommenders are tackled, focusing on deep learning-based approaches and building a comprehensive summary of current research. The work by [8] provides an excellent categorization of recommendation tasks and goals for sequence-aware recommenders, which have to deal with sequentially-ordered interactions. In [9], a unified framework on session-based RSs is provided (often considered a subset of sequence-aware recommenders), describing in depth the unique characteristics and challenges posed by session data. The excellent article by [3] details item recommendation in implicit settings, with a large focus on challenges faced during training and various techniques (mainly sampling) utilized to solve them. In [10], a framework of recommendation from the point of view of explainable recommendations is described. The interesting formulation of RSs as systems trying to solve a Multi-Armed Bandit problem is surveyed in [11]. Both [12] and [13] cover the usage of knowledge graphs in RSs. Finally, [14] characterize and formalize graph learning-based RSs, their challenges, and main progress in the sub-field.

We found many recent surveys addressing the usage of RSs in specific domains. For instance, in [20] the authors discuss algorithms that make use of user-assigned tags to predict item relevance, often in social network platforms. Much research has been published on the recommendation of scientific texts, like in [16], [18], [21]. Applications of RSs in the tourism and travel industry, like accommodation and food recommendations, are explored in [19], while [17] showcases the importance of location-based services and social networks in this domain. Finally, RSs can be beneficial in education for recommendation of teaching resources, for instance on e-learning platforms [15], [22]. Table 1 provides an overview of the surveys analyzed.

TABLE 1. Recent related surveys, sorted chronologically.

Survey	Year	Main focus
[6]	2018	Traditional and hybrid collaborative filtering
[8]	2018	Sequence-aware recommenders
[15]	2018	Ontology-based RSs for e-learning services
[7]	2019	Neural recommenders
[16]	2019	Research article recommendation systems
[17]	2019	Location recommendation for tourism
[10]	2020	Explainable recommendations
[18]	2020	Deep learning-based citation recommendation
[19]	2020	Tourism destination recommenders
[9]	2021	Session-based recommenders
[3]	2021	Item recommendation in implicit settings
[5]	2021	Neural recommenders with the accuracy goal
[14]	2021	Graph learning based recommenders
[20]	2021	Social tag-based recommenders
[11]	2022	Multi-Armed Bandit based recommenders
[21]	2022	Taxonomy of recommenders for research material

D. CONTRIBUTIONS OF THIS SURVEY

In contrast, our survey is organized from a more generic point of view, attempting to collate much of this information into a single, foundational overview. We attempt to highlight how the field has evolved over the years, such as to give a realistic and up-to-date view of the recommendation landscape. By analyzing challenges and points of contention on recent progress, we aim to incorporate theoretical knowledge with an authentic representation of the current state of this research field. This will help researchers discover new ideas to design better solutions in the future, while also being conscious of possible disputes about recent progress in the recommendation area [23]–[25]. Relatedly, we discuss how several different evaluation protocols are currently adopted to test the performance of RSs, and how possible issues in such protocols affect the assessment of the state-of-the-art.

In summary:

- We provide an overview of the recommendation task, its various facets, and possible design choices to be made when developing a recommender;
- We propose an updated taxonomy of RSs, based both on traditional categorizations and new emerging trends, clearly characterizing different approaches through popular representatives;
- We briefly describe a wide array of recently proposed methods, such as to provide an easily accessible overview of recent research in this area;
- We study the evaluation process of a RS and its critical issues, highlighting examples in recent literature.

How Papers Are Collected:

As our survey aims to capture the latest advancements and proposed ideas in the field, we retrieved the most related top conferences such as NEURIPS, ICML, ICLR, RECSYS, SIGIR, KDD, WWW, WSDM, AAAI and IJCAI, the same that were surveyed in [5]. Due to the very large number of retrieved results, we only reviewed a selection of contributions from each conference, matching the keyword "recommendation", "recommender" and "recommendation system", and preferring the ones surrounded by a larger amount of academic discourse. Among our goals, we wished to perform an analysis of testing protocols in recent works, a procedure which in many cases requires access to the code implementation of the experiments. We found that conference papers tend to publish the code of the experiments more frequently than journal publications. As such, we decided to select mainly conference papers, as was done in [23], [26]. As this field of work is particularly dynamic, we limited our search to papers published after the year 2019, though we also consulted particularly influential and distinguished publications from years prior.

However, in order to provide a more thorough analysis, we also complement our search with queries to Google Scholar[1] and DBLP.[2] We have first searched for the most influential works with an unfiltered search sorted by relevance, and then applied a more fine-grained search of recent works in the period of time 2018-2022. While we still included works published in conferences with this procedure, we tried to put a particular emphasis in searching works published in related journals rather than conference papers. These include journals such as Knowledge-Based Systems, Expert Systems with Applications and IEEE Access. The total number of works retrieved by the end of our research was of roughly 200 works, of which about 150 were conference papers. Another large portion of our references is from cross-referencing particularly important works mentioned within the corpus of our analysis. Over 120 recent or influential works are briefly presented in the methods overview.

E. STRUCTURE OF THE SURVEY

This survey is organized with the following structure:

- Section II provides an introduction to the main design choices to be made towards the optimization of RSs;
- Building on such information, Section III provides an overview of recommendation models and explains a data-dependent taxonomy, tying it with standard taxonomies and clarifying these approaches by giving influential examples;
- Section IV goes in-depth into an exploration of the recently proposed methods and approaches, which are largely based on neural networks;
- Section V describes the main evaluation protocols adopted in research, and reports the most popular metrics and datasets, with considerations on various testing strategies found in the literature on three of them;
- The survey draws to an end in Section VI, analyzing possible new and long-standing challenges as well as future research directions of this field;
- Lastly, our conclusions are reported in Section VII.

II. DESIGN CHOICES

Before diving into a taxonomy of RSs, it is useful to introduce a few propaedeutic concepts related to the field, all of which relate to the general design of a recommendation framework.

[1] https://scholar.google.com
[2] https://dblp.org

In this section, we first provide some clarity on why a categorization of RSs is not simple, and what considerations should be taken when constructing such a system. In a related fashion, we then proceed to introduce some of the main challenges faced by RSs, fundamental in order to better understand the design choices that differentiate the methods within the taxonomy. We introduce some of the most popular choices of learning objectives used to frame the recommendation problem into supervised Machine Learning (ML) problems. Lastly, we briefly touch on the ''retrieval and ranking'' approach for designing recommender frameworks, as well as a short mention to sampling approaches.

A. CONSIDERATIONS TO BE MADE

RSs have been long studied with great interest, and are generally considered an important subclass of ML and information filtering. However, we find that, unlike many traditional fields of study, they lack a robust definition and classification. This is not without reason; while an intuitive notion of what a RS should do is easily identifiable, the process of developing a careful characterization is soon met with an abundance of questions. Here we attempt to identify some of the main reasons why a consistent categorization of RSs can be difficult to achieve.

Firstly, it is important to consider (1) *what type of data is available to the system*. It is often not trivial to decide what information should be used, and how to treat missing or not readily available data points. Secondly, one should also consider (2) *how user interactions are treated*. For example, an e-commerce website might want to consider the action of ''adding to cart'' differently from the ''buy'' action. In a similar vein, one might ask (3) *what interaction is being sought*. In video recommendation scenarios, one might want to decide between maximizing watch time and CTR; this latter objective may favor ''click-bait'' videos, resulting in many videos that were opened, but abandoned shortly after. Last but not least, considering (4) *how the task is framed* is also of utmost importance. Learning strategies can differ depending on whether the algorithm objective is to approximate a user-dependent function that describes the level of affinity with items (classification or regression problem), or to populate a list of items of probable interest (retrieval problem). Furthermore, the specific application might have additional requirements, such as having at least one relevant item (or, conversely, as many as possible in a less precise manner).

Clearly, these considerations only cover part of the large number of facets of this design process. The ones presented above were chosen as we found them to capture some of the most relevant and thoroughly studied issues within the field. Throughout this survey, we will introduce and explain the various concepts necessary to answer these questions.

B. MAIN CHALLENGES

Throughout the years, RSs have had to deal with a staple set of challenges that are important to consider whenever discussing both new and old approaches. This section provides a brief introduction to the most common: *data sparsity*, the *cold start problem* and *scalability*. While we will address other important challenges in Section VI, we briefly anticipate these core issues, as we deem it necessary to wholly understand the methods that will be illustrated.

1) DATA SPARSITY

One of the most severe complications associated with RSs is the sparsity problem [4], a natural consequence of the fact that it is very unlikely for users to have interacted with more than a small fraction of the available items. In turn, the representations of such systems — which are, one way or another, based on interactions — will contain a large number of missing entries, i.e., will be very sparse. This causes severe complications, most notably the difficulty to create accurate representations for users and items, as most of the interactions will not have occurred [6]. Unobserved interactions are inherently *weak* negatives, as we have no information on whether the user has actively avoided them or has simply not come across them yet.

Moreover, not only are interactions sparse, but they are also commonly concentrated around popular items, meaning such sparsity is also highly localized [2], [27]. This property often satisfied by real-world recommendation datasets is referred to as the *long-tail*. Datasets with such property will have the vast majority of their interactions related to a restricted fraction of highly popular items. This creates a long-tail distribution when plotting the number of interactions against the items sorted by interaction frequency (Fig. 1), where the vast majority of items reside in such long-tail, yet have the least number of interactions overall.

2) COLD START

The *cold start* problem [4], [28] describes situations in which a recommender has to deal with either users or items that have few or no interaction histories, which is usually the case when they have just entered the system. Approaches based solely on interaction histories are inherently sensitive to this issue, since they have no other foundation to characterize users or items. While new users can be trivially suggested popular items, new items might end up never being recommended because of how they have never been part of any interaction. Utilizing side information (e.g., based on the item and user data) is usually an effective way to mitigate this problem [6].

3) SCALABILITY

Practical properties such as *scalability* [29], [30] are fundamental in RSs, as recommendations should be generated quickly — usually as soon as the user enters the system or each time they interact with an item. A scalable system should be able to handle often massively large amounts of information, which will likely only grow in time. It is notable that this issue leads to many real-world applications relying on methods that are not very recent (though they have obviously been refined) [23], [24], [31], [32], yet perform

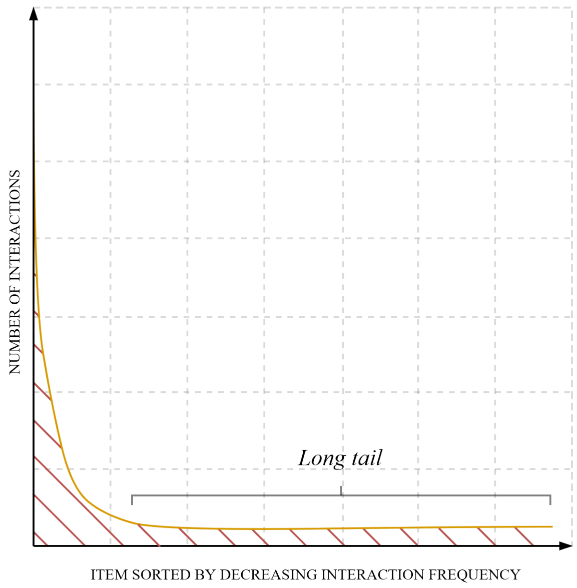


FIGURE 1. Characteristic long-tail distribution of interaction frequencies.

undeniably well when we consider their scalability. New approaches should be mindful of this constraint; trading accuracy for performance through approximations is often a necessity.

C. LEARNING OBJECTIVES

Recommendations are rarely provided as single items and instead are usually presented as a ranked list, with items deemed more relevant placed on top and vice versa. An important point of divergence between different recommendation approaches, then, is the choice of optimization task [3]. Functions that optimize a single affinity score between a user and an item are defined as *pointwise*, while other methods, namely *pairwise* and *listwise* approaches, fall within the "*learning-to-rank*" category. In general, this popular class of algorithms in information retrieval (IR) contains methods that sort items according to their predicted degree of relevance, putting less focus on a predicted score and more emphasis on a well-ordered result. The last part of this section discusses the *multiclass* approach highlighted by some authors [3], [33].

1) POINTWISE OPTIMIZATION

A large number of traditional RSs rely on *pointwise* optimization functions. Methods based on a pointwise criteria can be generally described as seeking to predict affinity scores between individual pairs of users and items. For instance, a method within this category might aim to predict the expected rating a user might give to a previously unseen item; the optimization process would therefore aim to minimize the error of each predicted rating against the real rating value. While the underlying task might be classification or regression, the result can be easily adapted to a top-*n* recommendation scenario by ordering the items by their predicted rating. This can very well incur infeasible computational costs as the number of training examples is very large ($\mathcal{O}(|U||I|)$), an open problem that has been tackled with various approaches, one of which is *negative sampling* [3], [33], [34].

2) PAIRWISE OPTIMIZATION

Researchers have long argued against the discrepancy between the objective of the optimization process and the final output of a RS. For example, [35] showcased how influential the choice of a properly chosen optimization criterion can be towards the end result, proving the potential of pairwise methods by introducing the widely popular Bayesian Personalized Ranking (BPR) optimization criterion. *Pairwise* methods compare pairs of interactions at train time. The learning task becomes one that must determine which of the two items would be preferred by the user, ultimately creating an ordering between items — leading to a personalized ranking for each user. Other works, such as the one by [36], have also argued against pointwise approaches, claiming that the application of accuracy-based metrics (such as rating prediction error, see Section V-B) is a sub-optimal fit to the recommendation task. Furthermore, they argue that pointwise optimization is by nature wasteful, as a good approximation is sought for items that are not to be suggested. In general, pairwise approaches are at least as expensive as pointwise approaches, having to consider a number of pairs in the order of $\mathcal{O}(|R||I|) \supseteq \mathcal{O}(|U||I|)$ [3], therefore also incurring in complexity issues.

Learning-to-rank approaches attempt to model the fact that, ideally, the algorithm should learn to directly maximize a ranking utility. However, maximizing utilities of this kind is not trivial, as they are often non-differentiable or otherwise uninformative gradient-wise, and this challenge has been studied widely by the academic community [37], [38]. The examples mentioned above fall within the class of solutions that devise surrogate, differentiable ranking losses to minimize, such as to indirectly maximize ranking metrics. To clarify, whenever we use the term "ranking loss", we refer to one that only considers relative preferences between items for each user, and does not care about maximizing absolute utility scores on single items. We also point out that, as an alternative approach, the pairwise ranking task has been reformulated as a classification problem (as in [39]), where pairs are labeled as positive if correctly ordered, negative if not.

3) LISTWISE OPTIMIZATION

Listwise approaches can be seen as a generalization of pairwise approaches to multiple items. Authors in this field argue that listwise approaches are more suitable for the learning-to-rank paradigm, as they directly address the problem of creating a list of objects as a prediction [39], [40]. Evidently, due to the fact that such approaches work with permutations that grow factorially in number, the learning task can quickly become intractable. This issue is often addressed by utilizing what is called a *score-and-sort* approach [41]. The task is then to learn a scoring function that, given a query (e.g., a user) and a set of items, produces a vector of relevance values, which can then be used to produce a ranking.

Many learning-to-rank methods further reduce the problem to that of learning a univariate scoring function to produce a score between a single query and an item [36], [37], [42].

4) MULTICLASS APPROACH

A related formulation frequently used in practice casts recommendation as an extreme multiclass classification problem where each item is a possible class. From a probabilistic point of view, this option models recommendation as a multinomial distribution over the items (conditional on the user, $\hat{y}(i|u)$). One of the most common functions utilized to translate the real valued score to a multinomial distribution is the *softmax* function, defined as:

$$p(i|u) = \frac{\exp\ (\beta \cdot \hat{y}(i|u))}{\sum_{j \in I} \exp\ (\beta \cdot \hat{y}(j|u))} \tag{2}$$

In the above equation, β is a temperature parameter used to control how much the output distribution will be concentrated around large values [3]. This class of methods is often paired with cross-entropy as a measure of the distance (or error) between predicted and target distribution [33].

The seminal work that introduced ListNet [39] reaches a similar formulation, which the authors call ''top-one probability'', while devising an approach to the otherwise intractable listwise approach based on permutations. The authors prove that the top-k probabilities over the items form a probability distribution, and propose to utilize any loss metric that measures the distance between score lists, such as cross-entropy.

Treating recommendation as a multiclass scenario utilizing the softmax and cross-entropy pairing might appear loosely related to ranking metrics. In [37], however, the authors find analytical connections (under certain conditions) between this loss and popular ranking metrics. Many modern methods do, in fact, utilize similar approaches (i.e., cross-entropy paired with softmax), and it is debatable whether these methods should be considered listwise, as most of the time they do not model the permutations within elements in the list explicitly. For the sake of clarity, throughout this article we consider ranking methods only those which openly address the learning-to-rank task, usually by applying ranking losses (or their approximations) directly.

D. RETRIEVAL, RANKING AND SAMPLING

Before moving on to the taxonomy, we note that many recommendation approaches are not applied directly as ''out-of-the-box'' solutions in practical settings. Modern recommendation architectures are usually much more complex and include multiple steps. For example, [43] describes a two-step pipeline in which items are gradually filtered and re-ranked into smaller groups, making it possible to exploit different approaches with different complexity requirements even in large settings.

Generalizing beyond a concrete framework, many approaches implicitly assume that complex methods are preceded by a *retrieval* model, whose job is to return a short (compared to the whole database) list of items. Retrieval models have to make a compromise between returning good items and obtaining them within the required serving latency requirements (commonly in the order of milliseconds). Complex models are then applied on the much smaller retrieved list, on which they act as a *ranking* model [44].

Another possible way to approach large-scale systems is that of *sampling*. While going into detail about such approaches is beyond the scope of this survey, the basic idea is to tackle the sparsity problem by coupling positive examples (which are typically much fewer in number) with a restricted pool of negative ones. This is commonly referred to as *negative sampling*. The choice of sampling distribution for the negatives and how the sampler weighs different examples is key to the design of proper sampling strategies [3].

III. TAXONOMY OF RECOMMENDATION SYSTEMS

In this section, we describe a data-oriented taxonomy for RSs. Based on our review of past and current works, we deem more appropriate an incremental taxonomy dependent on the amount and type of side information available inspired by works such as [5] and [45]. Note that what is meant by incremental is *not* that categories must necessarily contain the ones before it, but rather that each category can be extended by combining it with the others, allowing for large areas of overlap between them. This better reflects how work in this field has progressed, rarely tying itself to a specific subset of data and instead attempting to utilize any bit of information as permitted by the situation.

This section will introduce influential, explanatory examples for the categories, while Section IV will provide a more in-depth exploration of the current landscape of proposed methods.

A. TRADITIONAL VS DATA-ORIENTED CATEGORIES

In the following, we provide an overview of the classic taxonomy of RSs and then extend it to showcase a more accurate categorization of current approaches.

1) TRADITIONAL CATEGORIES

Traditional categorizations are helpful in providing a general perspective of the most prominent methods, even though we argue they no longer wholly capture how current methods approach the recommendation task. RSs have been traditionally divided in three main categories: *collaborative filtering*, *content-based* and *hybrid* recommenders:

- *Collaborative filtering* [46] methods predict user-item affinity by considering past interactions from other known users. This is commonly referred to as leveraging the ''wisdom of the crowd'', as suggestions made to a user will be based upon similar users. Similarity measures might differ, but are only ever based on past interactions and an expressed preference/feedback towards them;

- *Content-based* [47] methods are used to predict user-item affinity by considering only the user or item features (i.e., their ''content''); approaches such as this are most commonly user-centered, in which the system builds profiles for individual users to make predictions on unseen items. It is also possible to create item-centered systems, which models individual items and predicts some sort of affinity score when provided with unseen users;
- *Hybrid* [48] methods utilize approaches that combine both of the above categories. Many different combination approaches have been proposed, as well as entirely new methods which fuse them into a single algorithm.

Though such labels still see use, many new categorizations have been adopted in the literature, each with different amounts of overlap between the other. Earlier works, such as the seminal work by [48], placed these groups beside *Demographic-based* (based on demographic attributes of users), *Utility-based* (based on a utility model of items with regards to users), and *Knowledge-based* (based on knowledge bases of items) systems. *Hybrid systems* would then be defined as combinations of two or more of these, with multiple possible combination strategies between them. While these categorizations are a good fit, they are not entirely balanced, and recent approaches have begun to incorporate them within larger categories.

In our research, we find that most new approaches were, in fact, either hybrid or collaborative-based. Thus, though the distinction between collaborative and content-based filtering is still useful, it should be noted how the modern recommendation landscape is much more focused on the area of overlap between the two. We believe that by giving a more data-oriented depiction of this field we can paint a more realistic picture of the current state of RSs.

2) DATA-ORIENTED TAXONOMY

Differentiating models based on specific sources of data can potentially spawn too many subcategories. Recent approaches, from which we draw inspiration, maintain the dichotomy of interaction and content data and add a third category, which covers what is defined as *contextual information*. The latter aggregates features that are not specific to users or items but rather to the interactions themselves (i.e., describes their context) [5]. Depending on the information utilized, methods may fall below one or multiple of these categories.

It has become common to define the agglomeration of content and context information as *side information*, which most commonly adds to the base information of interactions given by collaborative filtering approaches. In other words, most methods are influenced by collaborative filtering methods, to which they possibly add available side information. This is due to the fact that interaction data (in particular, in its implicit form), is by and large the most common type of interaction information available [35], [49], [50].

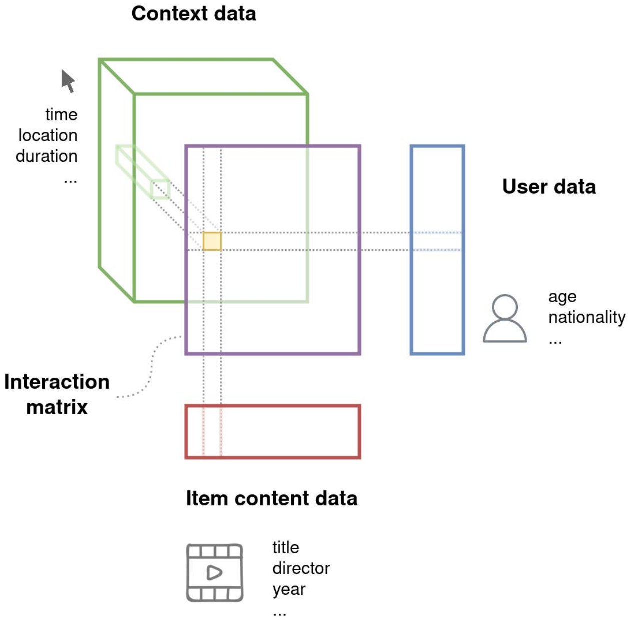


FIGURE 2. Representation of interactions, content and context data in a 3D space.

In the following sections, we describe a taxonomy based on three main categories:

- *Collaborative filtering*, which are based solely on the interactions between items and users, ignoring all types of information that describe either or both;
- *Content-enriched models*, which integrate content data into a recommendation, including all descriptive attributes that may be associated directly with items or users;
- *Context-aware models*, which utilize those types of information associated with interactions but not exclusive to the user or item involved, such as time or location.

Fig. 2 depicts the differences in the type of information used in the three families of methods. Again, many parts of such categories overlap based on specific availability and considerations of a particular context.

B. COLLABORATIVE FILTERING METHODS

Collaborative filtering (CF) methods model users and items solely based on the interactions of a population of users. As mentioned, users' interests are usually presented as a numerical rating in a small range (explicit feedback) or as a binary value that simply indicates whether the interaction has occurred (implicit feedback). We reiterate that, in practical scenarios, implicit feedback is far more common.

Memory-Based vs. Model-Based: Collaborative filtering approaches have often been divided into two subcategories, namely *memory-based* (or *heuristic-based*) and *model-based* [4], [45].

Early approaches calculate the behavior similarity of users or items directly, operating over the collection of interactions in order to make a suggestion. They are termed *memory-based* because of how they store computed similarities between users or items as a sort of "memory" to produce new recommendations. Memory-based models may also be further sub-categorized based on whether they compare users or items. Similarity may be identified through metrics such as the Pearson correlation or the cosine similarity [45]. The most popular memory-based approaches fall into the neighborhood search category, which we discuss next.

Model-based approaches, on the other hand, train prediction models based on the user-item interaction matrix (in contrast to using ratings directly), hence transforming the task into one of estimating the model's parameters. Latent Factor Models (LFMs), discussed in Section III-B2, are the most popular representatives of this category, though the idea is not exclusive to them and includes approaches such as cluster models and Bayesian networks [4], [51].

1) NEIGHBORHOOD METHODS

A popular memory-based approach is the conceptually simple nearest-neighbor (k-NN) algorithm [4], [45]. A *user-based* approach of this kind finds the k users with the highest similarity in terms of ratings and bases its expected affinity between a given user u and an unseen item i on the ratings of the k neighboring users. For example, a simple formulation to produce a predicted rating $\hat{r}$ might be:

$$\hat{r}_{ui} = \frac{1}{C} \sum_{k \in K_u} \text{sim}\,(u, k) \cdot r_{ki} \tag{3}$$

In the above equation, C is a normalizing constant, *sim* is a chosen similarity measure and K_u and r are the set of neighboring (similar) users to u and the true ratings, respectively. The above is merely a simple example, and much more refined methods exist. A similar, mirrored approach can be taken towards *item-based* neighborhood methods, where an unknown rating is predicted by averaging the ratings of similar items rated by the same test user [52].

Nearest-neighbor methods (and memory based approaches in general) can run into scalability issues. While the underlying implementation and what is being compared (items, users, a combination of both) obviously matters, there is an inescapable complexity in calculating similarities between all pairs of users and/or items. Most approaches, for $|U|$ users and $|I|$ items, have a worst-time complexity of $\mathcal{O}(|U||I|)$ — though it is empirically usually closer to $O(|U| + |I|)$ thanks to the sparsity of most user vectors [53]. This might still be prohibitive for large datasets, but appropriate preprocessing paired with sampling techniques can make these systems more viable at large scale, though recommendation quality is likely to be reduced.

2) LATENT FACTOR MODELS

LFMs have risen to be the most indicative representatives of collaborative filtering-based RSs, attracting great deals of attention ever since their impressive results in the Netflix contest [54], [55]. The idea behind such approaches is to find representations for users and items in a shared latent space, deriving them from the interaction matrix. The general objective is to learn two embedding matrices $\boldsymbol{P}$ and $\boldsymbol{Q}$ for users and items, where $\boldsymbol{p_u}$ and $\boldsymbol{q_i}$ are the parameters of the corresponding matrices for user u and item i, respectively. Latent models aim to find the underlying relationships between users and items by learning what are defined as their "*latent factors*".

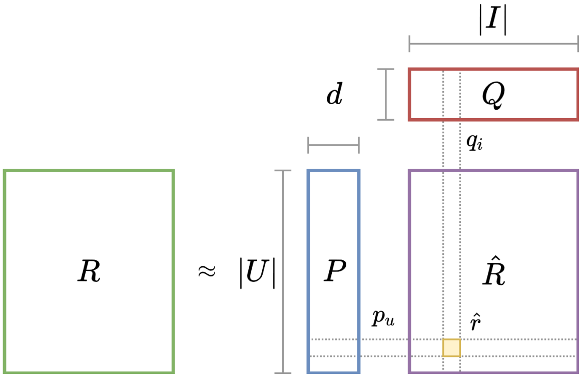


FIGURE 3. Visual representation of the matrix decomposition process, typical of MF.

The most well-known method in this category is Matrix Factorization (MF), and its basic idea is at the foundation of other LFMs. This model attempts to decompose the interaction matrix into the respective embedding matrices, whose combination is a good approximation of the original feedback matrix (Fig. 3). The most basic type of MF, proposed in [56], predicts a rating $\hat{r}$ by performing a dot product between user and item embeddings:

$$\hat{r} = \boldsymbol{p_u q_i}^\intercal \tag{4}$$

Though already an effective formulation, many developments have been proposed for it since (such as the different variants of the SVD [57], [58] and iALS [32] methods). One of the advantages of MF is its compact representation; given $|U|$ users, $|I|$ items and d-dimensional embeddings, the theoretical space complexity of the embedding matrices is $\mathcal{O}((|U|+|I|)d)$ in total. Given that d is typically much smaller than both $|U|$ and $|I|$, the resulting complexity is much more affordable than methods that have to consider all user-item pairs.

As a side note, the dot product has become a popular combination strategy for latent representations in many applications within the field of RSs (as we will showcase in Section IV). Indeed, it can be applied to any system that produces embedding representations for users and items. While we do not dive into details, it should suffice to know that dot product models are widely popular because they provide an efficient and effective way of combining embeddings, with many well-studied approximations that improve their practical applicability [3].

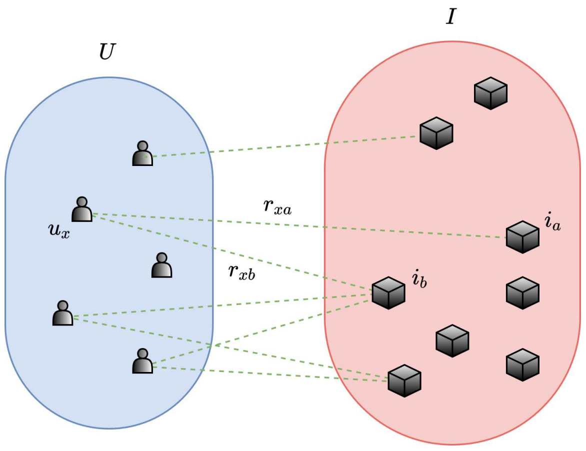


FIGURE 4. Bipartite graph representation of users, items and ratings.

3) GRAPH-BASED ALGORITHMS

Graph-based algorithms, as the name suggests, opt out of the traditional data representation based on feedback matrices, instead adopting one based on graphs. Not only is this a natural fit to interaction data, but it also lends itself quite conveniently to the integration of additional side information.

Indeed, the interaction relationship between users and items can be easily translated to a graph representation. Formally, consider the interaction relationship S as composed of user and item pairs, i.e., $(u, i) \in S$ if an interaction has occurred between the two. We can then define the graph as graph $G = (U \cup I, S)$ (Fig. 4). This representations is inherently *bipartite*, since user nodes can only be connected to item nodes and vice versa. Edges within the graph (i.e., interactions) might also be weighted, depending on available information. The weight might be based on feedback data such as explicit ratings, but might also be a straightforward opportunity to introduce more nuanced influences — such as the one derived from content information based on the features of the pair representing the edge.

In general, the objective of graph-based recommenders is to discover a ranking of item vertices in I for a user vertex u based on their respective similarities, as defined by the structure of the graph. If the interactions are implicit and the graph unweighted, the task can be framed as a more generic link prediction problem [14]. The bipartite graph representation grants several advantages; most notably, information can be propagated through nodes to mitigate sparsity and cold start issues. However, the true challenge resides in finding an effective way to enact such propagation. Moreover, this is particularly challenging in bipartite graphs, as user-user or item-item edges do not exist, requiring multiple hops from neighboring nodes for certain communications to happen.

Many graph-based approaches exist, and the recent resurgence in popularity of graph-based methods has revitalized this category of methods. A popular example of such approaches consists of *random walk-based* algorithms [59], [60]. In short, these methods operate through a stochastic process that lets a random walker move between nodes based on a transition probability (established from known feedback). The probability that a walker lands on an item node after a certain number of steps is utilized as a means to rank the candidate nodes. Examples of methods that fall within this category include P^3_α [61] and RP^3_β [62], which have obtained excellent results. An interesting consideration to be made is that P^3_α can be framed as equivalent to a k-NN item-based approach, which highlights how similar certain approaches can be despite a different representation [23].

More recently, graph embeddings have been proposed as a way to exploit graph structures, mapping nodes into low-dimensional embedding vectors to capture the structural information of the graph. Such embeddings can then be used as representations for users and items. With the advent of neural approaches, Graph Neural Networks (GNNs) have also been proposed, which we further detail in Section IV-F.

C. CONTENT-ENRICHED METHODS

We define as content-enriched those methods that integrate information about the main agents of interactions within a RS (i.e., users and items). Differently from the traditional class of content-based RSs, we consider this group almost as a natural extensions to collaborative filtering models. Empirically speaking, we find that many modern content-enriched methods utilize interaction data as a basis.

Therefore we find, similarly to [5], that describing them as complementary to collaborative filtering approaches is more suitable. Most content-enriched methods indeed "enrich" base interactions with auxiliary data related to users or items. This category can be further dissected into sub-classes that encapsulate the specific variety of data being incorporated, as will be described in the remainder of this section. As always, these categorizations can overlap and may be integrated as deemed appropriate.

1) PURELY CONTENT-BASED APPROACHES

We begin by providing a brief overview of purely content-based approaches. Empirically, we find these to be less common in modern systems, where hybridized models are by far the most prevalent. Still, in light of the previously mentioned fact that RSs are oftentimes complex, multi-step processes with multiple algorithms involved, they are worth mentioning, and can still be useful in certain contexts [63], [64]. A positive side of such algorithms is that they often produce more accurately tailored predictions to single users when compared to purely collaborative approaches. On the other hand, purely content-based systems suffer from *overspecialization* [4], [65], which describes a system whose recommendations are *strictly* similar to previous interactions, as well as sometimes being *too* similar (and hence, not interesting). Furthermore, while they fare better than CF in item cold start scenarios, they struggle with *user cold start*, as a sufficient number of interactions is necessary before a user profile can be built.

A standard example of a purely content-based approach would be an item-based k-nn approach [47]. This is similar to the one detailed in Section III-B1, where item similarities would be computed utilizing content attributes rather than ratings. To improve scalability, many content-based approaches resort to a projection of features into some type of low-dimensional space, which is then utilized to perform, for example, a search of nearby items. It is also possible to develop predictive models, inducing a similar dichotomy between memory and model based CF approaches (though it is less common to make this distinction in content-based approaches). These include various types of classifiers, decision trees, and clustering methods [4].

2) CATEGORICAL FEATURES AND ATTRIBUTES

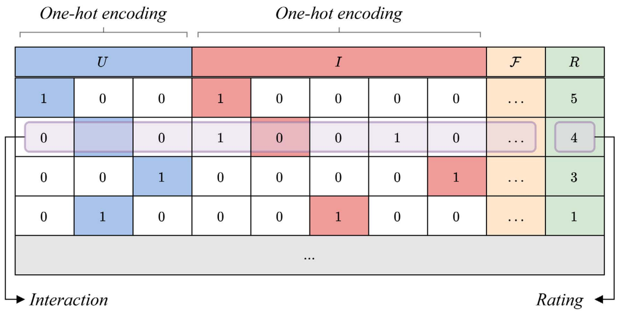
The sources of side information regarding users and items are broad and varied. Categorical and other similarly quantifiable generic types of attribute information are among the most commonly found, describing users and items to some degree (e.g., the genre of a movie or the gender of a user).

A strong representative of the models in this category are Factorization Machines (FMs) [66], which extend factorization models by integrating ideas and advantages of Support Vector Machines (SVMs) [67]. FMs are general predictors but, in contrast to SVMs, and thanks to their modeling of interactions between variables through factorized parameters, can estimate interactions in settings with high sparsity (which is practically always the case in RSs), all at an affordable computational cost [68]. FMs are not applied directly to the interaction matrix, instead requiring a data representation that more closely reflects their predictive nature. Concretely, FMs are provided a matrix in which each row is a feature vector that describes a specific interactions and its features. An interaction matrix can be easily transformed into such a form by creating a one-hot encoding of items and users (Fig. 5).

Additional features may then be concatenated to this feature vector. As each of these rows has a target value $y \in Y$ (e.g., rating), this framework is easily understandable as a standard prediction task. Notably, a FM without any auxiliary data is identical to a MF model, and it has also been shown that FMs can mimic most factorization models with appropriate feature engineering [69].

3) MULTIMEDIA CONTENT

Not all types of features can be introduced straightforwardly. When it comes to multimedia content, a dedicated approach may be necessary to first extract a good representation. This is the case for textual and visual content, which have been vastly studied of their own accord; the developments in the fields of Natural Language Processing (NLP) and Computer Vision (CV), respectively, can be combined with recommendation frameworks to obtain better user and item representations. Similarly, the same approach can be taken in regards to audio and video content.



U			I					$\mathcal{F}$	R
1	0	0	1	0	0	0	0	...	5
0		0	1	0	0	1	0	...	4
0	0	1	0	0	0	0	1	...	3
0	1	0	0	0	1	0	0	...	1
...									

FIGURE 5. The matrix representation utilized in a FM. Here, $\mathcal{F}$ stands in lieu of an arbitrary set of feature columns.

Textual content. The past decade has seen revolutionary advancements in the field of NLP. In particular, neural network-based approaches have enabled the automatic extraction of syntactically and semantically meaningful representations for text, most recently with the development of contextualized word embeddings based on Transformer architectures [70]–[72]. These embeddings can be combined with user and item embeddings produced by CF approaches to produce more accurate representations, or to produce more explainable recommendations [73]. As an example, content descriptions (such as abstracts for articles) can be utilized in this fashion [74].

Image content. RSs based on image content are suitable for scenarios that rely heavily on visual influence, such as clothing recommendation [75]. Image-based models may attempt to extract textual tags from images, which may then be processed as discussed previously. Another approach is to project both users and items in the same visual space; items are trivially projected through their pictorial representation, while users may be projected through the items they previously liked or by more advanced encoding procedures. Approaches based on Convolutional Neural Networks (CNNs) are, as of now, some of the most popular and prolific in terms of extracting features from images [76]–[79].

Audio and video. RSs based on rich visual and auditory information have also been proposed, both as purely content-based models as well as integrated to hybrid, content-enriched architectures [80], [81]. These can be particularly useful to recommend new audio and video content that has no historical behavior data by comparing its similarity to other well-known items, mitigating cold start issues. The more straightforward approach might be to utilize metadata related to such types of media (e.g., titles or descriptions), as it is more easily manageable. Nonetheless, some recent approaches have developed deep neural networks to extract image and audio features, projecting items into a low-dimensional feature space in which it is easier to operate (for example, by searching for similar videos in this space) [5]. It is worth noting that working with video media can be difficult because of the underlying computational expensiveness, as well as space storage requirements [80].

4) SOCIAL NETWORKS

Recommenders based on social networks (sometimes more broadly termed as ''community'' based) leverage the preferences of a user's friends or otherwise closely tied users to make a recommendation. Such methods attempt to model the underlying social influence among ''neighbors'', which is seen as the driving force that correlates users' interest in the network. Social sciences have long studied principles such as *homophily*, a property suggesting that contact between similar people occurs at a higher rate than among dissimilar people. In turn, this provides reason to believe that capturing social interactions can lead to better recommendations, as friends are likely to show preference to similar things.

The integration of social networks has been often devised as a countermeasure towards cold start issues and to integrate information into particularly sparse environments. For instance, [82] integrates social influences into probabilistic LFMs as regularization terms. Some recent approaches have also introduced social influences in neural models. An example is the work by [83], which pairs latent model-inspired user embeddings with social embeddings learned from an unsupervised deep learning approach, applying regularization techniques based on social correlation theories.

Social connections are also a natural extension to graph-based approaches, where they can be seen as edges that relate users to other users. While other approaches that integrate social networks might limit themselves to local first-order social neighbors (i.e., the direct friends of a user), recent approaches (notably GNNs) have been proposed as more accurate models to describe the global social diffusion process for recommendation. These methods have been applied to user-user social graphs (i.e., no interactions), but perhaps more interestingly have also been applied to heterogeneous graphs where both social connections and interactions are present [84]. The leftmost side of Fig. 6 represents an example social interaction graph.

5) KNOWLEDGE GRAPHS

Knowledge Graphs (KGs) are another effective way to represent entities and the relationships between them. There has been some debate [85] on the exact definition of this term; in the context of RSs, the denomination *knowledge graph* refers to directed graphs containing nodes $e \in E$ which represent entities, and edges $s \in S$ to denote the relationships between them. A KG is then formally defined as $\mathcal{G} = \{(h, s, t) \,|\, h, t \in E,\ s \in S\}$, where each triplet indicates the existence of a relationship s between head entity h and tail entity t [86]. Users and items will have relationships with their describing features (or otherwise connected data), which might also be related to other entities.

Earlier approaches utilized KGs to extract a representation (e.g. in the form of embeddings for users and items), but recent models have proposed to enrich such graphs by adding interaction relationships to the graph itself, arguing that it provides a more complete representation [87]. In other words, such methods add interactions to the set of edges of the graph. Fig. 6 showcases an explicative example. As before, the user-item interaction data is most commonly presented as a bipartite graph. Note that, while the example shown is mixed with information from a social interaction graph, this need not be necessarily the case. The concept outlined above where KGs are enriched with interaction information can be formally expressed as $\mathcal{G} = \{(h, s, t) \,|\, h \in E, t \in E \cup I,\ s \in S \cup R\}$, where we abuse notation and refer to R as the rating/interaction relationship.

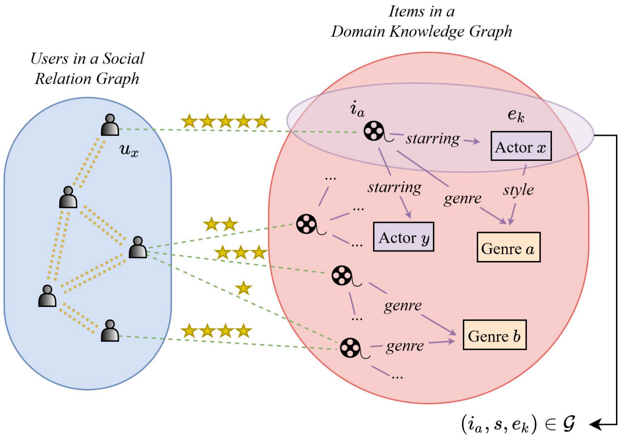


FIGURE 6. A possible (simplified) graph representation for a RS, including elements of a KG as well social relationships. Double dotted (yellow) edges between users represent social relationships between users, while straight directed edges (purple) indicate relationships between entities. As before, dotted directed (green) edges between users and items represent ratings.

KG-based methods are attractive because they allow for a more interpretable system [12], as it is possible to verify the reasoning behind a recommendation and, thus, create an explanation for it. In a similar fashion to standard graph approaches, these approaches can be used as regularization terms, as input for predictive path-based methods, and, more recently, have been explored in the frame of GNNs to model higher-order connectivity representations.

D. CONTEXT-AWARE METHODS

The category of context-aware methods includes approaches that integrate information sources that can describe the environment where an interaction happens. Because of this, this *context* is sometimes called ''interaction-associated information'' [45]. Some authors make the distinction between *representational context* [88], which is defined by a predefined set of ''observable'' context variables (e.g., time, location, weather), and *interactional context* [89], [90], which instead is more dynamic and has to be derived from the user's most recent actions and is not directly observable (user mood, current shopping intent). The latter set of context data is considered particularly important in settings where users are anonymous or new, as there is no historical data in such scenarios. Some types of side-information might not fall clearly within one category or the other, such as textual reviews for an item (which are content for the item, but contextual to the interaction that prompted the review).

The most widely studied type of contextual information is by far of the *temporal* kind [4], [5]; as such, the next sections examine in more detail approaches that take into consideration the temporal domain. Though we do not go into detail about other types of context, we point out that several approaches are possible in those cases, and are often similar to those described in the previous section. Methods that are not usually designed to work outside of the two-dimensional $User \times Item$ space have been generalized to multidimensional spaces, with approaches such as Tensor Factorization [91]. Tensor Factorization generalizes MF to arrays of higher orders, where, intuitively, they factorize interactions in the generic form *(user, item, interaction context, rating)*.

Categorization of Temporal Methods: Users' preferences change and evolve over time; due to this fact, static recommendations are likely to be less effective. Instead, it may be possible to discern patterns within the sequential behavior of users, which is the aim of methods that incorporate time in the recommendation process. Methods in this field usually differentiate between a *sequence*, which is considered a list of chronologically ordered interactions with no explicit time intervals, and a *session*, a list of interactions with a clear boundary, either ordered or unordered, which most commonly spans a relatively brief interval of time.

However, while research in this particular area is extensive, it is also surprisingly scattered, making it hard to find a categorization commonly agreed upon. To cite a few notable surveys, [8] classifies these methods based on the importance given to historical interactions, distinguishing between *last-n interactions-based recommendation*, which considers only the last few user interactions, *session-based recommendation*, in which only the last sequence of interactions (contained in a session) is available, and *session-aware recommendation*, which contains both knowledge about the current session as well as historical information. On the other hand, in [9] such categorizations are deemed to be more fitting of a "sequence-based" class. The authors argue that session-based methods are not only those that consider single, anonymous sessions, but instead include approaches that consider historical sessions as well.

Based on these ideas as well as other works [45], [92], [93], we differentiate between three loosely separated classes:

- *Time-aware RSs* utilize time information directly to recommend appropriate items, with a focus that is more largely tied to the exact point of time of past user interactions (e.g., time of day, day of the week). These methods are still related to a sequential environment as with the other two classes, as they might discover patterns within, for example, temporal cycles;
- *Sequence-based RSs*, sometimes defined *time-dependent* or *sequential recommenders*, instead put a much larger focus on the sequential order of events. Such methods aim to predict the next items a user might interact with given a sequence of historical interactions;
- *Session-based RSs* instead group interactions within sessions and tackle tasks more closely related to the sessions in question (further detailed later).

Again, these categories are not separated by hard lines and should be taken as purely functional to a better understanding of this sub-field. Indeed, sequence- and session-based RSs are often considered as special cases of a broader category, that of *sequence-aware* methods [8], [94].

We do not discuss time-aware approaches directly, pointing out that many of them rely on matrix completion approaches (i.e., common CF approaches), of which [92] provides an excellent overview. Instead, in the next sections, we briefly introduce sequence- and session-based recommenders, which have seen rising popularity in recent research. Notably, both of these classes frequently differentiate between various *types of interactions* [93], i.e., different with regards to the concrete action logged at that specific time (e.g., view vs buy). This is particularly relevant in a sequential context, as a specific sequence of actions might deliver further insights on the intent of a user.

As a side note, sequence- and session-based RSs will sometimes have to predict a utility value on a list of items, rather than for a specific item. This is different from Equation 1, where the utility was based on a single item, as the utility of the list is calculated on the *entire* sequence, and the sequence with maximum score should be found.

1) SEQUENCE-BASED RECOMMENDERS

Sequence-based systems try to explicitly discover the sequential dependencies among interactions, such as to discover behavioral patterns and other information that can only be understood when viewing the interactions as a succession of events. There are different types of patterns that might be sought; for example, [8] differentiates between *sequential*, *co-occurrence* and *distance* patterns. Sequential patterns relate interactions in a specific order, while co-occurrence patterns only care that two interactions have happened together. Distance patterns are less common and try to identify good lapses of time necessary before recommending something (e.g., a reminder).

2) SESSION-BASED RECOMMENDERS

Session-based recommenders consider (usually short) sequences of interactions within clearly bounded periods of time. A user's sessions are usually separated by non-identical time intervals. Sessions themselves have been categorized in multiple ways depending on their internal characteristics (length, internal order, action type). While the most common type of session is totally ordered and contains interactions of a single type, heterogeneous and partially ordered (or unordered) sessions have also been researched. Further considerations have also been made in regards to the length of the sessions and the amount of content (user) information available [93].

We note that some of the most popular session-based RSs are limited to the interactions of the current user session (i.e., only the last one), which is the case in anonymous or new-user scenarios. Most of the time, researchers will use the term ‘‘session-based’’ to refer to this situation, where user attributes and histories are usually scarce or not present [95]. As there is no consensus, we do not make a clear-cut distinction, clarifying when sessions are deemed as *anonymous* (only the last session is present) and when the system is instead *session-aware*, i.e. has knowledge about historical sessions (a term we also borrow from [95]).

Common Approaches:

There is a wide range of approaches that have been proposed for sequence- and session-based recommendation, and a detailed overview is provided by [9]. In general, the most popular approaches are, as expected, based on the exploitation of the sequential item transition patterns. These include conventional sequential approaches (e.g., Markov chains) [96], LFMs, and neural network-based approaches (e.g., RNNs, CNNs) [95]. Notably, graph-based approaches have also been successful, integrating sessions as chains (sequences of nodes) within the graph [97]–[99]. Methods also differ depending on the task being faced, which, especially in the case of session-based recommenders, can be of various types. The most common categorization separates them based on whether the system is trying to predict the next item in the session, all the remaining items (until the end of the session), or even the entirety of the next session [9].

E. SUMMARY

We described an extended taxonomy to classify RSs based on the amount and type of information they exploit to make recommendations. We consider this categorization as a reframing of the traditional classification schema, which we deem to have become less informative due to the abundance of hybrid methods that have been proposed. The taxonomy is inspired by the one devised in [5] and consists of three broad categories. The first makes exclusive use of user-item interaction histories, while the other two families of methods are characterized by the usage of additional information, namely user and item content and any environmental data describing the context in which the interaction took place. These categories are further specialized into more fine-grained sub-classes of methods, to exemplify how practical methods fit in this classification. It’s easy to see how such taxonomy puts much more emphasis on hybrid approaches; moreover, practical implementations usually fall in the large areas of overlap between the first category and one (or both) of the others.

IV. METHODS OVERVIEW

This section provides an overview of the most recent proposals for the improvement of RSs. When vital to the understanding of more recent methods, earlier influential approaches are introduced.

To reduce the amount of redundancy, methods are introduced based on which generic model they are based on — many methods span across different categories, so the distinction is not clear-cut. Whenever describing methods, we will clarify where they lie in the data-oriented taxonomy: collaborative filtering (CF), content-enriched (CE), or context-aware (CA). In particular, a method will be marked as CF if it only acts in a purely collaborative setting, while methods that use side information can be tagged as either CE, CA, or both. In those cases, we do no write CF, but we find that the vast majority of the algorithms analyzed have a collaborative foundation (in most neural approaches, the collaborative signal is implicitly embedded in the learning process).

Unsurprisingly, most of the discussed methods will be neural methods; the application of neural network frameworks is undeniably the most popular new approach to ML tasks. The section is also loosely ordered in terms of optimization processes. The widest class, described first, is mixture of pointwise and multiclass approaches. In Section IV-G, we will instead discuss approaches that directly tackle the learning-to-rank paradigm, while Section IV-H makes a few notable mentions of approaches not included elsewhere.

1) CONTROVERSY ON PROGRESS

Before going into detail on the most recent methods, we deem necessary a word of caution. The particular field of RSs has seen a vast amount of proposed improvements and proclaimed advancements, but these have been sometimes disputed by fellow researchers. It has been demonstrated that, at times, much simpler methods can compete or surpass complex, deep learning-based methods which were deemed superior because of poor testing practices or non-standardized metric evaluation [23]–[25]. This, in turn, causes a ripple effect throughout publications that use the latter methods as baselines, inadvertently basing improvements off of a false belief. We tried, to the best of our knowledge, to factor this within the explanation of individual recent approaches, such as to provide an intellectually honest representation of the recommendation landscape. The ideas and research directions taken by different fellow researchers are still important to study, but a careful examination of the baselines is necessary before declaring new approaches as state-of-the-art. We further discuss some of the issues at the root of these controversies in Section V-E.

2) COMMON ABBREVIATIONS

For the sake of clarity, Table 2 summarizes common technical acronyms used in tables throughout this section and in the rest of the survey. Whenever appropriate, the abbreviations will be explained in the text.

A. MATRIX FACTORIZATION-BASED METHODS

Ever since Funk’s MF [56] achieved third place in the Netflix Prize challenge [54], many LFMs based on Matrix Factorization (MF) principles have been proposed. SVD++ [57] is a

TABLE 2. Summary of acronyms used throughout this survey.

Abbreviation	Meaning
CTR	Click-Through Rate
MNAR	Missing Not At Random
MF	Matrix Factorization
NMF	Non-negative Matrix Factorization
PMF	Probabilistic Matrix Factorization
FM	Factorization Machine
HIN	Heterogeneous Information Network
ReLU	Rectified Linear Unit
MLP	Multilayer Perceptron
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
GRU	Gated Recurrent Unit
GNN	Graph Neural Network
GCN	Graph Convolutional Network
ATT	Attention
HATT	Hierarchical Attention
Tran	Transformer
DR	Dynamic Routing
PNN	Product-based Neural Network

notably popular example, extending the previous algorithm by creating an integrated model that allows for the benefits of neighborhood models (e.g., explainability), as well as allowing the use of implicit feedback in place of explicit item ratings. The reasoning behind this choice is that if a user rates an item, that is in itself an indication of preference. Non-negative Matrix Factorization (NMF) [120] has also been long used as a powerful tool able to identify meaningful substructures underlying the data. In particular, variants have been successfully applied into diverse fields and extended to analyze multiple matrices jointly [121].

1) RECENT DEVELOPMENTS IN MF

With the recent popularization of neural network-based approaches, some researchers have proposed MF methods that replace the original dot product between factorized matrices with learnable functions, most often feed-forward neural networks [104]. However, recent research concluded that these strategies are not trivial to fine-tune, and that dot-product should still be considered when developing MF methods, since it cannot be easily approximated using a feed-forward neural network [24]. Other approaches have proposed to consider feedback data as ordinal rather than binary; the Ordinal NMF (OrdNMF) [100] is a notable example, introducing a NMF approach that generalizes Poisson factorization and can be used with ordinal data, making it applicable to big sparse matrices of explicit ratings. On a different note, there is also great recent interest in creating extensions for MF techniques that focus on improving the interpretability and mathematical properties of the decomposed matrices [101]–[103].

2) HYBRIDIZED MF APPROACHES

As mentioned, purely CF methods do not natively support the incorporation of side information available from users and items. Content-enriched and context-aware methods, which the literature commonly regards as hybrid methods, have been studied extensively in recently proposed research. Within the spectrum of MF approaches, [122] proposes the usage of a Quantile Random Forest to model the effect of side information and combines it with MF in a Bayesian framework. In [123], Probabilistic Matrix Factorization (PMF) [124] is extended by integrating information derived from item descriptions, extracting a latent representation through a shallow CNN with max pooling.

TABLE 3. Recent MF-based methods. (E/I) = (Explicit/Implicit) ratings, no tag represents a method that can be used for both.

Method	Base	Cat.	Enhancements
OrdNMF [100]	NMF	CF	NMF for ordinal values by generalization of Poisson factorization
Fb-NMF [101]	NMF	CF	Factor-bounded NMF with guaranteed convergence
TopicMF [102]	MF, NMF	CE	Topic model to detect latent topic factors in review texts
MF-CNN-Interpr [103]	MF, CNN	CE	MF interpretability with CNN for extraction of salient words
NCF [104]	MF, MLP	CF (I)	Replacement of inner product in MF with neural architecture
HybridFM [105]	FM	CE, CA	Impact analysis of embedding of different types of side information
LightFM [106]	FM	CE	Linear combinations of content features' latent factors
xLightFM [107]	FM, LightFM	CE	Memory footprint reduction through quantization-based FM
MF/NMF-DR-JL [108]	FM	CF (E)	Bias reduction in MNAR ratings through joint learning of robust estimator with rating predictor and error imputation
AFM [109]	FM, ATT	CE, CA	Attention mechanism on feature interactions for neural-based FM
HNAFM [110]	FM, MLP, HATT	CE (HIN)	User and item features generations based on HIN meta-paths
ENSFM [111]	FM (opt)	N/A	Efficient FM implementation without negative sampling
NIS [112]	Agnostic	N/A	Embedding strategy for dynamic multi-size embeddings
MDRS2 [113]	MF	CE	Exploit social relations influence between users
Joint-MF [114]	MF	CE	Similarity-augmented MF based on Word2Vec embeddings
ER-MF [115]	MF	CE	Ensemble of similarity-augmented MFs with Doc2Vec embeddings
EnSocialMF [116]	MF	CE	Social factors and users trust for recommendation
TSCMF [117]	CMF (E)	CA	Jointly factorize ratings and social relation matrices with collective MF
SSTPMF [118]	PMF (I)	CA	Factorization of POI- and user-similarity
POI-SMF [119]	MF (I)	CA	Explore the influence of user preference, check-in time, trust relationships, and POI's location on recommendation

Information derived from social structures have also been successfully applied to MF-based methods. For instance, [113] integrates information from social relations between users, with the underlying assumption that different kind of relations should have different impact on the recommendation process. In a similar manner, the authors of [116] develop EnSocialMF, a model which derives social factors from social network data and uses it to influence recommendations. In particular, the algorithm attempts to fuse three factors, namely user trust relationships, user interest similarities, and item similarities, all within a PMF framework. Several works [117]–[119] propose to extend MF by considering temporal dynamics in user preference, as well as social factors and geo-spatial information. An example of joint MF framework is proposed by the authors of [114] in the context of AIoT. They propose a hybridized RS to learn user similarity, API similarity and user-API relevance matrices using three MF models that are jointly trained. API invocation histories for each user are embedded through a Word2Vec model [125]. ER-MF [115] is a similar approach, using Doc2Vec [126] to obtain user and item representations and training two models to compute the final recommendation score based respectively on user-similarity and item-similarity. The final model is the ensemble the previous two.

3) FACTORIZATION MACHINES

In Section III-C2, we discussed FMs as straightforward reformulations of the recommendation task through general-purpose linear predictors, able to work under huge sparsity. Recent studies have proposed FMs that are able to work with both categorical and arbitrary real-valued features [105], [106]. The authors of [107] propose the usage of product quantization to compress the memory usage (their model is based on [106]). In [110], Heterogeneous Information Networks and a hierarchical attention mechanisms are explored to capture relationships between objects. Finally, while many existing FM-based methods adopt negative sampling for training efficiency, [111] proposes a non-sampling method that is relatively efficient compared to the selected baselines.

B. FEED-FORWARD AND MULTILAYER PERCEPTRON-BASED METHODS

Feed-forward networks are some of the conceptually simplest and most widely explored types of neural networks. In particular, Multilayer Perceptrons (MLPs) have seen wide use throughout ML. Thanks to their flexibility, they have often been used as starting points or combined with other architectures, and have laid the foundation for some of the most influential earlier neural works on recommendation.

The term MLP is sometimes used ambiguously, usually referring to feed-forward networks with fully connected hidden layers. Since the vast majority of methods utilize fully connected structures, this section addresses them as MLPs directly. We thus review prominent methods based largely on feed-forward networks and MLPs, with various types of augmentations, most notably attention mechanisms (further detailed in Section IV-D and Appendix A-A).

1) POPULAR MLP APPROACHES

The influential work by [43] proposes a 2-step recommendation procedure to recommend YouTube videos. Side information about users (watch history, search keywords, demographic information) is embedded and passed through a feed-forward NN with ReLU activation to learn user and item representations. To train the classifier to discriminate between possibly huge numbers of items, a negative sampling strategy is used to generate multiclass probabilities over millions of candidate items, using a softmax to generate normalized probability scores. At serving time, the model is used to generate user and video embeddings, and an approximate nearest-neighbor algorithm can be used for low-latency constrained predictions. The Wide & Deep Learning framework [44], which has gained similar popularity, combines the advantages of memorization and generalization using two MLP-based branches. The "wide" component is a linear model that works with various combinations of manually-created features (responsible for memorization). The "deep" component, on the other hand, is a feed-forward neural network that converts sparse, high-dimensional categorical features into low-dimensional dense embeddings for all user and item features (responsible for generalization). The model attempts to learn nonlinear interactions through the combination of embeddings via neural networks rather than a dot product, a matter which still stands as a controversial topic. DeepFM [127] later expanded on this idea by introducing a framework that integrates FM

TABLE 4. Recent methods based on MLPs.

Method	Base	Cat.	Enhancements
DeepFM [127]	MLP, PNN	CF, CE	Wide & Deep plus FM for linear combination of features
DropoutNet [128]	MLP	CE	Tackle the cold start problem by randomly masking input
DLRM [129]	MLP	CE	FM-based method supporting continuous and discrete features
AutoCF [130]	MLP, MF	CF	Automated model hyperparameters search for CF methods
MKR [131]	MLP	CE (KG)	Jointly learned multi-task KG-enhanced model
AutoFIS [132]	MLP, FM	CE	Automated selection of low- and high-order feature interactions
AutoInt [133]	MLP, ATT	CE	Self-attention to learn high-order feature interactions
MAMO [134]	MLP, Memory	CE	Initialization of global embeddings for cold start users
MetaHIN [135]	MLP	CE (HIN)	HIN in meta-learning for cold start recommendation
DeezerMLP [136]	MLP	CE	Improvements for cold start and scalability through projection in known embedding space
EX3 [137]	MLP, CNN	CE, CA	Attribute-aware recommendation with attribute-based explanations
JTM [138]	Tree-based	CE	Joint training of tree index and deep neural architecture
UMEC [139]	MLP	CE	Embedding compression with jointly learned feature selection
CompDLRM [140]	MLP, DLRM	CE	Compositional embeddings to reduce memory requirements
DHE [141]	MLP	N/A	Hash-embeddings to reduce memory requirements for sparse data
LightRec [142]	N/A	CE	Product quantization, recurrent composite encoding of users/items
DNN-tourism [143]	MLP	CA	Tourist attraction recommendation with smart city data

and deep neural networks. Differently from Wide & Deep, the proposed model jointly learns both low- and high-order feature interactions without the need for handcrafting feature combinations. Both the wide and deep parts utilize the same input, enabling efficient training.

2) MLPs FOR HIGHER-ORDER FEATURES

Recent contributions mostly focus on strategies to embed side information such as to create more robust recommenders. The recent Deep Learning Recommendation Model (DLRM) [129] addresses the problem of using dense features in addition to categorical interaction features. While the latter are processed using an embedding table and projected in a dense feature space using a MLP, dense features are imputed in a disjoint MLP to learn expressive and properly sized representations. The authors of AutoInt [133] propose a new model to learn expressive higher-order features through a self-attention mechanism. In the proposed method, both categorical and continuous features are firstly projected in a low-dimensional embedding space. Different nonlinear combinations of features are then extracted and their relevance is weighted using the self-attention mechanism.

3) MLPs AND COLD START

In order to address the cold start problem, researchers have experimented with meta-learning solutions, approaches that use previous learning experiences to train new models [134], [135]. The core idea of meta-learning algorithms is to learn a global representation (shared initialization parameters) for all users, which are then used to learn local, personalized parameters for individual users. The work by [134] improves this by utilizing memory matrices that can to store task- and feature-specific memories. Also in the context of cold start solutions, [136] uses a MLP-based strategy to produce vector representations for warm (known) users using both interaction data and side information. To obtain approximate cold-user representations, they use averaged embeddings from a pool of warm users from the same geographical area and with similar age. Additionally, they leverage the interaction data from the same warm users during the registration day.

4) EFFICIENT MLPs FOR RECOMMENDATION

Several works adapt neural-based recommenders to satisfy specific resource constraints. The authors of [138] propose to jointly learn a tree-index and a deep neural model. The tree structure allows to efficiently retrieve user representations with logarithmic time complexity w.r.t. the corpus size. Jointly optimizing the tree-based retrieval problem with the deep recommendation model gains improvements in the overall recommendation accuracy. In [139] and [140], model compression methods are explored. The first proposes a unified framework to jointly optimize a network compression task as well as feature extraction from input interactions. The latter defines a new memory-efficient feature projection technique that relies on several smaller embedding tables to dynamically generate unique embeddings for every user. The Deep Hash Embedding (DHE) [141] framework replaces embedding tables with deep NNs that compute embeddings on the fly, utilizing multiple hash functions to generate unique identifiers for every feature value. This work aims to reduce memory requirements imposed by the usage of embedding tables.

TABLE 5. Recent methods based on CNNs.

Method	Base	Cat.	Enhancements
Weave&Rec [144]	CNN, Word2Vec	CE, CA	3D CNNs for spatio-temporal information in user reading history
3DSession [145]	CNN	CE, CA	3D CNNs to discover sequential patterns in click history
NPA [146]	CNN, ATT	CE, CA	News titles recommendation with user-personalized attention
NRPA [147]	CNN, ATT	CE, CA	User/item encoding on review texts and user-specific attention
MBCN [148]	CNN (dilated), FM	CE	User-personalized feature interactions through dilated convolutions
RCNN [149]	CNN, LSTM	CE, CA	Vertical and horizontal convolutions on LSTM hidden states
LSTUR [150]	CNN, GRU, ATT	CE, CA	Recurrent model with attentive CNN for news feature extraction
Caser [151]	CNN	CA	Vertical and horizontal convolutions for sessions
NextItNet [152]	CNN (dilated)	CA	Masked residual CNN to capture long-range dependencies
GRec [153]	CNN (dilated)	CA	Bidirectional model with gap-filling to mask future sequence
CoCNN [154]	CNN	CF (I)	Find item co-occurrence patterns

C. CONVOLUTIONAL NEURAL NETWORK-BASED METHODS

CNNs have been thoroughly explored as an efficient and effective way to extract latent representations from various types of media. Though popularized in the context of CV, they are not only applicable to RSs that involve visual content but also to other types of data, such as textual and temporal information.

1) CNNs FOR TEXTUAL INFORMATION

As it is common to encounter bodies of text (such as reviews, descriptions, and news articles) in various recommendation scenarios, a wide range of new approaches proposes to utilize CNNs to learn contextual representations efficiently. For instance, [146] proposes the Neural news recommendation model with Personalized Attention (NPA), which uses a CNN to learn the hidden representations of news articles based on their titles. Furthermore, two personalized attention mechanisms are introduced, at the word- and article-level respectively. This is intended to model how different users might perceive the same words in a title differently (with similar reasoning being applied to whole articles). Similarly, the Neural Recommendation with Personalized Attention (NRPA) proposed by [147] applies a hierarchical personalized attention mechanism to generate both user and item representations, considering textual user reviews of items as additional information. CNNs are used to extract semantic features of text reviews, while attention mechanisms are applied hierarchically over words and entire reviews. [150] proposes a neural news recommendation approach with long- and short-term user representations (LSTUR), which combines CNNs with GRUs to capture both long- and short-term dependencies between interactions. In a similar vein to previous approaches, news are encoded by passing their titles' embeddings through a CNN and an attention layer. Topics and sub-topics (represented by tags) are also projected to embeddings and concatenated with this representation. GRU networks are utilized to learn short-term user representations from their recently browsed news, which are combined with long-term representations (based on user embeddings) through either initialization of the GRU hidden states or by concatenation.

2) CNNs FOR SEQUENTIAL RECOMMENDATION

Multiple approaches embed the sequences (or sessions) of a user's interaction into a 2-dimensional latent matrix and treat them as an image. For instance, [149] proposes a Recurrent CNN model (RCNN), mixing LSTM and CNN networks to capture both long- and short-range user preferences from the user's interaction sequence. Recent hidden states of the recurrent layers are regarded as the ''image'', which convolutional filters search for local sequential features. The authors of [144] propose Weave&Rec, a 3D CNN applied on word embeddings extracted from news articles. This approach aims to learn ''spatial'' features (i.e., content of the article) as well as temporal features (across different articles, seen as a sequence). Test articles are instead passed through a 2D CNN, also working on word embeddings, and the interaction between a user and the item is obtained through element-wise product of the 3D (user) and 2D (item) CNN outputs. The usage of 3D CNNs had also been explored by [145], which addresses session-based recommendation. In their approach, content features are modeled with character-level encoding to avoid expensive feature engineering steps.

The Convolutional Sequence Embedding Recommendation Model (Caser) proposed by [151] represents users as a $L \times d$ image where L is the length of the interaction sequence and d is the embedding dimension (embeddings are learned throughout the training process). Sequential patterns are regarded as local features and extracted through 2D convolutions, while vertical convolutions (i.e., filter size $L \times 1$) are used to capture point-level sequential patterns across item representations. The authors of [152] address the issue of generative models in modeling long-range dependencies in item sequences, directly showcasing some limitations within the Caser model. Their model, named NextItNet, utilizes a stack of dilated 1D convolutional layers, as well as residual blocks to enable the training of deeper networks. Inspired by previous methods, [153] define a general framework for training encoder-decoder recommenders named Gap-filling based Recommender (GRec). The authors showcase a CNN-based encoder-decoder architecture, where the two parts are jointly trained with a gap-filling mechanism (inspired by NLP's masked language modeling [155]), such as to introduce bidirectionality without data leakage. Similar to NextItNet, both the encoder and the decoder use stacked 1D dilated convolutional layers with skip connections.

3) OTHER CNN-BASED APPROACHES

Lastly, we introduce a few interesting CNN-based approaches that do not fall within other categories. In [148], the authors aim to explicitly model feature interactions of arbitrary order, deemed particularly important to express context-aware semantics. They propose a Multi-Branch Convolutional Network (MBCN) with three specialized branches. The first branch is a standard 1D convolutional layer that learns feature correlations in a vector-wise manner. The second branch is a dilated convolutional layer that was added with the idea of generating interactions among features in non-neighboring positions. The last layer models user, item, and context bias (e.g., a user that tends to give mostly positive ratings) for better recommendation.

In [156], a framework to bridge content- and collaborative-based representations is proposed. Textual information is utilized (though extensions to other types of metadata are proposed) to extract representations for completely cold items, i.e., with no prior interactions. The resulting Content Based to Collaborative Filtering (CB2CF) model learns a mapping from the word embeddings of item descriptions (the ''CB'' representation) to a representation learned through BPR [35] (the ''CF'' representation). This multi-view mapping is learned with a CNN, though the authors claim that both this architecture and the BPR model can be replaced with similar approaches, focusing on the connection between representations.

The independence assumption between items is challenged in [154], which focus on the importance of item-item relationships in a CF problem. The authors propose the Co-occurrence pattern combined with CNN (CoCNN); this model is based on the assumption that the more two items appear together in a users' interaction history and are co-rated (i.e., similarly rated) by similar users, the more their representations should be close. The CNN learns representations from the co-occurrence matrix, directly applied to the embeddings. A different CNN model is used to learn pointwise user-item affinity and is jointly optimized with the previously described model.

D. RECURRENT NEURAL NETWORK-BASED METHODS

Recurrent Neural Networks (RNNs), by virtue of their intrinsic advantages in modeling sequential dependencies, are a strong candidate whenever dealing with interactions organized in sequences or sessions. Many recent approaches use more sophisticated recurrent units within their architecture, most popularly implementing a gating mechanism such as Long Short-Term Memory (LSTM) units [170] and Gated Recurrent Units (GRU) [171], such as to solve the various challenges faced by vanilla RNNs (e.g., the

TABLE 6. Recent methods based on RNNs.

Method	Base	Cat.	Enhancements
HRNN-meta [157]	GRU, RNN	CE, CA	Hierarchical sessions for inter- and intra-session relations
CTA [158]	bi-RNN, ATT	CA	Modeling temporal dynamics with contextualized self-attention
CSRM [159]	RNN, ATT	CF (I)	Collaborative information in session-based recommendation
DASL [160]	GRU, ATT	CA	Cross-domain data recommendation with dual attention
CAML [161]	GRU, ATT, FM	CE	Enhance accuracy and explainability through cross-task learning
CatDM [162]	LSTM, ATT	CE, CA	Integration of POI categories, geographical and temporal influences
ESRM-KG [163]	Tran, GRU	CE, CA	Multi-task learning by generating keywords to extract intent
SDM [164]	LSTM, ATT	CE, CA	Dynamic preference modeling using short- and long-term behaviors
SSRM [165]	GRU, MF	CA	Streaming sessions, attentive model based on MF
TailNet [166]	GRU, ATT	CA	Long-tail recommendation through a preference mechanism
Hi-RNN [167]	GRU, ATT	CA	Hierarchical model to handle irregularly spaced long- and short-term sequences
Meta-SKR [168]	GRU, ATT	CA, CE	Meta learning framework exploiting sequential, spatio-temporal and social knowledge
Deep-RegionRs [169]	Conv-LSTM, ATT	CA, CE	Capture spatio-temporal user preferences and correlations between locations

vanishing gradient problem). Furthermore, recent years have seen a dramatic increase in the proposal of approaches utilizing the *attention mechanism* [172], which was indeed first popularized in the context of recurrent networks. This enhancement can be summarized as a weighting strategy for different numerical components; a more detailed explanation is provided in Appendix A-A.

1) RNNs FOR ANONYMOUS SESSION-BASED RECOMMENDATION

Many candidate solutions for session-based recommendation are based on RNNs, a large portion of which deal with anonymous users (i.e., no historical information other than the current session is available). As a first example, [173] introduces the Neural Attentive Recommendation Machine (NARM), proposing an item-level attention mechanism to encode the user's global purpose in a session. The model presents itself as a neural encoder-decoder, where two encoders based on GRU layers encode global and local signals. An attention mechanism is applied to the hidden representations of each time-step t to emphasize it or ignore it. A collaborative framework is introduced in [159], which focuses on exploiting neighboring (but also anonymous) sessions. Two neural-based modules are applied in parallel: an "Inner Memory Encoder", that follows the NARM architecture, as well as an "Outer Memory Encoder". The first is composed of two sub-modules: one to capture global behavior from the user interaction sequence, the other to pay attention to specific behaviors and linearly combine them into a summary of the user's main purpose in a session. On the other hand, the Outer Memory Encoder extracts knowledge from similar sessions, effectively integrating a CF approach in session-based recommendation. The information from the two memory encoders are selectively combined through a fusion gating mechanism, and the recommendation score is computed through a bi-linear layer. In [163], a multi-task learning approach is proposed, incorporating keywords from product titles as soft supervision signals. Such signals are used in a keyword-generation module, which extracts the intent from the session and integrates it in the final prediction, improving performance as well as explainability. A Transformer module is used for keyword generation, while the next-click predictor module is based on a recurrent framework that utilizes GRU layers. A bi-linear layer with softmax, as with the previous approach, is used to get the probability for each item. Keyword generation is integrated into the learning process by connecting the Transformer encoder to the item predictor. An interesting topic is addressed by TailNet [166], which addresses the long-tail problem in anonymous session-based recommendation. The authors propose a "preference mechanism" to learn to balance recommendations between popular and niche (i.e., within the long tail) items.

2) RNNs FOR SHORT- AND LONG-TERM MODELING

Some approaches tackle instead scenarios where user profiles are available and propose various approaches to factor in long-term dependencies. The authors of [157], for instance, implement the Hierarchical Recurrent Network with metadata (HRNN-meta), which utilizes two different GRU models to learn intra- and inter-session representations. The idea of utilizing hierarchical recurrent architectures was first proposed by [95], which HRNN-meta builds on and extends by encoding time information as a learned embedding, allowing for more flexibility and efficiency. The authors integrate meta-data information by utilizing "field-aware" MLPs, allowing for multiple types of contextual data (other than time) to be integrated. The Sequential Deep Matching (SDM) [164] model also focuses on the evolving preferences of users by observing short- and long-term behaviors. In particular, we highlight the usage of multi-head attention on the output of a LSTM network to model the multiple interests of a user within the current session. A gated fusion module is utilized to merge global and local preference features. Intuitively, the latter combines a user representation with the short- and long-term representations, learning a gate vector that controls fusion behaviors in a similar fashion to LSTM gates. The authors recall a resemblance to attention-like models, though they argue this approach has more representational power. Similarly, the Hierarchical RNN model (Hi-RNN) [167] uses multiple GRU-based layers to represent both short- and long-term interactions, taking in consideration the time interval between inputs. Finally, the Streaming Session-based Recommendation Machine (SSRM) [165] incorporates a MF into a GRU-based encoder model, intending to integrate collaborative information into a session-based model. They focus on a streaming session-based environment, in which they enhance the short-term representation captured by the RNN encoder with the historical long-term preferences captured by MF.

3) OTHER RNN-BASED APPROACHES

Lastly, we introduce some RNN-based methods which explore different research directions. The work by [160], for instance, explores cross-domain sequential recommendations to improve CTR accuracy. The proposed Dual Attentive Sequential Learning (DASL) learns cross-domain user representations using a dual embedding strategy, which extracts latent embeddings in both domains simultaneously through metric learning. The dual embeddings are then used to initialize a GRU layer, that updates its hidden state consuming the sequence of interacted items. The dual attention mechanism then matches the embeddings with candidate items to provide cross-domain recommendations, which are obtained through a final MLP block.

The Co-Attentive Multi-task Learning (CAML) model [161] tackles recommendation explainability through an encoder-selector-decoder architecture. An encoder network is used to obtain latent representations for users and items, utilizing what they call "implicit factors" (user embeddings) as well as words item reviews. Then, a multi-pointer co-attention selector module is used to identify relevant features within reviews and concepts for both users and items. A multi-head decoder is used to generate predictions as well as a sentence explaining the recommendation in natural language. A FM is used for predictions, while a GRU-based module is used for sentence generation.

In [162], a deep LSTM-based model is proposed, meant to incorporate geographical and category information for next POI recommendation, such as to enrich sequential information. A personalized attention mechanism is used to weigh the importance of different time windows to improve recommendation accuracy.

Lastly, the authors of [158] propose to model the context of historical interactions more precisely, by factoring in "what", "when", and "how" the action took place. Most notably, they argue that session-based approaches could create a bottleneck in the way they aggregate data points in sessions, and hence distance their approach from such assumption. Their three-step approach starts by applying self-attention to the input sequence, meant to capture item correlation and long-term dependencies ("what action"). The second stage is used to learn temporal influence between interactions and the current moment of recommendation, for which multiple kernel functions are proposed ("when"). The last stage is concerned with using the temporal scores and the item representations to understand the user purpose ("how") in the session, tuning out noisy interactions that are probably less relevant and imputed to somewhat casual browsing, and it is implemented with a bi-directional RNN to capture event contexts from the past and the future.

TABLE 7. Recent methods based purely on attention mechanisms.

Method	Base	Cat.	Enhancements
CoCoRec [174]	ATT	CE, CA	Capture transition patterns within item categories
SAMN [175]	ATT	CE	Attention as measure of relevance for social signals
EATNN [176]	ATT	CE	Efficient, non-sampling attention measure of social signals
RCF [177]	HATT	CE	Use multiple relations between item for representation learning
AMM [178]	Tran, ATT	CE	Capture semantic match between historical and candidate news articles
ASReP [179]	Tran	CA	Augment short sequences with pseudo-prior items
AETN [180]	Tran	CE, CA	Overcome long-tailed app usage and sparse activity data
GeoSAN [181]	Tran	CA	Transformer for sequential POI recommendation
STAN [182]	ATT	CA	Spatio-temporal correlations with bi-attention
SSE-PT [183]	Tran	CE	Personalized user embeddings through novel regularization
SASRec [184]	Tran	CE, CA	Scalability using self-attention for next item recommendation
SASRec+ [185]	SASRec	CE, CA	Continuous time embedding in self-attention network
SMLayer [186]	SASRec	CE, CA	Attention in continuous time space with time regularization
CDR [187]	Tran, MLP	CE, CA	Disentangled learning of users' intentions from multi-feedback noisy data
DSS [188]	SASRec	CE, CA	Disentangled sequence intentions with self-supervision
SANST [189]	SASRec	CA, CE	Embed spatio-temporal information of user check-ins into self-attentive networks
PRM [190]	Tran	N/A	Modular component to re-rank items based on inter-item influences and user-level preferences
Transformer4Rec [191]	Tran	CE, CA	Multiple, focused on sequential and session-based RSs
BERT4Rec [192]	Tran	CA	Masked item prediction using bidirectional context

E. PURELY ATTENTION-BASED METHODS

As previously introduced, the attention mechanism has seen widespread use as an enhancement to various neural approaches. The authors of [70] introduced the Transformer, an architecture that has revolutionized the field of NLP and that crucially makes no use of recurrence, relying on attention as its main learning mechanism. RSs based on transformers (or its idea of basing themselves largely on attention) have naturally been a popular new approach to this task.

1) SELF-ATTENTION FOR SEQUENTIAL RECOMMENDATION

The application of self-attention modules, inspired by transformers, has been widely popular in recent proposals. A noteworthy example is the Self-Attention-based Sequential model (SASRec) [184], which attempts to capture long-term semantics in the interaction process while also being able to base the prediction on relatively few interactions. The attention mechanism seeks to identify relevant items within a user's history of interactions, basing the network's prediction on them. The Disentangled Self-Supervision (DSS) training strategy [188] aims to enhance SASRec's ability to capture multiple intentions. This approach utilizes self-supervision to reconstruct the sequence of future items as a whole (seq2seq), instead of individual items (seq2item). Moreover, the authors propose a disentanglement layer, which clusters intentions according to their distance to a set of prototypes. This is followed by an attention mechanism to encourage the model to learn user intentions over a number of latent categories.

In [185], it is argued that self-attention does not account for the time span between events, thus capturing sequential signals rather than patterns. They thus introduce various functional time feature mappings, from which they develop time embeddings compatible with self-attention. In a similar vein, [186] attempts to model both sequential behaviors as well as continuous timestamps (which measure a distance between those behaviors) with self-attention. They propose a self-modulating attention approach, which involves the re-weighting of attention coefficients according to the intensity function of temporal point processes, as well as continuous-time regularization to penalize the intensity of largely time-independent behavior data. The intuition is to adaptively and predictively re-weight past behaviors in their impact on the current score. In the same context but with a different approach, [174] proposes to tackle the sparsity of item-to-item transitions by examining the categories of items. They utilize self-attention to capture transition patterns within the same category (e.g., clothing, toys). A separate context encoder is used to predict the next interacted category, applying self-attention to interaction sessions. Finally, a collaborative module compares the users' category-specific preferences and integrates collaborative information based on users' similarities.

GeoSAN [181] also uses self-attention to model long-range dependencies, framing it in the context of sequential location recommendation. Here, the task is to predict the next location position based on the user trajectory and behaviors. The model is based on a Transformer architecture, with several modifications to handle geographical data. They also propose a new loss function based on importance sampling to obtain more informative negative samples. The Spatio-Temporal Attention Network (STAN) [182] improves over the previous work's performance by explicitly considering spatio-temporal information and the personalized item frequency (the number of times a user visits a location), using a bi-layer attention architecture.

2) OTHER ATTENTION-BASED APPROACHES

Lastly, we outline two notable Transformer-based frameworks. The Personalized Re-ranking Model (PRM) authored by [190] is a modular component that can be stacked on top of existing recommendation approaches to perform a re-ranking of item candidate lists. It uses a Transformer structure to capture item-to-item influences and a personalized module to integrate user-level preferences. A likewise worthwhile mention is the proposal by researchers at NVIDIA, which recently open-sourced the Transformers4Rec [191] library. Built upon the popular HuggingFace Transformers library [193], Transformers4Rec has the goal of encouraging the development of Transformer-based RSs, especially in sequential and session-based recommendation. The library includes various enhancements specific to the recommendation settings, and a general framework for training and evaluating different models on several built-in datasets with an incremental strategy.

F. GRAPH NEURAL NETWORK-BASED METHODS

GNNs have gained increasing popularity in recent years. Graphs have long been studied as particularly expressive structures, able to effectively capture dependencies and relationships between nodes [14], [219]; whenever a problem has an intuitive representation as a graph, approaches based on them may be able to reveal higher-order connectivity between its vertices. Many well-established approaches of popular neural network architectures have been generalized to arbitrarily structured graphs, most notably convolutions [220], [221], and have been shown to effectively propagate auxiliary information throughout the graph. There is also great interest in the application of GNNs to KGs, as we will showcase in this section. We refer to [220] for further details on graph convolutions.

1) GRAPH CONVOLUTIONAL NETWORKS

In recent years, works such as Neural Graph Collaborative Filtering (NGCF) [194] have paved the way for neural graph approaches in recommendation through the application of Graph Convolutional Networks (GCNs). The authors argue that earlier methods based on vectorial representations (i.e., embeddings), such as MF and other LFMs, can be lacking as they do not encode the collaborative signal expressed by interactions. The proposed bipartite graph structure allows the expressive modeling of high-order connectivity, which is injected in and propagated through the embedding process by utilizing an architecture akin to a standard GCN. LightGCN [203] simplifies the previous approach, yet obtains substantial improvements. The authors argue that, in the context of collaborative filtering, neighborhood aggregation is the most essential component of the GCN. The resulting network learns user and item embeddings by linear propagation on the user-item interaction graph, using a weighted sum of all layers' embeddings as the final embedding. The Self-supervised Graph Learning (SGL) paradigm [204] expands on the idea of LightGCN and explores the idea of self-supervision to supplement node representation learning via self-discrimination. In theory, this approach should mitigate bias, increase robustness to noise and encourage learning from hard negatives.

The GCN-based PinSage [205] combines efficient random walks and graph convolutions to generate node (item) embeddings, such as to incorporate both information about the graph structure and node feature information. It is a particularly worthwhile mention because of the work done

TABLE 8. Recent methods based on GNNs.

Method	Base	Cat.	Enhancements
NGCF [194]	GCN	CF (I)	Bipartite graph representation for high-order connectivity
IMfOU [195]	LSTM, GRU	CE, CA	Latent intention modeling from interaction sequences
BGCN [196]	GCN	CF (I)	User-item, user-bundle and bundle-item heterogeneous GCN
GAG [98]	GCN	CE	Session graphs with global attributes for streaming settings
LESSR [99]	GCN, GRU	CA	Lossless sessions graph encoding, long-term nodes dependencies
IMP-GCN [197]	Light-GCN	CF (I)	Address oversmoothing by applying convolutions to subgraphs
KGNN-LS [198]	GCN	CE (KG)	User-specific KGs to learn a scoring function for relationships
CKAN [199]	ATT	CE (KG)	Heterogeneous propagation of signals, attentive embedding strategy
KGIN [200]	GNN, ATT	CE	Combine KG relations attentively to model user intent
KGCN [201]	GCN	CE	Mine KG attributes with GCN for neighborhood information
MBGCN [202]	GCN	CE	Interactions as behavioral types against sparsity and cold start
TAGNN [97]	GNN, ATT	CA	Sessions as directed graphs, target-aware attention
LightGCN [203]	GCN	CF (I)	Simplify NGCF, focus on neighborhood aggregation
SGL [204]	Light-GCN	CF (I)	Self-supervised node representation learning
PinSage [205]	GCN	CE	Efficient training procedures, convolutions and random walks
FIVES [206]	GNN (conv)	CE	Automatic feature generation through adjacency tensor
MvDGAE [207]	ATT	CE	Cold start as missing link problem with denoising graph auto-encoder
KGAT [208]	GCN, ATT	CE	Information propagation, attention weighting of neighbors in KGs
DG-ENN [209]	GCN	CE	Dual graph embedding for attributes and collaborative signals
MixGCF [210]	N/A	CF (I)	Negative sampling plugin to generate hard negatives
M2GRL [211]	GCN	CE, CA	Session recommendation with side information embedding
Gemini [212]	GCN	CF (I)	Double semi-homogeneous graph with shared embeddings
NIRec [213]	GCN, ATT	CE	Efficient neighborhood-based interaction modeling on HINs
BGCF [214]	GCN	CF (I)	Bayesian Graph Convolutional Neural Network framework
LCFN [215]	GCN	CF	Low-pass collaborative filter GCN improvement
GraphRec [216]	GNN (conv)	CE	Joint framework for interactions and social user-user signals
IDCF-NN/GC [217]	ATT, GCN	CF	Inductive CF framework to learn hidden relational graphs
SR-GNN [218]	GCN, ATT	CF	Modeling of session sequences as graph data

towards architectural and training choices that make the method viable in massive graphs, with billions of nodes and edges. In [197], the oversmoothing problem is addressed directly — where node embeddings converge to a single set of values and become indistinguishable, resulting in poor performances. While the authors argue that works such as LightGCN partially address this issue by simplifying the structure, they argue that it is still largely present, and propose a novel Interest-aware Message-Passing GCN (IMP-GCN), where convolutions are performed inside subgraphs. The subgraphs consist of users with similar interests (as well as their interacted items), which should avoid transmitting information between users with little in common. The subgraphs are generated by a dedicated model based on user features and graph structure information. By limiting the amount of "negative" information, the model is proved to be more resistant to the oversmoothing issue. In [202], the type of interactions are diversified into multiple behaviors such as to contrast the data sparsity and cold start issues. The authors integrate this concept into a GCN over a heterogeneous graph based on multiple types of behavioral data, arguing that GNNs are a strong candidate in learning the difficult semantics and impact of multiple types of behaviors.

2) GNNs AND KNOWLEDGE GRAPHS

As mentioned, KGs have been studied with increasing interest as effective solutions to sparsity and cold start problems. As a prime example, [208] utilizes interactions within KGs in order to break down the interaction independence assumption. This is achieved by exploiting the links between items and their attributes (which may then be connected to other items, acting as bridges). They propose a Knowledge Graph Attention network (KGAT), which explicitly models high-order connectivities in an end-to-end fashion. Embeddings for nodes (which may be users, items, or attributes) share information through recursive propagation, regulated by a discriminative attention mechanism that weighs the importance of neighbors. In [201] an end-to-end framework inspired by GCNs on a KG representation (KGCN) is proposed. The system is able to capture inter-item relatedness by mining their associated attributes in the KG, aggregating and incorporating neighborhood information with bias when calculating the representation of the items within the graph. An extended GNN architecture is proposed by [198], aimed at simultaneously capturing user preferences as well as relationships between items. The KG is transformed into a user-specific weighted graph to address the relational heterogeneity, which, in layman's terms, attempts to learn a scoring function to weight particular relationships for users. For instance, in a movie recommendation setting, some users might be more interested in a "directed by" relationship, while others in the "lead actor" relation. They also develop a regularization technique based on label smoothness to counter overfitting (the model is hence called KGNN-LS). The Collaborative Knowledge-aware Attentive Network (CKAN) [199] extends the previous two methods and describes a novel way to integrate KG information with latent collaborative signals. This is achieved through heterogeneous propagation (collaboration and KG) and a novel attentive embedding strategy to model different conditions affecting neighboring KG entities. The authors of the Knowledge Graph-based Intent Network (KGIN) [200] propose an attentive combination of KG relations to model the intents that lie behind a user-item interaction. A newly proposed information scheme for GNNs allows for the integration of such intent information within user and item representations. This framework also allows for interpretable results (through an understanding of the intent).

3) GNN FOR SESSION-BASED RECOMMENDERS

Graph-based approaches have also been used in context-aware environments, encoding sessions within a graph structure. The Target Attentive GNN (TAGNN) [97] investigates temporal transitions of items within a session. The authors argue that prior sequence-based approaches often compress sessions into a single fixed representation, failing to consider the target items to be predicted. By representing sessions as directed graphs and introducing a target-aware attention mechanism, their GNN architecture should instead be able to activate different user interests concerning varied target items. In [98], session-based recommendation is tackled in an environment where data is produced in an online manner (in ''streams''). The authors argue that previous online learning approaches do not model sequences adequately and may easily overfit new data, losing important historical information on long-term preferences. They propose to model sessions as session graphs, where user embeddings are treated as a global attribute for the graph (Global Attributed Graph, GAG for short), and perform graph convolutions to update such global attributes. They also develop a reservoir technique based on the Wasserstein distance, which they deem more effective in sampling streaming session data. The LESSR model (Lossless Edge-order preserving aggregation and Short-cut graph attention for Session-based Recommendation) [99] addresses two issues with previous graph representations of sessions. The first issue they explore is the fact that such representations are lossy, as multiple sessions could map to an identical graph structure. The authors argue for a directed multigraph representation whose information is aggregated in an edge-order preserving manner through a GRU module. The second issue is related to the propagation of long-term dependencies, which they address by introducing attention-based shortcut connections.

G. LEARNING-TO-RANK

As mentioned, it has been argued that approaches that try to directly predict ratings may be non-optimal [42], [229]. Ideally, solving a ranking problem should require the objective function to depend on the relative distances between candidates (preference or rank), rather than the absolute rating value, which should instead have little importance. However, as we mentioned, IR metrics that are often used in the context of recommendation evaluation cannot be easily used as optimization criteria due to their non-smooth nature [38]. In this section, we explore recent approaches that put a larger focus on the learning-to-rank side, often devising surrogate ranking loss functions in an attempt to bridge the gap between training and evaluation objectives.

Various influential works have been proposed in earlier years, devising proxy approaches to optimize ranking scores directly, most commonly the Normalized Discounted Cumulative Gain (NDCG) metric. COFIRANK [230] uses Maximum Margin Matrix Factorization to this end, while [36] crafts surrogate ranking losses in both a pointwise and pairwise scenario (proposing a heuristic approach for the latter's complexity). Other notable classes of algorithms that work towards this end were proposed by the authors of SoftRank [38] and LambdaRank [231].

TABLE 9. Recent methods presenting learning-to-rank strategies.

Method	Base	Cat.	Enhancements
RankBoost+ [222]	RankBoost	CF (I)	Theoretical soundness of RankBoost and improved loss function
JoVA [223]	VAE	CF (I)	Jointly trained VAEs for user and item representation
SQL-Rank [40]	ListNet	CF	Listwise objective function to better handle ties and missing data
SetRank [224]	PMF, MLP	CF (I)	Relaxation of list ordering constraints in set representations
Set2SetRank [225]	Agnostic	CF (I)	Set-level distance measure as loss function for ranking
PushCR [226]	CR	CF (I)	Low rank representation, ranking losses for top-of-the-list accuracy
DeepRank [227]	MLP, MF	CF (I)	Neural-based listwise and pairwise model with cross-entropy and BPR losses
ListRank-MF [228]	MF	CF (E)	MF for learning to rank with cross-entropy loss for top-1 item prediction

1) PAIRWISE APPROACHES

Pairwise ranking approaches consider pairs of interactions rather than attempting to model the affinity between a single user and an item. It's worth noting that many pairwise approaches are based on the idea of Bayesian Personalized Ranking (BPR) [35], a general optimization criterion that tries to maximize the probability of binary comparison between an observed and an unobserved item, assuming the observed item will always be preferred. An example is that of DeepRank [227], which proposes a neural network model using the BPR loss for implicit feedback recommendation.

Earlier approaches such as RankBoost [232] have recently been revisited; as the name suggests, the original algorithm consists in the application of a boosting algorithm (an ensemble meta-learning algorithm widely used in classification) to the ranking framework. Effectively, this approach combines a collection of weak rankers into a single, more powerful ranking procedure. The original work proposed two pairwise ranking losses as optimization criteria; RankBoost+ [222] rectifies some issues related to the theoretical soundness of one of these approaches. Another example of pairwise approach is given by JoVA [223], a VAE-based model which we discuss in Section IV-H1. Finally, we mention PushCR [226], an approach based on collaborative ranking (CR) that experiments with three convex loss functions for ranking to emphasize the top positions of the results list.

2) LISTWISE APPROACHES

While pairwise approaches have seen great advances, authors have argued against the fact that this class of algorithms implicitly assumes that the item comparisons (the pairs) are independent. The problem, however, remains hard to solve, because of the aforementioned non-smoothness of ranking functions, hence making them unsuitable as direct loss functions. Notable works such as ListNet [39] address this by projecting labels and scores onto the probability simplex, minimizing the cross-entropy between resulting distributions. LambdaMART [233], on the other hand, dispatches the loss function entirely and formulates the gradients heuristically. While not recent, the latter approach is still considered to be among the best.

Due to its heuristic nature, LambdaMART's loss function is unknown, and it can only be *assumed* to be smooth — making theoretical analysis difficult. The work by [234] attempts to close this gap by defining a listwise ranking loss function based on cross-entropy. This modified cross-entropy loss is similar to ListNet's, and proven to provide an upper bound over the NDCG in general IR settings, hence allowing NDCG-driven optimization for retrieval problems. The Stochastic Queuing Listwise Ranking (SQL-Rank) [40] is a listwise approach that applies probabilities to permutations of the set of interacted items for every user. This work, which extends the earlier ListNet, can handle both implicit and explicit feedback, as well devising a graceful method to break ties through a stochastic shuffling process. The authors define a custom listwise loss for collaborative ranking, defined using the permutation probabilities, and highlight advantages over listwise methods that utilize the cross-entropy loss. The aforementioned DeepRank [227] also tests a listwise loss function, derived from the one used in ListRank-MF [228]. This method estimates the probability of an item being in the top position in a ranked list (i.e., top-one probability). The relation between users and items is modeled with the inner product, through a MF model. To introduce non-linearity in users and items representations, DeepRank replaces MF with a MLP with nonlinear activation functions. Differently from ListRank-MF, cross-entropy is used to optimize the top-k probability of items in the ranked list.

3) SETWISE APPROACHES

Some works have begun to incorporate setwise comparison in listwise approaches. While [224] praise the approach of SQL-Rank, they identify a weakness in the fact that only the upper bound — rather than the original negative log-likelihood — is optimized. To solve this, they propose SetRank, a setwise Bayesian approach for collaborative ranking that exploits set structures to better adapt to the recommendation with implicit feedback data (in which ties are particularly difficult to break). Their preference structure assumes users always prefer observed items over the *set* of unobserved ones. Thus, there is no need to order unobserved items. They experiment with two different models, MF-SetRank and Deep-SetRank. The first one utilizes PMF, while the second one is based on the DeepMF method [235] (an earlier MLP-based approach). The authors of Set2SetRank [225] also explore ideas based on considering sets of items, proposing a model-agnostic framework which leverages both an item-to-set and a set-to-set comparison. The first is achieved by encouraging each observed item to be ranked higher than the set of unobserved ones. The second works on setwise distances, by assuming that the sum of distances between positive instances should be less than the distance between the set of observed items and the closest unobserved item (“hard negative”). Both utilize sampling approaches for the two sets.

TABLE 10. Recent methods based on Autoencoder architectures.

Method	Base	Cat.	Enhancements
JoVA [223]	VAE	CF (I)	Jointly trained VAEs for user and item representation
BLOB [236]	VAE, Bandit	CF (I)	Combination of interaction history with “bandit” data
MacridVAE [237]	VAE	CF (I)	Disentangled user representation on fixed number of latent interests
EASE [238]	AE	CF (I)	Simple, linear autoencoder model

H. OTHER METHODS

Lastly, we make a briefer mention to two other classes of methods that are seeing much interest in recent years.

1) AUTOENCODER-BASED METHODS

Autoencoders are a type of encoder-decoder architecture in which the decoder maps back to the input space. This process forces the encoder to compress information and maintain the most important features in a low-dimensional space. Therefore, the task is to reconstruct the input with the least possible error. While generally unsupervised approaches, these architectures are often utilized with supervised learning methods to learn improved representations (embeddings) for raw input features, such as users and items in the context of RSs.

Different types of autoencoders exist, and we point to the comprehensive review from [7] for in-depth coverage. In our research, we found a rising interest in the application of a particular class of autoencoders, namely Variational Autoencoders (VAE), introduced in [239]. VAEs have a distinct probabilistic formulation, in which input samples are encoded as a probability distribution over the latent space factors, rather than a single value for each latent state attribute. This results in a representation of input data that resides in a smooth latent space.

In the aforementioned Joint Variational Autoencoder (JoVA) [223], two VAEs are assembled and jointly trained to understand user-user and item-item relationships with implicit feedback. One block reconstructs the rating matrix row-by-row (user representation), while the other reconstructs it column-by-column (item representation). The authors also propose a pairwise hinge-based loss function, to further specialize the method for top-n recommendation

TABLE 11. Recent methods based on Capsule Networks.

Method	Base	Cat.	Enhancements
MIND [240]	DR	CE	Extract multi-interests and use approximate k-nn at serve time
CARP [241]	CNN, ATT, DR	CE	Routing by bi-agreement for binary sentiment analysis of reviews
ComiRec [242]	MIND, ATT, DR	CE, CA	Multi-interest extraction with controllable aggregation module and CapsNet DR mechanism
FAT [243]	LSTM, ATT, DR	CA	Dynamic modeling of trends within user interests and user representation learning

tasks. The Macro-micro Disentangled Variational Auto-Encoder (MacridVAE) [237] tackles the complex problem of entangled representations. Briefly, an entangled representation identifies latent factors that each map to more than one generative factor; in the context of recommendation, this can be roughly understood as the learned representation for interactions being related to many different facets of the users' decision-making process. The authors therefore explore the development of a more interpretable and robust disentangled representation, based on VAEs and an information-theoretic interpretation of such models to obtain macro (e.g., user intention) and micro (e.g., descriptive factors of the item being sought) disentanglement. The Bayesian Latent Organic Bandit model (BLOB) [236] shows how to combine ''bandit'' data, information that describes how the user reacted to a sequence of recommendations, with ''organic'' data, which are sequences of naturally occurring interactions. The proposed probabilistic algorithm makes use of both these data sources, integrating advantages of VAEs and bandit-based approaches.

2) CAPSULE NETWORK-BASED METHODS

Recent work explores the usage of Capsule Networks to model dynamic user interests. The base unit in Capsules Networks (CN) is the capsule, which can be seen as a group of standard neurons (i.e., perceptrons). Differently from a perceptron, the output of a capsule is a vector instead of a scalar. An introduction on CN is given in Appendix A-B, and we refer to [244] and [245] for further details on these architectures. In the RS domain, capsules attempt to model the reasonable assumption that each user is a composition of different intents and multi-domain interests that should be recognizable by looking at their interaction sequence [240]–[242], [246].

The authors of [240] use capsules to generate multiple interest embeddings for every user. The multi-interest layer receives average pooled item embeddings as well as user embeddings, and outputs a variable number of interest vectors generated through a Dynamic Routing (DR) approach. Then, scaled dot-product attention is used to compute the importance of user interests with respect to the target item. At serve time, the capsule module is used to generate user interests, and a nearest neighbor procedure is run to generate recommended candidates. In [241], a novel routing by bi-agreement algorithm is proposed, optimized for a binary sentiment analysis task over review texts. By using a self-attention mechanism over embedding and convolutional layers, the method also aims to provide insight on which expressions and aspects of user reviews are most determining for the predicted sentiment. A general framework to extract multiple user representations is proposed by [242], such as to better capture a user's multiple interests. Both DR and self-attention mechanisms are used to generate these embeddings. The model is trained to predict the next interacted item in a sequential recommendation setting. At serve time, an approximate nearest neighbor is used to find the top-n candidates for every user interest, and an aggregation module selects the best candidates for the user. The work by [243] leverages future user behavior using DR to aggregate users' future behaviors into trend representations. A LSTM is used to compute sequence-aware user vectors. Then, a CF-inspired approach is used to select similar users and extract behavioral trends from them. A time-aware attention layer is applied to compute the future trend representation that is concatenated with the user history embedding and used to predict next item probability with a softmax operation.

3) NOTABLE MENTIONS

Lastly, we mention the existence of other noteworthy categories of methods, for which we however do not include a full section but rather point to other sources.

Reinforcement learning approaches have begun to garner attention in their deep learning variants, and the same can be said for *adversarial network-based* recommenders; [7] provides an overview of these methods. The multi-armed bandit is a reinforcement learning problem that exemplifies the exploitation-exploration dilemma. In the context of RSs, *bandit-based* algorithms [247], [248] have shown to be effective tools to promptly react to user feedback and trade-off between two goals: pleasing users by making safer bets based on historical behaviors (exploitation) and gaining knowledge about their tastes (exploration). The latter encourages showing more diverse recommendations in order to further improve user satisfaction in the long run. Reinforcement learning can also be used as an enhancement to other ML methods, as it is the case in the previously mentioned BLOB model [236]. An excellent resource on this topic is provided in [11].

Counterfactual learning has recently attracted much interest as a strategy to learn more robust representations for users and items. For instance, CauseRec [249] is a sequential model that uses contrastive learning by modeling counterfactual data distributions. They focus on denoising user representation learning, intuitively considering the retrospect question ''how would the user representation change if we intervened on the observed (historical) behavior sequence?''. The ''counterfactual'' part lies in changing the behavior sequence to observe how the representation changes.

Recent approaches are also furthering the class of *neighborhood* methods, such as [250], which apply a *k*-NN model with item frequency data and temporal dynamics to a next-basket recommendation environment. Related to distance-based methods, the authors of [251] argue that factorization and neural models, though effective, violate the triangle inequality, losing valuable fine-grained preference information. They propose to approximate users and items with Gaussian distributions use and the Wasserstein distance as a distance (preference) function between users and items. The set-based model proposed in [252] (which we refer to as "SetBased") is a straightforward and explainable method, where every user is represented as a weighted bag of interests (tags). A conceptually simple probability model is used to estimate the likelihood of tags for each user, based on the set of personalized preferences as well as the item priors (i.e., the probability of an item being liked by any user).

V. EXPERIMENTAL FACTORS

A. DATASETS AND CONSIDERATIONS

In this section, we highlight several popular datasets and their statistics, as well as describing some considerations to be made whenever splitting a dataset for the recommendation task.

1) POPULAR DATASETS

We report in Table 12 some statistics of the most popular datasets used within the research works we reviewed. Along with the number of users, items, and interactions, the table further indicates:

- whether sessions are defined explicitly;
- whether interactions are in the form of explicit ratings (EX) or implicit feedback (IM);
- the availability of additional feature for users (U) — e.g., age — items (I), — e.g., title, description — or the interaction itself (C) — e.g., context and type of interaction;
- the domain of the dataset.

We noticed that in some cases, namely with datasets like Epinions and Foursquare, researchers often crawl the data themselves. Therefore, many different versions of these datasets exist, but not all of them are published and some may be customarily built for a particular work. In such cases, datasets listed in Table 12 describe the most common version that is publicly available.

2) APPLYING EVALUATION METRICS

In order to better understand the application of evaluation metrics, which will be discussed in Sections V-B and V-C, it is important to understand how datasets in this context are utilized and partitioned.

First and foremost, evaluation procedures are assumed to be applied in an "offline" scenario, i.e., on historical data. Data is also assumed to be split as is common in most ML scenarios, i.e., a training split utilized for model building, a validation one used for parameter tuning, and a testing set that is used exclusively for evaluation (Fig. 7). Typical approaches, such as hold-out and cross-validation, may be applied.

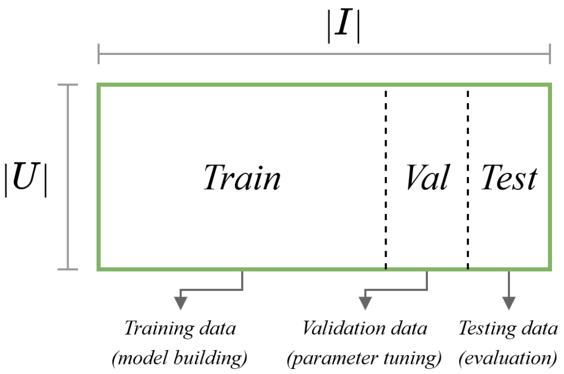


FIGURE 7. Common partitioning of a ratings matrix to accommodate for training and evaluation procedures. The validation and test portions might be selected with different procedures (e.g., random or based on time). The shown procedure evaluates weak generalization.

Validation and test portions of the data are not truly missing ratings but rather simulated through various hold-out procedures. This assumption has been widely studied [2], and such evaluation items are often characterized as *Missing Not At Random* (MNAR) or subject to a *selection bias* [275], which can lead to possibly inaccurate evaluations. This is a lengthy topic with various complications, some of which are explored in the following sections. For now, we mention that common approaches include random splits, temporal splits (utilizing more recent ratings as test data), and pre-made, fixed splits. The approach we found to be most common is the temporal one, which is, however, not entirely devoid of issues, as it does assume a certain sequential behavior model in the data; regardless, it is usually considered a reasonable choice [2].

3) STRONG AND WEAK GENERALIZATION

Another consideration to be made about evaluation procedures is the choice between "strong" or "weak" generalization protocols [276]. As discussed previously, in order to evaluate a model's generalization abilities, users (or, in some contexts, anonymous sessions) should be divided into a training and testing set. *Strong* generalization refers to a split that ensures the model is tested against completely novel user profiles. However, not all methods (especially in the case of collaborative filtering) are designed to work with novel user profiles. Such approaches are tested on a *weak* generalization protocol, where the test set is comprised of interactions from users that have already been characterized by the model (such as in Fig. 7).

In datasets where the interaction timestamp is available, a chronological split (e.g., first 80% of interactions in training, the last 20% for testing) is frequently used, though more traditional CF methods often prefer a random splitting strategy. The number of interactions reserved for testing is largely dataset- and method-specific. We found that

TABLE 12. List of most commonly used datasets.

Name	Users	Items	Interactions	Sessions	Type	Features	Domain
Netflix [54]	480K	18K	100M		EX	I	movies
Amazon 14 [253], [254]	20M	6M	143M		EX	U/I	e-comm
ML-25M [255]	163K	62K	25M		EX	I	movies
ML-20M [255]	138K	27K	20M		EX	I	movies
ML-10M [255]	72K	11K	10M		EX	I	movies
ML-1M [255]	6040	3900	1M		EX	U/I	movies
ML-100K [255]	943	1682	100K		EX	U/I	movies
Last.FM [256]	1892	18K	93K		IM	U/I (friends)	music
Douban M [257]	129K	59K	17M		EX	U (friends)	movies
Douban B [87]	5576	2680	66K		EX	N/A	books
Epinions [258]	22K	296K	922K		EX	I	e-comm
Ciao [258]	12K	107K	484K		EX	I	e-comm
Criteo [259], [260]	N/A	N/A	46M		IM	C	ads
Criteo TB [260]	N/A	N/A	4B		IM	C	ads
Avazu [261]	N/A	N/A	40M		IM	C	ads
Tmall [262], [263]	424K	1M	55M		IM	U/I	e-comm
Gowalla [264]	107K	1.3M	6.4M		IM	U/I (friends)	POI
Taobao [265]	988K	4.2M	100M		IM	U/I/C	e-comm
CiteULike [128]	5551	17K	205K		IM	U/I	news
Foursquare (NYC+TKY) [266], [267]	2763	100K	801K		IM	I	POI
MIND [268]	1M	161K	24M	✓	IM	I	news
Adressa [268], [269]	3M	48K	27M	✓	IM	I	news
Diginetica [270]	33K	44K	223K	✓	IM	N/A	e-comm
Yelp [271]	2M	150K	7M		EX	U/I/C	POI
YooChoose [272]	9.5M	50K	33M	✓	IM	I	e-comm
Steam [184], [273], [274]	2.6M	15.5K	7.8M		EX	U/I	games

session-based methods prefer to use one or a few target interactions [141], [188], while there is no clear preferred strategy among other methods.

B. ACCURACY METRICS

Here and in the following section we provide a description of the main evaluation metrics utilized in RSs. Note that, in discussing these metrics, it is common to use the terms ''relevant'' and ''irrelevant'' as an abstraction from the various types of interactions possible. Intuitively, a relevant item should be recommended (e.g., a positive implicit signal or a high explicit rating). Moving forward, we define all metrics for a generic user u, but in practice, reported metrics for a RS are always averaged over all users:

$$Average\ metric = \frac{1}{|U|}\sum_{u\in U} metric(u)$$

Though throughout this survey we frequently mentioned various critiques of accuracy-based evaluation procedures, they are often still preferred because of their simplicity. This is particularly common in contexts such as CTR prediction or next item prediction. These metrics measure the error of a predicted rating w.r.t. the real rating, i.e., for a user u and an unseen item i, $e_{ui} = \hat{r}_{ui} - r_{ui}$.

1) ROOT MEAN SQUARED ERROR

The Root Mean Squared Error (RMSE) is a metric commonly utilized in regression tasks, and is used to measure the difference between predicted and true values. A smaller RMSE indicates better performance, and the square-rooted version is usually preferred to plain the MSE, as its units are aligned with those of the ratings. Given a vector of predicted ratings $\hat{\boldsymbol{r}_u}$ and the ground truth $\boldsymbol{r}_u$, it may be defined as:

$$\mathrm{RMSE}(\hat{\boldsymbol{r}_u}, \boldsymbol{r_u}) = \left(\frac{\sum_{i=1}^{n}(\hat{r}_{ui} - r_{ui})^2}{n}\right)^{\frac{1}{2}} \tag{5}$$

where n is the number of test items (i.e., $n = |\hat{\boldsymbol{r}}| = |\boldsymbol{r}|$), u is a user and i is an item.

2) MEAN ABSOLUTE ERROR

The Mean Absolute Error (MAE) is another accuracy-based metric that is frequently used as an alternative. Notably, while RMSE tends to penalize large errors disproportionately (because of the squared term), MAE is more lenient in this regard:

$$\mathrm{MAE}(\hat{\boldsymbol{r}_u}, \boldsymbol{r_u}) = \frac{\sum_{i=1}^{n}|\hat{r}_{ui} - r_{ui}|}{n} \tag{6}$$

MAE tends to better reflect accuracy when outliers have limited importance, while RMSE values the robustness of the prediction across various ratings more highly.

C. RETRIEVAL METRICS

Though comparably not as simple in terms of direct performance feedback, retrieval (or ranking) metrics based on information retrieval theory provide a more realistic perspective of the true usefulness of a RS. These metrics typically restrict the evaluation to the first k item, and are hence commonly referred to as top-k metrics.

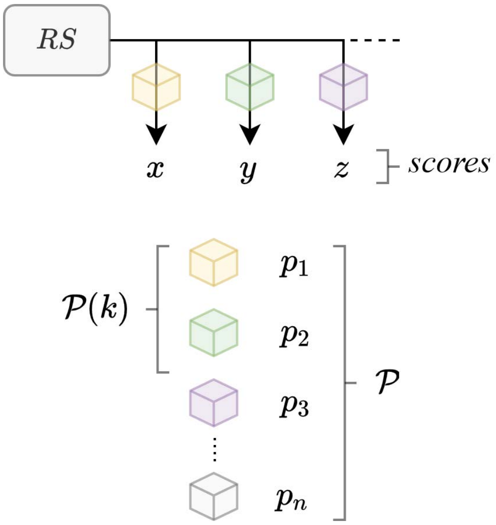


FIGURE 8. Visualization of a ranked list predicted by a RS.

Given a catalog of n items, consider a recommendation algorithm that produces a ranked list of such items. In order to make the formulations more digestible, we introduce the following notations: let $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$ with $|\mathcal{P}| = n$ be an ordered set of predicted items, generated by a scoring function, for a single user u (which we omit in the notation for the sake of simplicity). Better scores imply a higher degree of relevance of the item for the user. $\mathcal{P}$ is sorted in descending order with respect to the scores predicted (i.e., the first item is the best candidate), and p_i indicates item ranked at position i (Fig. 8). Notably, the predicted score only matters for sorting purposes. Let $\mathcal{G}$ with $|\mathcal{G}| = m$ be the list of true relevant items for the same user. We call $\mathcal{I}$ the set of all available items, either irrelevant and relevant. Whenever limiting such sets to the top k elements, we will indicate the value as a parameter, e.g., $\mathcal{P}(k) = \{p_i \in \mathcal{P} \mid i \leq k\}$. Also define $\nVdash$ as an indicator function, formally defined as:

$$\nVdash(condition) = \begin{cases} 1 & \iff condition \\ 0 & otherwise \end{cases}$$

1) NORMALIZED DISCOUNTED CUMULATIVE GAIN

The Discounted Cumulative Gain (DCG) is an overall measure of the usefulness (also called *gain*) of a list of retrieved items, weighted by how well the list is sorted. As mentioned, this is commonly restricted up to an arbitrary position $k \leq n$. The relevance score of singular items is summed, while a logarithmic discount factor is used to give more weight to higher positions and penalize lower ones.

While different approaches exist, it is common to express a utility function *util* as an exponential function of the relevance, such as to place a stronger emphasis on retrieving relevant items:

$$\text{util}(p_i) = 2^{\text{rel}\ (p_i)} - 1 \tag{7}$$

where $rel(p_i)$ is the relevance of item p_i (e.g., true rating/relevance of the item or a heuristic function thereof). For the sake of generality, we write util (p_i) rather than specifying a particular utility function.

Formally, DCG at k can be understood as an inverse logarithmic reward on all positions i that hold a relevant item:

$$\text{DCG@}k(\mathcal{P}) = \sum_{i=1}^{k} \frac{\text{util}\ (p_i)}{\log_2\ (i+1)} \tag{8}$$

The Normalized DCG (NDCG) further normalizes the score in the $0-1$ range: the DCG score is divided by the ideal DCG score (IDCG@k), which is obtained by calculating the DCG on the ground truth of relevant items:

$$\text{NDCG@}k = \frac{\text{DCG@}k}{\text{IDCG@}k} \tag{9}$$

NDCG is defined as *standard* when utilizing the inverse logarithmic decay (i.e. $\frac{1}{\log\ (i+1)}$. Note that the base of the logarithm is not important, as constant scaling will cancel out due to normalization [277].

2) RECALL

The Recall at k is the fraction of relevant items in $\mathcal{P}$ that are correctly recommended in the top-k scoring items, out of the set of relevant items $\mathcal{G}$:

$$\text{Recall@}k(\mathcal{P}) = \frac{|\mathcal{P}(k)\ \cap\ \mathcal{G}|}{|\mathcal{G}|} \tag{10}$$

As a side note, it must be considered that, if the total number of relevant items is greater than the cutoff value k (i.e., $|\mathcal{G}| > k$), the value of this metric will be lower than 1 even for perfect rankings.

3) PRECISION

The Precision at k is the fraction of relevant items in $\mathcal{P}$ that are correctly recommended among the top-k scoring items:

$$\text{Prec@}k(\mathcal{P}) = \frac{|\mathcal{P}(k)\ \cap\ \mathcal{G}|}{k} \tag{11}$$

In this case, if $|\mathcal{G}| > k$, multiple lists can achieve a perfect score as long as the top k items are relevant.

4) AVERAGE PRECISION

The Average Precision (AP) is defined as the average Precision at k over all k values that hold a true relevant item:

$$\text{AP@}k(\mathcal{P}) = \frac{\sum_{i=1}^{k} \nVdash(p_i \in \mathcal{G}) \cdot \text{Prec@}i\ (\mathcal{P})}{\min\ (|\mathcal{G}|\ , k)} \tag{12}$$

where the indicator function is used to enforce a value of 1 if the item at position k is truly relevant, 0 otherwise. The average precision also has an interpretation as the area under the precision-recall curve. We note that, as this metric is most commonly utilized in its averaged (over users) form, it is often used interchangeably as a synonym to Mean Average Precision (MAP@k), as it is implied that it is only a useful statistic when the mean of AP@k over all users is taken.

5) F-SCORE

The F-score (or F-measure) combines the Precision and Recall score in a single value, and their relative importance can be controlled with a β factor:

$$F_{\beta}(\mathcal{P}) = (1+\beta^2)\frac{\text{Prec }(\mathcal{P}) \cdot \text{Recall }(\mathcal{P})}{(\beta^2 \cdot \text{Prec }(\mathcal{P})) + \text{Recall }(\mathcal{P})} \tag{13}$$

If $\beta = 1$, both terms are equally weighted, resulting in the harmonic mean of precision and recall (usually termed F_1-score). This measure can be generalized to a F-measure@k using the previously defined Precision and Recall at k.

6) RECEIVER OPERATING CHARACTERISTIC

The Receiver Operating Characteristic (ROC) is one of the possible approaches for the evaluation of the trade-off between the length of the recommendation list (k) and the percentage of relevant items. Note that the ROC evaluates a binary setting, and hence is best suited for implicit feedback environments. The ROC depends on two measures, namely the *true-positive rate* (TPR), which is the same as recall, and the *false-positive rate* (FPR, also called inverse recall), which measures the fraction of ground truth negatives (items not interacted with) incorrectly captured in the prediction:

$$\text{FPR@}k(\mathcal{P}) = \frac{|\mathcal{P}(k) \setminus \mathcal{G}|}{|\mathcal{I} \setminus \mathcal{G}|} \tag{14}$$

The latter can be seen as a "negative" recall. The ROC curve is obtained by plotting the FPR on the x-axis and the TPR on the y-axis for varying values of k.

7) AREA UNDER CURVE

The Area under the ROC Curve (simplified to AUC) measures the likelihood that a random relevant item is ranked higher (scored better) than a random irrelevant item [278]:

$$\text{AUC}(\mathcal{P}) = \frac{\sum_{g^+ \in \mathcal{G}} \sum_{g^- \in \mathcal{I} \setminus \mathcal{G}} \mathbb{1}\left(\text{util }(g^-) < \text{util }(g^+)\right)}{|\mathcal{G}| \;\cdot\; (|\mathcal{I} \setminus \mathcal{G}|)} \tag{15}$$

Alternatively, it can also be expressed in terms of ranks rather than utilities (i.e. the *util* function), therefore requiring that the rank values (i.e., positions in the list) be sorted correctly [3].

Though AUC provides an objective and quantitative evaluation of the effectiveness of a particular method, as well as having many intuitive interpretations, this metric should be valued carefully. Among its most notable weaknesses stands the fact that it is not always the case that a method with higher AUC is strictly better than another, as the two ROC curves could cross (and, practically, they often do) at different thresholds [279]. In that case, it is hard or impossible to determine which method dominates the other. Furthermore, it should be considered that the ROC treats higher and lower ranked items equally, and is thus unable to give greater importance to higher-ranked items [2].

8) HIT RATE

We make a brief mention to Hit Rate (HR), a metric that often appears with different definitions. In many cases, it is defined as analogous to recall. Here, we describe it as measuring whether the prediction contains least *one* relevant item in the top-k results. For a single prediction:

$$\text{HR@}k(\mathcal{P}) = \mathbb{1}\,(|\mathcal{P}(k) \;\cap\; \mathcal{G}| > 0) \tag{16}$$

Generally, the hit rate is more meaningful when averaged among users. The similarity with recall is obvious; when exactly 1 relevant item exists for every user, this metric is equivalent to Recall@k.

9) MEAN RECIPROCAL RANK

The Mean Reciprocal Rank (MRR) is defined strictly on a pool of queries Q (i.e., prediction lists), where "rank" refers to the position of the first relevant item in the prediction. In other words, it measures, on average, where the first correct prediction lies. Assume the existence of a function *rank*, which returns the position of the first relevant item for a given prediction $\mathcal{P}$:

$$\text{MRR} = \frac{1}{|Q|}\sum_{\mathcal{P} \in Q} \frac{1}{\text{rank }(\mathcal{P})} \tag{17}$$

If there is no relevant item, the reciprocal rank for that prediction is 0. Since ranking positions are explicitly taken into consideration, this metric emphasizes the order of the recommendations, whereas the hit rate only cares about the existence of a relevant item.

10) SUMMARY

There is no pre-defined "best" metric for evaluation, as each metric values different aspects of the final ranking differently. Accuracy-based metrics only care about the distance between the actual and the predicted score, without directly considering the actual ranking. For ranking metrics, NDCG is comparatively better than other approaches at distinguishing between higher- and lower-ranked items. In order to verify the trade-off between precision and recall for different values of k, F-scores, average precision, and AUC can give intuitive evaluation estimates. HR and MRR can also be useful, especially if the situation requires the predicted list to contain at least one relevant item (HR) or if it is particularly important for a relevant item to be present in the higher ranks (MRR).

D. SAMPLED METRICS

While discussing metrics, we referred to the generic set of all items, containing both observed and unobserved items. However, the total number of items available in a practical setting is often up and above the hundreds of millions, which makes evaluation in real-world conditions challenging. Pointwise models, for example, would have to evaluate each user-item pair, resulting in substantial time requirements. Therefore, downsizing the set of items may be considered not only for training (with the aforementioned negative sampling approach) but also for the evaluation process, though this choice has significant influence over the results of the metrics. Keeping such considerations in mind, it comes as no surprise that many researchers use sampling strategies to speed up the evaluation process; since datasets are usually very sparse, some approaches sample a subset of unobserved items for every user. However, several studies have pointed out the difficulty of obtaining reliable performance results using metrics with sampling strategies [280]–[283].

Specifically, [280] demonstrates how sampled metrics are, in fact, *not* good indicators of the model performance when compared to the same global, non-sampled metrics. As a consequence, it is not possible to reliably compare the performance of two methods using sampled metrics, even if the two adopt the same sampling strategy for evaluation. The authors also introduce corrected versions of popular IR metrics that account for the sampling bias at the cost of higher variance. They point out that obtaining statistically significant results is still challenging, and requires at the very least the execution of several evaluation runs, such as to reduce variance. However, they conclude by saying that the only way to remove sampling bias is to avoid sampling altogether.

A recent work by [282] studies the impact of sampling on the recall@k measure, used frequently in implicit feedback recommendation settings. They demonstrate how this metric paired with sampling can be used to approximate the global metric, hence providing a more reliable measure. In another work, [284] proposes new methods to estimate the true unbiased rank distribution with approaches based on Maximal Likelihood Estimation and Maximal Entropy. However, it is still unclear how many samples should be used for a reliable evaluation.

E. EVALUATION IN RECENT WORKS

To provide practical insight to the discussion on evaluation procedures, this section presents our findings of the usage of different evaluation procedures in recent methods as applied to three popular datasets, with a foreword on reproducibility issues. We briefly introduce the most relevant preprocessing choices and evaluation strategies they describe, referring to the published code used for experiments when available. We selected methods from Tables 13 and 14 as applied to two large review datasets (Netflix, ML-20) and a common dataset for POI recommendation (Gowalla).

Foreword: The Issue of Reproducibility: In theory, research works that introduce new methods and deem them to be at a state-of-the-art level should clearly describe all the relevant details to make it possible for other researchers to validate their claim. Reproducibility of results should be ensured by publishing the training and evaluation code and, if possible, relevant data splits. Alternatively, authors ought to give precise instructions on how to generate and preprocess the data [26].

However, several studies, such as the ones in [23], [26], showcase that this is not always the case. Moreover, retrieval top-k metrics are often used with different parameters (i.e., different values of k). In many situations, we found it impossible to make a solid comparison between methods by looking at the reported performance metrics, even when they were reported on the same datasets. The main issues that caused this impossibility concerned different dataset splits or lack of enough details on how data were preprocessed and adapted to different tasks. We occasionally found it not possible to determine whether reported metrics had been computed using comparable data splits or whether they relied on sampling strategies, which would prevent direct comparison. As pointed out in [283], [311], the selected strategy to split data in training and testing set (and possibly to generate sessions) from user's historical behavior can have a considerable impact on the measured performance. Hence, methods using different data splits, even when created from the same datasets, cannot always be compared reliably without repeating tests on the same preprocessed data.

Another point of contention can be found with the conversion of datasets with explicit rating into an implicit feedback setting, most often by interpreting higher rating as a positive signal (e.g., applying a threshold such as ⊮ $(r \geq 4)$). This practice is widely diffused, seemingly with disregard of the fact that explicit ratings are a much stronger preference indicator than implicitly-gathered signals, which in turn are by nature ambiguous and thus weaker. As an example, consider a movie RS utilizing the previously described procedure. While a rating above a high threshold (e.g., $4/5$) describes a movie the user liked, an implicit signal only reveals that the movie has been watched or interacted with in specific ways. On behalf of this practice, [23] points out that there seems to be no rationale on the choice of the threshold beyond the fact that others used it before. Metrics and datasets seem to be conveniently chosen and paired with inadvertently weak baselines, giving the impression of improving a few performance metrics, despite several works warning there may not be a direct correlation between accuracy and improved recommendations [23], [312], [313]. While we understand how many of these works are custom-tailored to solve domain-specific problems and may well be worthy of attention, our goal is to show how slight variations in the problem formulation, data processing, and metrics of choice create a very fragmented landscape that lacks established benchmarking strategies of reference [25], [283].

1) THE NETFLIX PRIZE DATASET

In our search of recent contributions from top conferences, four works using the Netflix Prize dataset [54] were selected. This dataset comprises about 100 million explicit rating values assigned to 17.700 movies by more than 450.000 users. All of the studied methods binarize the dataset to emulate implicit feedback by considering movies with rating ≥ 4 as observed interactions and evaluate the models with retrieval metrics.

We start by considering the following two methods. The Embarrassingly Shallow Autoencoder (EASE) [238] is a linear model geared towards sparse data, for which the

TABLE 13. Tabulation of methods with employed datasets and code availability.

Method	Dataset																				Code
	Netflix	Amazon 14	ML-20M	ML-10M	ML-1M	ML-100K	Last.FM	Douban M	Douban B	Epinions	Tmall	Gowalla	Taobao	CiteULike	Foursquare	MIND	Diginetica	Yelp	YooChoose	Steam	
MacridVAE [237]	✓		✓		✓	✓															✓
JoVA [223]	✓				✓													✓			✓
EASE [238]	✓		✓																		
DGR [285]	✓		✓							✓								✓			
CDR [187]		✓			✓																✓
LCFN [215]		✓			✓																✓
IDCF-GC [217]		✓			✓	✓		✓													✓
NIRec [213]		✓			✓		✓														✓
DecRS [286]		✓			✓																✓
SASRec [184]		✓			✓															✓	✓
FAT [243]		✓			✓															✓	✓
SSE-PT [183]		✓		✓	✓															✓	✓
BERT4Rec [192]		✓	✓		✓															✓	✓
KGAT [208]		✓					✓											✓			✓
KGIN [200]		✓					✓														✓
LightRec [142]		✓		✓								✓		✓							✓
JTM [138]		✓											✓								✓
NGCF [194]		✓										✓						✓			✓
POI-SMF [119]												✓						✓			
ComiRec [242]		✓											✓								✓
MINM [287]		✓											✓								✓
DASL [160]		✓																			✓
CAML [161]		✓																✓			✓
SGL [204]		✓																✓			✓
BGCF [214]		✓																			✓
J3R [288]		✓																✓			✓
CARP [241]		✓																✓			✓
IMP-GCN [197]		✓										✓									✓
LightGCN [203]		✓										✓						✓			✓
NRPA [147]		✓																✓			✓
ASReP [179]		✓																			✓
DSS [188]		✓	✓		✓															✓	
AutoCF [130]		✓			✓	✓												✓			
CASR [270]		✓			✓		✓								✓		✓				
DHE [141]		✓	✓																		
DisAlign [289]		✓						✓	✓												
MF-DR-JL [108]		✓		✓																	
SMLayer [186]		✓									✓										
MIND [240]		✓																			
CauseRec [249]		✓										✓						✓			
PMLAM [251]		✓										✓									
RULE [290]		✓																✓			
EX3 [137]		✓																			
IMfOU [195]		✓																			
MixGCF [210]		✓																✓			✓
DESR [291]		✓			✓																
GAN-CDQN [292]					✓		✓						✓					✓	✓		✓
MetaHIN [135]					✓				✓									✓			✓
KTUP [87]					✓				✓												✓
MKR [131]					✓		✓														✓
ENSFM [111]					✓		✓														✓
CoCNN [154]					✓	✓	✓														✓
FedFast [293]					✓	✓												✓			
MvDGAE [207]					✓				✓									✓			
PURE [294]					✓													✓			
SCPR [295]							✓											✓			✓
SML [296]																		✓			✓
RCR [297]																		✓			
Fastformer [298]		✓														✓					✓
SQL-Rank [40]		✓			✓										✓						✓
SetRank [224]		✓			✓									✓							✓
Set2SetRank [225]		✓	✓															✓			✓
Hi-RNN [167]					✓															✓	
Meta-SKR [168]												✓						✓			
DeepRank [227]					✓	✓															✓

TABLE 14. Tabulation of methods with employed datasets and code availability (continuation).

Method	Dataset																			Code
	ML-20M	ML-10M	ML-1M	ML-100K	Last.FM	Epinions	Ciao	Criteo	Criteo TB	Avazu	Tmall	Gowalla	Taobao	CiteULike	Foursquare	MIND	Adressa	Diginetica	YooChoose	
KGCN [201]	✓				✓															✓
CKAN [199]	✓				✓															✓
KGNN-LS [198]	✓				✓															✓
Gemini [212]			✓																	
NIS-ME [112]			✓																	
P-MOIA-RS [299]			✓																	
SASRec+ [185]			✓																	✓
MAMO [134]			✓																	✓
RCF [177]				✓																✓
pAp-k [300]				✓																✓
LESSR [99]					✓							✓						✓		✓
Meta-Tree [301]			✓	✓																✓
AutoInt [133]			✓					✓		✓										✓
PEP [261]			✓					✓		✓										✓
CSRM [159]					✓														✓	✓
GAG [98]					✓							✓								✓
NextItNet [152]					✓														✓	✓
dcPF [302]					✓															✓
MBCN [148]					✓															
GLIDER [303]								✓		✓										✓
CompDLRM [140]								✓												✓
xLightFM [107]								✓		✓										✓
FIVES [206]								✓												
GeoSAN [181]												✓								✓
STAN [182]												✓			✓					✓
SSTPMF [118]												✓			✓					
SSRM [165]												✓								
Deep-RegionRs [169]												✓								
UMEC [139]								✓	✓											✓
DLRM [129]								✓	✓											✓
AutoFIS [132]									✓	✓										✓
DG-ENN [209]											✓									
ASLI [304]											✓									
MBGCN [202]											✓									
M2GRL [211]	✓																			✓
HRNN-meta [157]	✓	✓	✓										✓						✓	
OrdNMF [100]	✓																			✓
SetBased [252]	✓																			
CoCoRec [174]													✓							✓
SDM [164]																				✓
CTA [158]													✓							✓
DropoutNet [128]														✓						✓
HTD [305]														✓						✓
EATNN [176]						✓	✓													✓
SAMN [175]						✓	✓													✓
GraphRec [216]						✓	✓													✓
SCML [306]						✓	✓													✓
MDRS2 [113]						✓														
TSCMF [117]						✓														
CatDM [162]															✓					✓
HME [307]												✓			✓					
AMM [178]																✓	✓			
XLNet [191]																	✓		✓	✓
TAGNN [97]																		✓	✓	✓
ADER [308]																		✓	✓	✓
LP-RS [309]				✓																✓
Caser [151]			✓								✓	✓			✓					✓
SANST [189]												✓								
RCNN [149]												✓	✓		✓					
Transformer4Rec [191]																	✓		✓	✓
RankBoost+ [222]				✓																✓
PushCR [226]			✓	✓																
REKC [310]				✓																
SR-GNN [218]																		✓	✓	✓
EnSocialMF [116]			✓			✓														

authors report better ranking accuracy over state-of-the-art and deep models. The authors of MacridVAE [237] instead use VAEs to capture disentangled user representations. Both these methods use the same preprocessing steps to extract implicit feedback from the explicit ratings. They also declare to follow the exact same splitting strategy, where 40.000 users are held out for evaluation and the rest is used for training. Therefore, this is a strong generalization protocol. Once trained, the model is given 80% of the click history of the held-out set, and the remaining 20% is used as target. By inspecting the code available for MacridVAE, we found that all unobserved items are considered for evaluation (i.e. no negative sampling is used in evaluation). Both methods report results using NDCG@100 and Recall@20/50. These methods are evaluated under the same settings, hence their results can indeed be natively compared.

The authors of JoVA [223] report results using NDCG@1/5/10, but a random sample of 70.000 users is kept for training. Moreover, after inspecting the train/test splits generated from the ML-1M dataset (since the Netflix dataset splits are not shared in their repository), we can only assume that this model has been similarly evaluated on a weak generalization task, where interactions are randomly split and results are reported over 10% of each user's observed items. No negative sampling is used during evaluation. The last method we analyzed, the Deep Generative Ranking (DGR) [285] model, does not share its implementation, and little detail on the splitting strategy is provided. However, their conference paper seems to suggest that their evaluation on the Netflix dataset is conducted using all the available users, hence using a weak generalization framework. They also perform a separate evaluation on users with less than five ratings, but only on other datasets.

2) THE Movielens20M DATASET

The second dataset we analyze is the popular Movielens datasets, specifically in its 20-million interactions form. This dataset consists of about 20 million explicit movie ratings on a $1-5$ scale (with half-points) given by 138.493 users on 27.278 items [255].

The DHE and DSS models [141], [188] both use the same splitting strategy, but do not publicly share the experiments implementation. In both cases, all user ratings (regardless of the rating value) are considered observed interactions and are sorted by timestamp. Then, the last two interactions are put in the validation and test set respectively, and the rest are used for training. The authors of DSS further specify that their evaluation procedure relies on a negative sampling strategy: 100 items a user has never interacted with are randomly sampled according to their popularity and added to the test set. DHE results are reported using AUC while DSS uses Recall@k, NDCG@k, and MRR. M2GRL [211] works with sessions of movie ratings, created by splitting sequences into sessions for a user when two consecutive ratings are more than one year apart. Additionally, sessions with more than 50 ratings are divided into two shorter ones. Ratings lower than 3 are deleted from the dataset, so this method uses a different threshold value than the previous ones. The MetaHIN [135] model does not share the code used for its experiments, and we were not able to clearly understand the adopted strategy. Results are reported using MAE, RMSE, and NDCG@5.

The graph-based approaches KGCN [201], KGNN-LS [198] and CKAN [199] all propose to enhance recommendation using knowledge bases. The dataset is binarized using 4 as a threshold value for ratings that should be considered observed interactions, resulting in about 13.5 million interactions. During training, an equal number of unobserved interactions is randomly sampled. For evaluation, 40% of the total interactions are reserved and equally split between validation and testing, and the rest are used for training. The three models adopt a weak generalization scheme with no negative sampling in the evaluation process. Results of these methods can be compared on the reported AUC score for the CTR task.

The SetBased model proposed by [252] uses a strong generalization protocol, with a restricted dataset of about 17 million interactions and 5.800 items. To evaluate the binary relevance of predicted items, 1000 users are held-out and 20% of their ratings are used for testing. Again, a predicted rating of at least 4 is considered the minimum threshold for relevant items. MAP, NDCG, and MRR are reported, considering the list of the top-100 items. In the Ordinal NMF [100] approach, only users and movies with more than 20 interactions are kept, resulting in a smaller dataset of 20.000 users and 12.000 movies. A weak generalization scheme is also used here, as it commonly is for MF methods, without negative sampling in evaluation. Results for this method are reported using NDCG@100, but the task is framed as prediction of explicit ratings in the original scale, so it is hard to make a fair comparison with the others. EASE and MacridVAE follow the same evaluation protocol described previously for the Netflix dataset, with 10.000 held-out testing users.

3) THE GOWALLA DATASET

The Gowalla dataset [264] is a popular dataset of check-ins, where the users' friendship network is made available in the form of a graph with 190.000 nodes and 950.000 edges.

Three related graph-based methods, namely NGCF [194], LightGCN [203] and IMP-GCN [197], preprocess the dataset removing users and locations with less than 10 check-ins, keeping about 1 million interactions and 40.000 locations. Performance on the test set is evaluated by using all not-visited locations for every user, and the test set is composed of 20% of randomly sampled interactions for every user, hence using a weak generalization protocol. Results are reported using Recall@20 and NDCG@20, making these methods comparable. GeoSAN [181] seems to work with a bigger version of the dataset, with 131.000 locations and almost 3 million interactions, obtained by removing locations visited less than 10 times and users with less than 20 interactions. For efficient evaluation, the last visited location is used as target (weak generalization), while the rest are used for training, and 500 of the closest locations to the target are selected as negative samples. Between these, 100 are selected by a model trained on the same task. The hit rate and NDCG@5/10 are hence reported on a pool of 101 candidates for each user.

In the SSRM and GAG [98], [165] models, the 30.000 top locations are kept, and user sessions are created by grouping all check-ins for a single day. Sessions with more than 20 or less than 2 items are then removed. Both methods simulate a streaming context for evaluation, in which 40% of the last check-ins in chronological order are split into 5 slices, and evaluation is conducted over each slice, giving all past interactions as input. Results are reported using Recall@20 and MRR. The LESSR [99] model uses the same preprocessing described for the two previous methods to generate sessions, but the evaluation protocol is different: the last 20% of interactions for every session is used as target set. Even if these methods are evaluated on weak generalization and results reported on MRR@20, their results are hardly comparable. Negative sampling is not used in evaluation here, since the model predicts scores for all items.

In PMLAM [251], interactions are not considered as sequential, so evaluation is done similarly to purely CF algorithms, with a cleaned dataset of 1.2 million interactions. Five-folds-cross validation is used: observed interactions are randomly divided into five folds, one fold used for test and the rest for training, and metrics are the average of test results over the five splits. Negative sampling is used for training, but we are not able to clearly understand if it is also used during testing, since the implementation is not available at the time of writing. The authors of LightRec [142] approach evaluation with a similar strategy, but here, for every user, 10% of interactions are used for validation. The dataset is also reduced to about 800.000 samples, by removing users with less than 3 interactions. The code is published, and we found that their weak generalization evaluation protocol does not use negative sampling. However, results can only be tested on the ML-10M dataset, and the Gowalla splits are not shared.

The authors of STAN [182] use a subset of the dataset with 121.000 locations, 53.000 users, and 3.3M interactions. For every user with m check-ins, $m - 3$ training sequences are created. Each sequence $i \in [1, m - 3]$ is composed of the first $\{1, \ldots, i\}$ items with item $i + 1$ as target item. The test set is composed of a single sequence of the first $m - 1$ items for each user, with the last check-in as target item. No evaluation negative sampling is used, as the model outputs prediction over the whole set of items. Likewise, Deep-RegionRs [169] uses a weak generalization protocol and predicts the next location given the sequence of previous check-ins. However, during testing, candidate locations appear to be sampled based on their distance from the correct one, and results are reported with different metrics, making it incomparable with STAN. HME [307] uses a subset of Gowalla with check-in data from Houston, and the most recent 10% of each user's check-in is used for testing. Therefore, the method adopts a weak generalization strategy. Code is not published for this work. The CauseRec [249] approach uses a strong generalization protocol, and 10% of users are held out for testing. During evaluation, for each test user, 80% of interactions are used to learn the new user representation and the remaining part are the target items. We were not able to find an official implementation, and no further details are given.

We briefly mention SSTPMF [118], POI-SMF [119] and Meta-SKR [168], methods which all appear to use different subsets of the Gowalla dataset. Interactions are chosen as selected within different time spans, or filtering check-ins on a subset of cities. The splitting strategies utilized are different from one another, and the authors do not publish the code for inspection of their results.

Summary and Discussion: Most methods we analyzed follow evaluation procedures defined by previous work. When in doubt on the evaluation procedure, we consulted the published code for experiments, which, to the best of our knowledge, is only available for 2/3 of the methods listed in Tables 13 and 14. However, with the exception of related methods or the ones that directly improve over one another, they are not directly comparable without extensively editing their implementations.

This is mainly due to two reasons. First, as showcased, it is common to find methods trained and tested on different subsets of the same dataset. Furthermore, these sometimes utilize different splitting strategies for training and testing sets. The second culprit resides in substantial variations on the evaluation objective: models that operate on the same dataset are often tested on different tasks. These include but are not limited to the prediction of the next n item(s), the selection of top-n items among candidates, and the incorporation of temporal ordering in both. For example, we showcased how the Movielens20M has been framed as a purely implicit collaborative filtering task, but also as a session-based context-aware problem. In two of the analyzed works (DHE, DSS) [141], [188], all explicit ratings were considered observed interactions, but it is easy to find other research in the literature in which a threshold value (e.g., $3 - 4$) is used to convert explicit feedback to a binary form. Additionally, there is always the possibility of framing the task as explicit rating prediction, of which we studied one example (OrdNMF) [100]. Though NDCG can still be calculated on the resulting ranked list, it might be unfair to compare two methods with different optimization tasks in mind.

Even between methods working with the same implicit data, we found performance estimations reported using various top-k metrics (often with incomparable k values), as well as AUC. Even when AUC is used, the evaluation protocol can be very different. For instance, DHE and DSS evaluate the ability of the model to recommend a single relevant item, while the graph-based approaches KGCN, KGNN-LS, and CKAN [198], [199], [201] all evaluate on 20% of the interactions, with a variable number of relevant items. A similar strategy is adopted by the SetBased approach, EASE, and MacridVAE [237], [238], [252], but using a strong generalization protocol and measuring different metrics, computed over a different subset of held-out users (1000 for the SetBased approach, 10.000 for the others). NDCG@100 is reported for all three latter methods. As evaluation metrics continue to be an already controversial topic because of phenomena such as MNAR and the significance of accuracy, this fragmentation makes evaluation all the more difficult. Negative sampling in evaluation was declared to be used in only two of the analyzed works, namely DSS and GeoSAN [181], [188].

Both weak and strong generalization settings present these issues. Ideally, datasets should be standardized in the way they are split and treated, at least in regards to their testing procedures. This is further discussed in Section VI-D. Ultimately, we find that this great variety in evaluation strategies is worsened by the lack of effective benchmarks and platforms that should be used for consistent evaluation of different models [314]. In the NLP field, for instance, GLUE, SuperGLUE [315], as well as other sibling initiatives, provide strong frameworks for the evaluation of newer language models, ensuring fair and consistent results that can be easily compared with other baselines. Similar initiatives should be sought for the improvement and betterment of recommendation methods. Fortunately, the emergence of works such as the previously mentioned [23] and [24] have raised much concern, and various proposals for comprehensive recommendation frameworks are starting to be proposed. Many of these issues are being addressed by the excellent works of [191], [283], [314], [316].

VI. CHALLENGES AND RESEARCH DIRECTIONS

The ubiquity of RSs in today's digital platforms motivates research and industry to monitor closely users' online experience with the aim to continuously improve it. At the same time, the influence of AI systems on users' behaviors raises legitimate concerns about the unwanted effects that a biased RS may have when used to deliver content (like, for instance, personalized news feeds, search results, and shopping advice). Furthermore, the growing need to adhere to strict data protection regulations has steered recent research towards the development of more reliable, transparent, and privacy-aware RSs. While we previously introduced the main technical difficulties encountered in the development of a RS, this section expands on this topic and introduces other major challenges and directions addressed in current research.

A. FACING BIAS AND FAVORING DIVERSITY

It is well known that, in data-driven approaches, the lack of sufficiently diverse data can create dangerous biases, especially on consumer-faced systems such as recommendation algorithms. In general, the concept of *diversity* implies that the set of proposed recommendations within a single recommended list should be as diverse as possible. The effects of *bias* are discussed in two recent surveys [317], [318], that systematically study the sources of bias (like input data and model design) and highlight how these flaws contribute to creating unfair results. For example, they argue that the user base contained in the training historical data usually reflects the behavior of an uneven user distribution, resulting in a tendency to under-represent smaller groups. Moreover, additional inductive biases exist within models, related to the assumptions about the nature of the target function of the method of choice.

These, however, are problems that concern every data-driven system. Nonetheless, it is also possible to find biases specific to RSs, such as the ones that may originate from the users' tendency to give feedback only to content that is particularly liked or disliked and to converge towards the majority behavior (a phenomenon termed "conformance bias"). Many works also highlight the influence of the long tail phenomenon we mentioned at the start of this work, where a small number of popular items represent a considerable part of user interactions. Feedback-loops used within RSs to update user preference may reinforce this effect, known as "Matthew's effect" [317], that reduces recommendation diversity in favor of the most "likely likable" items.

Recent works try to mitigate bias effects, mostly through regularization techniques using multi-task objective functions or explicitly capturing the concept of diversity from past user interactions [291], [319]–[321]. Other notable works include the one from [322], which performs an empirical study on this phenomenon on a news dataset using different recommendation logic, finding that careful algorithm design can lead to diverse recommendations in line with manually curated news feeds. The authors of [323] test a serendipity-oriented approach based on a topic diversification algorithm to improve the variety of retrieved items.

B. EXPLAINABLE RECOMMENDER SYSTEMS

Explainability refers to the ability of a user to understand why it has received a recommendation. This can be directly related to the concept of *user trust*, which can be thought of as similar to accuracy (though not entirely the same because of its intrinsic subjectivity, among other things). Many of the works we presented rely on various artificial neural network architectures to generate recommendations. In recent years, researchers have tried to make the results of these "black-box" architectures more understandable to human subjects. Surveys from [10], [324] comprehensively cover the latest efforts and emphasize the desirability of robust RSs that can be perceived as reliable and transparent by the users. Some works focus on explicitly modeling latent factors or user profiles [325] and propose the usage of template-based systems to generate user data [252], [326]. Many works use disentangled representation learning to assist in the separation of contextual representation into a number of disjoint user and item factors that support factor-based explanations [187], [237]. Some works have gone as far as proposing the usage of language models to generate a natural language explanation base on the internal user/item representations [73], [161], [288].

C. TOWARDS FEDERATED LEARNING

Changes in data policies laws have recently pushed for the development of recommendation solutions that strike a balance between personalization and user privacy. In contrast with traditional systems, where all data is processed by a centralized infrastructure, *federated learning* enables a distributed approach. Personalized models are updated directly on user devices and then transferred to the server to be aggregated in a global model [293]. We found several recent works proposing new solutions for federated RSs [293], [327]–[329]. One recently published work explores the effectiveness of FedAttack [330], a method for launching ‘‘poisoning’’ attacks on federated RSs. This work suggests that this paradigm may be vulnerable to specific adversarial attacks that may compromise the functioning of the target RS. In [290], the authors study memory-efficient recommenders to tackle the limitations of resource-constrained edge devices, proposing ‘‘elastic embeddings’’. Such embeddings are composed of smaller blocks (sub-embeddings), similar to compositional embeddings, though its components are exclusive and not shared.

D. IMPROVING EVALUATION PROTOCOLS FOR COMPREHENSIVE EVALUATION

As already discussed, few datasets are used consistently throughout different studies and results are often difficult to compare. For example, only one of the presented datasets provides an ‘‘official’’ data split for training and testing. As a consequence, most works adopt different splitting strategies or reuse datasets from previous work without clear indication of how to retrieve them.

More importantly, this strategy can be seen as somehow compelling new research to select the dataset and evaluation strategy in function of the methods that it is seeking to improve, since results would be not comparable otherwise. This would not necessarily be a negative thing if it were not for the highly fragmented dataset and evaluation landscape. The only exception we could find is the MIND dataset [268], a relatively recent dataset whose train, test, and validation data splits have been made readily available ever since its origin. The dataset portal[3] also allows the submission of predictions on an undisclosed test set with results published on an official leader-board. Benchmarks such as this ensure that models are evaluated fairly and always using the same evaluation protocol. We find that initiatives of this kind, close in spirit to the ones that have now become standard in domains such as NLP, can effectively mitigate the fragmentation and reproducibility issues that are becoming more frequent in current research.

E. OTHER ISSUES AND EXTENSIONS

There exist other research issues and possible extensions that we did not address, some of which we briefly introduce here. We mentioned, though did not address it directly, how researchers have sought algorithms that are both *stable* and *robust*, which should imply they are not affected by fake ratings or when patterns in data evolve significantly over time [2]. Some studies have addressed *multi-criteria ratings*, i.e., approaches that distinguish between (for example) like, dislike, and no interaction at all. Other properties related to user experience such as *non-intrusiveness*, *trustworthiness* and other matters related to privacy have also been discussed with great interest [4]. Finally, much could be said about metrics related to *fairness* and *novelty*, partly related to the matters of bias and diversity we discussed before, and that are seeing more and more interest in recent research [283].

VII. CONCLUSION

In this work, we provide a comprehensive overview of the main topics necessary to develop an understanding of recent developments in recommendation systems research. We begin by discussing the relevant factors that impact the design of a recommendation algorithm, like data availability and evaluation metrics of choice. We describe a data-oriented taxonomy in line with new developments in this area and present a selection of recent traditional and neural-based approaches classified using the newly introduced categorization. We provide statistics for the most popular datasets and discuss the most common evaluation metrics used to measure an algorithm's performance. We examine the various evaluation protocols used in the researched works and make an empirical analysis concerning three datasets. Our findings highlight a lack of clearly defined testing protocols and benchmarks of reference, suggesting a dire need for systematic evaluation procedures. Finally, the survey closes with a description of the latest research trends and open challenges addressed in recent works.

APPENDIX A
ARCHITECTURAL DETAILS

This section briefly outlines two influential architectural paradigms that are extensively used in neural-based methods and also in RSs research.

A. THE ATTENTION MECHANISM

Recent research has made extensive use of various types of attention mechanisms, which can be summarized as weighting strategies for different numerical components.

1) ORIGINS OF ATTENTION

Attention has become truly ubiquitous when it saw applications in the domain of NLP, being used first in machine translation tasks [172] and later in the Transformer architecture [70]. The seminal work by [172] introduced additive attention as an enhancement over an encoder-decoder architecture based on bi-directional RNNs. Previous to this work,

[3] https://msnews.github.io

the standard approach to such encoder-decoder structures was to use a single, fixed-size context (the compressed hidden representation) as input of the decoding stage. However, long-term dependencies between tokens in the input sequence were difficult to encapsulate in such representation, as the context was, in practice, not able to compress all relevant information when it came to particularly long sequences. The authors therefore proposed to enrich the context vector fed to the decoder by instead providing all hidden states of the encoder, obtaining a different context $\boldsymbol{c_i}$ for each target position of the sequence (sentence). The context vector $\boldsymbol{c_i}$ for each target word y_i is a weighted sum over all the hidden states $\boldsymbol{h_j}$, which are the concatenation of backward and forward hidden states for input word j, as defined in Equation 18.

$$\boldsymbol{c_i} = \sum_{j=1}^{N} \alpha_{ij} \boldsymbol{h_j} \tag{18}$$

The weights α_{ij} that effectively measure the attention score between word j and target word i are computed by the attention model. These depend on the previous decoder state $\boldsymbol{s_{i-1}}$ (before generating word y_i) and the hidden state $\boldsymbol{h_j}$, as in Equation 19.

$$\alpha_{ij} = \text{softmax}\left(\text{attention}\left(\boldsymbol{s_{i-1}}, \boldsymbol{h_j}\right)\right) \tag{19}$$

In the above equation, the attention function (originally termed as the *alignment model*) was parameterized as a feed-forward neural network jointly trained with the rest of the system. This approach allows for the hidden states from each input word to influence, to different degrees, the generated word y_i (that depends on previously generated words) as well as the context $\boldsymbol{c_i}$.

2) THE TRANSFORMER ARCHITECTURE

In the Transformer architecture [70], a similar mechanism is applied to a different framework, notably without any recurrence involved. Having dispatched with recurrence, the sequential processing restrictions are lifted, allowing the authors to propose a novel encoder-decoder model that can process all input tokens in parallel. While a detailed description can be found in the original paper, we introduce the most important part of the architecture, which is the multi-head attention (MHA) layer. This layer uses "scaled dot-product" attention in order to achieve efficient computation of attention weights. In the regular attention function proposed, all input tokens are inputted and embedded simultaneously since the architecture makes no use of recurrence, and each embedding matrix $\boldsymbol{X} \in \mathbb{R}^{N \times dim}$ is projected in three different spaces through different linear transformations, generating three different input representations with values $\in \mathbb{R}^{N \times d_k}$, as in Equation 20. These are dubbed query (matrix $\boldsymbol{Q}$), keys ($\boldsymbol{K}$), and values ($\boldsymbol{V}$) following an information retrieval naming convention.

$$\boldsymbol{Q} = \boldsymbol{X}\boldsymbol{W_Q}, \quad \boldsymbol{K} = \boldsymbol{X}\boldsymbol{W_K}, \ \boldsymbol{V} = \boldsymbol{X}\boldsymbol{W_V} \tag{20}$$

Then, in a few efficient matrix operations defined in Equation 21, the whole self-attention matrix $\boldsymbol{Z} \in \mathbb{R}^{N \times d_k}$ is computed, producing the context vector for every decoded position. The authors define this mechanism as "self-attentive" because of how keys, values, and queries all come from the same place (in their case, the output of an encoder layer).

$$\boldsymbol{Z} = \text{Attention}\left(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}\right) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}}\right)\boldsymbol{V} \tag{21}$$

In the above Equation, the denominator d_k is a scaling factor, used to improve the gradient stability. Intuitively, in the NLP context, the query is the word being looked at, while keys and values both represent the past memory. The query is checked against the key matrix; the output of the matrix multiplication is passed through a softmax, obtaining a mask that allows to find the values corresponding to those keys. The idea of multi-head attention is simply to linearly project the queries, keys, and values h times with a set of learned linear projections. These operations are performed in parallel and operate on a (usually smaller) sub-space, which can learn multiple diverse representations. Their output is then concatenated and passed through a linear layer to obtain the summarized representation from all heads.

B. CAPSULE NETWORKS

Recent work explores the usage of Capsule Networks [244] to model dynamic user interests. The base unit in Capsules Networks (CN) is the capsule, which can be seen as a group of standard neurons (i.e., perceptrons). Differently from a perceptron, the output of a capsule is a vector instead of a scalar. Capsules have been first introduced in CV [331], and their operation on images is probably the easiest way to explain them. Every object in an image can be considered a composition of several sub-objects, all in a predictable position with respect to each other (e.g., eyes, mouth, and nose in a face). In the RS domain, authors translate this metaphor into the reasonable assumption that each user is a composition of different intents and multi-domain interests that should be recognizable by looking at its interaction sequence.

1) CAPSULES

A capsule is a specialized unit with a dual task (contextualized to images):

- recognize the presence of a single sub-object (estimate how likely a part of a whole object is present in an image);
- estimate the instantiation parameters of this part, computing a vector that describes the sub-object orientation in space, like its dimension, position, rotation, etc.

Hence, with respect to perceptrons, a capsule can capture much richer information about each object's spatial properties, and this information is propagated in the network and exploited in the training process. This stands in stark contrast with other lossy operations often used in CV like pooling [245]. Capsules are organized in layers, and their output is fed to the next capsule using weighted connections. Every layer of capsules specializes on recognizing more high-level objects, by using the sub-objects information captured by lower-level capsules.

2) DYNAMIC ROUTING

Since every capsule in a specific layer learns to recognize specific parts of an image with spatial information, the next layer must decide how to organize these parts consistently. The routing-by-agreement algorithm known as Dynamic Routing [245] is the key solution to this issue. To learn connection weights, each capsule tries to predict the output of every capsule in the following layer. This can be intended as an ''educated guess'' of the capsule about the object that is most likely made up of the recognized parts, and that should be found by the higher-level capsules. The entire process can be seen as a soft-clustering algorithm that creates clusters of capsules based on the agreement between their predictions and the target vectors. Predictions made by capsule i in layer l about the output of capsule j in layer $l+1$ is computed:

$$\hat{\boldsymbol{u}}_{j|i} = \boldsymbol{W_{ij}} \boldsymbol{u_i} \tag{22}$$

In the equation above, $\boldsymbol{u_i}$ is the activation vector of capsule i and matrix $\boldsymbol{W_{ij}}$ is used to learn the part-to-whole relationship between sub-objects and higher-level objects recognized by the next layer. A weight matrix $\boldsymbol{b}$ stores the connection weights between capsule i in layer l and capsule j in layer $l+1$ (entry b_{ij}). All entries of this matrix are initialized to 0. Then, a fixed number of iterations is performed to update weights for each layer. At each iteration the coupling coefficients are computed as follows:

$$\boldsymbol{c_i} \leftarrow \text{softmax}\,(\boldsymbol{b_i}), \qquad \forall\,\text{capsule} \;\; i \in l \tag{23}$$

Then, for each capsule j in layer $l+1$, the weighted sum of predictions made from capsules in layer l is computed. Here the vector $\boldsymbol{s_j}$ depends on the ''guesses'' of all lower capsules i:

$$\boldsymbol{s_j} = \sum_{i \in l} c_{ij} \hat{\boldsymbol{u}}_{j|i} \quad \boldsymbol{v_j} = \frac{||\boldsymbol{s_j}||^2}{1 + ||\boldsymbol{s_j}||^2} \cdot \frac{\boldsymbol{s_j}}{||\boldsymbol{s_j}||} \tag{24}$$

The raw sum in $\boldsymbol{s_j}$ creates an un-normalized vector with values potentially bigger than 1. Since we want the vector magnitude (norm-2) to represent the probability of the capsule ''being right'' on the recognized part, a squashing non-linear activation is applied to obtain $\boldsymbol{v_j}$, as in Equation 24.

To measure the agreement between capsules from subsequent layers, the dot product is computed between actual output $\boldsymbol{v_j}$ and predicted output $\hat{\boldsymbol{u}}_{j|i}$. This agreement score is used to update the connection weights:

$$b_{ij} \leftarrow b_{ij} + \hat{\boldsymbol{u}}_{j|i} \boldsymbol{v_j} \tag{25}$$

This way, capsules in l that were more in agreement with capsules in level $l+1$ can send a stronger signal than capsules that made a wrong prediction, with respect to higher-level capsules. After a few update rounds for each layer l, the algorithm proceeds to the next layer, until all capsule connections are weighted. This routing mechanism has been recently improved using the Expectation-Maximization algorithm [332] in order to overcome some of the limitations of the former approach.

ACKNOWLEDGMENT

(Matteo Marcuzzo and Alessandro Zangari are co-first authors.)

REFERENCES

[1] P. Resnick and H. R. Varian, ''Recommender systems,'' *Commun. ACM*, vol. 40, no. 3, pp. 56–58, 1997, doi: 10.1145/245108.245121.

[2] C. C. Aggarwal, *Recommender Systems—The Textbook*, 1st ed. Cham, Switzerland S pringer, 2016, doi: 10.1007/978-3-319-29659-3.

[3] S. Rendle, *Item Recommendation from Implicit Feedback*. New York, NY, USA: Springer, 2022, pp. 143–171, doi: 10.1007/978-1-0716-2197-4_4.

[4] G. Adomavicius and A. Tuzhilin, ''Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions,'' *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005, doi: 10.1109/TKDE.2005.99.

[5] L. Wu, X. He, X. Wang, K. Zhang, and M. Wang, ''A survey on accuracy-oriented neural recommendation: From collaborative filtering to information-rich recommendation,'' *IEEE Trans. Knowl. Data Eng.*, early access, Jan. 25, 2022, doi: 10.1109/TKDE.2022.3145690.

[6] R. Chen, Q. Hua, Y.-S. Chang, B. Wei, L. Zhang, and X. Kong, ''A survey of collaborative filtering-based recommender systems: From traditional methods to hybrid methods based on social networks,'' *IEEE Access*, vol. 6, pp. 64301–64320, 2018, doi: 10.1109/ACCESS.2018.2877208.

[7] S. Zhang, L. Yao, A. Sun, and Y. Tay, ''Deep learning based recommender system: A survey and new perspectives,'' *ACM Comput. Surv.*, vol. 52, no. 1, pp. 1–38, Jan. 2020, doi: 10.1145/3285029.

[8] M. Quadrana, P. Cremonesi, and D. Jannach, ''Sequence-aware recommender systems,'' *ACM Comput. Surv.*, vol. 51, no. 4, pp. 1–36, Jul. 2019, doi: 10.1145/3190616.

[9] S. Wang, L. Cao, Y. Wang, Q. Z. Sheng, M. A. Orgun, and D. Lian, ''A survey on session-based recommender systems,'' *ACM Comput. Surv.*, vol. 54, no. 7, pp. 1–38, Sep. 2022, doi: 10.1145/3465401.

[10] Y. Zhang and X. Chen, ''Explainable recommendation: A survey and new perspectives,'' *Found. Trends Inf. Retr.*, vol. 14, no. 1, pp. 1–101, 2020, doi: 10.1561/1500000066.

[11] N. Silva, H. Werneck, T. Silva, A. C. M. Pereira, and L. Rocha, ''Multi-armed bandits in recommendation systems: A survey of the state-of-the-art and future directions,'' *Expert Syst. Appl.*, vol. 197, Jul. 2022, Art. no. 116669, doi: 10.1016/j.eswa.2022.116669.

[12] Q. Guo, F. Zhuang, C. Qin, H. Zhu, X. Xie, H. Xiong, and Q. He, ''A survey on knowledge graph-based recommender systems,'' *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 8, pp. 3549–3568, Aug. 2022, doi: 10.1109/TKDE.2020.3028705.

[13] J. Liu and L. Duan, ''A survey on knowledge graph-based recommender systems,'' in *Proc. IEEE 5th Adv. Inf. Technol., Electron. Automat. Control Conf. (IAEAC)*, vol. 5, Mar. 2021, pp. 2450–2453, doi: 10.1109/IAEAC50856.2021.9390863.

[14] S. Wang, L. Hu, Y. Wang, X. He, Q. Z. Sheng, M. A. Orgun, L. Cao, F. Ricci, and P. S. Yu, ''Graph learning based recommender systems: A review,'' in *Proc. 13th Int. Joint Conf. Artif. Intell. (IJCAI)*, Z.-H. Zhou, Ed. Montreal, QC, Canada: IJCAI, Aug. 2021, pp. 4644–4652, doi: 10.24963/ijcai.2021/630.

[15] J. K. Tarus, Z. Niu, and G. Mustafa, ''Knowledge-based recommendation: A review of ontology-based recommender systems for e-learning,'' *Artif. Intell. Rev.*, vol. 50, no. 1, pp. 21–48, Jun. 2018, doi: 10.1007/s10462-017-9539-5.

[16] X. Bai, M. Wang, I. Lee, Z. Yang, X. Kong, and F. Xia, ''Scientific paper recommendation: A survey,'' *IEEE Access*, vol. 7, pp. 9324–9339, 2019, doi: 10.1109/ACCESS.2018.2890388.

[17] A. Menk, L. Sebastia, and R. Ferreira, ''Recommendation systems for tourism based on social networks: A survey,'' 2019, *arXiv:1903.12099*.

[18] Z. Ali, P. Kefalas, K. Muhammad, B. Ali, and M. Imran, "Deep learning in citation recommendation models survey," *Expert Syst. Appl.*, vol. 162, Dec. 2020, Art. no. 113790, doi: 10.1016/j.eswa.2020.113790.

[19] K. Chaudhari and A. Thakkar, "A comprehensive survey on travel recommender systems," *Arch. Comput. Methods Eng.*, vol. 27, no. 5, pp. 1545–1571, Nov. 2020, doi: 10.1007/s11831-019-09363-7.

[20] T. Bogers, *Tag-Based Recommendation*. Cham, Switzerland: Springer, 2018, pp. 441–479, doi: 10.1007/978-3-319-90092-6_12.

[21] A. Ghannadrad, M. Arezoumandan, L. Candela, and D. Castelli, "Recommender systems for science: A basic taxonomy," in *Proc. 18th Italian Res. Conf. Digit. Libraries (IRCDL)*, Feb. 2022, pp. 1–8. [Online]. Available: http://ircdl2022.dei.unipd.it/downloads/papers/IRCDL_2022_paper_17.pdf

[22] N. Cheong, "Personalized learning in science recommendation system based on learners' preferences," in *Proc. 3rd Int. Conf. Educ. Develop. Stud.*, New York, NY, USA, 2022, pp. 22–27, doi: 10.1145/3528137.3528161.

[23] M. Ferrari Dacrema, P. Cremonesi, and D. Jannach, "Are we really making much progress? A worrying analysis of recent neural recommendation approaches," in *Proc. 13th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2019, pp. 101–109, doi: 10.1145/3298689.3347058.

[24] S. Rendle, W. Krichene, L. Zhang, and J. Anderson, "Neural collaborative filtering vs. matrix factorization revisited," in *Proc. 14th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2020, pp. 240–248, doi: 10.1145/3383313.3412488.

[25] S. Rendle, L. Zhang, and Y. Koren, "On the difficulty of evaluating baselines: A study on recommender systems," 2019, *arXiv:1905.01395*.

[26] M. Ferrari Dacrema, S. Boglio, P. Cremonesi, and D. Jannach, "A troubling analysis of reproducibility and progress in recommender systems research," *ACM Trans. Inf. Syst.*, vol. 39, no. 2, pp. 1–49, Apr. 2021, doi: 10.1145/3434185.

[27] P. Cremonesi, Y. Koren, and R. Turrin, "Performance of recommender algorithms on top-n recommendation tasks," in *Proc. 4th ACM Conf. Recommender Syst. (RecSys)*, New York, NY, USA, 2010, pp. 39–46, doi: 10.1145/1864708.1864721.

[28] J. Bobadilla, F. Ortega, A. Hernando, and J. Bernal, "A collaborative filtering approach to mitigate the new user cold start problem," *Knowl.-Based Syst.*, vol. 26, pp. 225–238, Feb. 2012, doi: 10.1016/j.knosys.2011.07.021.

[29] C. C. Aggarwal, J. L. Wolf, K.-L. Wu, and P. S. Yu, "Horting hatches an egg: A new graph-theoretic approach to collaborative filtering," in *Proc. 5th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, New York, NY, USA, 1999, pp. 201–212, doi: 10.1145/312129.312230.

[30] G. Takács, I. Pilászy, B. Németh, and D. Tikk, "Scalable collaborative filtering approaches for large recommender systems," *J. Mach. Learn. Res.*, vol. 10, pp. 623–656, Dec. 2009.

[31] Y. Koren and R. Bell, *Advances in Collaborative Filtering*. Boston, MA, USA: Springer, 2015, pp. 77–118, doi: 10.1007/978-1-4899-7637-6_3.

[32] S. Rendle, W. Krichene, L. Zhang, and Y. Koren, "IALS++: Speeding up matrix factorization with subspace optimization," 2021, *arXiv:2110.14044*.

[33] G. Blanc and S. Rendle, "Adaptive sampled softmax with kernel based sampling," in *Proc. 35th Int. Conf. Mach. Learn.* (Proceedings of Machine Learning Research), vol. 80, J. Dy and A. Krause, Eds. Stockholm, Sweden: PMLR, 10–15, Jul. 2018, pp. 590–599. [Online]. Available: https://proceedings.mlr.press/v80/blanc18a.html

[34] Y. Bai, S. Goldman, and L. Zhang, "TAPAS: Two-pass approximate adaptive sampling for softmax," 2017, *arXiv:1707.03073*.

[35] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," in *Proc. 25th Conf. Uncertainty Artif. Intell.* Arlington, VA, USA: AUAI Press, 2009, pp. 452–461.

[36] S. Balakrishnan and S. Chopra, "Collaborative ranking," in *Proc. 5th ACM Int. Conf. Web Search Data Mining (WSDM)*, New York, NY, USA, 2012, pp. 143–152, doi: 10.1145/2124295.2124314.

[37] S. Bruch, X. Wang, M. Bendersky, and M. Najork, "An analysis of the softmax cross entropy loss for learning-to-rank with binary relevance," in *Proc. ACM SIGIR Int. Conf. Theory Inf. Retr.*, New York, NY, USA, Sep. 2019, pp. 75–78, doi: 10.1145/3341981.3344221.

[38] M. Taylor, J. Guiver, S. Robertson, and T. Minka, "SoftRank: Optimizing non-smooth rank metrics," in *Proc. Int. Conf. Web Search Web Data Mining (WSDM)*, New York, NY, USA, 2008, pp. 77–86, doi: 10.1145/1341531.1341544.

[39] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li, "Learning to rank: From pairwise approach to listwise approach," in *Proc. 24th Int. Conf. Mach. Learn. (ICML)*, New York, NY, USA, 2007, pp. 129–136, doi: 10.1145/1273496.1273513.

[40] L. Wu, C.-J. Hsieh, and J. Sharpnack, "SQL-rank: A listwise approach to collaborative ranking," in *Proc. 35th Int. Conf. Mach. Learn.* (Proceedings of Machine Learning Research), vol. 80, J. Dy and A. Krause, Eds. PMLR, Jul. 2018, pp. 5315–5324. [Online]. Available: https://proceedings.mlr.press/v80/wu18c.html

[41] S. E. Robertson, "The probability ranking principle in IR," *J. Document.*, vol. 33, pp. 294–304, Apr. 1977, doi: 10.1108/eb026647.

[42] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, "Learning to rank using gradient descent," in *Proc. 22nd Int. Conf. Mach. Learn. (ICML)*, New York, NY, USA, 2005, pp. 89–96, doi: 10.1145/1102351.1102363.

[43] P. Covington, J. Adams, and E. Sargin, "Deep neural networks for Youtube recommendations," in *Proc. 10th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2016, pp. 191–198, doi: 10.1145/2959100.2959190.

[44] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, R. Anil, Z. Haque, L. Hong, V. Jain, X. Liu, and H. Shah, "Wide & deep learning for recommender systems," in *Proc. 1st Workshop Deep Learn. Recommender Syst.*, New York, NY, USA, 2016, pp. 7–10, doi: 10.1145/2988450.2988454.

[45] Y. Shi, M. Larson, and A. Hanjalic, "Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges," *ACM Comput. Surv.*, vol. 47, no. 1, pp. 1–45, Jul. 2014, doi: 10.1145/2556270.

[46] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, "Using collaborative filtering to weave an information tapestry," *Commun. ACM*, vol. 35, no. 12, pp. 61–70, 1992, doi: 10.1145/138859.138867.

[47] P. Lops, M. de Gemmis, and G. Semeraro, *Content-Based Recommender Systems: State of the Art and Trends*. Boston, MA, USA: Springer, 2011, pp. 73–105, doi: 10.1007/978-0-387-85820-3_3.

[48] R. Burke, "Hybrid recommender systems: Survey and experiments," *User Model. User-Adapted Interact.*, vol. 12, no. 4, pp. 331–370, Nov. 2002, doi: 10.1023/A:1021240730564.

[49] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *Proc. 8th IEEE Int. Conf. Data Mining*, Dec. 2008, pp. 263–272, doi: 10.1109/ICDM.2008.22.

[50] D. Oard and J. Kim, "Implicit feedback for recommender systems," in *Proc. AAAI Workshop Recommender Syst.*, 1998, pp. 81–83.

[51] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Proc. 14th Conf. Uncertainty Artif. Intell.* San Francisco, CA, USA: Morgan Kaufmann Publishers, 1998, pp. 43–52, doi: 10.48550/arXiv.1301.7363.

[52] J. Wang, A. P. de Vries, and M. J. T. Reinders, "Unifying user-based and item-based collaborative filtering approaches by similarity fusion," in *Proc. 29th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR)*, New York, NY, USA, 2006, pp. 501–508, doi: 10.1145/1148170.1148257.

[53] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet Comput.*, vol. 7, no. 1, pp. 76–80, Jan./Feb. 2003, doi: 10.1109/MIC.2003.1167344.

[54] J. Bennett and S. Lanning, "The Netflix prize," in *Proc. KDD Cup Workshop*, Aug. 2007, pp. 1–4.

[55] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *IEEE Comput.*, vol. 42, no. 8, pp. 30–37, Aug. 2009, doi: 10.1109/MC.2009.263.

[56] S. Funk. (2006). *Netflix Update: Try This at Home*. Accessed: Mar. 25, 2022. [Online]. Available: https://sifter.org/simon/journal/20061211.html

[57] Y. Koren, "Factorization meets the neighborhood: A multifaceted collaborative filtering model," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, New York, NY, USA, 2008, pp. 426–434, doi: 10.1145/1401890.1401944.

[58] Y. Koren, "Collaborative filtering with temporal dynamics," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, 2009, pp. 447–456, doi: 10.1145/1557019.1557072.

[59] F. Fouss, A. Pirotte, and M. Saerens, "A novel way of computing similarities between nodes of a graph, with application to collaborative recommendation," in *Proc. IEEE/WIC/ACM Int. Conf. Web Intell. (WI)*, Sep. 2005, pp. 550–556, doi: 10.1109/WI.2005.9.

[60] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens, ''Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation,'' *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 3, pp. 355–369, Mar. 2007, doi: 10.1109/TKDE.2007.46.

[61] C. Cooper, S. H. Lee, T. Radzik, and Y. Siantos, ''Random walks in recommender systems: Exact computation and simulations,'' in *Proc. 23rd Int. Conf. World Wide Web*, New York, NY, USA, Apr. 2014, pp. 811–816, doi: 10.1145/2567948.2579244.

[62] B. Paudel, F. Christoffel, C. Newell, and A. Bernstein, ''Updatable, accurate, diverse, and scalable recommendations for interactive applications,'' *ACM Trans. Interact. Intell. Syst.*, vol. 7, no. 1, pp. 1–34, Mar. 2017, doi: 10.1145/2955101.

[63] P. Lops, D. Jannach, C. Musto, T. Bogers, and M. Koolen, ''Trends in content-based recommendation: Preface to the special issue on recommender systems based on rich item descriptions,'' *User Model. User-Adapted Interact.*, vol. 29, no. 2, pp. 239–249, Mar. 2019, doi: 10.1007/s11257-019-09231-w.

[64] X. Li, J. Yang, and J. Ma, ''Recent developments of content-based image retrieval (CBIR),'' *Neurocomputing*, vol. 452, pp. 675–689, Sep. 2021, doi: 10.1016/j.neucom.2020.07.139.

[65] P. Adamopoulos and A. Tuzhilin, ''On over-specialization and concentration bias of recommendations: Probabilistic neighborhood selection in collaborative filtering systems,'' in *Proc. 8th ACM Conf. Recommender Syst. (RecSys)*, New York, NY, USA, 2014, pp. 153–160, doi: 10.1145/2645710.2645752.

[66] S. Rendle, ''Factorization machines,'' in *Proc. IEEE Int. Conf. Data Mining*, New York, NY, USA, Dec. 2010, pp. 995–1000, doi: 10.1109/ICDM.2010.127.

[67] C. Cortes and V. Vapnik, ''Support-vector networks,'' *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sep. 1997, doi: 10.1007/BF00994018.

[68] S. Rendle, Z. Gantner, C. Freudenthaler, and L. Schmidt-Thieme, ''Fast context-aware recommendations with factorization machines,'' in *Proc. 34th Int. ACM SIGIR Conf. Res. Develop. Inf. (SIGIR)*, New York, NY, USA, 2011, pp. 635–644, doi: 10.1145/2009916.2010002.

[69] S. Rendle, ''Factorization machines with libFM,'' *ACM Trans. Intell. Syst. Technol.*, vol. 3, no. 3, pp. 1–22, May 2012, doi: 10.1145/2168752.2168771.

[70] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, ''Attention is all you need,'' in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.* Red Hook, NY, USA: Curran Associates, Dec. 2017, pp. 6000–6010. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[71] A. Gasparetto, M. Marcuzzo, A. Zangari, and A. Albarelli, ''A survey on text classification algorithms: From text to predictions,'' *Information*, vol. 13, no. 2, p. 83, Feb. 2022, doi: 10.3390/info13020083.

[72] A. Gasparetto, A. Zangari, M. Marcuzzo, and A. Albarelli, ''A survey on text classification: Practical perspectives on the Italian language,'' *PLoS ONE*, vol. 17, no. 7, pp. 1–46, Jul. 2022, doi: 10.1371/journal.pone.0270904.

[73] P. Sun, L. Wu, K. Zhang, Y. Fu, R. Hong, and M. Wang, ''Dual learning for explainable recommendation: Towards unifying user preference prediction and review generation,'' in *Proc. Web Conf.*, New York, NY, USA, 2020, pp. 837–847, doi: 10.1145/3366423.3380164.

[74] L. Zheng, V. Noroozi, and P. S. Yu, ''Joint deep modeling of users and items using reviews for recommendation,'' in *Proc. 10th ACM Int. Conf. Web Search Data Mining*, New York, NY, USA, Feb. 2017, pp. 425–434, doi: 10.1145/3018661.3018665.

[75] J. McAuley, C. Targett, Q. Shi, and A. van den Hengel, ''Image-based recommendations on styles and substitutes,'' in *Proc. 38th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Aug. 2015, pp. 43–52, doi: 10.1145/2766462.2767755.

[76] X. Yang, Y. Ma, L. Liao, M. Wang, and T.-S. Chua, ''TransNFCM: Translation-based neural fashion compatibility modeling,'' *Proc. AAAI Conf. Artif. Intell.*, Jul. 2019, vol. 33, no. 1, pp. 403–410. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/3811

[77] A. Gasparetto, L. Cosmo, E. Rodola, M. Bronstein, and A. Torsello, ''Spatial maps: From low rank spectral to sparse spatial functional representations,'' in *Proc. Int. Conf. 3D Vis. (3DV)*, Oct. 2017, pp. 477–485, doi: 10.1109/3DV.2017.00061.

[78] M. Pistellato, L. Cosmo, F. Bergamasco, A. Gasparetto, and A. Albarelli, ''Adaptive albedo compensation for accurate phase-shift coding,'' in *Proc. 24th Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2018, pp. 2450–2455, doi: 10.1109/ICPR.2018.8545465.

[79] A. Gasparetto, G. Minello, and A. Torsello, ''Non-parametric spectral model for shape retrieval,'' in *Proc. Int. Conf. 3D Vis.*, Oct. 2015, pp. 344–352, doi: 10.1109/3DV.2015.46.

[80] J. Lee and S. Abu-El-Haija, ''Large-scale content-only video recommendation,'' in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 987–995, doi: 10.1109/ICCVW.2017.121.

[81] J. Lee, S. Abu-El-Haija, B. Varadarajan, and A. Natsev, ''Collaborative deep metric learning for video understanding,'' in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Jul. 2018, pp. 481–490, doi: 10.1145/3219819.3219856.

[82] H. Ma, H. Yang, M. R. Lyu, and I. King, ''SoRec: Social recommendation using probabilistic matrix factorization,'' in *Proc. 17th ACM Conf. Inf. Knowl. Mining (CIKM)*, New York, NY, USA, 2008, pp. 931–940, doi: 10.1145/1458082.1458205.

[83] L. Wu, P. Sun, R. Hong, Y. Ge, and M. Wang, ''Collaborative neural social recommendation,'' *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 1, pp. 464–476, Jan. 2021, doi: 10.1109/TSMC.2018.2872842.

[84] L. Wu, J. Li, P. Sun, R. Hong, Y. Ge, and M. Wang, ''DiffNet++: A neural influence and interest diffusion network for social recommendation,'' *IEEE Trans. Knowl. Data Eng.*, early access, Dec. 31, 2021, doi: 10.1109/TKDE.2020.3048414.

[85] L. Ehrlinger and W. Wöß, ''Towards a definition of knowledge graphs,'' in *Proc. Joint Posters&Demos@SEMANTiCS Workshop Co-Located 12th Int. Conf. Semantic Syst. (SuCCESS)*, vol. 1695, Sep. 2016, pp. 1–4. [Online]. Available: http://ceur-ws.org/Vol-1695/paper4.pdf

[86] X. Wang, D. Wang, C. Xu, X. He, Y. Cao, and T.-S. Chua, ''Explainable reasoning over knowledge graphs for recommendation,'' in *Proc. 33rd AAAI Conf. Artif. Intell. 31st Innov. Appl. Artif. Intell. Conf. 9th AAAI Symp. Educ. Adv. Artif. Intell.*, Feb. 2019, pp. 1–8. [Online]. Available: https://doi.org/10.1609/aaai.v33i01.33015329

[87] Y. Cao, X. Wang, X. He, Z. Hu, and T.-S. Chua, ''Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences,'' in *Proc. World Wide Web Conf.*, New York, NY, USA, May 2019, pp. 151–161, doi: 10.1145/3308558.3313705.

[88] P. Dourish, ''What we talk about when we talk about context,'' *Pers. Ubiquitous Comput.*, vol. 8, no. 1, pp. 19–30, Feb. 2004, doi: 10.1007/s00779-003-0253-8.

[89] N. Hariri, B. Mobasher, and R. Burke, ''Context-aware music recommendation based on latenttopic sequential patterns,'' in *Proc. 6th ACM Conf. Recommender Syst. (RecSys)*, New York, NY, USA, 2012, pp. 131–138, doi: 10.1145/2365952.2365979.

[90] N. Natarajan, D. Shin, and I. S. Dhillon, ''Which app will you use next?: Collaborative filtering with interactional context,'' in *Proc. 7th ACM Conf. Recommender systems*, New York, NY, USA, 2013, pp. 201–208. [Online]. Available: https://doi.org/10.1145/2507157.2507186

[91] T. G. Kolda and B. W. Bader, ''Tensor decompositions and applications,'' *SIAM Rev.*, vol. 51, no. 3, pp. 455–500, 2009, doi: 10.1137/07070111X.

[92] P. G. Campos, F. Díez, and I. Cantador, ''Time-aware recommender systems: A comprehensive survey and analysis of existing evaluation protocols,'' *User Model. User-Adapted Interact.*, vol. 24, nos. 1–2, pp. 67–119, 2014, doi: 10.1007/s11257-012-9136-x.

[93] H. Fang, D. Zhang, Y. Shu, and G. Guo, ''Deep learning for sequential recommendation: Algorithms, influential factors, and evaluations,'' *ACM Trans. Inf. Syst.*, vol. 39, no. 1, pp. 1–42, Jan. 2021, doi: 10.1145/3426723.

[94] D. Jannach, B. Mobasher, and S. Berkovsky, ''Research directions in session-based and sequential recommendation,'' *User Model. User-Adapted Interact.*, vol. 30, no. 4, pp. 609–616, Sep. 2020, doi: 10.1007/s11257-020-09274-4.

[95] M. Quadrana, A. Karatzoglou, B. Hidasi, and P. Cremonesi, ''Personalizing session-based recommendations with hierarchical recurrent neural networks,'' in *Proc. 11th ACM Conf. Recommender Syst.*, New York, NY, USA, Aug. 2017, pp. 130–137, doi: 10.1145/3109859.3109896.

[96] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, ''Factorizing personalized Markov chains for next-basket recommendation,'' in *Proc. 19th Int. Conf. World Wide Web (WWW)*, New York, NY, USA, 2010, pp. 811–820, doi: 10.1145/1772690.1772773.

[97] F. Yu, Y. Zhu, Q. Liu, S. Wu, L. Wang, and T. Tan, ''TAGNN: Target attentive graph neural networks for session-based recommendation,'' in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2020, pp. 1921–1924, doi: 10.1145/3397271.3401319.

[98] R. Qiu, H. Yin, Z. Huang, and T. Chen, ''GAG: Global attributed graph neural network for streaming session-based recommendation,'' in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2020, pp. 669–678, doi: 10.1145/3397271.3401109.

[99] T. Chen and R. C.-W. Wong, ''Handling information loss of graph neural networks for session-based recommendation,'' in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 1172–1180, doi: 10.1145/3394486.3403170.

[100] O. Gouvert, T. Oberlin, and C. Févotte, ''Ordinal non-negative matrix factorization for recommendation,'' in *Proc. 37th Int. Conf. Mach. Learn.* (Proceedings of Machine Learning Research), vol. 119. H. D. III and A. Singh, Eds. PMLR, Jul. 2020, pp. 3680–3689. [Online]. Available: https://proceedings.mlr.press/v119/gouvert20a.html

[101] K. Liu, X. Li, Z. Zhu, L. Brand, and H. Wang, ''Factor-bounded nonnegative matrix factorization,'' *ACM Trans. Knowl. Discovery Data*, vol. 15, no. 6, pp. 1–18, May 2021, doi: 10.1145/3451395.

[102] Y. Bao, H. Fang, and J. Zhang, ''TopicMF: Simultaneously exploiting ratings and reviews for recommendation,'' in *Proc. 28th AAAI Conf. Artif. Intell.*, 2014, pp. 2–8, doi: 10.1609/aaai.v28i1.8715.

[103] Y. Lu, R. Dong, and B. Smyth, ''Convolutional matrix factorization for recommendation explanation,'' in *Proc. 23rd Int. Conf. Intell. User Interfaces Companion*, New York, NY, USA, Mar. 2018, doi: 10.1145/3180308.3180343.

[104] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, ''Neural collaborative filtering,'' in *Proc. 26th Int. Conf. World Wide Web*, New York, NY, USA, Apr. 2017, pp. 173–182, doi: 10.1145/3038912.3052569.

[105] S. Geuens, ''Factorization machines for hybrid recommendation systems based on behavioral, product, and customer data,'' in *Proc. 9th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2015, pp. 379–382, doi: 10.1145/2792838.2796542.

[106] M. Kula, ''Metadata embeddings for user and item cold-start recommendations,'' in *Proc. 2nd Workshop New Trends Content-Based Recommender Syst.*, Vienna, Austria, Sep. 2015, pp. 1–8.

[107] G. Jiang, H. Wang, J. Chen, H. Wang, D. Lian, and E. Chen, ''xLightFM: Extremely memory-efficient factorization machine,'' in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2021, pp. 337–346, doi: 10.1145/3404835.3462941.

[108] X. Wang, R. Zhang, Y. Sun, and J. Qi, ''Doubly robust joint learning for recommendation on data missing not at random,'' in *Proc. 36th Int. Conf. Mach. Learn.* (Proceedings of Machine Learning Research), vol. 97, K. Chaudhuri and R. Salakhutdinov, Eds. Long Beach, CA, USA: PMLR, Jun. 2019, pp. 6638–6647. [Online]. Available: https://proceedings.mlr.press/v97/wang19n.html

[109] J. Xiao, H. Ye, X. He, H. Zhang, F. Wu, and T.-S. Chua, ''Attentional factorization machines: Learning the weight of feature interactions via attention networks,'' in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 3119–3125.

[110] L. Chen, Y. Liu, Z. Zheng, and P. Yu, ''Heterogeneous neural attentive factorization machine for rating prediction,'' in *Proc. 27th ACM Int. Conf. Inf. Knowl. Manage.*, New York, NY, USA, Oct. 2018, pp. 833–842, doi: 10.1145/3269206.3271759.

[111] C. Chen, M. Zhang, W. Ma, Y. Liu, and S. Ma, ''Efficient non-sampling factorization mach. for optim. context-aware recommendation,'' in *Proc. Web Conf.*, New York, NY, USA, 2020, pp. 2400–2410, doi: 10.1145/3366423.3380303.

[112] M. R. Joglekar, C. Li, M. Chen, T. Xu, X. Wang, J. K. Adams, P. Khaitan, J. Liu, and Q. V. Le, ''Neural input search for large scale recommendation models,'' in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, 2020, pp. 2387–2397, doi: 10.1145/3394486.3403288.

[113] S. Bin and G. Sun, ''Matrix factorization recommendation algorithm based on multiple social relationships,'' *Math. Problems Eng.*, vol. 2021, Feb. 2021, Art. no. 6610645, doi: 10.1155/2021/6610645.

[114] Y. Xu, Y. Wu, H. Gao, S. Song, Y. Yin, and X. Xiao, ''Collaborative Apis recommendation for artificial intelligence of things with information fusion,'' *Future Gener. Comput. Syst.*, vol. 125, pp. 471–479, Dec. 2021, doi: 10.1016/j.future.2021.07.004.

[115] Y. Xu, H. Zhang, H. Gao, S. Song, Y. Yin, L. Hei, Y. Ding, and R. J. D. Barroso, ''Preference discovery from wireless social media data in Apis recommendation,'' *Wireless Netw.*, vol. 27, no. 5, pp. 3441–3451, Jul. 2021, doi: 10.1007/s11276-021-02543-z.

[116] R. Chen, Y.-S. Chang, Q. Hua, Q. Gao, X. Ji, and B. Wang, ''An enhanced social matrix factorization model for recommendation based on social networks using social interaction factors,'' *Multimedia Tools Appl.*, vol. 79, nos. 19–20, pp. 14147–14177, May 2020, doi: 10.1007/s11042-020-08620-3.

[117] H. Tahmasbi, M. Jalali, and H. Shakeri, ''TSCMF: Temporal and social collective matrix factorization model for recommender systems,'' *J. Intell. Inf. Syst.*, vol. 56, no. 1, pp. 169–187, Feb. 2021, doi: 10.1007/s10844-020-00613-w.

[118] M. Davtalab and A. A. Alesheikh, ''A POI recommendation approach integrating social spatio-temporal information into probabilistic matrix factorization,'' *Knowl. Inf. Syst.*, vol. 63, no. 1, pp. 65–85, Jan. 2021, doi: 10.1007/s10115-020-01509-5.

[119] C. Xu, A. S. Ding, and K. Zhao, ''A novel POI recommendation method based on trust relationship and spatial–temporal factors,'' *Electron. Commerce Res. Appl.*, vol. 48, Jul./Aug. 2021, Art. no. 101060, doi: 10.1016/j.elerap.2021.101060.

[120] D. D. Lee and H. S. Seung, ''Learning the parts of objects by non-negative matrix factorization,'' *Nature*, vol. 401, no. 6755, pp. 788–791, Oct. 1999, doi: 10.1038/44565.

[121] L. Zhang and S. Zhang, ''A general joint matrix factorization framework for data integration and its systematic algorithmic exploration,'' *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 9, pp. 1971–1983, Sep. 2020, doi: 10.1109/TFUZZ.2019.2928518.

[122] A. Babkin, ''Incorporating side information into robust matrix factorization with Bayesian quantile regression,'' *Statist. Probab. Lett.*, vol. 165, Oct. 2020, Art. no. 108847. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167715220301504

[123] D. Kim, C. Park, J. Oh, S. Lee, and H. Yu, ''Convolutional matrix factorization for document context-aware recommendation,'' in *Proc. 10th ACM Conf. Recommender Syst.*, New York, NY, USA, 2016, pp. 233–240. [Online]. Available: https://doi.org/10.1145/2959100.2959165

[124] R. Salakhutdinov and A. Mnih, ''Probabilistic matrix factorization,'' in *Proc. 20th Int. Conf. Neural Inf. Process. Syst* Red Hook, NY, USA: Curran Associates, Dec. 2007, pp. 1257–1264.

[125] T. Mikolov, K. Chen, G. Corrado, and J. Dean, ''Efficient estimation of word representations in vector space,'' in *Proc. 1st Int. Conf. Learn. Represent. (ICLR), Workshop Track*, Y. Bengio and Y. LeCun, Eds. Scottsdale, AZ, USA, May 2013, pp. 1–12.

[126] Q. Le and T. Mikolov, ''Distributed representations of sentences and documents,'' in *Proc. 31st Int. Conf. Int. Conf. Mach. Learn.*, vol. 3, 2014, pp. II-1188–II-1196.

[127] H. Guo, R. Tang, Y. Ye, Z. Li, X. He, and Z. Dong, ''DeepFM: An end-to-end wide & deep learning framework for CTR prediction,'' 2018, *arXiv:1804.04950*.

[128] M. Volkovs, G. Yu, and T. Poutanen, ''DropoutNet: Addressing cold start in recommender systems,'' in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2017, pp. 1–10. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/dbd22ba3bd0df8f385bdac3e9f8be207-Paper.pdf

[129] M. Naumov *et al.*, ''Deep learning recommendation model for personalization and recommendation systems,'' 2019, *arXiv:1906.00091*.

[130] C. Gao, Q. Yao, D. Jin, and Y. Li, ''Efficient data-specific model search for collaborative filtering,'' in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2021, pp. 415–425, doi: 10.1145/3447548.3467399.

[131] H. Wang, F. Zhang, M. Zhao, W. Li, X. Xie, and M. Guo, ''Multi-task feature learning for knowledge graph enhanced recommendation,'' in *Proc. World Wide Web Conf.*, New York, NY, USA, May 2019, pp. 2000–2010, doi: 10.1145/3308558.3313411.

[132] B. Liu, C. Zhu, G. Li, W. Zhang, J. Lai, R. Tang, X. He, Z. Li, and Y. Yu, ''AutoFIS: Automatic feature interaction selection in factorization models for click-through rate prediction,'' in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 2636–2645, doi: 10.1145/3394486.3403314.

[133] W. Song, C. Shi, Z. Xiao, Z. Duan, Y. Xu, M. Zhang, and J. Tang, ''AutoInt: Automatic feature interaction learning via self-attentive neural networks,'' in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, New York, NY, USA, Nov. 2019, pp. 1161–1170, doi: 10.1145/3357384.3357925.

[134] M. Dong, F. Yuan, L. Yao, X. Xu, and L. Zhu, ''MAMO: Memory-augmented meta-optimization for cold-start recommendation,'' in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 688–697, doi: 10.1145/3394486.3403113.

[135] Y. Lu, Y. Fang, and C. Shi, ''Meta-learning on heterogeneous information networks for cold-start recommendation,'' in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, 2020, pp. 1563–1573, doi: 10.1145/3394486.3403207.

[136] L. Briand, G. Salha-Galvan, W. Bendada, M. Morlon, and V.-A. Tran, ''A semi-personalized system for user cold start recommendation on music streaming apps,'' in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2021, pp. 2601–2609, doi: 10.1145/3447548.3467110.

[137] Y. Xian, T. Zhao, J. Li, J. Chan, A. Kan, J. Ma, X. L. Dong, C. Faloutsos, G. Karypis, S. Muthukrishnan, and Y. Zhang, ''EX3: Explainable attribute-aware item-set recommendations,'' in *Proc. 15th ACM Conf. Recommender Syst.*, New York, NY, USA, 2021, pp. 484–494. [Online]. Available: https://doi.org/10.1145/3460231.3474240

[138] H. Zhu, D. Chang, Z. Xu, P. Zhang, X. Li, J. He, H. Li, J. Xu, and K. Gai, ''Joint optimization of tree-based index and deep model for recommender systems,'' in *Adv. Neural Inf. Process. Syst.*, vol. 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2019, pp. 1–10. [Online]. Available: https://proceedings.neurips.cc/paper/2019/file/1c6a0198177bfcc9bd93f6aab94aad3c-Paper.pdf

[139] J. Shen, H. Wang, S. Gui, J. Tan Z. Wang, and J. Liu, ''UMEC: Unified model and embedding compression for efficient recommendation systems,'' in *Proc. Int. Conf. Learn. Represent.*, May 2021, pp. 1–13. [Online]. Available: https://openreview.net/forum?id=BM—bH_RSh

[140] H.-J.-M. Shi, D. Mudigere, M. Naumov, and J. Yang, ''Compositional embeddings using complementary partitions for memory-efficient recommendation systems,'' in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 165–175, doi: 10.1145/3394486.3403059.

[141] W.-C. Kang, D. Z. Cheng, T. Yao, X. Yi, T. Chen, L. Hong, and E. H. Chi, ''Learning to embed categorical features without embedding tables for recommendation,'' in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2021, pp. 840–850, doi: 10.1145/3447548.3467304.

[142] D. Lian, H. Wang, Z. Liu, J. Lian, E. Chen, and X. Xie, ''LightRec: A memory and search-efficient recommender system,'' in *Proc. Web Conf.*, New York, NY, USA, 2020, pp. 695–705, doi: 10.1145/3366423.3380151.

[143] J. C. Cepeda-Pacheco and M. C. Domingo, ''Deep learning and Internet of Things for tourist attraction recommendations in smart cities,'' *Neural Comput. Appl.*, vol. 34, no. 10, pp. 7691–7709, May 2022, doi: 10.1007/s00521-021-06872-0.

[144] D. Khattar, V. Kumar, V. Varma, and M. Gupta, ''Weave&Rec: A word embedding based 3-D convolutional network for news recommendation,'' in *Proc. 27th ACM Int. Conf. Inf. Knowl. Manage.*, New York, NY, USA, 2018, pp. 1855–1858, doi: 10.1145/3269206.3269307.

[145] T. X. Tuan and T. M. Phuong, ''3D convolutional networks for session-based recommendation with content features,'' in *Proc. 11th ACM Conf. Recommender Syst.*, New York, NY, USA, Aug. 2017, pp. 138–146, doi: 10.1145/3109859.3109900.

[146] C. Wu, F. Wu, M. An, J. Huang, Y. Huang, and X. Xie, ''NPA: Neural news recommendation with personalized attention,'' in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Jul. 2019, pp. 2576–2584, doi: 10.1145/3292500.3330665.

[147] H. Liu, F. Wu, W. Wang, X. Wang, P. Jiao, C. Wu, and X. Xie, ''NRPA: Neural recommendation with personalized attention,'' in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2019, pp. 1233–1236, doi: 10.1145/3331184.3331371.

[148] W. Guo, C. Zhang, H. Guo, R. Tang, and X. He, ''Multi-branch convolutional network for context-aware recommendation,'' in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2020, pp. 1709–1712, doi: 10.1145/3397271.3401218.

[149] C. Xu, P. Zhao, Y. Liu, J. Xu, V. S. S. S. Sheng, Z. Cui, X. Zhou, and H. Xiong, ''Recurrent convolutional neural network for sequential recommendation,'' in *Proc. World Wide Web Conf. (WWW)*, New York, NY, USA, 2019, pp. 3398–3404, doi: 10.1145/3308558.3313408.

[150] M. An, F. Wu, C. Wu, K. Zhang, Z. Liu, and X. Xie, ''Neural news recommendation with long- and short-term user representations,'' in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 336–345. [Online]. Available: https://aclanthology.org/P19-1033

[151] J. Tang and K. Wang, ''Personalized top-N sequential recommendation via convolutional sequence embedding,'' in *Proc. 11th ACM Int. Conf. Web Search Data Mining*, New York, NY, USA, Feb. 2018, pp. 565–573. [Online]. Available: https://doi.org/10.1145/3159652.3159656

[152] F. Yuan, A. Karatzoglou, I. Arapakis, J. M. Jose, and X. He, ''A simple convolutional generative network for next item recommendation,'' in *Proc. 12th ACM Int. Conf. Web Search Data Mining*, New York, NY, USA, Jan. 2019, pp. 582–590, doi: 10.1145/3289600.3290975.

[153] F. Yuan, X. He, H. Jiang, G. Guo, J. Xiong, Z. Xu, and Y. Xiong, ''Future data helps training: Modeling future contexts for session-based recommendation,'' in *Proc. Web Conf.*, New York, NY, USA, Apr. 2020, pp. 303–313, doi: 10.1145/3366423.3380116.

[154] M. Chen, T. Ma, and X. Zhou, ''CoCNN: Co-occurrence CNN for recommendation,'' *Expert Syst. Appl.*, vol. 195, Jun. 2022, Art. no. 116595. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417422000902

[155] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, ''BERT: Pre-training of deep bidirectional transformers for language understanding,'' in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, Minneapolis, MN, USA, vol. 1, Jun. 2019, pp. 4171–4186. [Online]. Available: https://aclanthology.org/N19-1423

[156] O. Barkan, N. Koenigstein, E. Yogev, and O. Katz, ''CB2CF: A neural multiview content-to-collaborative filtering model for completely cold item recommendations,'' in *Proc. 13th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2019, pp. 228–236, doi: 10.1145/3298689.3347038.

[157] Y. Ma, B. Narayanaswamy, H. Lin, and H. Ding, ''Temporal-contextual recommendation in real-time,'' in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 2291–2299, doi: 10.1145/3394486.3403278.

[158] J. Wu, R. Cai, and H. Wang, ''Déjà vu: A contextualized temporal attention mechanism for sequential recommendation,'' in *Proc. Web Conf.*, New York, NY, USA, 2020, pp. 2199–2209, doi: 10.1145/3366423.3380285.

[159] M. Wang, P. Ren, L. Mei, Z. Chen, J. Ma, and M. de Rijke, ''A collaborative session-based recommendation approach with parallel memory modules,'' in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2019, pp. 345–354, doi: 10.1145/3331184.3331210.

[160] P. Li, Z. Jiang, M. Que, Y. Hu, and A. Tuzhilin, ''Dual attentive sequential learning for cross-domain click-through rate prediction,'' in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2021, pp. 3172–3180, doi: 10.1145/3447548.3467140.

[161] Z. Chen, X. Wang, X. Xie, T. Wu, G. Bu, Y. Wang, and E. Chen, ''Co-attentive multi-task learning for explainable recommendation,'' in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 2137–2143, doi: 10.24963/ijcai.2019/296.

[162] F. Yu, L. Cui, W. Guo, X. Lu, Q. Li, and H. Lu, ''A category-aware deep model for successive POI recommendation on sparse check-in data,'' in *Proc. Web Conf.*, New York, NY, USA, Apr. 2020, pp. 1264–1274, doi: 10.1145/3366423.3380202.

[163] Y. Liu, Z. Ren, W.-N. Zhang, W. Che, T. Liu, and D. Yin, ''Keywords generation improves E-commerce session-based recommendation,'' in *Proc. Web Conf.*, New York, NY, USA, Apr. 2020, pp. 1604–1614, doi: 10.1145/3366423.3380232.

[164] F. Lv, T. Jin, C. Yu, F. Sun, Q. Lin, K. Yang, and W. Ng, ''SDM: Sequential deep matching model for online large-scale recommender system,'' in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, New York, NY, USA, Nov. 2019, pp. 2635–2643, doi: 10.1145/3357384.3357818.

[165] L. Guo, H. Yin, Q. Wang, T. Chen, A. Zhou, and N. Quoc Viet Hung, ''Streaming session-based recommendation,'' in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Jul. 2019, pp. 1569–1577, doi: 10.1145/3292500.3330839.

[166] S. Liu and Y. Zheng, ''Long-tail session-based recommendation,'' in *Proc. 14th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2020, pp. 509–514, doi: 10.1145/3383313.3412222.

[167] B. Choe, T. Kang, and K. Jung, ''Recommendation system with hierarchical recurrent neural network for long-term time series,'' *IEEE Access*, vol. 9, pp. 72033–72039, 2021, doi: 10.1109/ACCESS.2021.3079922.

[168] Y. Cui, H. Sun, Y. Zhao, H. Yin, and K. Zheng, ''Sequential-knowledge-aware next POI recommendation: A meta-learning approach,'' *ACM Trans. Inf. Syst.*, vol. 40, no. 2, pp. 1–22, Apr. 2022, doi: 10.1145/3460198.

[169] H. Xu, W. Ding, W. Shen, J. Wang, and Z. Yang, ''Deep convolutional recurrent model for region recommendation with spatial and temporal contexts,'' *Ad Hoc Netw.*, vol. 129, Apr. 2022, Art. no. 102545, doi: 10.1016/j.adhoc.2021.102545.

[170] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997, doi: 10.1162/neco.1997.9.8.1735.

[171] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder–decoder approaches," in *Proc. 8th Workshop Syntax, Semantics Struct. Stat. Transl. (SSST)*, Doha, Qatar, Oct. 2014, pp. 103–111. [Online]. Available: https://aclanthology.org/W14-4012

[172] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*.

[173] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma, "Neural attentive session-based recommendation," in *Proc. ACM Conf. Inf. Knowl. Manage.*, New York, NY, USA, Nov. 2017, pp. 1419–1428, doi: 10.1145/3132847.3132926.

[174] R. Cai, J. Wu, A. San, C. Wang, and H. Wang, "Category-aware collaborative sequential recommendation," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2021, pp. 388–397, doi: 10.1145/3404835.3462832.

[175] C. Chen, M. Zhang, Y. Liu, and S. Ma, "Social attentional memory network: Modeling aspect- and friend-level differences in recommendation," in *Proc. 12th ACM Int. Conf. Web Search Data Mining*, New York, NY, USA, Jan. 2019, pp. 177–185, doi: 10.1145/3289600.3290982.

[176] C. Chen, M. Zhang, C. Wang, W. Ma, M. Li, Y. Liu, and S. Ma, "An efficient adaptive transfer neural network for social-aware recommendation," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2019, pp. 225–234, doi: 10.1145/3331184.3331192.

[177] X. Xin, X. He, Y. Zhang, Y. Zhang, and J. Jose, "Relational collaborative filtering: Modeling multiple item relations for recommendation," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2019, pp. 125–134, doi: 10.1145/3331184.3331188.

[178] Q. Zhang, Q. Jia, C. Wang, J. Li, Z. Wang, and X. He, "AMM: Attentive multi-field matching for news recommendation," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2021, pp. 1588–1592, doi: 10.1145/3404835.3463232.

[179] Z. Liu, Z. Fan, Y. Wang, and P. S. Yu, "Augmenting sequential recommendation with pseudo-prior items via reversely pre-training transformer," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2021, pp. 1608–1612, doi: 10.1145/3404835.3463036.

[180] J. Zhang, B. Bai, Y. Lin, J. Liang, K. Bai, and F. Wang, "General-purpose user embeddings based on mobile app usage," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 2831–2840, doi: 10.1145/3394486.3403334.

[181] D. Lian, Y. Wu, Y. Ge, X. Xie, and E. Chen, "Geography-aware sequential location recommendation," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 2009–2019, doi: 10.1145/3394486.3403252.

[182] Y. Luo, Q. Liu, and Z. Liu, "STAN: Spatio-temporal attention network for next location recommendation," in *Proc. Web Conf.*, New York, NY, USA, Apr. 2021, pp. 2177–2185, doi: 10.1145/3442381.3449998.

[183] L. Wu, S. Li, C.-J. Hsieh, and J. Sharpnack, "SSE-PT: Sequential Recommendation Via Personalized Transformer," in *Proc. 14th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2020, pp. 328–337, doi: 10.1145/3383313.3412258.

[184] W.-C. Kang and J. McAuley, "Self-attentive sequential recommendation," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2018, pp. 197–206, doi: 10.1109/ICDM.2018.00035.

[185] D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, and K. Achan, "Self-attention with functional time representation learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32. H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2019, pp. 1–11. [Online]. Available: https://proceedings.neurips.cc/paper/2019/file/cf34645d98a7630e2bcca98b3e29c8f2-Paper.pdf

[186] C. Chen, H. Geng, N. Yang, J. Yan, D. Xue, J. Yu, and X. Yang, "Learning self-modulating attention in continuous time space with applications to sequential recommendation," in *Proc. 38th Int. Conf. Mach. Learn.* (Proceedings of Machine Learning Research), vol. 139. M. Meila and T. Zhang, Eds. PMLR, Jul. 2021, pp. 1606–1616. [Online]. Available: https://proceedings.mlr.press/v139/chen21h.html

[187] H. Chen, Y. Chen, X. Wang, R. Xie, R. Wang, F. Xia, and W. Zhu, "Curriculum disentangled recommendation with noisy multi-feedback," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34. Red Hook, NY, USA: Curran, Dec. 2021, pp. 26924–26936. [Online]. Available: https://proceedings.neurips.cc/paper/2021/file/e242660df1b69b74dcc7fde711f924ff-Paper.pdf

[188] J. Ma, C. Zhou, H. Yang, P. Cui, X. Wang, and W. Zhu, "Disentangled self-supervision in sequential recommenders," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 483–491, doi: 10.1145/3394486.3403091.

[189] Q. Guo and J. Qi, "SANST: A self-attentive network for next Point-of-Interest recommendation," 2020, *arXiv:2001.10379*.

[190] C. Pei, Y. Zhang, Y. Zhang, F. Sun, X. Lin, H. Sun, J. Wu, P. Jiang, J. Ge, W. Ou, and D. Pei, "Personalized re-ranking for recommendation," in *Proc. 13th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2019, pp. 3–11, doi: 10.1145/3298689.3347000.

[191] G. de Souza Pereira Moreira, S. Rabhi, J. M. Lee, R. Ak, and E. Oldridge, "Transformers4Rec: Bridging the gap between NLP and sequential/session-based recommendation," in *Proc. 15th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2021, pp. 143–153, doi: 10.1145/3460231.3474255.

[192] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang, "BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, New York, NY, USA, Nov. 2019, pp. 1441–1450, doi: 10.1145/3357384.3357895.

[193] T. Wolf, L. Debut, V. Sanh, and J. Chaumond, "Transformers: State-of-the-art natural language processing," in *Proc. Conf. Empirical Methods Natural Lang. Process., Syst. Demonstrations*, Oct. 2020, pp. 38–45. [Online]. Available: https://aclanthology.org/2020.emnlp-demos.6

[194] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2019, pp. 165–174, doi: 10.1145/3331184.3331267.

[195] X. Guo, C. Shi, and C. Liu, "Intention modeling from ordered and unordered facets for sequential recommendation," in *Proc. Web Conf.*, New York, NY, USA, Apr. 2020, pp. 1127–1137, doi: 10.1145/3366423.3380190.

[196] J. Chang, C. Gao, X. He, D. Jin, and Y. Li, "Bundle recommendation with graph convolutional networks," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2020, pp. 1673–1676, doi: 10.1145/3397271.3401198.

[197] F. Liu, Z. Cheng, L. Zhu, Z. Gao, and L. Nie, "Interest-aware message-passing GCN for recommendation," in *Proc. Web Conf.*, New York, NY, USA, Apr. 2021, pp. 1296–1305, doi: 10.1145/3442381.3449986.

[198] H. Wang, F. Zhang, M. Zhang, J. Leskovec, M. Zhao, W. Li, and Z. Wang, "Knowledge-aware graph neural networks with label smoothness regularization for recommender systems," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Jul. 2019, pp. 968–977, doi: 10.1145/3292500.3330836.

[199] Z. Wang, G. Lin, H. Tan, Q. Chen, and X. Liu, "CKAN: Collaborative knowledge-aware attentive network for recommender systems," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2020, pp. 219–228, doi: 10.1145/3397271.3401141.

[200] X. Wang, T. Huang, D. Wang, Y. Yuan, Z. Liu, X. He, and T.-S. Chua, "Learning intents behind interactions with knowledge graph for recommendation," in *Proc. Web Conf.*, New York, NY, USA, Apr. 2021, pp. 878–887, doi: 10.1145/3442381.3450133.

[201] H. Wang, M. Zhao, X. Xie, W. Li, and M. Guo, "Knowledge graph convolutional networks for recommender systems," in *Proc. World Wide Web Conf. (WWW)*, New York, NY, USA, 2019, pp. 3307–3313, doi: 10.1145/3308558.3313417.

[202] B. Jin, C. Gao, X. He, D. Jin, and Y. Li, "Multi-behavior recommendation with graph convolutional networks," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2020, pp. 659–668, doi: 10.1145/3397271.3401072.

[203] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "LightGCN: Simplifying and powering graph convolution network for recommendation," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2020, pp. 639–648, doi: 10.1145/3397271.3401063.

[204] J. Wu, X. Wang, F. Feng, X. He, L. Chen, J. Lian, and X. Xie, "Self-supervised graph learning for recommendation," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2021, pp. 726–735, doi: 10.1145/3404835.3462862.

[205] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Jul. 2018, pp. 974–983, doi: 10.1145/3219819.3219890.

[206] Y. Xie, Z. Wang, Y. Li, B. Ding, N. M. Gürel, C. Zhang, M. Huang, W. Lin, and J. Zhou, "FIVES: Feature interaction via edge search for large-scale tabular data," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2021, pp. 3795–3805, doi: 10.1145/3447548.3467066.

[207] J. Zheng, Q. Ma, H. Gu, and Z. Zheng, "Multi-view denoising graph auto-encoders on heterogeneous information networks for cold-start recommendation," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2021, pp. 2338–2348, doi: 10.1145/3447548.3467427.

[208] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua, "KGAT: Knowledge graph attention network for recommendation," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Jul. 2019, pp. 950–958, doi: 10.1145/3292500.3330989.

[209] W. Guo, R. Su, R. Tan, H. Guo, Y. Zhang, Z. Liu, R. Tang, and X. He, "Dual graph enhanced embedding neural network for CTR prediction," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2021, pp. 496–504, doi: 10.1145/3447548.3467384.

[210] T. Huang, Y. Dong, M. Ding, Z. Yang, W. Feng, X. Wang, and J. Tang, "MixGCF: An improved training method for graph neural network-based recommender systems," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2021, pp. 665–674, doi: 10.1145/3447548.3467408.

[211] M. Wang, Y. Lin, G. Lin, K. Yang, and X.-M. Wu, "M2GRL: A multi-task multi-view graph representation learning framework for web-scale recommender systems," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 2349–2358, doi: 10.1145/3394486.3403284.

[212] J. Xu, Z. Zhu, J. Zhao, X. Liu, M. Shan, and J. Guo, "Gemini: A novel and universal heterogeneous graph information fusing framework for online recommendations," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 3356–3365, doi: 10.1145/3394486.3403388.

[213] J. Jin, J. Qin, Y. Fang, K. Du, W. Zhang, Y. Yu, Z. Zhang, and A. J. Smola, "An efficient neighborhood-based interaction model for recommendation on heterogeneous graph," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 75–84, doi: 10.1145/3394486.3403050.

[214] J. Sun, W. Guo, D. Zhang, Y. Zhang, F. Regol, Y. Hu, H. Guo, R. Tang, H. Yuan, X. He, and M. Coates, "A framework for recommending accurate and diverse items using Bayesian graph convolutional neural networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 2030–2039, doi: 10.1145/3394486.3403254.

[215] W. Yu and Z. Qin, "Graph convolutional network for recommendation with low-pass collaborative filters," in *Proc. 37th Int. Conf. Mach. Learn.* (Proceedings of Machine Learning Research), vol. 119, H. D. III and A. Singh, Eds. PMLR, Jul. 2020, pp. 10936–10945. [Online]. Available: https://proceedings.mlr.press/v119/yu20e.html

[216] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, "Graph neural networks for social recommendation," in *Proc. World Wide Web Conf. (WWW)*, New York, NY, USA, 2019, pp. 417–426, doi: 10.1145/3308558.3313488.

[217] Q. Wu, H. Zhang, X. Gao, J. Yan, and H. Zha, "Towards open-world recommendation: An inductive model-based collaborative filtering approach," in *Proc. 38th Int. Conf. Mach. Learn.* (Proceedings of Machine Learning Research), vol. 139, M. Meila and T. Zhang, Eds. PMLR, Jul. 2021, pp. 11329–11339. [Online]. Available: https://proceedings.mlr.press/v139/wu21j.html

[218] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan, "Session-based recommendation with graph neural networks," in *Proc. AAAI Conf. Artif. Intell.*, Jul. 2019, vol. 33, no. 1, pp. 346–353. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/3804

[219] M. Schiavinato, A. Gasparetto, and A. Torsello, "Transitive assignment kernels for structural classification," in *Similarity-Based Pattern Recognition*, vol. 9370, A. Feragen, M. Pelillo, and M. Loog, Eds. Cham, Switzerland: Springer, 2015, pp. 146–159, doi: 10.1007/978-3-319-24261-3_12.

[220] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*.

[221] A. Torsello, A. Gasparetto, L. Rossi, L. Bai, and E. Hancock, "Transitive state alignment for the quantum Jensen-Shannon kernel," in *Structural, Syntactic, and Statistical Pattern Recognition* (Lecture Notes in Computer Science), vol. 8621. Berlin, Germany: Springer, 2014, pp. 22–31, doi: 10.1007/978-3-662-44415-3_3.

[222] H. Connamacher, N. Pancha, R. Liu, and S. Ray, "Rankboost+: An improvement to Rankboost," *Mach. Learn.*, vol. 109, no. 1, pp. 51–78, Jan. 2020, doi: 10.1007/s10994-019-05826-x.

[223] B. Askari, J. Szlichta, and A. Salehi-Abari, "Variational autoencoders for top-K recommendation with implicit feedback," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2021, pp. 2061–2065, doi: 10.1145/3404835.3462986.

[224] C. Wang, H. Zhu, C. Zhu, C. Qin, and H. Xiong, "SetRank: A Setwise Bayesian approach for collaborative ranking from implicit feedback," in *Proc. 34th AAAI Conf. Artif. Intell.*, Feb. 2020, vol. 34, no. 4, pp. 6127–6136. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/6077

[225] L. Chen, L. Wu, K. Zhang, R. Hong, and M. Wang, "Set2setRank: Collaborative set to set ranking for implicit feedback based recommendation," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2021, pp. 585–594, doi: 10.1145/3404835.3462886.

[226] K. Christakopoulou and A. Banerjee, "Collaborative ranking with a push at the top," in *Proc. 24th Int. Conf. World Wide Web*, Geneva, Switzerland, May 2015, pp. 205–215, doi: 10.1145/2736277.2741678.

[227] M. Chen and X. Zhou, "DeepRank: Learning to rank with neural networks for recommendation," *Knowl.-Based Syst.*, vol. 209, Dec. 2020, Art. no. 106478, doi: 10.1016/j.knosys.2020.106478.

[228] Y. Shi, M. Larson, and A. Hanjalic, "List-wise learning to rank with matrix factorization for collaborative filtering," in *Proc. 4th ACM Conf. Recommender Syst. (RecSys)*, New York, NY, USA, 2010, pp. 269–272, doi: 10.1145/1864708.1864764.

[229] T.-Y. Liu, "Learning to rank for information retrieval," *Found. Trends Inf. Retr.*, vol. 3, no. 3, pp. 225–331, 2009, doi: 10.1561/1500000016.

[230] M. Weimer, A. Karatzoglou, Q. V. Le, and A. Smola, "COFI RANK—Maximum margin matrix factorization for collaborative ranking," in *Proc. 20th Int. Conf. Neural Inf. Process. Syst.* Red Hook, NY, USA: Curran Associates, Dec. 2007, pp. 1593–1600.

[231] C. J. C. Burges, R. Ragno, and Q. V. Le, "Learning to rank with nonsmooth cost functions," in *Proc. 19th Int. Conf. Neural Inf. Process. Syst.* Cambridge, MA, USA: MIT Press, Dec. 2006, pp. 193–200.

[232] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer, "An efficient boosting algorithm for combining preferences," *J. Mach. Learn. Res.*, vol. 4, pp. 933–969, Nov. 2003.

[233] C. Burges, "From ranknet to lambdarank to lambdamart: An overview," Microsoft Res., Redmond, WA, USA, Tech. Rep. MSR-TR-2010-82, Jun. 2010. [Online]. Available: https://www.microsoft.com/en-us/research/publication/from-ranknet-to-lambdarank-to-lambdamart-an-overview/

[234] S. Bruch, "An alternative cross entropy loss for learning-to-rank," in *Proc. Web Conf.*, New York, NY, USA, Apr. 2021, pp. 118–126, doi: 10.1145/3442381.3449794.

[235] H.-J. Xue, X. Dai, J. Zhang, S. Huang, and J. Chen, "Deep matrix factorization models for recommender systems," in *Proc. 26th Int. Joint Conf. Artif. Intell. (IJCAI)*, Aug. 2017, pp. 3203–3209, doi: 10.24963/ijcai.2017/447.

[236] O. Sakhi, S. Bonner, D. Rohde, and F. Vasile, "BLOB: A probabilistic model for recommendation that combines organic and bandit signals," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 783–793, doi: 10.1145/3394486.3403121.

[237] J. Ma, C. Zhou, P. Cui, H. Yang, and W. Zhu, "Learning disentangled representations for recommendation," in *Proc. 33rd Int. Conf. Neural Inf. Process. Syst.* Red Hook, NY, USA: Curran Associates, Dec. 2019, pp. 1–12.

[238] H. Steck, "Embarrassingly shallow autoencoders for sparse data," in *Proc. World Wide Web Conf. (WWW)*, New York, NY, USA, 2019, pp. 3251–3257, doi: 10.1145/3308558.3313710.

[239] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, *arXiv:1312.6114*.

[240] C. Li, Z. Liu, M. Wu, Y. Xu, H. Zhao, P. Huang, G. Kang, Q. Chen, W. Li, and D. L. Lee, ''Multi-interest network with dynamic routing for recommendation at tmall,'' in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, New York, NY, USA, Nov. 2019, pp. 2615–2623, doi: 10.1145/3357384.3357814.

[241] C. Li, C. Quan, L. Peng, Y. Qi, Y. Deng, and L. Wu, ''A capsule network for recommendation and explaining what you like and dislike,'' in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2019, pp. 275–284, doi: 10.1145/3331184.3331216.

[242] Y. Cen, J. Zhang, X. Zou, C. Zhou, H. Yang, and J. Tang, ''Controllable multi-interest framework for recommendation,'' in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 2942–2951, doi: 10.1145/3394486.3403344.

[243] Y. Lu, S. Zhang, Y. Huang, L. Wang, X. Yu, Z. Zhao, and F. Wu, ''Future-aware diverse trends framework for recommendation,'' in *Proc. Web Conf.*, New York, NY, USA, Apr. 2021, pp. 2992–3001, doi: 10.1145/3442381.3449791.

[244] G. E. Hinton, A. Krizhevsky, and S. D. Wang, ''Transforming auto-encoders,'' in *Artificial Neural Networks and Machine Learning–(ICANN)*, T. Honkela, W. Duch, M. Girolami, and S. Kaski, Eds. Berlin, Germany: Springer, 2011, pp. 44–51, doi: 10.1007/978-3-642-21735-7_6.

[245] S. Sabour, N. Frosst, and G. E. Hinton, ''Dynamic routing between capsules,'' in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.* Red Hook, NY, USA: Curran Associates, Dec. 2017, pp. 3859–3869.

[246] M. Pistellato, F. Bergamasco, A. Albarelli, L. Cosmo, A. Gasparetto, and A. Torsello, ''Robust phase unwrapping by probabilistic consensus,'' *Opt. Lasers Eng.*, vol. 121, pp. 428–440, Oct. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0143816618317044

[247] Y. Ban, J. He, and C. B. Cook, ''Multi-facet contextual bandits: A neural network perspective,'' in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2021, pp. 35–45, doi: 10.1145/3447548.3467299.

[248] J. Sanz-Cruzado, P. Castells, and E. López, ''A simple multi-armed nearest-neighbor bandit for interactive recommendation,'' in *Proc. 13th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2019, pp. 358–362, doi: 10.1145/3298689.3347040.

[249] S. Zhang, D. Yao, Z. Zhao, T.-S. Chua, and F. Wu, ''CauseRec: Counterfactual user sequence synthesis for sequential recommendation,'' in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2021, pp. 367–377, doi: 10.1145/3404835.3462908.

[250] H. Hu, X. He, J. Gao, and Z.-L. Zhang, ''Modeling personalized item frequency information for next-basket recommendation,'' in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2020, pp. 1071–1080, doi: 10.1145/3397271.3401066.

[251] C. Ma, L. Ma, Y. Zhang, R. Tang, X. Liu, and M. Coates, ''Probabilistic metric learning with adaptive margin for top-K recommendation,'' in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 1036–1044, doi: 10.1145/3394486.3403147.

[252] K. Balog, F. Radlinski, and S. Arakelyan, ''Transparent, scrutable and explainable user models for personalized recommendation,'' in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2019, pp. 265–274, doi: 10.1145/3331184.3331211.

[253] R. He and J. McAuley, ''Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering,'' in *Proc. 25th Int. Conf. World Wide Web*, New York, NY, USA, Apr. 2016, pp. 507–517, doi: 10.1145/2872427.2883037.

[254] J. McAuley, C. Targett, Q. Shi, and A. van den Hengel, ''Image-based recommendations on styles and substitutes,'' in *Proc. 38th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Aug. 2015, pp. 43–52, doi: 10.1145/2766462.2767755.

[255] F. M. Harper and J. A. Konstan, ''The MovieLens datasets: History and context,'' *ACM Trans. Interact. Intell. Syst.*, vol. 5, no. 4, pp. 1–19, Jan. 2016, doi: 10.1145/2827872.

[256] I. Cantador, P. Brusilovsky, and T. Kuflik, ''Second workshop on information heterogeneity and fusion in recommender systems (HetRec2011),'' in *Proc. 5th ACM Conf. Recommender Syst.*, New York, NY, USA, Oct. 2011, pp. 387–388, doi: 10.1145/2043932.2044016.

[257] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King, ''Recommender systems with social regularization,'' in *Proc. 4th ACM Int. Conf. Web Search Data Mining (WSDM)*, New York, NY, USA, 2011, pp. 287–296, doi: 10.1145/1935826.1935877.

[258] J. Tang, H. Gao, and H. Liu, ''MTrust: Discerning multi-faceted trust in a connected world,'' in *Proc. 5th ACM Int. Conf. Web Search Data Mining (WSDM)*, New York, NY, USA, 2012, pp. 93–102, doi: 10.1145/2124295.2124309.

[259] P. Zhao, K. Xiao, Y. Zhang, K. Bian, and W. Yan, ''AMEIR: Automatic behavior modeling, interaction exploration and MLP investigation in the recommender system,'' 2020, *arXiv:2006.05933*.

[260] *Criteo Research Datasets*. Criteo AI Lab. Accessed: Mar. 25, 2022. [Online]. Available: https://ailab.criteo.com/ressources

[261] S. Liu, C. Gao, Y. Chen, D. Jin, and Y. Li, ''Learnable embedding sizes for recommender systems,'' 2021, *arXiv:2101.07577*.

[262] J. Qin, W. Zhang, X. Wu, J. Jin, Y. Fang, and Y. Yu, ''User behavior retrieval for click-through rate prediction,'' in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2020, pp. 2347–2356, doi: 10.1145/3397271.3401440.

[263] Alibaba Cloud. *IJCAI-15 Contest*. Accessed: Mar. 25, 2022. [Online]. Available: https://tianchi.aliyun.com/dataset/dataDetail?dataId=42

[264] E. Cho, S. A. Myers, and J. Leskovec, ''Friendship and mobility: User movement in location-based social networks,'' in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, New York, NY, USA, 2011, pp. 1082–1090, doi: 10.1145/2020408.2020579.

[265] *User Behavior Data From Taobao for Recommendation*. Alibaba Cloud. Accessed: Mar. 25, 2022. [Online]. Available: https://tianchi.aliyun.com/dataset/dataDetail?dataId=649

[266] D. Yang, D. Zhang, V. W. Zheng, and Z. Yu, ''Modeling user activity preference by leveraging user spatial temporal characteristics in LBSNs,'' *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 45, no. 1, pp. 129–142, Jan. 2015, doi: 10.1109/TSMC.2014.2327053.

[267] D. Yang. *Foursquare Dataset*. Accessed: Mar. 25, 2022. [Online]. Available: https://sites.google.com/site/yangdingqi/home/foursquare-dataset

[268] F. Wu, Y. Qiao, J.-H. Chen, C. Wu, T. Qi, J. Lian, D. Liu, X. Xie, J. Gao, W. Wu, and M. Zhou, ''MIND: A large-scale dataset for news recommendation,'' in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, Jul. 2020, pp. 3597–3606, doi: 10.18653/v1/2020.acl-main.331.

[269] J. A. Gulla, L. Zhang, P. Liu, Ö. Özgöbek, and X. Su, ''The adressa dataset for news recommendation,'' in *Proc. Int. Conf. Web Intell.*, New York, NY, USA, Aug. 2017, pp. 1042–1048, doi: 10.1145/3106426.3109436.

[270] Z. Wang, J. Zhang, H. Xu, X. Chen, Y. Zhang, W. X. Zhao, and J.-R. Wen, ''Counterfactual data-augmented sequential recommendation,'' in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2021, pp. 347–356, doi: 10.1145/3404835.3462855.

[271] *Yelp Open Dataset: An All-Purpose Dataset for Learning*. Accessed: Mar. 25, 2022. [Online]. Available: https://www.yelp.com/dataset

[272] D. Ben-Shimon, A. Tsikinovsky, M. Friedmann, B. Shapira, L. Rokach, and J. Hoerle, ''RecSys challenge 2015 and the YOOCHOOSE dataset,'' in *Proc. 9th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2015, pp. 357–358, doi: 10.1145/2792838.2798723.

[273] M. Wan and J. McAuley, ''Item recommendation on monotonic behavior chains,'' in *Proc. 12th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2018, pp. 86–94, doi: 10.1145/3240323.3240369.

[274] A. Pathak, K. Gupta, and J. McAuley, ''Generating and personalizing bundle recommendations on steam,'' in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Aug. 2017, pp. 1073–1076, doi: 10.1145/3077136.3080724.

[275] B. M. Marlin and R. S. Zemel, ''Collaborative prediction and ranking with non-random missing data,'' in *Proc. 3rd ACM Conf. Recommender Syst. (RecSys)*, New York, NY, USA, 2009, pp. 5–12, doi: 10.1145/1639714.1639717.

[276] B. Marlin, ''Collaborative filtering: A machine learning perspective,'' M.S. thesis, Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, 2004. [Online]. Available: https://api.semanticscholar.org/CorpusID:11455170

[277] Y. Wang, L. Wang, Y. Li, D. He, T.-Y. Liu, and W. Chen, ''A theoretical analysis of NDCG type ranking measures,'' in *Proc. 26th Annu. Conf. Learn. Theory*, vol. 30, S. Shalev-Shwartz and I. Steinwart, Eds. Princeton, NJ, USA: PMLR, Jun. 2013, pp. 25–54. [Online]. Available: https://proceedings.mlr.press/v30/Wang13.html

[278] T. Calders and S. Jaroszewicz, ''Efficient AUC optimization for classification,'' in *Proc. 11th Eur. Conf. Princ. Data Mining Knowl. Discovery*, vol. 4702, Sep. 2007, pp. 42–53, doi: 10.1007/978-3-540-74976-9_8.

[279] D. Hand, ''Measuring classifier performance: A coherent alternative to the area under the ROC curve,'' *Mach. Learn.*, vol. 77, no. 1, pp. 103–123, 2009, doi: 10.1007/s10994-009-5119-5.

[280] W. Krichene and S. Rendle, ‘‘On sampled metrics for item recommendation,’’ in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 1748–1757, doi: 10.1145/3394486.3403226.

[281] S. Rendle, ‘‘Evaluation metrics for item recommendation under sampling,’’ 2019, *arXiv:1912.02263*.

[282] D. Li, R. Jin, J. Gao, and Z. Liu, ‘‘On sampling top-K recommendation evaluation,’’ in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 2114–2124, doi: 10.1145/3394486.3403262.

[283] V. W. Anelli, A. Bellogin, A. Ferrara, D. Malitesta, F. A. Merra, C. Pomo, F. M. Donini, and T. Di Noia, ‘‘Elliot: A comprehensive and rigorous framework for reproducible recommender systems evaluation,’’ in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2021, pp. 2405–2414, doi: 10.1145/3404835.3463245.

[284] R. Jin, D. Li, B. Mudrak, J. Gao, and Z. Liu, ‘‘On estimating recommendation evaluation metrics under sampling,’’ in *Proc. 35th AAAI Conf. Artif. Intell.*, May 2021, vol. 35, no. 5, pp. 4147–4154. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/16537

[285] H. Liu, J. Wen, L. Jing, and J. Yu, ‘‘Deep generative ranking for personalized recommendation,’’ in *Proc. 13th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2019, pp. 34–42, doi: 10.1145/3298689.3347012.

[286] W. Wang, F. Feng, X. He, X. Wang, and T.-S. Chua, ‘‘Deconfounded recommendation for alleviating bias amplification,’’ in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2021, pp. 1717–1725, doi: 10.1145/3447548.3467249.

[287] Q. Pi, W. Bian, G. Zhou, X. Zhu, and K. Gai, ‘‘Practice on long sequential user behavior modeling for click-through rate prediction,’’ in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Jul. 2019, pp. 2671–2679, doi: 10.1145/3292500.3330666.

[288] P. V. S. Avinesh, Y. Ren, C. M. Meyer, J. Chan, Z. Bao, and M. Sanderson, ‘‘J3R: Joint multi-task learning of ratings and review summaries for explainable recommendation,’’ in *Proc. Mach. Learn. Knowl. Discovery Databases*, Würzburg, Germany, U. Brefeld, E. Fromont, A. Hotho, A. Knobbe, M. Maathuis, and C. Robardet, Eds. Berlin, Germany: Springer, Sep. 2020, pp. 339–355. [Online]. Available: https://dl.acm.org/doi/abs/10.1007/978-3-030-46133-1_21

[289] W. Liu, J. Su, C. Chen, and X. Zheng, ‘‘Leveraging distribution alignment via stein path for cross-domain cold-start recommendation,’’ in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds. Red Hook, NY, USA: Curran Associates, 2021, pp. 19223–19234. [Online]. Available: https://proceedings.neurips.cc/paper/2021/file/a0443c8c8c3372d66 2e9173c18faaa2c-Paper.pdf

[290] T. Chen, H. Yin, Y. Zheng, Z. Huang, Y. Wang, and M. Wang, ‘‘Learning elastic embeddings for customizing on-device recommenders,’’ in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2021, pp. 138–147, doi: 10.1145/3447548.3467220.

[291] X. Li, W. Jiang, W. Chen, J. Wu, G. Wang, and K. Li, ‘‘Directional and explainable serendipity recommendation,’’ in *Proc. Web Conf.*, New York, NY, USA, Apr. 2020, pp. 122–132, doi: 10.1145/3366423.3380100.

[292] X. Chen, S. Li, H. Li, S. Jiang, Y. Qi, and L. Song, ‘‘Generative adversarial user model for reinforcement learning based recommendation system,’’ in *Proc. 36th Int. Conf. Mach. Learn.* (Proceedings of Machine Learning Research), vol. 97, K. Chaudhuri and R. Salakhutdinov, Eds. Long Beach, CA, USA: PMLR, Jun. 2019, pp. 1052–1061. [Online]. Available: https://proceedings.mlr.press/v97/chen19f.html

[293] K. Muhammad, Q. Wang, D. O'Reilly-Morgan, E. Tragos, B. Smyth, N. Hurley, J. Geraci, and A. Lawlor, ‘‘FedFast: Going beyond average for faster training of federated recommender systems,’’ in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 1234–1242, doi: 10.1145/3394486.3403176.

[294] Y. Zhou, J. Xu, J. Wu, Z. Taghavi, E. Korpeoglu, K. Achan, and J. He, ‘‘PURE: Positive-unlabeled recommendation with generative adversarial network,’’ in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2021, pp. 2409–2419, doi: 10.1145/3447548.3467234.

[295] W. Lei, G. Zhang, X. He, Y. Miao, X. Wang, L. Chen, and T.-S. Chua, ‘‘Interactive path reasoning on graph for conversational recommendation,’’ in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 2073–2083, doi: 10.1145/3394486.3403258.

[296] M. Li, S. Zhang, F. Zhu, W. Qian, L. Zang, J. Han, and S. Hu, ‘‘Symmetric metric learning with adaptive margin for recommendation,’’ in *Proc. 34th AAAI Conf. Artif. Intell.*, Feb. 2020, vol. 34, no. 4, pp. 4634–4641. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/5894

[297] R. Zhang, T. Yu, Y. Shen, H. Jin, and C. Chen, ‘‘Text-based interactive recommendation via constraint-augmented reinforcement learning,’’ in *Proc. Adv. Neural Inf. Process. Syst.*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Red Hook, NY, USA: Curran Associates, 2019, pp. 1–11. [Online]. Available: https://proceedings.neurips.cc/paper/2019/file/52130c418d4f02c74f74a5bc1f8020b2-Paper.pdf

[298] C. Wu, F. Wu, T. Qi, Y. Huang, and X. Xie, ‘‘Fastformer: Additive attention can be all you need,’’ 2021, *arXiv:2108.09084*.

[299] Z. Chai, Y. Li, and S. Zhu, ‘‘P-MOIA-RS: A multi-objective optimization and decision-making algorithm for recommendation systems,’’ *J. Ambient Intell. Humanized Comput.*, vol. 12, no. 1, pp. 443–454, Jan. 2021, doi: 10.1007/s12652-020-01997-x.

[300] G. Hiranandani, W. Vijitbenjaronk, S. Koyejo, and P. Jain, ‘‘Optimization and analysis of the pAp@k metric for recommender systems,’’ in *Proc. 37th Int. Conf. Mach. Learn.* (Proceedings of Machine Learning Research), vol. 119, H. D. III and A. Singh, Eds. PMLR, Jul. 2020, pp. 4260–4270. [Online]. Available: https://proceedings.mlr.press/v119/hiranandani20a.html

[301] E. Shulman and L. Wolf, ‘‘Meta decision trees for explainable recommendation systems,’’ in *Proc. AAAI/ACM Conf. AI, Ethics, Soc.*, New York, NY, USA, 2020, pp. 365–371, doi: 10.1145/3375627.3375876.

[302] O. Gouvert, T. Oberlin, and C. Févotte, ‘‘Recommendation from raw data with adaptive compound Poisson factorization,’’ in *Proc. Conf. Uncertainty Artif. Intell. (UAI)*, Tel Aviv, Israel, Jul. 2019, pp. 1–11. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02392075

[303] M. Tsang, D. Cheng, H. Liu, X. Feng, E. Zhou, and Y. Liu, ‘‘Feature interaction interpretability: A case for explaining ad-recommendation systems via neural interaction detection,’’ in *Proc. Int. Conf. Learn. Represent.*, 2020, pp. 1–19.

[304] M. M. Tanjim, C. Su, E. Benjamin, D. Hu, L. Hong, and J. McAuley, ‘‘Attentive sequential models of latent intent for next item recommendation,’’ in *Proc. Web Conf.*, New York, NY, USA, Apr. 2020, pp. 2528–2534, doi: 10.1145/3366423.3380002.

[305] S. Kang, J. Hwang, W. Kweon, and H. Yu, ‘‘Topology distillation for recommender system,’’ in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2021, pp. 829–839, doi: 10.1145/3447548.3467319.

[306] S. Zhang, H. Chen, X. Ming, L. Cui, H. Yin, and G. Xu, ‘‘Where are we in embedding spaces?’’ in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2021, pp. 2223–2231, doi: 10.1145/3447548.3467421.

[307] S. Feng, L. V. Tran, G. Cong, L. Chen, J. Li, and F. Li, ‘‘HME: A hyperbolic metric embedding approach for next-POI recommendation,’’ in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2020, pp. 1429–1438, doi: 10.1145/3397271.3401049.

[308] F. Mi, X. Lin, and B. Faltings, ‘‘ADER: Adaptively distilled exemplar replay towards continual learning for session-based recommendation,’’ in *Proc. 14th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2020, pp. 408–413, doi: 10.1145/3383313.3412218.

[309] M. Mladenov, E. Creager, O. Ben-Porat, K. Swersky, R. Zemel, and C. Boutilier, ‘‘Optimizing long-term social welfare in recommender systems: A constrained matching approach,’’ in *Proc. 37th Int. Conf. Mach. Learn.* (Proceedings of Machine Learning Research), vol. 119, H. D. III and A. Singh, Eds. PMLR, Jul. 2020, pp. 6987–6998. [Online]. Available: https://proceedings.mlr.press/v119/mladenov20a.html

[310] Z. Jiang, C. Chi, and Y. Zhan, ‘‘Let knowledge make recommendations for you,’’ *IEEE Access*, vol. 9, pp. 118194–118204, 2021, doi: 10.1109/ACCESS.2021.3106914.

[311] Z. Meng, R. McCreadie, C. Macdonald, and I. Ounis, ‘‘Exploring data splitting strategies for the evaluation of recommendation models,’’ in *Proc. 14th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2020, pp. 681–686, doi: 10.1145/3383313.3418479.

[312] A. Maksai, F. Garcin, and B. Faltings, ‘‘Predicting online performance of news recommender systems through richer evaluation metrics,’’ in *Proc. 9th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2015, pp. 179–186, doi: 10.1145/2792838.2800184.

[313] M. Rossetti, F. Stella, and M. Zanker, ''Contrasting offline and online results when evaluating recommendation algorithms,'' in *Proc. 10th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2016, pp. 31–34, doi: 10.1145/2959100.2959176.

[314] Z. Sun, D. Yu, H. Fang, J. Yang, X. Qu, J. Zhang, and C. Geng, ''Are we evaluating rigorously? Benchmarking recommendation for reproducible evaluation and fair comparison,'' in *Proc. 14th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2020, pp. 23–32, doi: 10.1145/3383313.3412489.

[315] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman, ''Superglue: A stickier benchmark for general-purpose language understanding systems,'' in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2019. [Online]. Available: https://proceedings.neurips.cc/paper/2019/file/4496bf24afe7fab6f046bf4923da8de6-Paper.pdf

[316] W. X. Zhao, S. Mu, Y. Hou, Z. Lin, Y. Chen, X. Pan, K. Li, Y. Lu, H. Wang, C. Tian, Y. Min, Z. Feng, X. Fan, X. Chen, P. Wang, W. Ji, Y. Li, X. Wang, and J.-R. Wen, ''RecBole: Towards a unified, comprehensive and efficient framework for recommendation algorithms,'' in *Proc. 30th ACM Int. Conf. Inf. Knowl. Manage.*, New York, NY, USA, Oct. 2021, pp. 4653–4664, doi: 10.1145/3459637.3482016.

[317] J. Chen, H. Dong, X. Wang, F. Feng, M. Wang, and X. He, ''Bias and debias in recommender system: A survey and future directions,'' 2020, *arXiv:2010.03240*.

[318] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, ''A survey on bias and fairness in machine learning,'' *ACM Comput. Surv.*, vol. 54, no. 6, pp. 1–35, Jul. 2021, doi: 10.1145/3457607.

[319] Y. Huang, W. Wang, L. Zhang, and R. Xu, ''Sliding spectrum decomposition for diversified recommendation,'' in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2021, pp. 3041–3049, doi: 10.1145/3447548.3467108.

[320] A. Jain, P. K. Singh, and J. Dhar, ''Multi-objective item evaluation for diverse as well as novel item recommendations,'' *Expert Syst. Appl.*, vol. 139, Jan. 2020, Art. no. 112857. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417419305597

[321] M. Abdool, M. Haldar, P. Ramanathan, T. Sax, L. Zhang, A. Manaswala, L. Yang, B. Turnbull, Q. Zhang, and T. Legrand, ''Managing diversity in airbnb search,'' in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 2952–2960, doi: 10.1145/3394486.3403345.

[322] J. Möller, D. Trilling, N. Helberger, and B. van Es, ''Do not blame it on the algorithm: An empirical assessment of multiple recommender systems and their impact on content diversity,'' *Inf., Commun. Soc.*, vol. 21, no. 7, pp. 959–977, Jul. 2018, doi: 10.1080/1369118X.2018.1444076.

[323] D. Kotkov, J. Veijalainen, and S. Wang, ''How does serendipity affect diversity in recommender systems? A serendipity-oriented greedy algorithm,'' *Computing*, vol. 102, no. 2, pp. 393–411, Feb. 2020, doi: 10.1007/s00607-018-0687-5.

[324] A. Vultureanu-Albişi and C. Bădică, ''A survey on effects of adding explanations to recommender systems,'' *Concurrency Comput., Pract. Exper.*, vol. 34, no. 20, p. e6834, Jan. 2022, doi: 10.1002/cpe.6834.

[325] O. Barkan, Y. Fuchs, A. Caciularu, and N. Koenigstein, ''Explainable recommendations via attentive multi-persona collaborative filtering,'' in *Proc. 14th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2020, pp. 468–473, doi: 10.1145/3383313.3412226.

[326] T. Chen, H. Yin, G. Ye, Z. Huang, Y. Wang, and M. Wang, ''Try this instead: Personalized and interpretable substitute recommendation,'' in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2020, pp. 891–900, doi: 10.1145/3397271.3401042.

[327] F. K. Khan, A. Flanagan, K. E. Tan, Z. Alamgir, and M. Ammad-ud-Din, ''A payload optimization method for federated recommender systems,'' in *Proc. 15th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2021, pp. 432–442, doi: 10.1145/3460231.3474257.

[328] D. Chai, L. Wang, K. Chen, and Q. Yang, ''Secure federated matrix factorization,'' *IEEE Intell. Syst.*, vol. 36, no. 5, pp. 11–20, Sep./Oct. 2021, doi: 10.1109/MIS.2020.3014880.

[329] J. Han, Y. Ma, Q. Mei, and X. Liu, ''DeepRec: On-device deep learning for privacy-preserving sequential recommendation in mobile commerce,'' in *Proc. Web Conf.*, New York, NY, USA, Apr. 2021, pp. 900–911, doi: 10.1145/3442381.3449942.

[330] C. Wu, F. Wu, T. Qi, Y. Huang, and X. Xie, ''FedAttack: Effective and covert poisoning attack on federated recommendation via hard sampling,'' 2022, *arXiv:2202.04975*.

[331] E. Xi, S. Bing, and Y. Jin, ''Capsule network performance on complex data,'' 2017, *arXiv:1712.03480*.

[332] G. E. Hinton, S. Sabour, and N. Frosst, ''Matrix capsules with EM routing,'' in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–15. [Online]. Available: https://openreview.net/forum?id=HJWLfGWRb

MATTEO MARCUZZO received the bachelor's degree in computer games technology from the University of the West of Scotland, Paisley, U.K., and the master's degree in computer science from the University of Padua, Italy. He is an Associate Researcher with Digital Strategy Innovation. His research interests include natural language processing, representation learning, interpretable AI, and high performance computing.

ALESSANDRO ZANGARI received the master's degree in computer science from the University of Padua, in 2020. He is an Associate Researcher with Digital Strategy Innovation and a Machine Learning Engineer with the Ca' Foscari University of Venice. His current research interests include deep learning applications for natural language processing algorithms, recommendation systems, computer vision, and interpretability of AI.

ANDREA ALBARELLI is currently a Professor for the multidisciplinary master program in data analytics for business and society with the Ca' Foscari University of Venice, where he is responsible for the artificial intelligence teaching. He is a Researcher in the field of artificial intelligence, with a special focus on the design of disruptive data-driven methodologies to be applied on real-world scenarios. To this end, he works in close collaboration with companies willing to undertake a radical digital transformation process. His approach is end-to-end, spanning from the co-design of digital-first business models to the scientific advising needed to fulfill their methodological and technological infrastructure. He has led several technological transfer projects, resulting in research papers published in top international journals and presented in key engineering conferences. He received several scientific and industrial recognitions, including the NVIDIA Best Paper Award, for his research on 3D data processing; and innovation grants from companies like Electrolux and TIM, for the technical contributions.

ANDREA GASPARETTO received the M.Sc. degree in computer science from the University of Venice, Italy, in 2012, and the Ph.D. degree in computer science from the Ca' Foscari University of Venice. Since 2016, he has been a Researcher and a Teaching Assistant with the Management Department, Ca' Foscari University of Venice. His research interests include in the artificial intelligence field, and more precisely in computer vision, shape analysis, retrieval and classification, and non-vectorial data models.