User Modeling and User-Adapted Interaction (2022) 32:685–746
https://doi.org/10.1007/s11257-021-09311-w

Rank-sensitive proportional aggregations in dynamic recommendation scenarios

Stepan Balcar[1] · **Vit Skrhak**[1] · **Ladislav Peska**[1]

Received: 6 March 2021 / Accepted in revised form: 5 November 2021 / Published online: 1 January 2022

Abstract

In this paper, we focus on the problem of rank-sensitive proportionality preservation when aggregating outputs of multiple recommender systems in dynamic recommendation scenarios. We believe that individual recommenders may provide complementary views on the user's preferences or needs, and therefore, their proportional (i.e. unbiased) aggregation may be beneficial for the long-term user satisfaction. We propose an aggregation framework (FuzzDA) based on a modified D'Hondt's algorithm (DA) for proportional mandates allocation. Specifically, we adjusted DA to register fuzzy membership of items and modified the selection procedure to balance both relevance and proportionality criteria. Furthermore, we propose several iterative votes assignment strategies and negative implicit feedback incorporation strategies to make FuzzDA framework applicable in dynamic recommendation scenarios. Overall, the framework should provide benefits w.r.t. long-term novelty of recommendations, diversity of recommended items as well as overall relevance. We evaluated FuzzDA framework thoroughly both in offline simulations and in online A/B testing. Framework variants outperformed baselines w.r.t. click-through rate (CTR) in most of the evaluated scenarios. Some variants of FuzzDA also provided the best or close-to-best iterative novelty (while maintaining very high CTR). While the impact of the framework variants on user-wise diversity was not so extensive, the trade-off between CTR and diversity seems reasonable.

The work on this paper has been supported by Czech Science Foundation (GAČR) Project 19-22071Y, Charles University Grant SVV-260588 and Charles University Grant Agency (GA UK) Project Number 1026120. Computational resources were partially supplied by the project "e-Infrastruktura CZ" (e-INFRA CZ ID:90140) supported by the Ministry of Education, Youth and Sports of the Czech Republic.

✉ Ladislav Peska
ladislav.peska@matfyz.cuni.cz

Stepan Balcar
stepan.balcar@matfyz.cuni.cz

[1] Department of Software Engineering, Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic

Keywords Proportionality preservation · Offline simulations · Unbiased aggregation · Recommender systems

1 Introduction

In recent years, we experienced a gradual shift in the perceived role of recommender systems (RSs). Until recently, RS was mostly viewed as algorithms that learn users' preferences (from their past feedback) and subsequently recommend items that fit to those preferences. We denote this as a *static* view on recommender systems. Nowadays, many researchers start to perceive recommender systems as a part of a dynamically changing environment. In contrast to the static view on RS, main premises of *dynamic* RS are that:

- User preferences are dynamic, they can develop over time or be influenced by current needs, contexts or available choices.
- Instead of a static feedback dataset, a stream of feedback events pours into the system. The most relevant information for users' preference estimation is often carried by the most recent events.
- Novel users and items continuously emerge in the system. Requests for recommendation may come at any moment, and sometimes, no novel (positive) feedback is received between two consecutive requests.

The dynamic setting for RS naturally brings a brand new set of challenges and problems. One of the main challenges is the ability of RS to quickly adapt to the gradual or sudden changes in user's preferences. Training of traditional RS such as matrix factorization would require a prohibitive amount of time. Therefore, various incremental learning approaches, e.g., Vinagre et al. (2014), or session-aware approaches, e.g., (Ludewig and Jannach 2018; Quadrana et al. 2018), were proposed to responsively adapt to most recent user needs. Considerable attention also received the problem of reliable offline evaluation under the constraints of dynamic RS (Ludewig and Jannach 2018).

A different set of challenges comes from the repetitions in user behavior. In *static* scenarios, items with existing user feedback are not recommended, because these cannot be present in the test set. Also, one list of recommendations is usually generated and evaluated for each test set user. Nonetheless, in real world, it is quite common that users re-visit or consume items several times (e.g., repeatedly playing a favorite song). Therefore, in some situations it may be relevant to repeatedly recommend already visited items (Lerche et al. 2016).

Furthermore, depending on the structure of the web, users may spend a significant portion of their visit outside pages dedicated to specific items (e.g., on homepage, various category pages, etc.). It is natural to recommend items on these pages as well. However, if common types of feedback are assumed (item ratings, item visits, item consumption), it may often happen that several consecutive requests for recommendation must be fulfilled without any new feedback from the user. The question is whether repeated recommendation of the same items would be of any use to the

user, or, e.g., we should consider the previous items' exposure as supporting (weakly negative) feedback.

The final set of challenges comes from the capability of dynamic RS to affect user's behavior. In *static* scenario, RS is viewed as simple responders to the (predefined) user preferences. Nonetheless, by (not) displaying certain content, user preferences (or at least our view of preferences inferred from the received feedback) may change gradually. This effect is more prevalent in systems, where the majority of content is personalized such as Facebook or Netflix, but it may appear on other sites as well. Once embracing that RS is partially responsible for the development of users' views and preferences, several concerns may be raised. One direction considers algorithmic fairness issues (i.e., is the RS biased against some groups of users or items? Mansoury 2021; Elahi et al. 2021). The other direction considers the long-term effects of RS exposure (Nguyen et al. 2014; Symeonidis et al. 2019; Ge et al. 2020; Lunardi et al. 2020; Sinha et al. 2016), which is more relevant for our work.

Let us, for instance, mention the feedback loops problem (Sinha et al. 2016). The problem can be formulated as follows: Suppose to have a system, where most of its content is personalized (recommended) according to (supposed) user preferences. Capability of users to find additional content independently is limited. Then, due to the lack of other choices, users tend to consume some of the recommended content no matter whether they are fully satisfied with it. The consumption behavior (considered as a positive feedback) in turn reinforces the user preference model, forming a feedback loop. Over time, RS become overfitted to the reinforced preference model, unable to deliver sufficiently novel and surprising results. Even though the recommendations might had been interesting and relevant at first, the lack of novelty and diversity eventually deteriorates user's experience. Related problems of filter bubbles (Nguyen et al. 2014), echo chambers (Ge et al. 2020) or popularity bias (Abdollahpouri and Burke 2019; Abdollahpouri et al. 2020) also continue to receive considerable attention from the research community.

1.1 Motivation

Various approaches were proposed to mitigate the problem of deteriorating long-term performance of RS. These approaches include maintenance of sufficient catalogue coverage, randomization of the recommendation procedure, considering diversity, novelty or serendipity of recommended items, calibration w.r.t. various axes of user preferences or some notion of fairness in items representation (Kotkov et al. 2016; Bertani et al. 2020; Kaminskas and Bridge 2016; Lathia et al. 2010; Steck 2018; Lathia et al. 2010).

A common denominator for many of these approaches is that they (directly or indirectly) utilize some concept of *proportionality*.

For instance, suppose that we are aiming on enhancing the diversity of provided recommendations. Such task can be up to some extent re-formulated as increasing the *proportionality of representation* for certain sub-areas in a metric space induced by the notion of items similarity (see, e.g., Dang and Croft (2012) or the *extended calibration* concept in Steck (2018)). Similar re-formulation can be used for variants

of catalogue coverage metrics as well if one-hot representation of items is considered. Studies focused on the popularity bias and calibration phenomena often aim to minimize some form of disproportionality between the proposed recommendations and user profiles (Abdollahpouri et al. 2020; Steck 2018). Also, methods dealing with the filter bubbles phenomenon mostly focus on some form of similarity relaxation among recommended items (Lunardi et al. 2020), or introduce additional optimization axis less correlated with the estimated relevance of items (Symeonidis et al. 2019). As a result, they indirectly manipulate with the proportionality of exploitation- and exploration-oriented recommendations.

Based on the previous paragraph, we can see that the proportionality of representation concept is (directly or indirectly) linked with many extensively researched phenomena in RS. However, to the best of our knowledge, the proportionality concept itself did not receive much attention by the recommender systems community yet.

The main contribution of this paper is a proposal of FuzzDA framework focusing on the proportionality preservation problem in dynamic recommending ecosystems. In this paper, we evaluate one possible use-case for the framework, namely proportional aggregation of multiple base recommending algorithms. However, let us already now highlight that the framework or its parts may be useful in other areas of RS research as well, e.g., group recommendation problem (Kaya et al. 2020), proportional representation of items' sub-components (Starychfojtu and Peska 2020) or calibrating various quality criteria (novelty, diversity, relevance) of recommendations.

The selected use-case is based on the assumption that individual recommending algorithms can be considered as different (latent) views on user's interests or preferences. Supposedly, the more the base recommending algorithms internally differ (e.g., collaborative vs. content-based vs. session-based), the more diverse viewpoints they take. Suppose further that as long as the hypotheses behind these algorithms are sound and they are trained on up-to-date data, their recommendations should be considerably different from one another and at least partially relevant for the user. Therefore, by giving users the access to multiple base recommenders, we can broaden their available choices and hopefully contribute towards both increasing perceived relevance of the system and decreasing the negative effects of feedback loop or filter bubble phenomenons. Figure 1 provides a schematic display of the desired effect.

There are several challenges hidden behind this illustrative example. First, the proper selection of the RS portfolio is crucial. Substantially inferior RS could hurt the overall performance of the system, while selecting too similar RS would not provide the required diversity. However, even if the selection was made properly, the performance of individual RS may vary greatly either in general or for particular users or contexts. Therefore, the proposed approach should be capable of assessing the performance of base RS and reflect it during the selection procedure (i.e., do not display too many items from inferior recommenders). Furthermore, if a certain item is agreed on by multiple base RS, it can be understood as an additional evidence of its relevance, and this should be also reflected during the selection procedure. Finally, many recommending algorithms cannot adapt quickly enough to the most recent user context and feedback (especially inactions on the recently recommended items, i.e., implicit negative feedback). Therefore, capability to react immediately on the negative feed-

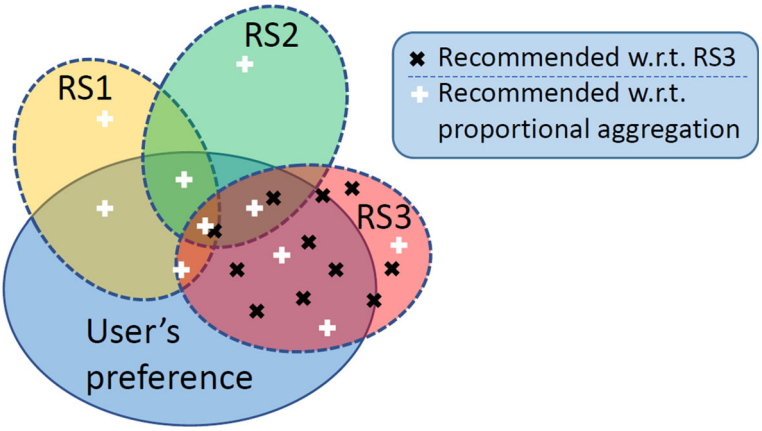

Fig. 1 Schematic view of the desired effect of proportionality-preserving aggregation of RS. Note that items recommended w.r.t. single best recommender (RS3) cover only a small section of user's preferences in contrast to the aggregated approach, while the relevance of both approaches remains similar. Aggregated approach aims to respect the relevance of individual recommenders (denoted by its intersection with user's preference) and prefers items on which more base RS agreed

back as well as current user's context should be also incorporated into the proposed approach.

1.2 FuzzDA framework

The proposed FuzzDA framework comprises of three main components. Votes assignment strategies observe the recent performance of base RS and derive their estimated relevance (i.e., votes) for the current context. Proportionality-preserving aggregator utilizes these votes together with the recommendations of base RS and returns the final list of recommendations. The final list should maintain the proportionality between assigned votes and recommended items,[1] but also represent those items that received best ratings overall. Thus, defined proportionality preservation can be considered as one possible approximation of fairness in recommender systems.[2] Finally, the negative feedback incorporation strategies focus on the recent feedback from the user and limit the relevance of those items that were recently ignored by the user.[3]

The proposed FuzzDA framework aims to address several of the previously described dynamic RS challenges, namely maintaining long-term usability of rec-

[1] We specifically consider the term rank-sensitive proportionality, which will be formally defined in Sect. 3.2.2. Intuitively, we want that each prefix of the results follows the votes distribution as closely as possible.

[2] Specifically, we utilize a component-based notion of fairness similarly to Steck (2018). Instead of evaluating fairness w.r.t. individuals or groups of individuals, we consider the fairness of representing various views on the user's needs and preferences with the goal to reflect them according to their corresponding proportions.

[3] To be more specific, we consider both the temporal distance of such events (i.e., what is the chance that the user has changed his/her mind in the meantime) and position of the item within the results (i.e., what is the chance that the user simply overlooked the recommendation).

ommendations, capability to adjust recommendations to the recent user feedback and context and ability to properly handle repeated requests for recommendations. We also propose innovative offline simulations to evaluate RS in a similar fashion to being deployed online in dynamic environments.

Some portions of FuzzDA framework were introduced in our previous work (Peška and Balcar 2019; Balcar and Peska 2020). In contrast to these publications, we propose several new alternatives to the individual components of FuzzDA framework and provide much more thorough evaluation w.r.t. multiple beyond-accuracy metrics.

In summary, the main contributions of this paper are:

- We propose FuzzDA framework for proportionality-preserving RS aggregation. The framework is modular, and we propose several variants of aggregation algorithm, votes assignment strategies and models of implicit negative feedback incorporation, which make the system suitable for usage under the constraints of dynamic RS.
- We propose innovative offline simulations to evaluate the performance of FuzzDA framework. Specifically, we focus on a not-yet explored phenomenon of repeated requests for recommendations.
- We evaluate variants of FuzzDA framework w.r.t. several metrics including various notions of novelty and diversity, popularity bias and estimated click-through rate (CTR) with favorable results on two datasets. Notably, variants of FuzzDA were able to maintain uncompromised CTR performance while increasing the iterative novelty of recommended items. FuzzDA also improved the relevance–diversity trade-off in many evaluated scenarios.
- We also performed an online A/B test on a small e-commerce enterprise, where a variant of FuzzDA received the highest CTR and diversity scores.

The rest of the paper is organized as follows: In Sect. 2, we provide an overview of the related work as well as some background knowledge utilized in the rest of the paper. The main content of the paper (i.e., FuzzyDA framework and its variants) is presented in Sect. 3. Sections 4 and 5 describe the evaluation protocol and results, respectively, and finally the conclusions and future work are outlined in Sect. 6.

2 Background and related work

In this section, we would like to discuss several areas related to our work and describe the background on which we build the FuzzyDA framework.

We start with an overview of aggregation methods in recommender systems (Sect. 2.1) with a specific focus on list-wise aggregation techniques. In Sect. 2.2, we provide an introduction into proportional mandates allocation strategies, especially D'Hondt's algorithm, which was a direct inspiration for our work. We follow with a more generic discussion on the concepts of proportionality and calibration in recommender systems (Sect. 2.3). Our work is framed into the dynamic recommendation problem. We discuss its properties and impact on the proposed framework in Sect. 2.4. We specifically focus on offline simulations of dynamic RS (Sect. 2.4.1) and beyond-accuracy metrics in the context of dynamic RS (Sect. 2.4.2). Finally, in Sect. 2.5 we

discuss assumptions made about the behavior of users and their implications on the proposed framework and offline evaluation procedures.

2.1 Aggregations in recommender systems

There is a considerable track of research that takes into account some sort of aggregation in various domains of recommender system. In an overview paper by Beliakov et al. (2015), authors describe various stages of the recommendation process, where aggregation may be applicable. Some of the variants are aggregation of features in content-based approaches, neighborhood formation in K nearest-neighbors algorithms (KNN) or weighted hybrid systems that aggregates output of multiple recommender systems. Our approach belongs to the last group.

The multiple RS aggregation problem may be formalized as follows: Suppose we have several base recommenders $R_1, \ldots, R_m$ and an aggregator A. Each recommender R_i provides an ordered list of recommended items $O_i = [o_{i,1}, \ldots, o_{i,n}]$; $o_{i,j} \in \mathcal{O}$, usually accompanied with scores assigned for each recommended object $S_i = [s_{i,1}, \ldots, s_{i,n}]$. Furthermore, each recommender has some assigned weight w_i, which reflects its performance.[4] The task of the aggregator is to provide the overall recommendation based on these inputs $A : \{(O_1, S_1, w_1), \ldots, (O_m, S_m, w_m)\} \longrightarrow [o_1, \ldots, o_k]$; $o_j \in \mathcal{O}$. Typically, the size of final recommendations list is smaller than the output of base recommenders ($k < n$).

Beliakov et al. (2015) further discuss properties of several aggregation functions. Nonetheless, all mentioned aggregations are *item-wise* only, i.e., they aggregate results of each item separately, irrespective of other recommended items. Several other works from various domains of RS focus on item-wise aggregations as well. Some examples are reciprocal RS (Neve and Palomares 2019), conference review assignments (Nguyen et al. 2018) or neighbors aggregation in KNN (Garcin et al. 2009). Some authors also focus on learning the correct aggregation, e.g., via genetic programming (Oliveira et al. 2018). Nature-inspired optimizations are nevertheless difficult to utilize in an online environment, and even in this case, the learned function is still item-wise.

One of the well known disadvantages of item-wise aggregations is the risk of a systematic bias against some of the aggregated parties (i.e., recommenders in our case) (Kaya et al. 2020; Serbos et al. 2017; Xiao et al. 2017). For instance, consider the following example with three recommender systems R_1, R_2 and R_3 that propose items as shown in Table 1. For simplicity, assume that all recommenders have the same weight and aggregation function is the (weighted) average of per-item scores. Now, if top-3 objects are recommended to the user, those will be o_1, o_3 and o_2. Specifically, not a single object preferred by R_3 would be displayed to the user.

Let us consider another situation with two recommenders R_4 and R_5. This time, both recommenders have assigned weights $w_4 = 0.7$ and $w_5 = 0.3$. Now, although the recommender R_5 should have certain importance (it has nonzero weight), it has absolutely no effect on the final ranking if weighted AVG of item's scores is used. In fact, the final ranking is exactly opposite to the ranking induced by R_5. Obviously, these examples are over-simplistic, but similar issues may be expected in real-world

[4] Note that later in the paper, we also use the term "votes" instead of "weight" with the same meaning.

Table 1 Example ratings and results of item-wise weighted average aggregations

Object	R_1	R_2	R_3	AVG	R_4; $w_4 = 0.7$	R_5; $w_5 = 0.3$	WAVG
o_1	0.9	0.8	0.0	0.567	0.9	0.0	0.63
o_2	0.6	0.5	0.0	0.367	0.8	0.2	0.62
o_3	0.5	0.9	0.0	0.467	0.6	0.3	0.51
o_4	0.0	0.1	0.9	0.333	0.3	0.8	0.45
o_5	0.1	0.1	0.8	0.333	0.1	0.9	0.34
o_6	0.2	0.0	0.7	0.300	0.0	0.9	0.27

situations as well. Also, similar adversary examples may be constructed for other item-wise aggregation functions.

The main difference between our approach and those described above is the fact that instead of relying on item-wise aggregations, we propose a list-wise approach to provide the aggregated ranking of items. (The proposed algorithm is denoted as EP-FuzzDA.) EP-FuzzDA aims to maximize the exactly proportional fraction of item's scores per base recommender and therefore simultaneously optimize for both relevance and proportionality w.r.t. base recommenders.[5] Considering EP-FuzzDA and the first example, the aggregated ranking of items would be $o_1, o_4, o_3, o_5, o_2, o_6$. For the second example, the aggregated ranking of items would be $o_2, o_3, o_4, o_1, o_5, o_6$. Note that in both cases, all considered recommenders have some high-rated items in the list of top-3 recommendations.

2.1.1 Aggregations in group recommender systems

There is one sub-area of RS where several approaches focused on aggregation strategies beyond simple item-wise approaches: group recommender systems (Masthoff 2011; Felfernig et al. 2018). The main paradigm of group RS is that instead of focusing on the preferences of an individual, recommendations should be provided according to the preference of several users. Group recommendation strategies usually operate on top of classical recommending algorithms that supplies them with preferences for individual users (usually in the form of user-item relevance estimations). Group recommendation strategies then process individual preferences of group members and output a list of recommended items for the group. To utilize group recommending strategies in our use-case, one can simply substitute group members (and their preferences) for different recommending algorithms providing recommendations for the same user.

Several group RS focused on the list-wise aggregations (Xiao et al. 2017; Sacharidis 2019; Serbos et al. 2017; Kaya et al. 2020), where some concept of fairness among group members is introduced and maintained in the final list of recommendations. An early example is FAI algorithm (Masthoff 2011), which regularly switches between users and selects the best remaining item of the current user. In this way, FAI indirectly balances the per-user volume of relevant items. More recent approaches usually

[5] Details of the algorithm will be given in Sect. 3.2.2.

explicitly define some per-user utility function, e.g., mean predicted relevance for user (Xiao et al. 2017) or similarity of aggregated ranking to the per-user ranking (Sacharidis 2019; Serbos et al. 2017). This utility function then serves as a base for some fairness metrics, such as Least Misery (minimal utility among users) or min–max ratio (ratio between minimal and maximal utility). Finally, some multicriterial optimization solver simultaneously optimizing for selected relevance and fairness metrics is employed to construct the list of recommendations.

One further enhancement to the above-described methodology was proposed by Kaya et al. (2020). Instead of focusing on the overall fairness of the whole list, they introduced a *rank-sensitive* fairness property of the recommendations and proposed a greedy optimization algorithm (GFAR) that maintains this property. In contrast to the list-wise fairness, *rank-sensitive* fairness considers ordering of items. Specifically, having a list of top-K recommended items, we denote it as fair in the rank-sensitive way if each prefix of the list maximizes a selected list-wise fairness metric. In another words, the first selected item should as much as possible balance the interests of all group members, so should the first two items, etc. Kaya et al. defined this fairness through the probability that at least one item is found relevant by the user. The proposed greedy algorithm in each step selects an item with maximal marginal gain to the sum of these probabilities.

Similarly as Kaya et al., our proposal maintains the rank-sensitive fairness. However, both approaches considerably differ in the definition of fairness and subsequently the selected items. GFAR defines relevance probability via items' ranks and employs rather severe penalty for items on lower ranks (which, however, may be still acceptable for the user). On the other hand, EP-FuzzDA focuses on the sum of predicted relevance scores obtained from each user and iteratively maximizes the proportional part of them. As such, EP-FuzzDA tends to select items with higher overall agreement, rather than (close to) best items for individual users as GFAR. Moreover, GFAR (similarly as other group RS) does not provide a natural way to define importance of individual users (i.e. individual RS in our use-case), which prevents them from being directly applicable on our problem.[6]

2.2 Proportional mandates allocation in public elections

One of the main inspirations for our work are mandates allocation algorithms utilized in public elections. These methods aim on mapping the volume of per-party votes into the volume of per-party mandates while maintaining the proportionality of constructed representation. Using a simple projection of parties to individual recommenders and candidates to recommended items, proportional mandates allocation algorithms may be applied to solve the RS aggregation problem. In this work, we specifically focus on D'Hondt's algorithm (DA, D'Hondt 1882; Medzihorsky 2019) and its extensions.

[6] To be more precise, GFAR, FAI and other group RS algorithms implicitly consider fixed uniform importance of all actors. Therefore, it is theoretically possible to apply them on our use-case, but any information about the quality of individual RS would be lost. We decided to incorporate FAI algorithm as one of the baseline aggregators to show the extent to which is this loss significant.

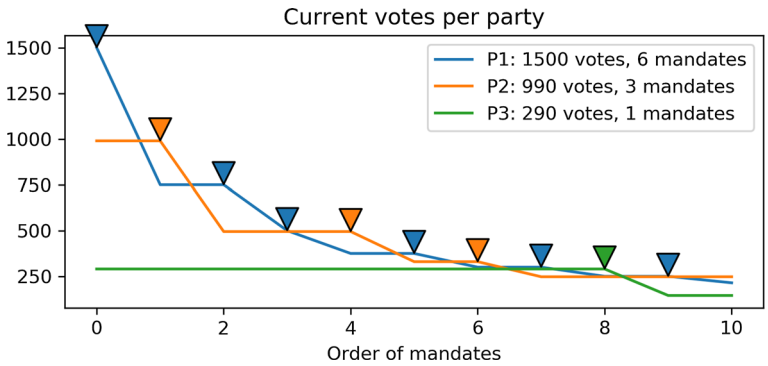

Fig. 2 An example of D'Hondt's candidate selection process. The example considers three parties (P1, P2 and P3), each assigned with different volume of votes. Lines denote the current values of per-party accountable votes; colored markers denote which party receives the next mandate

DA belongs to the class of greedy selection algorithms. On input, it expects a set of parties $p \in \mathcal{P}$, per-party votes v_p and an ordered list of per-party candidates $C_p = [c_{p,1}, \ldots, c_{p,k}]$. At each step, DA performs the following operations:

1. Select party p_{best} with the highest volume of accountable votes, $best = \text{argmax}_{\forall i}\, a_i$.
2. Append the first not-yet-selected candidate of the currently best party into the list of representatives.
3. Decrease the volume of accountable votes of party p_{best} as follows: $a_{best} = v_{best}/(m_p + 1)$, where m_p is the volume of mandates received by party p.

Figure 2 depicts an example of the D'Hondt's candidate selection process. One can observe that candidates of individual parties are interleaved approximately according to the share of per-party votes. The sequence of selections can be understood as a rank-sensitive proportional ordering of candidates w.r.t. parties popularity.

Although theoretical guarantees of proportionality were not discussed in the original DA description, there were several attempts to provide them subsequently. Sainte-Laguë (1910) shows that DA always selects a candidate that minimizes the largest per-party advantage ratio $adv_p = m_p/v_p$. In another words, DA iteratively minimizes the level of over-representation of the most over-represented party. Furthermore, Medzihorsky (2019) showed that if this criterion is considered as the metric of proportionality, DA separates the votes into a fraction exactly proportionally represented by the assigned mandates and some nonnegative residual votes (i.e., per-party under-representation). Subsequently, Medzihorsky shows that DA iteratively maximizes the volume of exactly proportionally represented votes. This observation (although we did not follow the same proportionality metric) was the main inspiration behind the proposed EP-FuzzDA aggregation algorithm.

Nonetheless, DA has several drawbacks which renders it less suitable for the recommendation aggregation task in online settings.

1. DA does not account for candidates shared among parties. However, in the context of recommender systems, we may expect that candidates proposed by multiple base RS should gain some relevance bonus.
2. Given the votes for individual parties, DA is fully deterministic. This may be problematic in the repeated recommendations settings (Balcar and Peska 2020), which is prevalent in real-world scenarios.
3. DA expects that the volume of votes per party is supplied externally. In the context of recommender systems, this brings the questions of who provides these votes and on what conditions should they depend. Considering online recommendation scenarios, we assume that the votes should be updated iteratively based on the recent performance of individual base RS and possibly also based on the gradually developing user profile.

In our preliminary works (Peška and Balcar 2019; Balcar and Peska 2020), we touched some of these challenges. We proposed a fuzzy extension to DA (FuzzDA) which considers (partial) relevance of candidates to multiple actors and modifies the selection procedure (Peška and Balcar 2019). Furthermore, in Balcar and Peska (2020) we proposed a stochastic version of FuzzDA and incorporation of recent implicit negative feedback to cope with the repeated recommendations problem. In both cases, the volume of per-recommender votes was not personalized, i.e., base RS started with uniform votes, and these were updated incrementally based on RS's recent performance (w.r.t. all users).

In contrast to our previous work, we re-visited not only the aggregation algorithm itself (see Sect. 3.2), but also the votes assignment strategy (Sect. 3.3) as well as the negative feedback incorporation (Sect. 3.4). We propose and evaluate several modifications to these components.

2.3 Calibration in recommender systems

The concept of calibrated recommendations was recently introduced by Steck (2018) with connotations to the fairness of RS. Both calibration and proportionality preservation are highly related concepts, which makes it interesting to compare the work of Steck (2018) with our own. Steck considers recommendations to be calibrated if various interests of the user are reflected in the list of recommended items w.r.t. their appropriate proportions. Specifically, the approach is demonstrated on MovieLens dataset, where the dimensions of user's interests correspond to the movie genres. The level of user's interest in particular genre is determined as a fraction of movies from user's profile that are of that genre. In order to measure the level of (mis)calibration, Steck utilize Kullback–Leibler (KL) divergence between two distributions: $p(g|u)$ genre's distribution for movies in the user profile u and $q(g|u)$ genre's distribution for recommended movies. Steck also proposed an analogy to maximal marginal relevance algorithm (MMR, Carbonell and Goldstein (1998)) to jointly optimize for relevance and calibration of the results.

Several papers focused on an interplay between calibration and other related concepts (Lin et al. 2020; Kaya and Bridge 2019; Abdollahpouri et al. 2020). Kaya and Bridge (2019) focus on the comparison between calibration-enhancing approaches

and intent-aware approaches, which aim to increase the diversity of results. Instead of performing the re-ranking w.r.t. pre-selected content-based (CB) feature as in Steck (2018), Kaya et al. suggested to utilize user's sub-profiles based on collaborative (CF) similarity of items. Despite the fact that calibration and diversity are inherently different as discussed in Steck (2018), Kaya et al. have shown that intent-aware approaches often increase calibration up to some extent and vice versa. These results are in line with our own previous observations on the recipes recommendation domain (Starychfojtu and Peska 2020).

Lin et al. (2020) focused on the calibration problem from the user's perspective. Authors utilize the same calibration metric as in Steck (2018). They show that for a small group of users, it is difficult to provide calibrated results across different recommending algorithms. Nonetheless, their results also show that quite often, if one RS provides highly miscalibrated results for a certain user, the level of miscalibration is not as severe in some of the other evaluated algorithms. Lin et al. further observed that the category-wise user profile entropy and several popularity-related metrics are the key factors affecting the amount of per-user miscalibration. This was one of the motivations for applying FuzzDA on multiple base RS. Moreover, in contrast to Steck (2018), Lin et al. noted that there may be some positive correlation between calibration and recommendation relevance, which we also observed in our work.

Abdollahpouri et al. (2020) also evaluated the relation between the calibration and the level of popularity in user profiles (i.e., propensity to block-busters). Authors utilized the *popularity lift* between the user profile and recommendations and showed that the popularity lift is most severe for users with lowest average popularity (i.e., users with niche tastes) regardless of the utilized recommending algorithm. Authors further show that popularity lift positively correlates with the miscalibration of recommendations across all evaluated algorithms. We utilized a similar definition of a popularity lift[7] in the evaluation.

Nonetheless, one considerable drawback of both Lin et al. (2020) and Abdollahpouri et al. (2020) is that authors focused solely on collaborative recommending algorithms. The effect of various content-based or hybrid recommending algorithms on calibration, popularity bias and related topics remains to be disclosed. Also, it is not clear whether CB algorithms provide miscalibrated recommendations for the same groups of users as CF algorithms do.

There are several directions in which our approach differs from the ones described in this section. First, we focus on recommender systems in dynamic environments with varying sets of users and objects, non-stationary user's interests, etc. Such setting provides several challenges that were not considered by the approaches described in this section.

Second, note that we fully agree with the general requirement of Steck (2018) that RS should provide results proportionally to the various interests of users. Nonetheless, instead of evaluating the proportionality through miscalibration and use of some multicriterial optimization such as MMR, we propose a different view on the problem.

[7] We focused on the overall popularity lift per-RS instead of per-user.

Specifically, we aim to maximize the exactly proportional fraction[8] of the estimated relevance scores and as result provide a single objective to optimize instead of treating relevance and calibration independently. As we do not directly utilize calibration property as defined in Steck (2018), we denote our approach as *"proportionality-preserving"* rather than "*calibrated*" to prevent confusion.

Third, instead of aiming on re-ranking the results of a single recommending algorithm and calibrating it w.r.t. preselected CB feature, we aim on proportional aggregation of results w.r.t. multiple base RS. There were several considerations behind this choice. Central point of our thoughts is the question of *"What is the user-perceived calibration?"* or, in another words *"How to distinguish among individual interests of users?"*. In some domains, it may be relatively simple to define meaningful features to base the calibration on (e.g., genres in movie RS). This may, however, not be so simple in other domains including, e.g., most of the e-commerce. For instance, note that user's buying intent often changes as a result of previous purchases (i.e., there is no need to buy a second dishwasher immediately after the previous one), so calibration w.r.t. product categories does not make much sense for many e-commerce domains. Collaborative sub-profiles utilized in Kaya and Bridge (2019) can be considered as well, but similarly as for an arbitrary selection of CB features, there is no guarantee that collaborative relations would have any meaning to the user.

Instead, we hypothesize that sufficiently diverse recommending algorithms (e.g., one member from each collaborative, content-based and item-based RS families) provide sufficiently diverse latent views on user's profiles. Therefore, proportional aggregation of recommendations induced by these *base RS* can be considered as a proxy to the proportional representation of (latent) user's interests. Although this hypothesis needs further validation in a dedicated future work, we believe that by employing sufficiently diverse base RS, one can reveal also those latent axes of user preferences that would be inaccessible if only a single collaborative RS with explicit CB or CF sub-profiles is considered.[9]

Furthermore, if a meaningful explicit division of user's interests is established, then, for arbitrary RS, it is possible to filter out only the results consistent with each such dimension. These results can be then treated as several base RS outputs, so the proposed FuzzDA framework can be extended to combine both latent and explicit user interests as well. Therefore, in a sense, our proposal is more generic than Steck (2018).

2.4 Dynamic recommender systems

We are framing our research in the context of online recommender systems for dynamically evolving scenarios with a specific interest in small e-commerce enterprises.

The common view on recommender systems problem is to have a sparse static matrix of $users \times items$, where users' feedback on items (e.g., views, ratings, purchases) is stored in each cell. The task of RS is then to train its inner model based

[8] Exactly proportional fraction with respect to the relevance of individual actors. The actors may be individual base RS as in our case or, e.g., values of certain CB property as in Steck (2018).

[9] In other words, multiple RS may provide a broader view of user preferences than a single RS with multiple sub-profiles.

on available data and subsequently provide predictions on the missing entries of the matrix. Quality of these predictions is usually evaluated offline w.r.t. entries withheld from the feedback matrix.

This *"static"* view of the recommendation problem, however, omits several important issues that have to be considered in real-world systems. In reality, the nature of the recommendation process is *dynamic*. Novel users and items emerge, some items may be repeatedly consumed by the user, user's needs or preferences may change over time. RS algorithms have to be able to quickly adapt to such changes because requests for recommendation as well as user's responses to them may pour in at any time. Those are just a few examples of situations that are not incorporated in the *static* view of the recommendation problem.

One way to model the dynamic nature of real-world recommendation problem is through *sequence-aware* RS (Quadrana et al. 2018). Sequence-aware RS considers the recommendation problem from a temporal perspective. A typical input for such system is a stream of feedback events interleaved with requests for recommendations. Feedback events may be connected to individual users, or merely to a session of an anonymous visitor. A request for recommendation may be received at any timepoint, which urges RS to maintain some form of online updates of their internal models (e.g., through incremental learning (Frigó et al. 2017; Vinagre et al. 2014) or by representing users through more permanent entities as in item-based or session-based RS Ludewig and Jannach 2018; Hidasi et al. 2016). Upon request, the typical output of *sequence-aware* RS is a list of recommended items similarly as in the *static* case. However, instead of primarily focusing on long-term preferences as in *static* RS, short-term goals and contexts are often being incorporated (Quadrana et al. 2018).

The shift from long term to short term is motivated by the volatility of user preferences (Cao et al. 2009). In domains such as multimedia, the drift of user's preferences is rather gradual with some short-term noise, so solutions such as time-aware matrix factorization (Koren 2009) may be suitable. However, in e-commerce and similar domains, user's preferences are foremost driven by his/her current shopping needs (i.e., if users aim to buy something, they tend to consider recommendation's suitability in the context of this intent) (Jannach et al. 2017). In such cases, models with the ability to adapt to short-term user needs are preferable. We considered this observation during the process of selecting base RS, which are in majority capable of incorporating short-term needs of users.

2.4.1 Offline simulations of dynamic recommender systems

A typical approach to evaluate sequence-aware RS offline is to re-play (i.e., simulate) the past feedback events in the corresponding order and observe responses of the recommender system (Quadrana et al. 2018). In this paper, we utilize such simulation-style offline evaluation protocol—with a few enhancements. Using the vocabulary of Quadrana et al. (2018), we perform an *event-level* partitioning on the whole set of users. The request for recommendation is initiated every time new feedback is received similarly to Hidasi et al. (2016), Vinagre et al. (2014). Target items are defined with a fixed *look-ahead* (i.e., several next items are considered as relevant).

In contrast to the related work, we further focus on situations, where multiple requests for recommendation are initiated before further (positive) feedback is received. Furthermore, we employ a user model estimating noticeability of recommended items based on their position, which allows us to evaluate estimated click-through rate (CTR) instead of proxy metrics such as nDCG or recall@top-k. We provide more detailed discussion on both topics in Sect. 2.5.

2.4.2 Long-term relevance of RS and beyond-accuracy metrics

Two of the desired effects of the proposed FuzzDA framework are the ability to integrate multiple views on user's preferences and the ability to quickly adjust recommendations based on negative feedback. Both of these features should contribute towards increased long-term novelty and diversity of the recommended content, while maintaining reasonable levels of accuracy w.r.t. user's current preferences. As such, these features may considerably decrease the effect of feedback loops (Sinha et al. 2016) and contribute towards long-term sustainability of RS.

One of the first studies considering the effect of novelty or diversity over time was Lathia et al. (2010). Authors define the iterative novelty of the current list of recommendations as the fraction of novel items in it (i.e., items not displayed to the user in any previous recommendations). Authors further shown that iterative novelty is rather low for three CF-based recommending algorithms and if only subsequent lists of recommendations are considered, the diversity considerably decreases with the user profile size. These findings correspond with our own previous work (Balcar and Peska 2020). Authors further propose two algorithms to increase the novelty: switching the output of several base recommenders and obtaining final top-k recommendations from the set of top-n ($k < n$) results at random. Both approaches were included as baselines in our study. Subsequent work on this topic included, e.g. applying time-aware xQuAD to increase diversity over time (Anelli et al. 2017; Abdollahpouri and Burke 2019) or diversity adjustments in session-based recommendations (Esmeli et al. 2020).

The effect of filter bubbles received considerable attention in the past. In an early work on the filter bubbles problem, Fleder and Hosanagar (2009) focused on the overall sales diversity, and by utilizing an analytical model of recommendation process, they shown that evaluated RS tends to direct various users to the same items and therefore on the global level lead to less diversity. These results were corroborated by a follow-up study on the Apple iTunes (Hosanagar et al. 2014). User-wise effects of filter bubbles were evaluated, e.g., by Nguyen et al. (2014) using the MovieLens data. Authors focused on the CB diversity of items recommended to users as well as items rated by these users. The diversity was evaluated for blocks of items per-user and therefore reflected the effect of user's seniority (i.e., the volume of visited items per user). Authors shown that the diversity of both rated and recommended items drops for more senior users, therefore corroborating the content diversity deterioration effect in RS. Similar approach was taken by Lunardi et al. (2020) for the news recommendation problem. Authors define a homogenity-level metric on items selected by users to assess the severity of the filter bubble effect. We utilize a similar metric in the results evaluation; however, we substituted the Jaccard similarity by the cosine similarity, which is more suitable for our CB data. Authors have also shown that by utilizing

MMR re-ranking, the level of homogenity can be decreased. This together with the work of Abdollahpouri and Burke (2019) was an inspiration for the iterative MMR extensions applied on the WAVG baseline aggregator (described in Sect. 4.4).

Both temporal diversity, iterative novelty and filter bubbles are highly correlated research topics. Based on the related work, the main difference seems to be in the applied evaluation metrics. The approaches focused on the temporal diversity enhancements mainly evaluate the user-wise diversity of provided recommendations. On the other hand, approaches focused on the filter bubble problem mostly focus on the diversity of user's choices, e.g., diversity of consumed items (either individually per-user or overall through some approximation of the catalogue coverage). One of our intentions is to observe the level of correlation between the selection and recommendation diversity; therefore, both metric variants were included in the evaluation procedure.

2.5 User behavior model

The proposed FuzzDA framework as well some properties of the offline simulations (Sect. 4.1) is based on assumptions we made about the behavior of users. This section summarizes these assumptions and provides the necessary background for the proposed models. We specifically focus on three aspects: noticeability of recommended items, stability of user's preferences and repeated requests for recommendations.

2.5.1 Noticeability of recommended items

In common RS offline evaluation scenarios, it is implicitly believed that once the relevant object is listed in the recommendations, it is observed by the user. This belief is reflected by the usage of metrics such as recall@top-k or precision@top-k, which simply counts relevant items in the list of top-k recommendations. However, the assumption of an all-seeing user is not very realistic in the real world. During the usage of the system, the user may be distracted in various ways and especially in complex GUIs, he/she may simply miss some portions of the page. Furthermore, various content fragments may be displayed for different periods of time, on various screen regions, or even not at all (Peska and Vojtas 2017). Applying this to the list of displayed recommendations, there is a chance that user misses a recommendation, even though it is relevant.

Some relevance metrics such as nDCG discount the relevance of items based on their rank. Such behavior may represent a belief that items lower in the list, although being relevant, may be unintentionally skipped (i.e., not noticed) by the user. The level of relevance discounts may be understood as the aggregated probability of the skip behavior. Such metrics work fine as long as the recommender is not expected to react on "ignore" behavior of users. However, such requirements are imposed by several groups of RS, e.g. multi-armed bandits (Brodén et al. 2018) or our own FuzzDA framework. Therefore, a different approach is needed.

Instead of utilizing aggregated skip estimation as in nDCG, we devised a stochastic approach. For each request for recommendation received in the offline simulation, we randomly sample whether the user did or did not observe an item at a specific

rank. There is a plethora of ways to create probabilistic distributions for such a task. To reduce the space, we focus on variants that use only the item's rank (k) on input and that are non-decreasing w.r.t. the rank. There are some related studies utilizing eye-tracking to quantify the eye fixations over designated areas of interest, which is one possible proxy metric for the noticeability of items (Zhao et al. 2016; Granka et al. 2008; Joachims et al. 2007). If the results are displayed sequentially below each other (e.g., as in search engines), the fixations often follow so-called *golden triangle* pattern (Granka et al. 2008). The volume of fixations per search result drops (close to) linearly for a major part of the displayed page (Joachims et al. 2007). This was a main inspiration for the *linear* noticeability model. More recently, Zhao et al. (2016) considered the problem of gaze distribution in a grid-based interface such as YouTube or MovieLens. Their results show almost identical fixation probabilities for first two rows of the results grid, while the fixation probability of the right part of the row remains fairly constant. Furthermore, authors denoted that given sufficient dwell time (approx. 60 s), all grid positions reach a reasonably high probability of being fixated (80% and more). This was a main inspiration for the *static* noticeability model.

There are certain limitations to the mentioned studies. First, authors considered rather simple page layouts (1D list or 2D grid of results), while, e.g., e-commerce pages often contain numerous GUI elements that can distract the user in various ways. Second, authors did not explicitly consider the effect of not initially visible content. Third, we lack the quantification of how many fixations imply that the object was actually noticed by the user and whether there are other factors affecting this relation. As a result, although we followed the overall tendencies of the described research in the proposed noticeability models, the exact settings are illustratory and should be re-considered when utilized for a different domain or page layout. Additionally, we devised a noticeability estimation model based on the difference between displayed and accessed items in one of the evaluated datasets.

To sum up, we evaluated three different noticeability models: *static*, *linear* and *power-law*.

The *static* model assumes that all items have the same chance of being noticed, specifically $P_{static}(noticed|k) = 0.8$. This model may correspond to smaller grid-based recommendation lists displayed on prominent positions (e.g., top-right) within the page layout. The model also assumes rather thorough users who spent sufficient time to evaluate the page.

The *linear* model assumes that probability of being noticed linearly drops with the rank of an item. In the evaluation, top-20 recommended items were considered and probabilities of the first and last (i.e., 20th) items were set to $max = 0.9$ and $min = 0.1$, respectively. Probabilities of the remaining items were linearly interpolated from min and max values. This model simulates a moderate drop of noticeability that may correspond to a vertical list of items, where most of the items are within initially visible area of the page.

Finally, the *power-law* model represents a conservative estimation of average noticeability on one of the datasets (SLAN tour, ST) used in this paper. Specifically, we utilized our earlier work on ST dataset (Peska and Vojtas 2020) and focused on the comparison between items that were recommended and *clicked* and items that were

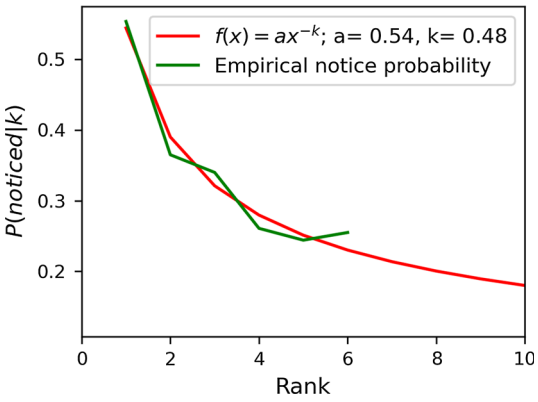

Fig. 3 Empirical notice probability and fitted power–law curve for ST dataset

recommended and later *accessed* by the user.[10] Note that $clicked \subset accessed$. Let $clicked[k]$ and $accessed[k]$ denote all items recommended on k-th position that were clicked and accessed, respectively. Based on a conservative assumption that *accessed* items represent all relevant items for a particular user, it is possible to define the probability of noticing relevant item as $P(noticed|k) = |clicked[k]|/|accessed[k]|$. These values are depicted in Fig. 3. However, as the original dataset provided data for only top-6 recommendations, we extended the empirical results via a fitted power–law curve as can be also seen in Fig. 3.

If sufficient data are available, our approach could be extended, e.g., by incorporating the page layout, actual response from users (scrolling, mouse motion, dwell time) or variable capability of items to draw user's attention. Nonetheless, we leave these to the future work.

2.5.2 Stability of user preferences

Another important consideration was how to determine the level of stability of user's preferences. In offline simulations, the stability of user's preferences has a key impact on which items (from user's future feedback) can be considered as relevant for the user at the current timepoint. As stated by Quadrana et al. (2018), there is no generally acceptable agreement on this point. Various authors applied different approaches ranging from considering only the next event as relevant to considering all future events as relevant [not to mention additional approaches aiming to detect changes in preferences directly, e.g. Eskandanian et al. (2018)]. Our stance is somewhat closer to the next-item prediction as preferences in e-commerce may change rapidly. On the other hand, at least in ST dataset, users tend to pursue a single goal for some time (i.e., we can observe short sequences of several related visited items or categories), so next-item strategy would tend to reject items already relevant from user's perspective. To cope with both constraints, we apply a next-k items strategy, i.e., considering all items occurred in the next k user's events as relevant for him/her. In evaluations, we set

[10] We also checked how far in the future user accessed the item. However, as the vast majority of events occurred within several minutes after the recommendation, we did not pursue this variable any further.

$k = 5$, which encompasses most of the sequence lengths observed in the ST dataset, and leave the effect of different next-k lengths for the future work.

2.5.3 Repeated requests for recommendations

Finally, we also considered how frequently does the user requests recommendations as compared to how frequently he/she provides feedback. The analysis of the ST dataset (see Sect. 4.2.2) revealed that less than a half of recorded feedback events got a specific target item. Other events were recorded on category pages, search results, etc. A vast majority of recommending algorithms cannot process a feedback not given on a specific item, so such events are only rarely present in RS datasets. However, it is quite common that some recommendations are offered throughout the website, including category pages or a homepage.

A typical approach in offline simulations is to initiate one request for recommendation each time a new event occurred. However, this may cause some discrepancies, because requests for recommendations that would normally occur, e.g., on category page visits are simply omitted. The main reason why this bothers us is the fact that multiple requests for recommendation may be triggered with exactly the same user profiles (with no additional feedback received in the meantime). While a conservative strategy would be to provide the same recommendations, there are other options, which we discuss in Sect. 3.4.

In order to accommodate for potentially missing requests for recommendations, we introduced a repeat parameter R to the offline simulations. Instead of triggering one request for recommendation between each two feedback events, we repeat the request R-times, giving algorithms a chance to adjust their recommendations. In evaluation, we focused on $R = 1, 2$ and 3.

Please note that the problem of repeated requests for recommendation essentially differs from the problem of repeated recommendations (Jannach et al. 2017; Lerche et al. 2016; Schedl et al. 2018). In the repeated requests for recommendation, we consider whether the recently recommended items (not consumed by the user) should be recommended again. In contrast, in repeated recommendation problem, authors focus on whether/when should be the already consumed/visited item recommended again.

3 FuzzDA framework

3.1 Overview

The proposed FuzzDA framework for RS aggregation has three main components: aggregation algorithm itself, votes assignment strategy and negative implicit feedback penalization strategy. Figure 4 provides an overview of the proposed approach.

The aggregation algorithm utilizes recommendations of base RS and aggregates them into a single list of recommendations. Formally, the input of aggregator A can be

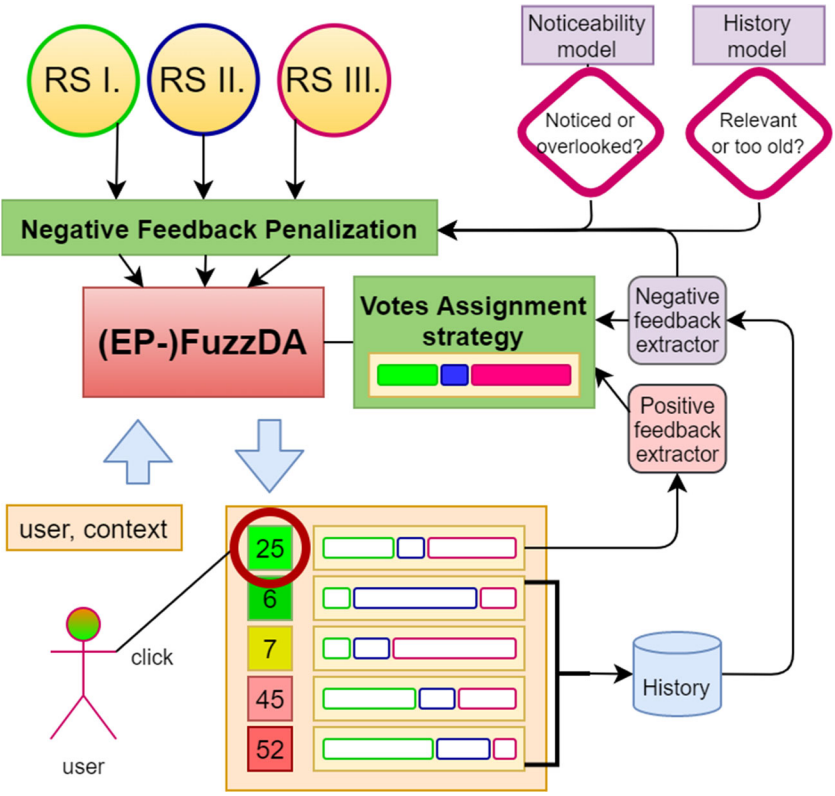

Fig. 4 Schematics of the proposed RS aggregator. The central part of the proposal is RS aggregator (FuzzDA or EP-FuzzDA). Results of base RS are supplied through the negative feedback penalization component, which utilizes models for user's noticeability and history (preference stability). Each recommended item carries the information on its support from individual RS (depicted as colored bars next to the item). Based on the individual support and user's positive or negative feedback on each item, votes assignment strategy adjust votes for base RS

viewed as a function $A : \{(O_1, S_1, v_1), \ldots, (O_m, S_m, v_m)\} \longrightarrow [o_1, \ldots, o_k]$, where O_i, S_i and v_i are recommended items, their scores and votes of i-th recommender system in the portfolio. We focused on aggregation variants capable of preserving the proportionality of votes fraction assigned to individual RS. While O_i and S_i are provided by the RS itself, votes have to be supplied externally, which is the responsibility of the votes assignment component. We focused on variants capable of small incremental updates to enable online votes updates. One variant is based on the stochastic gradient descent; other two are borrowed from the domain of reinforcement learning. Finally, the penalization strategies are based on the recent negative implicit feedback received from the current user. They decrease the S_i scores of those items that seem (based on the negative feedback) irrelevant for the user at current time.[11]

[11] I.e., at the time the recommendations are generated.

In the rest of this section, we describe the proposed variants for each of these components. We start with the variants of proportional aggregators (Sect. 3.2), followed by votes assignment strategies (Sect. 3.3) and penalization models for implicit negative feedback (Sect. 3.4). Finally, in Sect. 3.5 we discuss limitations of the proposed framework.

3.2 Proportional RS aggregations

3.2.1 FuzzDA aggregation algorithm

In Peška and Balcar (2019), we proposed a fuzzifying extension to the D'Hondt's election algorithm (FuzzDA). The main aim of FuzzDA is to enable fuzzy membership of candidates in multiple parties. Although this requirement is not of much use in the original domain of public elections, it is well suited to describe the reality of RS aggregations. Assume to have a list of several base RS. Upon the request for recommendation, each recommender R_i is represented with the ordered list of recommended items O_i, scores assigned to them S_i and votes assigned to each recommender v_i. It is not uncommon for a pair of base RS to share some of the recommended items, i.e., $O_i \cap O_j \neq \emptyset$. The working hypothesis of FuzzDA is that if an object is jointly recommended by multiple base RS, this should be considered as an additional evidence of its relevance (i.e., some relevance bonus should be gained by the object). The original DA algorithm would ignore objects' co-occurrences and treat each base RS's list independently.

Algorithm 1 Fuzzy D'Hondt's Aggregation

Input: $R_i \in \mathcal{R}$: set of base RS, $o \in \mathcal{O}$: set of candidate objects, $\bar{s}_{i,o} \in [0, 1]$: normalized recommender-object relevance scores; $\forall R_i : \sum_{\forall o} \bar{s}_{i,o}^2 = 1$, k: number of items to select, $v_i \in [0, 1]$: votes for base RS; $\sum_{\forall R_i} v_i = 1$
Output: list of aggregated recommendations O_A

$O_A = []; \forall R_i : a_i = v_i; \forall R_i : k_i = 1;$
for $i \in [0, ..., k]$ **do**
  for $o \in \mathcal{O} \setminus O_A$ **do**
    $gain_o = \sum_{\forall R_i} a_i * \bar{s}_{i,o}$
  end for
  select $o_{best} = \text{argmax}_{\forall o}(gain_o)$; append o_{best} to O_A
  $\forall R_i : k_i = k_i + \bar{s}_{i,best}$
  $\forall R_i : a_i = v_i / k_i$
end for
return O_A

The pseudocode of FuzzDA is depicted in Algorithm 1. In order to account for objects' co-occurrences, we consider S_i scores to be fuzzy membership indicators. The desired semantics of the scores is: *To what extent does the object belong to the recommendations according to a particular base RS?* Not all scoring mechanisms of base RS are "off-the-shelf" compatible with such semantics. Furthermore, various

base RS may provide scores of different magnitude, which may cause unjustified advantages over other RS. To cope with these obstacles, we first discard all scores that may correspond to the negative preference.[12] Then, for each base RS, the remaining scores are scaled to have unit L2 norm. We can assume that objects not ranked by a particular RS can be considered as irrelevant to it and therefore zero scores can be assigned to them.[13] We denote the resulting normalized scores as $\bar{S}_i$.

Scores for individual objects are calculated on line 4. We are using an *and-like* connection between the current accountable votes of particular recommender (a_r) and relevance of object for that particular recommender ($\bar{s}_{r,o}$). On the other hand, *or-like* connections are applied while aggregating scores of different recommenders, which accounts for the modality of views on user's preferences as applied by the base RS.

It can be seen that FuzzDA provides the same results as the original D'Hondt's mandates allocation algorithm (DA, Sect. 2.2), if two conditions are met:

- The lists of recommended items are disjoint.
- The relevance score for a candidate on k-th position is defined as $1 - k * \epsilon$ for some sufficiently small epsilon. [14]

As such, FuzzDA represents one possible way to extend DA for items co-occurrence. If FuzzDA is used for for RS aggregation problem, it would particularly prefer items with high scores for those base RS that are relevant, but were neglected so far (and therefore have high a_i values). In this way, FuzzDA reflects both relevance and proportionality metrics. However, it does not provide any theoretical guarantees for either of those metrics. This was the main reason behind the proposal of the EP-FuzzDA algorithm.

3.2.2 EP-FuzzDA aggregation algorithm

Similarly as FuzzDA, *E*xactly-*P*roportional FuzzDA (EP-FuzzDA) is a greedy algorithm that iteratively selects the next best candidate to the list of final recommendations. The preprocessing steps and inputs are the same for both algorithms. However, in contrast to FuzzDA, EP-FuzzDA provides a rank-sensitive optimization of the *E*xactly-*P*roportional *rel*evance *sum* (or EP-rel-sum for short). Let us first describe how we derive the EP-rel-sum criterion and what do we mean by rank-sensitive optimization and then continue with the description of EP-FuzzDA algorithm.[15]

The basic idea for EP-rel-sum was sparked by the work of Medzihorsky (2019). Medzihorsky showed that if certain proportionality criterion is applied, DA separates the per-party votes into two parts: a fraction that is exactly proportionally represented by the assigned mandates and some nonnegative residual votes (i.e., per-party underrepresentation). Medzihorsky also showed that DA iteratively maximizes the volume of exactly proportionally represented votes.

[12] We approximate this task by keeping only top-N most preferred objects per base RS.

[13] Internally, sparse matrices of scores are used.

[14] So that the difference in relevance scores would only affect the ordering within a single party, not the sequence of selected party members.

[15] Note that we simultaneously proposed the EP-FuzzDA algorithm also for the group recommendation problem (Malecek and Peska 2021).

Now, the mandates allocation problem considers neither the varying relevance of candidates (i.e., $\bar{s}_{i,j}$ scores assigned to items) nor the possible intersections in the lists of proposed candidates. However, both of these conditions are common in RS aggregations. A natural consequence is that some candidates may be better in average or even Pareto-dominating[16] to some other candidates. This may pose a following problem. While seeking for the proportionality preservation, one can find a combination of items that perfectly balance the relevance of base RS. Nonetheless, it is possible that one of the selected items is inferior (w.r.t. all base RS) to another item and yet this combination is not used, because it provides slightly worse balance.

Our solution to this problem is as follows (see Algorithm 2 for details). At each iteration, the overall utility of all candidate items is calculated. By default, the sum of the item's scores w.r.t. all base RS is utilized, but we limit the accountable relevance scores per base RS (i.e., if a particular RS has too large share in the so-far constructed results, we ignore its relevance scores while selecting the next item). The per-RS limit is the exactly proportional portion (w.r.t. assigned votes) of the total relevance of all items in the so far constructed list of recommendations.

Let us describe this idea with an example. Consider three base recommenders R_1, R_2 and R_3 with assigned votes share 0.5, 0.3 and 0.2, respectively. Furthermore, consider that they recommend the items as stated in Table 1. Starting at the first iteration with an empty list of recommendations, we consider to add o_1 to the list. Total relevance of o_1 is $0.9+0.8+0.0 = 1.7$. Proportional fractions of the total relevance are 0.85, 0.51 and 0.34 for R_1, R_2 and R_3, respectively. Therefore, the prospective gain of adding o_1 is $0.85 + 0.51 + 0.0 = 1.36$. After evaluating all candidate objects, it turns out to be the best gain and therefore we select o_1.

Lets move to the next iteration, considering o_5. Its total relevance is $0.1 + 0.1 + 0.8 = 1.0$. Proportional shares of the base recommenders are calculated from the total relevance of already selected items plus the prospective one, i.e., $1.7 + 1.0$. The proportional shares are 1.35, 0.81 and 0.54 for R_1, R_2 and R_3, respectively. Given the already selected objects, the not-yet-accounted part of these shares are $1.35 - 0.9 = 0.45$, $0.81 - 0.8 = 0.01$ and $0.54 - 0.0 = 0.54$. The prospective gain of o_5 is therefore $0.1 + 0.01 + 0.54 = 0.65$. In this case it turns out that o_6 obtains slightly higher gain than o_5 and therefore is selected. The sum of per-RS relevance scores after the selection of o_1 and o_6 are 1.1, 0.8 and 0.7, which fairly corresponds with the vote shares of base RS.

To be more formal, let the overall relevance score for recommender R_i be defined as $s_i = \sum_{o_j \in O_A} \bar{s}_{i,j}$, let TOT be the total relevance of the list of recommendations $TOT = \sum_{\forall R_i} s_i$ and let the per-RS votes be normalized to unit sum ($\sum_{\forall R_i} v_i = 1$). Now the EP-rel-sum can be defined as follows.

$$\text{EP-rel-sum}(O_A) = \sum_{\forall R_i} \min(s_i, v_i * TOT) \tag{1}$$

[16] I.e., be equal or better with respect to all considered dimensions—base recommenders' scores in our case.

We can easily observe that for two lists O_A and $\bar{O_A}$ with the same total relevance, O_A will receive higher EP-rel-sum score if the relevance distribution is more proportional to the votes distribution. Similarly, if we have two lists with the same relevance distributions, higher EP-rel-sum score will receive the one with higher total relevance.

For the rank-sensitive optimization, we follow the definition of Kaya et al. (2020).[17] Specifically, having a partially constructed list of recommendations O_A, the rank-sensitive optimization strategy should select such item o_j that would provide the best balance between relevance and proportionality in the newly constructed $O_A \cup o_j$ list. As such, the best possible balance between relevance and proportionality is maintained for all prefixes of the list of recommendations. One way to construct the aggregated recommendations in a rank-sensitive way is to utilize the marginal *gains* on the EP-rel-sum criterion as follows.

$$gain(O_A, o_j) = \text{EP-rel-sum}(O_A \cup \{o_j\}) - \text{EP-rel-sum}(O_A) \tag{2}$$

To sum up, Algorithm 2 contains the pseudo-code of EP-FuzzDA aggregator. Note that instead of using marginal gains (Eq. 2) directly, they are constructed incrementally for the sake of performance.

Also note that EP-FuzzDA does not explicitly penalize for excesses over the proportional share of relevance scores (see *max* on line 5). Such penalties would result in situations, where good items are penalized just because they are good also for "not currently wanted" RS. Instead, we simply ignore any excesses of relevance for those particular RS. This may lead to provisional disproportionalities, which are usually repaired during the next steps of the algorithm.

Algorithm 2 Exactly-Proportional Fuzzy D'Hondt's Aggregation

Input: $r \in \mathcal{R}$: set of base RS, $o \in \mathcal{O}$: set of candidate objects, $\bar{s}_{r,o} \in [0, 1]$: normalized recommender-object relevance scores; $\forall r : \sum_{\forall o} \bar{s}_{r,o}^2 = 1$, k: number of items to select, $v_r \in [0, 1]$: votes for base RS; $\sum_{\forall r} v_r = 1$
Output: list of aggregated recommendations O_A

1: $O_A = []$; $TOT = 0$; $\forall r : s_r = 0$;
2: **for** $i \in [0, ..., k]$ **do**
3: **for** $o \in \mathcal{O} \setminus O_A$ **do**
4: $TOT_o = TOT + \sum_{\forall r} \bar{s}_{r,o}$
5: $\forall r : e_r = max(0, TOT_o * v_r - s_r)$
6: $gain_o = \sum_{\forall r} min(\bar{s}_{r,o}, e_r)$
7: **end for**
8: select $o_{best} = \text{argmax}_{\forall o}(gain_o)$; append o_{best} to O_A
9: $\forall r : s_r = s_r + \bar{s}_{r,best}$
10: $TOT = \sum_{\forall r} s_r$
11: **end for**
12: **return** O_A

[17] Note that Kaya et al. used a different criterion to balance the relevance and proportionality.

3.3 Votes assignment strategies

Both FuzzDA and EP-FuzzDA aggregation algorithms expect that the votes for base RS are supplied externally. One requirement for the votes assignment strategies is that recommender systems that provide better recommendations (w.r.t. user consumption statistics) should receive more votes. In this context, we consider recommendations quality from the user's consumption point of view, i.e., better recommendations are those for which we received more positive feedback. For this purpose, we interpret a click on the recommended item as an evidence of positive user preference and no-click as a (weak) evidence of negative preference.

Another requirement is, given the nature of dynamic RS, votes assignment strategies should be capable of online incremental updates to reflect the received feedback immediately.

Let us now proceed with the definition of individual votes assignment strategies.

3.3.1 Gradient descent

The gradient descent (*GD*) votes assignment strategy was employed already in our preliminary work (Peška and Balcar 2019; Balcar and Peska 2020). The approach was motivated by the proposal of incremental updates in matrix factorization (Frigó et al. 2017; Vinagre et al. 2014). Specifically, consider the per-RS votes v_i to be variables and suppose that we are trying to maximize the criterion from Eq. 3, i.e., simultaneously maximize the sum of votes of preferred items and minimize the same (weighted) criterion for ignored items.

$$\max_{\forall v_i} \Big(\sum_{\forall o_j \in \mathcal{F}^+} \big(\sum_{\forall R_i \in \mathcal{R}} v_i * s_{i,j} \big) - \lambda_{neg} \sum_{\forall o_j \in \mathcal{F}^-} \big(\sum_{\forall R_i \in \mathcal{R}} v_i * s_{i,j} \big) \Big) \tag{3}$$

$\mathcal{F}^+$ and $\mathcal{F}^-$ denote lists of items that occurred in positive and negative feedback events, respectively. Similarly as in Vinagre et al. (2014), a single stochastic gradient descent step is performed each time a new feedback occurs. The update steps for positive and negative feedback are as follows:

$$\begin{aligned} v_i &= v_i + \eta_{pos} * \big(s_{i,j} - \sum_{\forall R_k \in \mathcal{R}, k \neq i} s_{k,j}\big) \\ v_i &= v_i - \eta_{neg} * \big(s_{i,j} - \sum_{\forall R_k \in \mathcal{R}, k \neq i} s_{k,j}\big) \end{aligned} \tag{4}$$

where η_{pos} and η_{neg} are learning rate hyperparameters. In order to prevent the divergence of votes, minimal and maximal bounds are employed and the sum of votes is linearly scaled to equal one. One of the benefits of GD algorithm is the capability to adapt on gradual changes in RS performance. In the preliminary work, the GD votes sampling performed adequately; however, we found it quite tricky to balance the η_{pos} and η_{neg}. Quite often, votes tend to regress to the uniform share or diverge to sup-

port just one RS. This was one of the main reasons why we searched for other votes assignment strategies.

3.3.2 Fuzzy Thompson sampling

Second variant is based on Thompson sampling (TS) strategy for multi-armed bandits (Brodén et al. 2018). It collects the volume of positive and negative feedback per base RS and then draw the votes at random from $Beta$ distribution based on the feedback. Specifically, consider that $\mathcal{F}_i^+$ is a list of clicked items recommended by R_i and $\mathcal{F}_i^-$ is a list of ignored items recommended by R_i. Thompson sampling scores is then defined as follows:

$$v_i = Beta(\alpha_0 + |\mathcal{F}_i^+|, \beta_0 + |\mathcal{F}_i^-|) \tag{5}$$

where α_0 and β_0 correspond to the prior distribution of score (uniform distribution is considered in evaluations). One of the baseline aggregators utilizes this approach (BEER(TS,SB), Brodén et al. 2018).

However, note that unlike BEER(TS,SB), FuzzDA framework considers the option that an item is jointly recommended by multiple base RS. Therefore, the responsibility for each recommended item is not binary, but rather *fuzzy*, and this should be also reflected by the votes assignment strategy. In order to comply with this requirement, we modified the per-RS feedback collection as follows:

$$|\mathcal{F}_i^+| = \sum_{\forall o_j \in \mathcal{F}^+} \bar{s}_{i,j}, \quad |\mathcal{F}_i^-| = \sum_{\forall o_j \in \mathcal{F}^-} \bar{s}_{i,j}$$

Now, upon each request for recommendation, votes for each base RS are drawn according to Eq. 5 and $\mathcal{F}_i^+$ and $\mathcal{F}_i^-$ values can be maintained easily by simply adding any feedback upon its reception.

3.3.3 Contextualized votes assignment

Both previously described variants lack the capability to adapt on a context under which the requests for recommendation are made. Several contextual axes may be relevant in the our use-cases, e.g. user's seniority, segment of the catalogue that he/she recently explored or type of page he/she currently visits. Such setting is quite similar to the work of Li et al. (2010) applying contextual bandits on the news recommendation problem. Specifically, Li et al. proposed LinUCB, a contextualized upper confidence bound algorithm, where individual context dimensions are required to have a linear dependence on the payoff of each arm (i.e., votes of base RS in our case). Coefficients for the linear model are estimated via Ridge regression from the matrix of contextual features of each trial and a list of corresponding rewards. For further details, please visit Section 3.1 in Li et al. (2010).

Algorithm 3 contains a pseudo-code of LinUCB votes assignment strategy (denoted as CX in evaluations). Parameters of the internal model are adjusted upon each received feedback (lines 5,6). Similarly as in TS, we utilize the relevance score of base RS instead of a binary reward, i.e., while updating parameters of R_i based on event

Algorithm 3 Contextualized Votes Assignment via LinUCB Li et al. (2010)

Notation: $cx \in \mathbf{R}^d$: context vector with d dimensions, s: reward value, R_i: i-th base recommender, v_i: votes assigned to i-th recommender, $\mathbf{A}_i$ and $\mathbf{b}_i$: parameters of the linear model for i-th recommender, $\mathbf{I}_{d\times d}$: identity matrix, $\mathbf{0}_d$: zero vector.

1: $\forall i : \mathbf{A}_i = \mathbf{I}_{d\times d}; \mathbf{b}_i = \mathbf{0}_d$

2: Function ASSIGN_VOTES(R_i, **cx**)
3: $\quad \hat{\Theta}_i = \mathbf{A}_i^{-1}\mathbf{b}_i$
4: $\quad v_i = \hat{\Theta}_i^T \mathbf{cx} + \sqrt{\mathbf{cx}^T \mathbf{A}_i^{-1} \mathbf{cx}}$
5: $\quad$ **return** v_i

6: Function UPDATE(R_i, **cx**, s)
7: $\quad \mathbf{A}_i = \mathbf{A}_i + \mathbf{cx} * \mathbf{cx}^T$
8: $\quad \mathbf{b}_i = \mathbf{b}_i + s * \mathbf{cx}$

occurred on o_j, the reward score $s = \bar{s}_{i,j}$ in the case of positive feedback and $s = 0$ in the case of negative feedback. Upon the request for recommendation, votes assignments are calculated for each recommender (lines 3–5). Note that the inverse of $\mathbf{A}$ matrices is re-calculated periodically as suggested in Li et al. (2010).

As for the employed contextual features, in both evaluated datasets (SLAN tour and MovieLens) we utilized an aggregation of CB features of recently visited objects as a proxy to the user's interests. Also, for both datasets we employed several notions of user's seniority (i.e., the level of experience users have with the website) based on the volume of visited objects. Finally, in MovieLens dataset, some basic demographic features were utilized as well and in online experiments on SLAN tour, we incorporated the type of currently visited page (*homepage*, *category page* or *object detail page*).

3.4 Implicit negative feedback penalty models

So far, given the assigned votes, the aggregation algorithm is completely deterministic. This fits the original purpose of the DA (i.e., mandates allocation in political elections), but the situation in dynamic RS is different.

Now, it is the truth that the negative implicit feedback is considered by all votes assignment strategies. However, they only utilize it on the level of individual recommending algorithms (not on the level of specific objects). Furthermore, upon receiving the feedback, votes assignment strategies introduce only small gradual changes to the votes.[18] As a result, if no new positive feedback was received, there would be only small (if any) changes between two consecutive recommendations to the same user.

On the one hand, such repeated recommendations may let user to re-consider the item (or actually observe it if he/she did not notice the item during previous recommendations). On the other hand, repeating the same recommendations over and over again may quickly annoy users. Furthermore, such recommendations are blocking available space for other items, potentially more appealing to the user.

[18] With an exception of early stages of *Contextualized* votes assignment.

In order to find some balance for repeated recommendations, we propose two different variants of penalization strategies based on the implicit negative feedback. The penalization is applied on items in a personalized manner (i.e., only the items with known negative feedback from the current user are affected). The amount of reduction is based on two variables: how certain we are that the user noticed the object and how far in the past this event occurred (in another words, what is the chance that the user changed his/her mind). Furthermore, we also evaluate a randomization baseline, which does not utilize negative feedback directly, but may have a similar effect.

3.4.1 Roulette-based randomization baseline

The Roulette-based randomization (*Rand*) is rather straightforward: we optionally amplify the object's scores (i.e., $gain_o = gain_o^q$) and normalize them to the unit sum, so they can be treated as a probability distribution. Then, in each step of FuzzDA or EP-FuzzDA, the next item is selected at random w.r.t. this distribution. The rest of the algorithm remains unchanged. The amplification parameter q governs the exploration vs. exploitation trade-off for the predicted scores (with higher q values the results would be more similar to the original algorithm).

3.4.2 Relevance discounts penalization

The relevance discounts penalization ($RelDisc$) was already described in our preliminary work (Balcar and Peska 2020). The method divides the original relevance scores of items with the sum of the evidence from negative feedback. To be more formal, let $\mathcal{F}^-_{u,j} = [(o_j, t, k)]$ be the list of recent implicit negative feedback events (i.e., ignored recommendations) from user u on object o_j. Furthermore, feedback events contain information on the rank of object o_j within the list of recommendations (k) and the temporal or interactional distance[19] (t). Then, the relevance discounts penalization modifies the score of object o_j as follows:

$$s_j = \frac{s_j}{1 + \sum_{\forall (o_j,t,k) \in \mathcal{F}^-_{u,j}} \left(noticed(k) * relevant(t) \right)} \tag{6}$$

where $noticed(k) \rightarrow [0, 1]$ function provides an estimation that the user *noticed* an object on k-th position and ignored it intentionally and $relevant(t) \rightarrow [0, 1]$ provides an estimation that an event with t distance from the current timepoint is still relevant (i.e., that user did not change his/her mind in the meantime).

3.4.3 Probabilistic penalization

The *Probabilistic* penalization applies a slightly different view on the problem. It tries to estimate a joint probability that despite all evidence from negative feedback, the item might still be relevant. This can be decomposed event-wise into the probability that

[19] By "interactional" we mean how many other events occurred between the specified one and the present.

either the user did not notice the item or that he/she already changed his/her mind from that timepoint. Assuming independence of these events and using the one-complement trick, the joint probability that the item might be relevant can be defined as follows:

$$P(rel|\mathcal{F}^{-}_{u,j}) = \prod_{\forall(o_j,t,k)\in\mathcal{F}^{-}_{u,j}} 1 - \Big(noticed(k) * relevant(t)\Big) \tag{7}$$

Finally, the adjusted score should jointly represent the belief of base RS that the item is relevant (this is represented by the original score) and the probability that it may be relevant despite received negative feedback. Using the independence assumption on these phenomena, the probabilistic penalization is as follows:

$$s_j = s_j * P(rel|\mathcal{F}^{-}_{u,j}) \tag{8}$$

As for the $noticed(k)$ and $relevant(t)$ functions, they should provide realistic estimations of the phenomena at hand, so their definitions are inherently domain dependent. In our work, we experimented with *static* and *linear*ly decreasing models. Static model simply estimates uniform probability irrespective of k or t parameters. Linear model for $noticed(k)$ function utilizes a simple assumption that the chance that user noticed an item linearly decreases with the position of the item in the list of recommendations. Linear model is defined as:

$$noticed_{lin}(k) = min + \frac{(\text{top-k} - k) * (max - min)}{\text{top-k}} \tag{9}$$

where min and max are parameters of the model and top-k value corresponds to the volume of recommended items (top-k $= 20$ in the experiments). Linear model for the $relevant(t)$ function is defined analogically, i.e., the events further in past are less probable to be still valid and a maximal size to the list of events is applied ($max_t = 100$, which corresponds to 5 lists of recommendations in current experiments).

3.5 Limitations of FuzzDA framework

There are several limitations to the usability of the proposed FuzzDA framework. The framework is built on two groups of assumptions: assumptions on base recommenders and assumptions on the user behavior.

As for the base recommenders, we work with the hypothesis that each base recommender provides a slightly different view on user's preferences and therefore their unbiased aggregation may improve the overall quality of recommendations. This hypothesis is plausible only if the base recommenders are sufficiently diverse and relevant at least to some extent. Furthermore, FuzzDA framework prefers items that are sufficiently relevant for multiple base RS. In order to utilize this feature, there should naturally be at least some intersection between the results of individual recommenders. In the evaluation scenario, we utilized highly diverse base RS w.r.t. working principles (i.e., collaborative, content-based, session-based, non-personalized), and we

also checked the level of intersection among individual approaches (see Sect. 4.6.1). Similar procedure should be considered while applying the framework on a novel domain.

In Sect. 2.5, we summarized several assumptions and observations on the behavior of users, i.e., noticeability of recommended items, stability of user preferences and repeated requests for recommendations. Both the uncertainty whether user noticed an item and the assumption of partial stability of user's preferences largely impacted the design of the framework. Furthermore, all three assumptions provided foundations for the offline evaluation scenario. In general, the framework is sufficiently parameterizable to deal with different variants of the presented user model. However, if the user behavior is largely inconsistent with the presented behavior model (e.g., there is a large volume of sudden, unpredictable changes in user preferences), applicability of the FuzzDA framework would be severely decreased.

Finally, the proposed FuzzDA framework does not provide any countermeasures if some particular user consistently behaves against some of the framework goals (e.g., to maintain diversity). If, for instance, user consistently ignores some groups of recommended items, INF penalization strategies may render such groups less accessible for the user, which in turn may decrease observed diversity of recommendations. The effect was not measurable on evaluation datasets due to rather low consumption ratio and shorter stable preference intervals. However, it may be worthwhile to consider it in situations where users tend to have high overall consumption, visit the site frequently and have rather stable preferences (e.g., news providers).

4 Evaluation protocol

4.1 Offline simulations

In order to evaluate the performance of RS in a dynamic environment, we proposed a following simulations methodology. Parameters of the simulation are train set size ($ts = 90\%$ in our evaluations), user's noticeability model (*linear*, *power-law*, or *static*), repeat parameter for the recommendation requests ($R \in \{1, 2, 3\}$) and the window size ($w = 5$) to determine currently relevant objects. Figure 5 provides an overview of the simulation process.

The feedback events are ordered according to their timestamps, and the head of the list (based on ts) is kept aside to train base RS. After training, the simulation iterates over the feedback events. For each feedback event, it determines the next-w items visited by the user as currently relevant. Items clicked in previous iterations of some RS variant are excluded from currently relevant items for that particular RS variant. Then, the request for recommendations is initiated R-times. For each request, the simulation selects which positions of items would be noticed by the user[20] based on the noticeability model. Then, the simulation forwards the request for recommendation to all evaluated RS variants and collects the recommendations. The simulator evaluates recommended item as *clicked*, if it is both noticed and currently relevant. Other items

[20] Note that the assignment is the same for all evaluated RS variants.

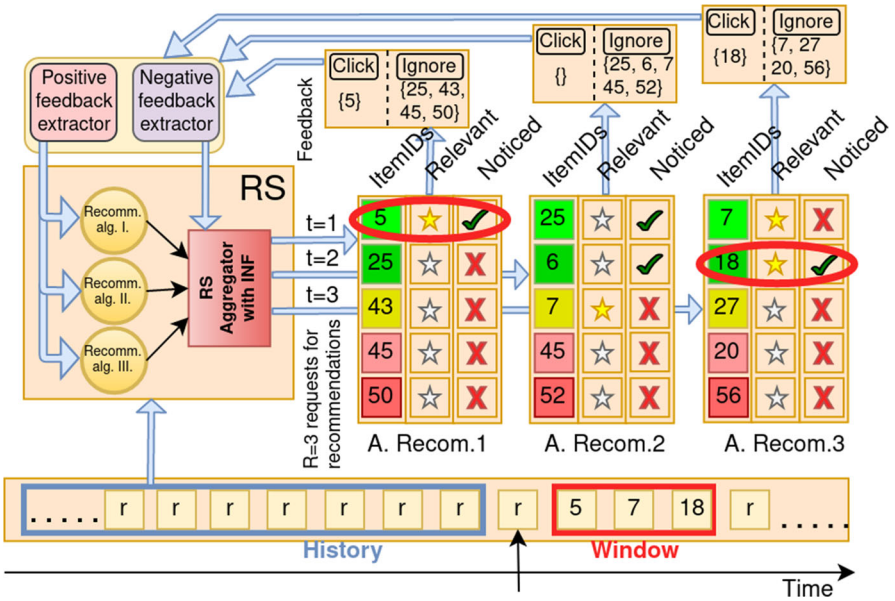

Fig. 5 Schematics of the offline simulations. Timeline of events is displayed on the bottom, where a window of current relevant results is depicted in red. RS has a chance to repeatedly recommend R-times for the same user's profile. (Implicit negative feedback may be utilized between each round.) For each recommended item, a probability of being noticed is defined (illustrated with cell's background color). "Noticed" indicator is generated at random based on this probability. Only if the item is both noticed and relevant, we consider it as a click. Finally, after all 3 requests for recommendation are processed, we move for the next event in the timeline

are evaluated as ignored by the user. Both ignore and click events are shipped to the corresponding RS aggregator, which may adjust its internal model based on this feedback (e.g., change vote assignments or employ some negative feedback penalty). After all R requests are processed, the simulation moves to the next event and discloses the current one to base RS, so they can update their models as well.

4.2 Datasets

4.2.1 MovieLens dataset

First series of simulation experiments were conducted on a well-known MovieLens 1M dataset (further denoted as ML1M) (Harper et al. 2015). The dataset contains explicit ratings on 1–5 scale and some basic content-based attributes of rated movies. The dataset does not contain explicit user sessions, but each rating is accompanied with a timestamp, so it is possible to perform a sequence-aware simulation experiments on it. Because in our experiments we focus on predicting clicking behavior, some adjustments to the dataset had to be made. Note that lower explicit ratings are expected to convey negative preferences of users. Assuming that the user can anticipate his/her

ratings up to some extent, such movies would be rarely clicked on if recommended.[21] Therefore, we keep only the events with 4* or 5* ratings in the dataset similarly as, e.g., Pan and Chen (2013).

ML1M contains only a very basic content-based (CB) attributes (year, genres and title). In order to boost the potential of CB recommendation, we extended the dataset with IMDB[22] data in the same way as in Peska (2018). To be more specific, we collected additional information on the contributing persons (actors, director, writer), language, country of origin, overall ratings, etc. These attributes were used as a base for a content-based recommender (Cosine CB) as well as CB diversity metric.

4.2.2 SLAN tour dataset

Second series of simulation experiments as well as an online A/B testing were conducted on SLAN tour—a medium-sized Czech travel agency. The agency sells tours of various types to several dozens of countries. Some tours (such as trips to major sport events) are one-time-only events, others, e.g., seaside holidays or sightseeing tours are offered on a similar schedule with only minimal changes for several years. All tours contain a range of content-based (CB) attributes, e.g., tour type, meal plan, type of accommodation, length of stay, prices, destination country, etc. Similarly as in ML1M, the same CB similarity was utilized in the CB recommender and as a base for the CB diversity metric.

Agency's website contains attribute and keyword search GUI as well as some browsing and sorting options. Recommendations are displayed throughout the website, specifically on main page, browsed categories, search results and tour details. However, due to the importance of other GUI elements, recommendations are placed on the edge of initially visible area.

The dataset utilized in this study comprises almost 12 months of the user visit events (mid January till late December 2020). The SLAN tour dataset (further denoted as ST dataset) contains approx. 210K events, out of which approx. 90K were object visits. This renders the appropriate R hyperparameter somewhere between 2 and 3. The dataset contains feedback from 80K users (identified via browser cookies) and over 2300 unique tours.

4.3 Base recommender systems

In order to evaluate the capability of the RS aggregation methods, several diverse base RS were considered.

Session (sequence)-based recommendation problem is often approximated through item-to-item recommender systems. In order to allow for this possibility, we implemented classical item-based KNN (*iKNN*, Jannach et al. 2017). For the current item (i.e., the last item visited by the user), iKNN recommends top-k items most similar to the current one based on the mutual user's visits and cosine similarity.

[21] The obvious exception are movies that seemed promising, but turned out to be a disappointment. Although such cases definitely exist, we suppose them to be rather rare.

[22] imdb.com.

In contrast to iKNN, Session KNN (SKNN Jannach et al. 2017) considers whole sessions and defines a similarity between individual sessions (Jaccard index on the set of visited objects). Algorithm then recommends novel items from sessions similar to the current one.

In order to evaluate also classical approaches for collaborative filtering, we utilized *BPR* matrix factorization (Rendle et al. 2009). During simulations, BPR was periodically re-trained after a fixed number of events to keep it as up-to-date as possible.

We added the *Most popular* non-personalized RS to see how aggregators can cope with a (probably) inferior RS in their portfolio. This recommender simply returns top-k items with the highest sum of visits/ratings up to date.

Content-based information are processed via *Cosine CB* recommender (Peska and Vojtas 2020). While preprocessing CB attributes, nominal attributes were binarized and numeric attributes were standardized. Cosine similarity is then applied on the vector of CB attributes. For a selected item, Cosine CB returns a list of most similar items.

Skip-gram *word2vec* model (Mikolov et al. 2013) utilizes the stream of user's visits. Similarly as in Barkan and Koenigstein (2016), the sequence of visited objects is used instead of a sentence of words; however, we kept the original window size parameter intact. The output of the algorithm is an embedding of a given size for each object, while similar embeddings denote objects appearing in similar contexts. For each item, word2vec RS returns items with the most similar embeddings to the current one (w.r.t. cosine similarity).

Both *CosineCB* and *word2vec* RS were extended to consider longer sequences of objects similarly as in Peska and Vojtas (2020). Recommendations are calculated for last-k visited items. Scores of individual items are then aggregated either w.r.t. *max* score, *mean* score or *weighted mean*, where items further in the past receive lower weights.

Note that the semantics of this implementation of word2vec (as well as the other base RS) is to provide alternatives rather than, e.g. complementary products. This fits with both domains (travel agency and movies) as the notion of complementarity does not make much sense here. In other domains, however, the selection of base RS should be adjusted appropriately.

4.4 Baseline RS aggregators

Apart from variants of FuzzDA algorithm, we evaluated several baseline algorithms. Random K from N (*RandKfromN*) and *Switching hybrid* RS were implemented according to Lathia et al. (2010).

When the recommendations are delivered to the user for the first time, *RandKfromN* returns top-k recommendations of the underlying base RS. On the next encounter, the final list of recommendation is selected at random from top-N candidates. In evaluations, we utilized $k = 20$ and $N = 100$ and used the best base RS for each dataset (*word2vec* for ST dataset and *iKNN* for ML1M) to provide top-N recommendations.

Random *Switching hybrid* works in a similar way; however, it randomly selects the base RS, whose output will be delivered. When the user requests recommendations

for the first time, the recommendations of the best base RS are utilized. Next time, the base RS is selected at random from the portfolio.

Another natural baseline is the weighted average ($WAVG$) of item-wise scores. In evaluations, we utilized the GD strategy to assign votes of base RS. We also implemented a maximal marginal relevance (MMR, Carbonell and Goldstein 1998) iterative diversity enhancements to $WAVG$ (denoted as *WAWG+MMR*). The recommendations are selected iteratively according to MMR score as follows:

$$s_{MMR}(o_j) = (1 - \lambda)s_j - \lambda \max_{\bar{o} \in O_A} \big(sim(o_j, \bar{o})\big), \tag{10}$$

where s_j denotes aggregated score of item o_j received from WAVG, sim denotes CB similarity of items and O_A denotes the list of recommended objects. In contrast to the common usage of MMR, O_A includes both the currently constructed list of recommendations and the whole list of recommendations from the previous user's request. The reason is that in this way, the recommendations in the current iteration should not only differ from each other, but also from the previous recommendations. We hypothesized that this adjustment will result into improved long-term diversity. In evaluations, we utilized $\lambda = 0.5$ to ensure an equal importance of both diversity and accuracy.

We also incorporated FAI algorithm (Masthoff 2011) that iteratively constructs the list of recommendations O_A by switching between base RS and each time selecting the best remaining item of the current RS. In essence, FAI algorithm is similar to *switching hybrid*, but the switch is performed for each position in the list of recommendations rather than for each list.

Finally, we utilized a multi-armed bandit algorithm for RS aggregation *BEER(TS,SB)* (Brodén et al. 2018). The algorithm iteratively utilizes Thompson sampling on the base RS level and each time selects the best not-yet-selected item from the winning RS. However, the algorithm does not consider any synergic effect for items recommended by multiple base RS.

4.5 Evaluation metrics

During the evaluation, we observed a wide range of metrics incorporating various notions of relevance, novelty, diversity, fairness and popularity.

We evaluated the relevance of recommendations through estimated click-through rate (CTR). We utilized a conventional definition:

$$CTR = \frac{|\text{clicked items}|}{|\text{recommended items}|}$$

In the online setting, the notion of clicks is obvious. In the offline simulations, we utilize the assumption that an item is clicked whether it is currently relevant and noticed by the user at the same time, i.e., $clicked = relevant(u, i) \wedge noticed(u, i)$, where $relevant$ and $noticed$ functions are defined in the user behavior model (Sect. 2.5).

We considered the problem of novelty versus diversity along several axes: the underlying similarity metric, iterative or user-wise measurements and the evaluation of recommended or clicked items. Specifically, we consider three variants of similarity metrics: collaborative (CF) cosine similarity defined on the rating vectors of respective items, content-based (CB) cosine similarity defined on CB attributes and identity (ID) metric simply stating whether two objects are the same.

In the *iterative* evaluation of novelty, we focus on how much the newest list of recommendations differs from the previous recommendations given to the user. Formally, lets have a sequence of recommendations given to the user $S_u = [O_1, \ldots, O_m]$, where $O_i = [o_{i,1}, \ldots, o_{i,k}]$. Further suppose that $S_u[l]; l < m$ denotes a prefix of the recommendation sequence $S_u[l] = [O_1, \ldots, O_l]$ and $sim(o_i, o_j)$ is a similarity metric (CF, CB or ID). Now, while evaluating the novelty of x-th iteration of recommendations, we compare similarity of all items from O_x with their closest relatives from the previous iterations $S_u[x-1]$ and report mean values.

$$nov_{sim}(u, x) = \frac{\sum_{\forall o_i \in O_x} 1 - \max_{o_j \in S_u[x-1]} sim(o_i, o_j)}{|O_x|} \tag{11}$$

If identity is considered as the similarity metric, iterative novelty is the same as the temporal novelty defined in Lathia et al. (2010). Another way how to look on the iterative novelty is that it plays the same role as the second component of MMR metric (see Eq. 10), only expanded from the list of recommendations to the sequence of lists of recommendations.

In contrast, the per-user diversity focuses on how much diverse are the recommendations to individual users (irrespective of time). In another words, it provides answers on whether only a single user's interest is being continually exploited, or the user received a diverse selection of items to choose from. Formally, let O_u be a concatenation of all recommendations received by the user $O_u = \{o_{1,1}, \ldots, o_{1,k}, o_{2,1}, \ldots, o_{m,k}\}$. Then per-user diversity is evaluated as mean diversity between individual items from O_u.

$$div_{sim}(u) = \frac{\sum_{\forall o_i, o_j \in O_u; i \neq j} 1 - sim(o_i, o_j)}{|O_u| * (|O_u| - 1)} \tag{12}$$

In practice, we approximate this diversity by randomly sampling the o_i, o_j pairs. We evaluated per-user diversity for both CF and CB similarity metrics and both for items recommended to user as well as items that were clicked by the user. This metric is similar to the content diversity proposed in Nguyen et al. (2014) including the distinction between recommended and consumed items (the underlying CB similarity is different, though).

The diversity metric can be utilized for ID similarity as well; however, in that case we opted for a simplified definition. Relative per-user coverage ($RelCov$) is defined as the ratio between the volume of unique items recommended to the user and the volume of all recommended items. The values of $RelCov = 1$ denote that all items were recommended to the user just once and so the recommendations covered maximal possible fraction of objects.

Next, although per-user recommendations may be both novel and diverse, RS may tend to provide generic recommendations by over-sampling the highly popular items. To evaluate this phenomenon, we utilized a popularity lift score ($PopLift$) similarly as in Abdollahpouri et al. (2020) (however, as we do not focus on per-user lift, but rather on per-RS lift).

$$PopLift = \frac{mPop_{rec} - mPop_{data}}{mPop_{data}} \tag{13}$$

The $mPop_{rec}$ and $mPop_{data}$ stand for the mean popularity of items that were recommended and items that occurs in the dataset, respectively. Formally, suppose to have a list of positive feedback events in a dataset $f_i(u, o) \in \mathcal{F}^+$. Each event is triggered by a user u on an item o. We can use the notation $o_j \in f_i$, meaning that the item o_j is a target in the event f_i. Then popularity of an item is defined as

$$pop(o_j) = \frac{|\{f_i : o_j \in f_i\}|}{|\mathcal{F}^+|}$$

Now, suppose that O_{rec} contains a concatenated list of all recommendations (irrespective of users) and O_{data} contains a list of target items for all events $f_i(u, o) \in \mathcal{F}^+$. Then

$$mPop_{rec} = \frac{\sum_{o_j \in O_{rec}} pop(o_j)}{|O_{rec}|} \text{ and } mPop_{data} = \frac{\sum_{o_j \in O_{data}} pop(o_j)}{|O_{data}|}.$$

Finally, we also focused on the level of fairness while representing individual base RS in the final list of recommendations. However, this is rather a challenging problem, because for many evaluated algorithms, the relevance assignments of base RS change over time based on the received feedback—which in turn depends on the previously displayed recommendations. For example, each variant of FuzzDA framework gives each base RS different amount of votes at each timepoint. Therefore, any "generic" fairness metric such as "the proportion of base RS that appear in the final recommendations" is inherently biased as it partially contradicts the fairness metric considered by the aggregation algorithms. In contrast, we aimed to measure whether aggregation algorithms are able to maintain the proportionality to whatever they currently consider to be a fair representation of base RS.

For FAI algorithm, the relevance of each base RS is uniform. For variants of FuzzDA as well as WAVG, we consider the ratio given by assigned per-RS votes. Finally, for BEER(TS,SB), we utilized the expected value of beta distribution (Eq. 5) for each base RS. Then, we can compare distributions induced by per-RS relevance assignment with the ones induced by actually recommended items.

Specifically, for the final list of recommendation $\mathcal{O}_A$ constructed by an aggregator A, the score of base RS R_i is defined as $s_{R_i} = \sum_{o \in \mathcal{O}_A} s_{i,o}$, where $s_{i,o}$ is the estimated relevance of item o by R_i. The fairness of the list is defined as follows:

$$fairness = KL\_div(norm([s_{R_1}, \ldots, s_{R_m}]), norm([v_{R_1}, \ldots, v_{R_m}]))$$

where v_{R_i} denote the estimated relevance of R_i by aggregator A at the current timepoint (e.g., the volume of votes in FuzzDA framework). KL_div denotes Kullback–Leibler divergence of probability distributions (the higher values indicate lower fairness of the representation). Note that s_{R_i} and v_{R_i} values were normalized to the unit sum as required by KL_div function. In results, we report mean values of $fairness$ over all lists of recommendations.

4.6 Hyperparameter tuning

During the hyperparameter tuning, we utilized the same offline simulation strategy as in the final experiment. However, in this case we utilized the first 80% of events for training and evaluated the models on the following 10% of the events, so the test-set of the final evaluations was not utilized in this phase.

4.6.1 Baseline RS selection and comparison

As for the base RS variants, hyperparameters of base RS were evaluated for each dataset w.r.t. CTR. As all base RS are non-randomized, we evaluated them only w.r.t. a single variant of user behavior. For *word2vec*, the window sizes were selected from {1, 3, 5}, embedding sizes from {32, 64, 128} and number of iterations from $\{50K, 100K, 200K\}$. For *Cosine CB*, two variants of similarity matrices (with or without TF-IDF weighting) were evaluated. Furthermore, for both *word2vec* and *Cosine CB*, we evaluated several profile aggregation variants, namely profile sizes were selected from {1, 3, 5, 7, 10 and *all*} and results aggregations from $\{max, mean$ and $weighted AVG\}$. For *BPR MF*, number of factors was selected from {10, 20, 50, 100}, learning rate and regularization from {0.003, 0.01, 0.03, 0.1} and number of iterations from {5, 10, 20, 50}. In *SKNN*, we selected the volume of neighbors from {25, 50, 75} and finally *iKNN* as well as *Most popular* recommenders are non-parametrized.

As for the results of best variants per base RS, $iKNN$ performed the best on ML1M dataset by a large margin, followed by *SKNN*, *word2vec*, *Cosine CB*, *BPR MF* and *Most popular*. The performance difference between *SKNN* and *word2vec* was minimal, similarly as between *BPR MF* and *Cosine CB*. Results on ST dataset were somewhat different. *Word2vec*, *SKNN*, *iKNN* and *Cosine CB* performed comparably to each other (in this order). Both *BPR MF* and *Most popular* performed inferiorly by a large margin to the previously mentioned algorithms.

Based on these results, we selected the best-performing variant for each base RS. Furthermore, given a large hyperparameter space for *word2vec*, *BPR MF* and *Cosine CB*, we selected a second candidate with close-to-best performance and sufficiently different hyperparameters to enhance the diversity of the portfolio. Base RS selected for the portfolios of RS aggregators are listed in Table 2.

Next, we evaluated whether the necessary conditions for the applicability of FuzzDA framework were met, i.e., whether individual base RS provide appropriately diverse recommendations. In order to do so, we compared corresponding[23] lists of top-20 recommended objects for each base RS w.r.t. Jaccard similarity. Figure 6 depicts

[23] By corresponding we mean responses given to the same events in the simulation timeline.

Table 2 Selected base RS for ML1M and ST datasets

Algorithm	ML1M parameters	CTR	ST parameters	CTR
Word2vec 1	f:32, w:1, i:50K, agg:wAVG, len:3	0.00630	f:32, w:1, i:100K, agg:max, len:1	**0.00801**
Word2vec 2	f:64, w:1, i:50K, agg:wAVG, len:7	0.00586	f:64, w:1, i:200K, agg:wAVG, len:5	0.00740
Cosine CB 1	agg:max, len:1	0.00431	agg:max, len:1	0.00571
Cosine CB 2	agg:wAVG, len:3	0.00438	agg:wAVG, len:5	0.00470
BPR MF 1	f:20, i:20, lr:0.003, reg:0.1	0.00378	f:20, i:50, lr:0.1, reg:0.01	0.00186
BPR MF 2	f:100, i:10, lr:0.003, reg:0.1	0.00379	f:50, i:20, lr:0.1, reg:0.03	0.00199
SKNN	k:25	0.00676	k:25	0.00683
iKNN	–	**0.01330**	–	0.00655
Most pop	–	0.00308	–	0.00122

f factors or embeddings size, *w* window size, *i* iterations, *agg* profile aggregation, *len* profile length limit, *lr* learning rate, *reg* regularization, k, number of neighbors. For each base RS, we depict its CTR on the validation sets. Best results are in bold

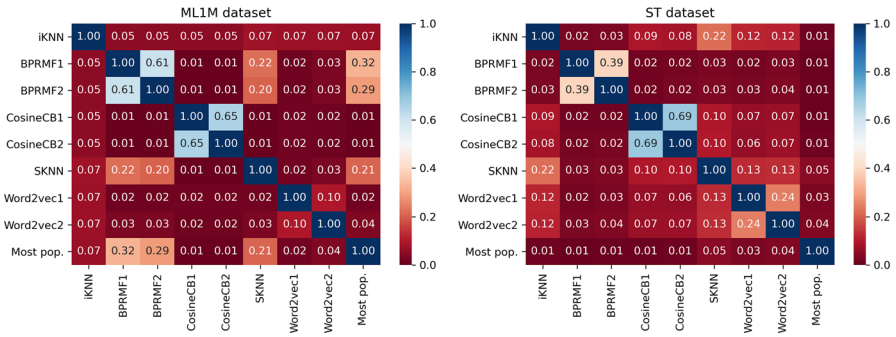

Fig. 6 Mean Jaccard similarity of top-20 recommendations for base RS

mean Jaccard similarities for both ST and ML1M datasets. We can observe that similarity of pairs with the same base algorithm (Word2vec, Cosine CB, BPR MF) is higher than other pairs. Nonetheless, even those pairs are sufficiently diverse to justify the inclusion of both members. Other algorithm pairs provide highly diverse recommendation supporting the assumption that each base RS pursues different (possibly latent) axis of user preferences. On the other hand, recommended lists have mostly non-empty intersection with some of the other base RS's recommendations. This enables the usage of aggregation techniques focusing on joint relevance such as FuzzDA framework.

4.6.2 Roulette amplification parameter

In the next stage of hyperparameter selection process, we focused on the role of the amplification hyperparameter q in the Roulette-based randomized selection for FuzzDA / EP-FuzzDA algorithm (see Sect. 3.4.1). With the higher values of q, the selection would become more uniform and closer to the default selection procedure (i.e., select items with currently highest relevance). We assumed that this would have a direct effect on the decrease of incremental novelty and indirectly also affect diversity and popularity metrics. On the other hand, having the selection procedure too uniform might negatively impact CTR. We evaluated these hypotheses on Roulette variants with $q = 1, 2, 3, 4$ and 5 and also compared them to the standard (*Fixed*) selection procedure.

Figure 7 depicts results of the evaluation w.r.t. CTR and iterative novelty (nov_{ID}). We can indeed observe a clear trade-off between CTR and incremental novelty. With increasing values of q, novelty gradually drops and CTR rises (unless the incremental novelty drops too much which in turn hurts CTR). Fixed selection procedure seemingly serves as a bound for increasing q values. As results w.r.t. both metrics were contradictory, we decided to utilize following variants in the offline evaluation: Roulette with $q = 1$ (best w.r.t. novelty), fixed selection (mostly best w.r.t. CTR) and Roulette with $q = 3$ as a reasonable compromise between the previous two variants.

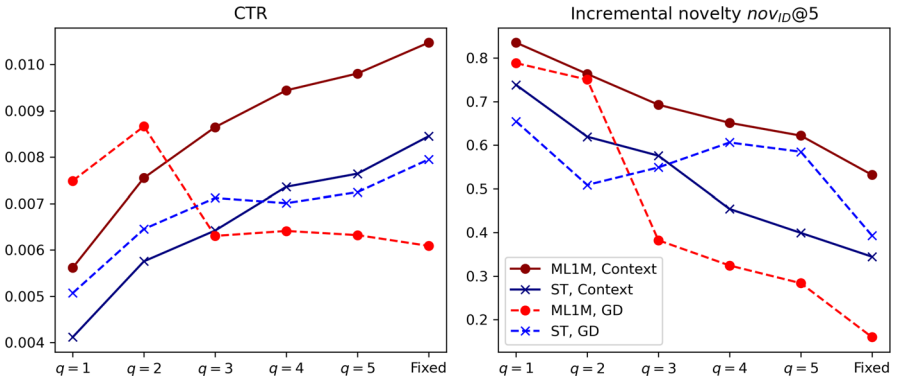

Fig. 7 CTR and incremental novelty for different values of Roulette amplification hyperparameter. Figure depicts results of ML1M and ST datasets with R = 1 and linear(0.9, 0.1) noticeability

4.6.3 Negative implicit feedback strategies

As for the penalization strategies, we considered linear and static models for $noticed(k)$ function and linear model for $relevant(t)$ function. Static penalties were selected from {0.75, 0.5, 0.25}. As for the linear models, in both cases we evaluated all plausible combinations of {1.0, 0.75, 0.5, 0.25, 0.0} values as min / max penalty values. This leads to in total 130 variants for both probabilistic ($Prob$) and relevance discount ($RelDisc$) penalizing strategies. While evaluating the results, we observed a strong dependence of the CTR values on the mean penalty score (mps) of individual items:

$$mps = \frac{(rank_{max} + rank_{min})}{2} * \frac{(hist_{max} + hist_{min})}{2} \tag{14}$$

Figure 8 depicts the dependence of CTR on mps. It can be seen that for both datasets, both penalty variants have the peak at roughly the same point around $mps = 0.2$. Nonetheless, while the CTR performance of $RelDisc$ strategy deteriorates slowly with increasing mps, the process is quicker for probabilistic strategy. We assume that this is the natural effect of the application procedure, where even highly penalized objects with $RelDisc$ scenario retain relatively lot of their original relevance compared to probabilistic penalties. Therefore, probabilistic penalties seem to be more prone towards over-penalizing the objects. On the other hand, at peak points, probabilistic penalties considerably outperform relevance discount ones in the case of ML1M dataset. As the peak points vary for different user behavior models and R hyperparameters, we decided to sample from the penalization strategies. Specifically, we divided penalization strategies along the mps score into four clusters. From each cluster, we selected the best-performing variant (w.r.t. CTR) for both probabilistic and relevance discount penalties. In total, we run the final series of experiments with 8 variants of penalization strategies.

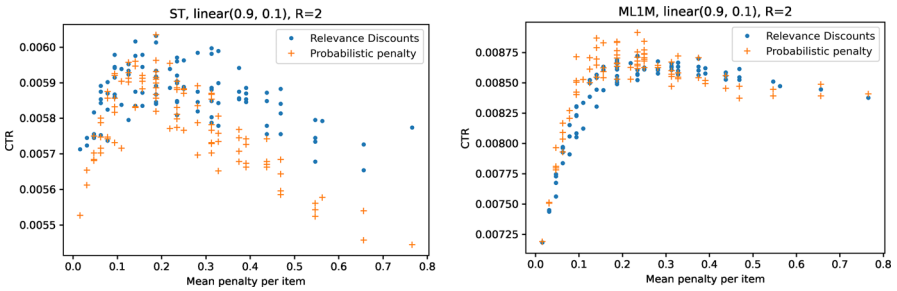

Fig. 8 CTR values of probabilistic vs. relevance discount-based penalties for various values of the mean penalty score per item

5 Experimental results

5.1 Offline simulation results

The offline simulation was performed on both ST and ML1M datasets with three different noticeability models (*static*, *linear* and *power-law*) and three variants of the request for recommendation repeat parameter ($R = 1$, $R = 2$ and $R = 3$).

5.1.1 Overall results

First, let us provide an overview of the results. Tables 3 and 4 depict the overall results on ML1M and ST datasets, respectively. Tables contain CTR and selected beyond-accuracy metrics of base RS, baseline aggregators as well as several groups of (EP-)FuzzDA framework variants. For each metric, we depict the average over nine evaluated scenarios (*static*, *linear* and *power-law* noticeability and request repeat parameter $R \in \{1, 2, 3\}$). For each scenario, we select the best-performing example of each group. As such, tables depict sort of an upper bound for each variant's results if the hyperparameter selection is consistent with the final evaluation of particular metric.

Notably, in both datasets a variant of FuzzDA framework significantly outperformed the vast majority of baselines w.r.t. CTR and iterative novelty. For the collaborative- and content-based diversity results, variants of FuzzDA were close to the best-performing approaches. In most cases, approaches that outperformed FuzzDA variants w.r.t. some diversity metric were inferior w.r.t. CTR. As for the popularity lift metric, FuzzDA variants did not improve much over the mean statistics of base RS or baseline aggregators.

In the results, we can observe a strong trade-off between CTR and (mostly) popularity lift on the one hand and diversity and novelty metrics on the other hand. This can be illustrated, e.g., on WAVG and WAVG+MMR results. The latter considerably improves both incremental novelty and content-based diversity, but at the cost of CTR reduced to approx. half of its original value. Similar tendency can be observed between RandKfromN aggregator and best base RS (on which RandKfromN is based) as well as, e.g., FuzzDA+Roulette, compared to INF models (Prob, RelDist).

Table 3 Results of offline evaluations on ML1M dataset

Algorithm	CTR	PopLift	$nov_{ID}@5$	div_{CB}	div_{CF}
Word2vec	0.00287⋆○	1.2⋆○	0.445⋆	0.382⋆	0.9⋆○
Cosine CB	0.00215⋆○	**0.091**	0.367⋆	0.329⋆○	**0.946**
BPR MF	0.00168⋆○	3.663⋆○	0.011⋆○	0.334⋆○	0.742⋆○
SKNN	0.00316⋆○	7.386⋆○	0.041⋆○	0.359⋆○	0.597⋆○
iKNN	0.00611⋆	3.677⋆○	0.219⋆○	0.366⋆○	0.771⋆○
Most pop	0.00112⋆○	12.132⋆○	0.0⋆○	0.341⋆○	0.481⋆○
RandKfromN	0.00489⋆	3.418⋆	0.657⋆	0.371⋆○	0.785⋆○
Switching hybrid	0.00425⋆○	3.778⋆○	0.573⋆○	0.369⋆○	0.825⋆
WAVG	0.00524	4.863⋆○	0.146⋆○	0.358⋆○	0.728
WAVG+MMR	0.00337⋆○	5.378⋆○	0.206⋆○	**0.388**	0.718⋆○
FAI	0.00354⋆○	4.884⋆○	0.219⋆○	0.36⋆○	0.798⋆○
BEER(TS,SB)	0.00611⋆	4.054⋆○	0.238⋆○	0.367⋆○	0.765⋆○
FuzzDA (no INF)	0.00566⋆○	5.481⋆○	0.203⋆○	0.354⋆○	0.725⋆○
EP-FuzzDA (no INF)	0.00552⋆	4.748⋆	0.275⋆	0.355⋆	0.771⋆
(EP-)FuzzDA+GD (no INF)	0.00434⋆	4.458⋆	0.152⋆○	0.353⋆	0.76⋆○
(EP-)FuzzDA+TS (no INF)	0.00568⋆	4.761⋆○	0.275⋆	0.355⋆	0.763⋆○
(EP-)FuzzDA+CX (no INF)	0.00523⋆○	4.799⋆○	0.266⋆○	0.354⋆○	0.771⋆
(EP-)FuzzDA+Prob	0.00726	3.757⋆○	**0.949**	0.368⋆○	0.807⋆○
(EP-)FuzzDA+RelDist	0.00721○	3.928⋆○	0.821⋆○	0.367⋆○	0.8⋆○
(EP-)FuzzDA+Roulette	0.00492⋆○	2.429⋆	0.832⋆○	0.374⋆	0.872⋆
FuzzDA+INF	0.00681	3.794⋆	**0.949**	0.368⋆	0.799⋆○
EP-FuzzDA+INF	**0.00735**	3.762⋆	0.898⋆○	0.367⋆○	0.807⋆
All base RS	0.00611○	0.091	0.451○	0.382○	0.946
All base agg.	0.00633○	3.418○	0.657○	0.388	0.825○
All (EP-)FuzzDA	0.00744	2.429○	0.949	0.374○	0.872○

$Nov_{ID}@5$ stands for iterative novelty at 5th user's iteration. GD, TS and CX stand for gradient descent, Fuzzy Thompson Sampling and Contextualized votes assignments, respectively. Prob and RelDist denote probabilistic and relevance discount penalty models for implicit negative feedback (INF). Best results are depicted in bold, while best results per category (divided by horizontal line) are underlined. With "⋆" we denote significant differences (p-value < 0.05) between the best and other approaches w.r.t. paired t-test, while with "○" we denote significant differences w.r.t. best approach per category

However, e.g., by including (any) INF strategy into FuzzDA framework, all observed metrics were considerably improved. Results also corroborate that GD votes assignment is outperformed by TS or CX in the majority of cases. As for using $Prob$ or $RelDisc$ noticeability models or simple Roulette selection, the results were slightly mixed. Roulette-based approaches were clearly inferior w.r.t. CTR, but provided better popularity lift, content-based and collaborative diversity. Probabilistic variants

Table 4 Results of offline evaluations on ST dataset

Algorithm	CTR	PopLift	$nov_{ID}@5$	div_{CB}	div_{CF}
Word2vec	0.00459⋆	2.542⋆○	0.223⋆	0.267⋆○	0.83⋆○
Cosine CB	0.00351⋆○	1.117⋆○	0.206⋆○	0.117⋆○	0.803⋆○
BPR MF	0.00095⋆○	**0.286**	0.036⋆○	**0.34**	0.906
SKNN	0.00422⋆○	5.088⋆○	0.085⋆○	0.201⋆○	0.64⋆○
iKNN	0.00346⋆○	1.051⋆○	0.11⋆○	0.239⋆○	0.562⋆○
Most pop	0.00074⋆○	20.831⋆○	0.0⋆○	0.308⋆○	0.859⋆○
RandKfromN	0.00323⋆○	1.513⋆	0.61⋆	0.295⋆○	0.866⋆
Switching hybrid	0.0048⋆○	3.758⋆○	0.502⋆○	0.285⋆○	0.872⋆
WAVG	0.00499⋆○	5.035⋆○	0.114⋆○	0.236⋆○	0.798⋆○
WAVG+MMR	0.00226⋆○	6.318⋆○	0.151⋆○	**0.34**	0.868⋆
FAI	0.0053⋆○	6.199⋆○	0.142⋆○	0.28⋆○	0.873⋆
BEER(TS,SB)	0.00571	3.311⋆○	0.205⋆○	0.258⋆○	0.831⋆○
FuzzDA (no INF)	0.00559⋆	5.775⋆	0.128⋆○	0.24⋆○	0.826⋆○
EP-FuzzDA (no INF)	0.0054⋆○	5.796⋆	0.176⋆	0.252⋆	0.836⋆
(EP-)FuzzDA+GD (no INF)	0.00508⋆○	5.366⋆	0.142⋆○	0.263⋆	0.831⋆
(EP-)FuzzDA+TS (no INF)	0.00559⋆	5.775⋆	0.167⋆	0.243⋆○	0.817⋆○
(EP-)FuzzDA+CX (no INF)	0.00508⋆○	6.241⋆○	0.147⋆	0.252⋆○	0.836⋆
(EP-)FuzzDA+Prob	0.00612	3.867⋆○	0.827	0.284⋆○	0.876⋆○
(EP-)FuzzDA+RelDist	0.00626	3.76⋆○	0.749⋆○	0.278⋆○	0.873⋆○
(EP-)FuzzDA+Roulette	0.00465⋆○	3.015⋆	0.755⋆○	0.327⋆	**0.91**
FuzzDA+INF	**0.00629**	3.639⋆	**0.833**	0.28⋆	0.879⋆
EP-FuzzDA+INF	0.00574⋆○	4.333⋆○	0.744⋆○	0.284⋆	0.862⋆○
All base RS	0.00459○	0.286	0.223○	0.34	0.906
All base agg.	0.00571	1.513○	0.61○	0.34	0.885○
All (EP-)FuzzDA	0.00632	2.958○	0.833○	0.327○	0.91

$Nov_{ID}@5$ stands for iterative novelty at 5th user's iteration. GD, TS and CX stand for gradient descent, fuzzy Thompson sampling and contextualized votes assignments, respectively. Prob and RelDist denote probabilistic and relevance discount penalty models for implicit negative feedback (INF). Best results are depicted in bold, while best results per category (divided by horizontal line) are underlined. With "⋆" we denote significant differences (p-value < 0.05) between the best and other approaches w.r.t. paired t-test, while with "○" we denote significant differences w.r.t. best approach per category

provided the best incremental novelty scores as well as slightly better diversity scores than Relevance discount ones, while CTR results were mostly similar.

As for the variants of aggregation algorithm, results w.r.t. both datasets were rather contradictory for individual observed metrics. Also the ordering of FuzzDA vs. EP-FuzzDA algorithms changed considerably when INF was/was not included. Proper

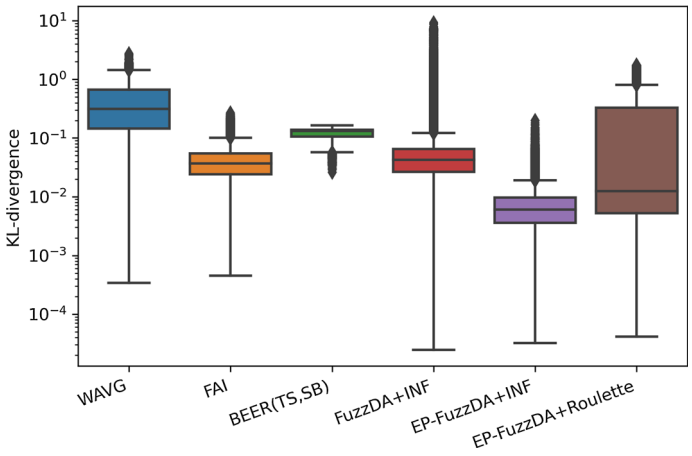

Fig. 9 Boxplot of KL-divergence between per-RS relevances as assigned by the aggregator and per-RS sums of recommended items' relevances on ML1M dataset. Note that y-axis is logarithmic

selection of the aggregation algorithm seems to depend on the particular parameters of the target website.

Evaluating fairness of FuzzDA framework

Before continuing to the detailed results, let us briefly report on the fairness of individual aggregators. Figure 9 depicts a boxplot of KL-divergence values (i.e., the fairness of base RS representation) for several variants of FuzzDA framework as well as selected baselines. Fairness of baseline aggregators varies greatly. Unsurprisingly, the lowest fairness was achieved by WAVG (KL-divergence of 0.45 in average). We assume that this was caused by the presence of systematic bias inherent for all item-wise approaches. BEER(TS,SB) achieved mean KL-divergence of 0.12. In this case, the main cause of elevated KL-divergence score was a simple fact that the RS selection procedure was not designed to be fair. Instead, BEER(TS,SB) tends to over-sample the best-performing RS (as compared to the proportion given by the estimated values of the beta distribution), which causes discrepancies in the distributions. The FAI algorithm was the best-performing baseline with mean KL-divergence of 0.042. We assume that the remaining discrepancies may be caused by the fact that FAI does not account for mutually preferred items and for varying preference intensities.

As for the FuzzDA variants, EP-FuzzDA outperformed FuzzDA approach almost by the order of magnitude (in average 0.007 vs. 0.056) as long as a fixed selection procedure is considered. It also significantly outperforms all other evaluated variants. Nonetheless, if a Roulette-based item selection procedure is employed, the fairness levels drop considerably (0.22 in average). Seemingly, the fairness-preserving selection process is rather fragile and any randomness responsible for the occasional selection of not so fair items hurts the targeted fairness beyond repair.

The overall evaluation revealed importance of considering the results in the context of more than just one metric. For instance, while high CTR values of some particular system indicate good level of short-term user goals fulfilment, reasonable values of incremental novelty and diversity may be needed to secure its long-term applicability.

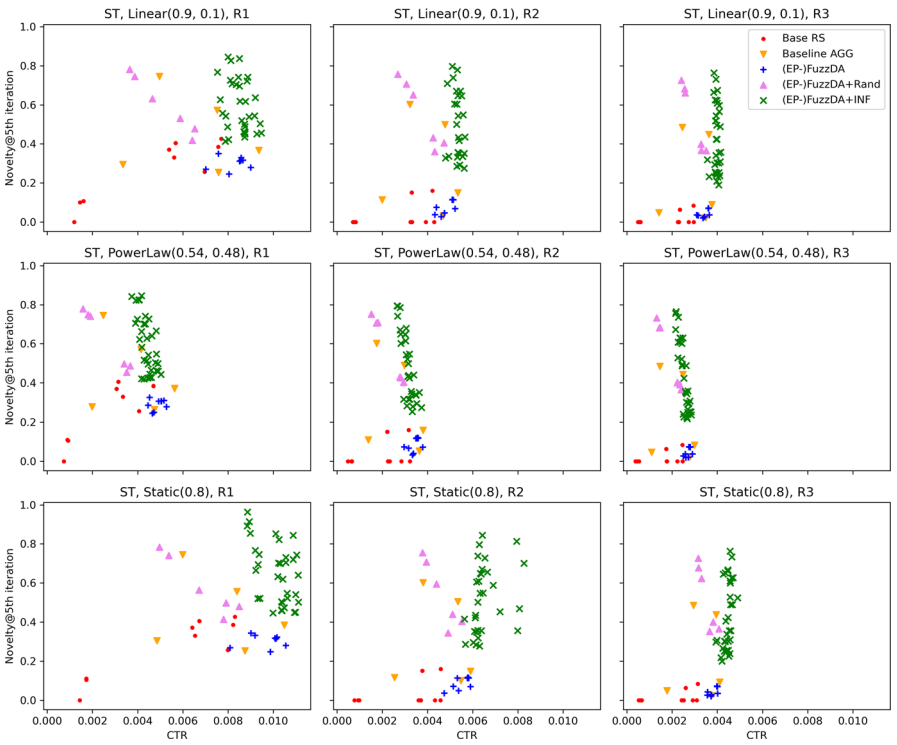

Fig. 10 Comparison of CTR with nov_{ID} at 5th iteration per user for all evaluated variants of ST dataset. Note that for the sake of comprehensibility, we only labeled individual results with corresponding groups, e.g., each red dot represents mean results of a single base RS in a respective evaluation scenario

On the other hand, several baseline algorithms provided significant improvements of some beyond-accuracy metric, but at the cost of large drops in CTR relevance score. Furthermore, individual evaluated scenarios may have a considerable effect on applicability of each strategy. Therefore, in the next several sections we focus on the interplay between evaluated metrics and on the effect of evaluation scenarios. Specifically, the effect of iterative novelty is observed in Sect. 5.1.2. Next, in Sect. 5.1.3 we focus on several variants of per-user diversity and on the effect of popularity bias in Sect. 5.1.4. Finally, in Sect. 5.1.5, we aim on the comparison of individual variants of FuzzDA framework w.r.t. different evaluation settings.

5.1.2 Iterative novelty

Figures 10 and 11 depict the interplay between the accuracy of RS (CTR) and iterative novelty of recommended items (nov_{ID}) for ST and ML1M datasets, respectively. Please note that in these figures (as well as in the next sections), we only reference individual approaches with a corresponding group they belong to (e.g., base RS, baseline aggregator, etc.). This is mainly for the sake of comprehensibility and also because we

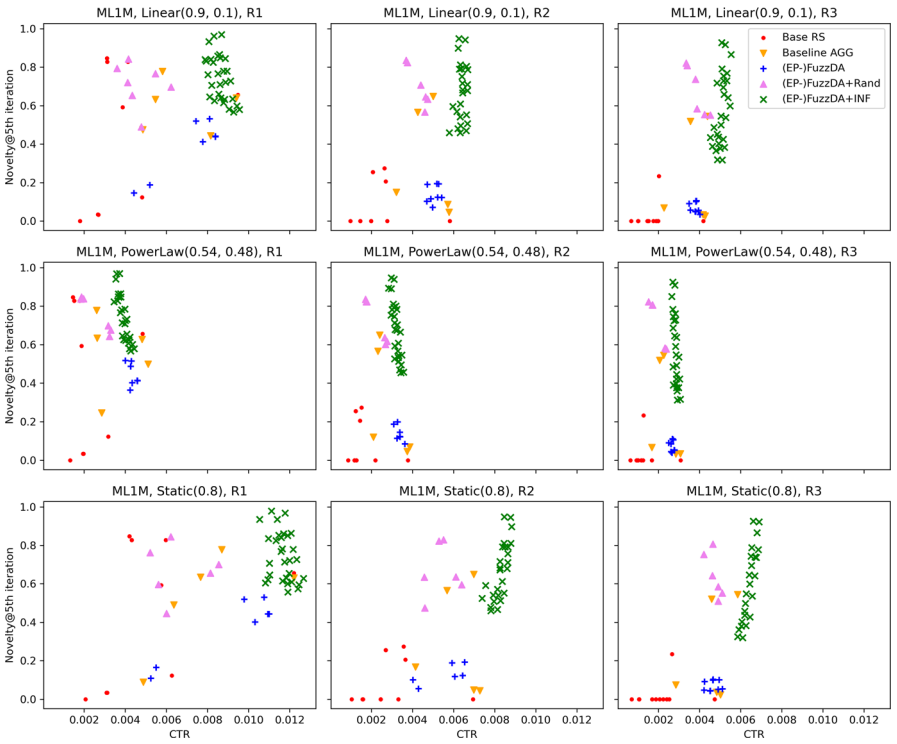

Fig. 11 Comparison of CTR with nov_{ID} at 5th iteration per user for all evaluated variants of ML1M dataset

want to focus on overall trends in the results. Nonetheless, if some particular approach performs exceptionally, it is mentioned in the text.

For most of the evaluation scenarios, FuzzDA approaches with implicit negative feedback penalization (INF) performed on or close to the Pareto front.[24] Various INF strategies have a considerable impact on the levels of iterative novelty. For *power-law* user's noticeability model, introducing more novelty via INF resulted in a slight decrease of CTR performance, even in $R = 2$ and $R = 3$ scenarios. This is quite understandable as this model assumes that there is a very high average chance that the user does not notice recommended items. In other two noticeability models, INF was able to improve novelty with only minimal CTR penalties, or even simultaneously increase both novelty and CTR (*static* models with $R = 2$ and $R = 3$ and *linear* model with $R = 3$ for ML1M dataset).

In ST dataset, most of the FuzzDA variants outperformed all base RS in terms of CTR for all evaluation scenarios. In ML1M dataset, one base RS (*iKNN*) significantly outperformed all other base RS in terms of CTR. This poses some difficulties for RS aggregators to adjust. BEER(TS,SB) algorithm was able to quickly adapt to the situation and mainly follow *iKNN* recommendations. Therefore, its performance is

[24] I.e., there are no other approaches that would increase the results w.r.t. one of the metrics, while the other would not be decreased. For more detailed description of Pareto front see https://en.wikipedia.org/wiki/Pareto_efficiency.

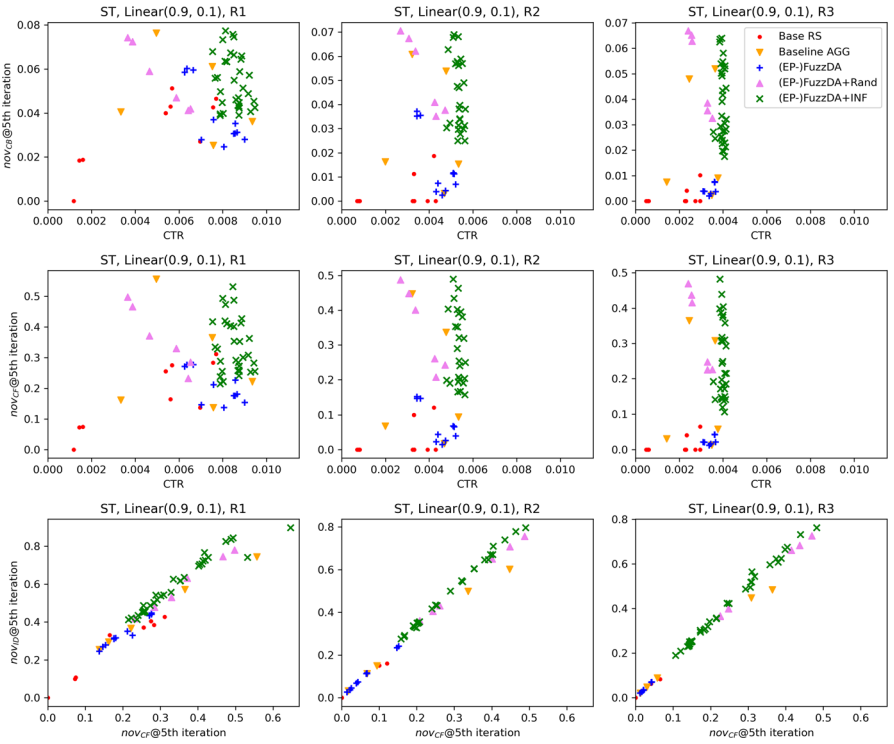

Fig. 12 Comparison of nov_{CF} and nov_{CB} with CTR and nov_{ID} at ST dataset. Only the results of *linear* user model are depicted

close (yet inferior) to *iKNN* in $R = 1$. Other RS aggregators (including variants of FuzzDA) are mostly rendered between $iKNN$ and other base RS. Nonetheless, the situation changed for $R = 2$ and 3.

Results of other two novelty metrics (nov_{CB} and nov_{CF}) closely resemble those of nov_{ID} as can be seen in Fig. 12. Seemingly, results of both metrics were overwhelmed by a simple presence of the same items in the previous iterations (resulting into zero novelty), so the presence of similar items in previous iterations (resulting into small, but nonzero novelty) does not change the ordering of items much. The last row of Figs. 12 and 13 shows the comparison of nov_{CB} and nov_{CF} for linear evaluation scheme on ST and ML1M datasets, respectively (similar results were obtained also for remaining ST and ML1M evaluation scenarios). Also, the results of per-user *relative coverage* were highly similar to those of iterative novelties, so we omit them as well.

Let us now focus a bit more on the development of nov_{ID} metric between iterations and on the effect of various INF penalization strategies on it. Figure 14 depicts results of FuzzDA with context-aware votes assignments together with several baselines.

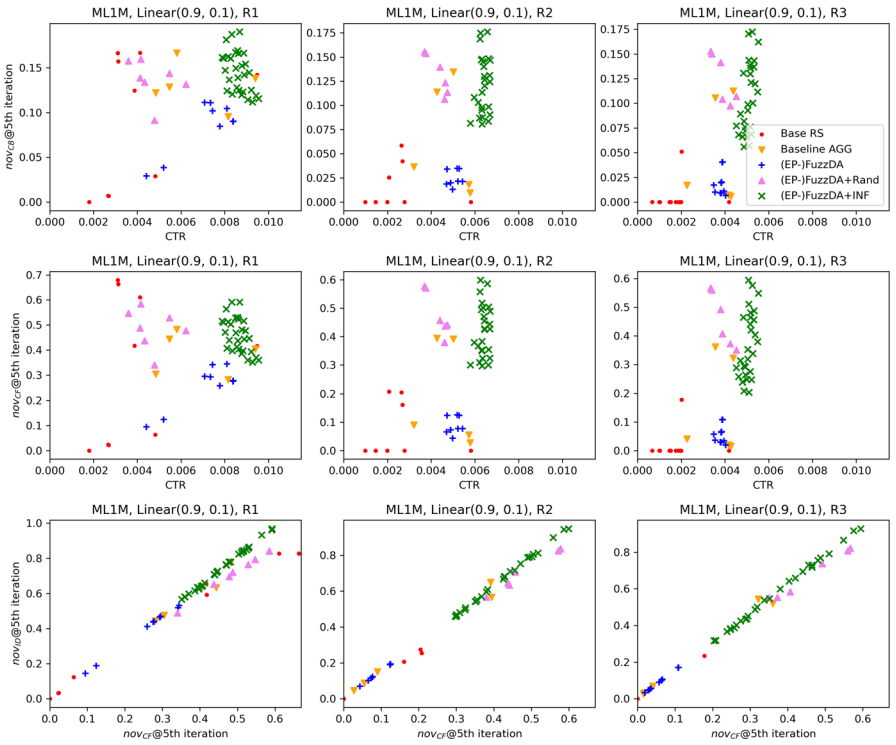

Fig. 13 Comparison of nov_{CF} and nov_{CB} with CTR and nov_{ID} at ML1M dataset. Only the results of *linear* user model are depicted

INF penalization clearly increases the iterative novelty scores, where mean penalty score has a significant impact on the level of increase. With the same levels of mps, probabilistic penalization provides slightly more novelty than relevance discounts, especially for a first couple of user's interactions. The novelty of all evaluated approaches gradually drops with additional interactions. This is not so surprising as the volume of potentially relevant items is not unlimited. Nonetheless, all INF penalization variants consistently increased the iterative novelty as compared to the FuzzDA baseline.

Methods with high mps experience a sudden drop around 6th iteration. This is supposedly an effect of the utilized length of history, which considers negative feedback obtained within last-5 iterations. From 6th iteration onward, already recommended items may tend to regularly re-appear in the recommendations results, which naturally hurts the iterative novelty metric. This behavior corresponds to the assumed level of user's preference stability as discussed in Sect. 2.5 and can be changed by simply considering longer or shorter feedback history if necessary.

The "zigzag" shape curves for $R = 2$ and $R = 3$ are caused by repeated requests for recommendation with the same user profile. I.e., recommenders are forced to

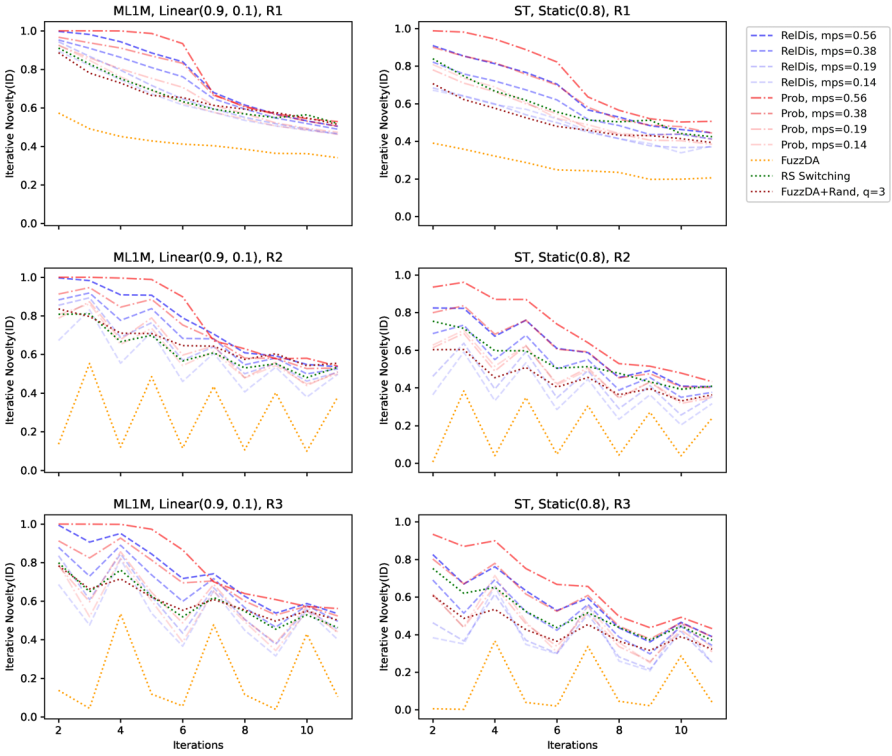

Fig. 14 Development of iterative novelty w.r.t. volume of recommendation requests per user. $RelDis$ stands for relevance discounts penalization, $Prob$ stands for probabilistic penalization and mps stands for mean penalty score (Eq. 14). For RelDis and Prob results, alpha color channel corresponds to the mps values; fully opaque line would correspond to $mps = 1$, fully transparent to $mps = 0$

recommend again for the same user profile, which often causes significant drops in the novelty (up to zero novelty for most base RS). Methods with higher mps can considerably reduce this novelty drop. For later user's interactions (5+), some additional baselines (*RandKfromN* or *FuzzDA+Rand* with $q = 1$) provided higher novelty than evaluated INF penalization variants. Nonetheless, these baselines perform inferior w.r.t. CTR by a large margin.

To conclude, variants of FuzzDA framework with INF provide the best trade-off between CTR and iterative novelty metrics. Especially the probabilistic INF model with higher mps can maintain high levels of iterative novelty for the whole period of supposedly stable user's preferences. As long as items' noticeability and users' preference stability are correctly inferred, employment of INF models can have beneficial effects on overall relevance of recommendations as well.

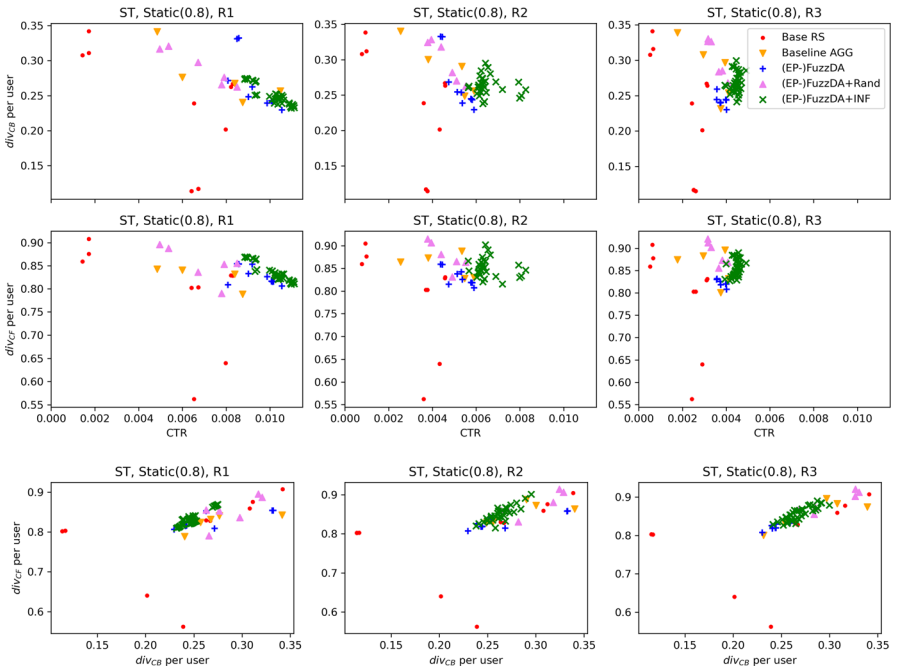

Fig. 15 Comparison of per-user diversities div_{CF} and div_{CB} with CTR on ST dataset. Only the results of *static* user model are depicted

5.1.3 Per-user diversity

Let us now focus on the diversity metrics. Figures 15 and 16 depict results w.r.t. mean per-user CB and CF diversities on ST and ML1M datasets, respectively. Again, FuzzDA variants with INF are on or close to the Pareto front in the most cases. However, unlike the novelty results, INF does not distribute results along the diversity axis too much, so there are other approaches offering more diverse results at the price of decreased relevance. FuzzDA algorithms without INF already provide reasonable levels of diversity (as compared to their results w.r.t. novelty) and INF does not provide so large improvements. For RS aggregators, CF and CB diversity scores are quite correlated (see the last rows of Figs. 15 and 16); however, there are considerable discrepancies in base RS's performance. This observation comes naturally from the fact that some collaborative or content-based paradigm is employed by each base RS, which makes it more susceptible to lower diversity w.r.t. this aspect. RS aggregators on the other hand can mitigate this effect by merging multiple base RS. This usually led to slightly lower diversities than the best base RS, but kept reasonable diversity levels for both metrics. Also note that the evaluated aggregators (except for WAVG+MMR) did not explicitly consider any diversity enhancement post-processing such as MMR (Carbonell and Goldstein 1998) or xQuad (Santos et al. 2010). Therefore, the achieved diversity levels can be attributed solely to the aggregation procedure and can be further increased via post-processing if necessary.

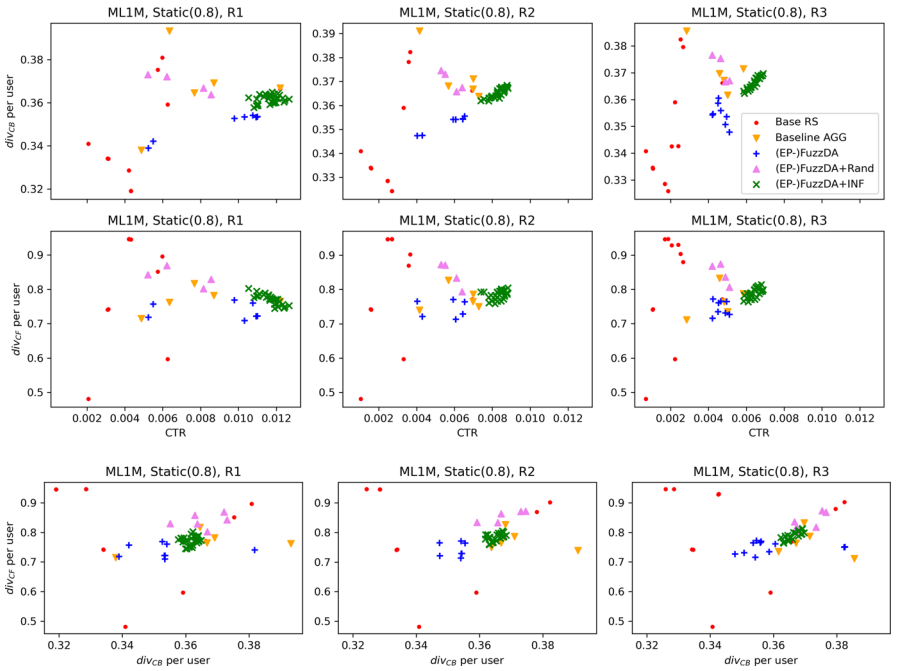

Fig. 16 Comparison of per-user diversities div_{CF} and div_{CB} with CTR on ML1M dataset. Only the results of *static* user model are depicted

As for the variants of the aggregation algorithm, EP-FuzzDA provided slightly higher per-user diversity values as compared to FuzzDA (in average 0.263 vs. 0.248, p-value: 3.0e–8 for div_{CB} and 0.841 vs 0.833, p-value: 0.03 for div_{CF}) on ST dataset. The trade-off was a slight decrease of CTR (0.00496 vs. 0.00524), which was, however, not statistically significant. Similar results were obtained for ML1M as well. This may be an effect of the fact that EP-FuzzDA provides more proportional representation of base RS, which in turn increases the diversity metrics.

Figures 17 and 18 depict results w.r.t. selection-wise diversity (i.e., the diversity of per-user clicked items) for ST and ML1M datasets, respectively.[25] We only depict the results of *static* model; the results were quite similar for other noticeability models as well.

Notably, diversity of user's selections was considerably higher for RS aggregators than for individual base RS (except for div_{CF} on ML1M dataset). This may be one argument for using the RS aggregation as an approach for penetrating the filter bubble. FuzzDA+INF variants secured the highest values of CF diversity of selected items on ST dataset and CB diversity of selected items on ML1M dataset. Interestingly, the dependence between collaborative diversity and CTR on ST dataset was almost linear similarly as CB diversity on ML1M dataset. However, results of the other two pairs were much more spread out. We assume this might be an artefact of recommenders utilized during the data collection process (content-based in ST dataset and collabora-

[25] Note that we only included results for users with 2+ clicked items.

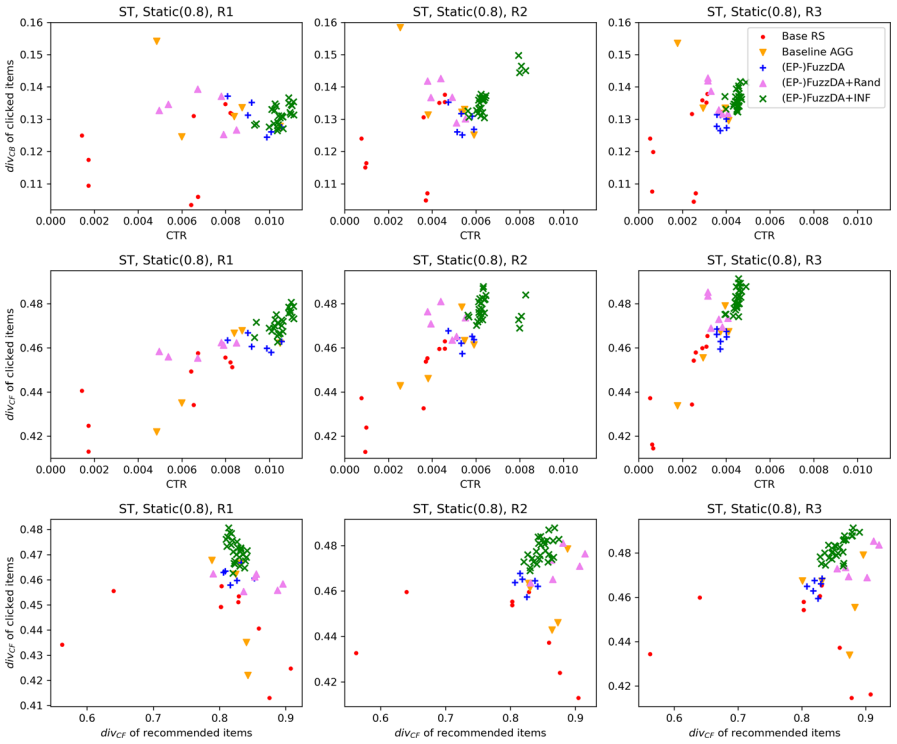

Fig. 17 Diversity of user's selection vs. CTR and diversity of recommendations on ST dataset. Only the results of *static* user model are depicted

tive in ML1M), but we plan to analyze this phenomenon in more detail as a part of our future work. Also note that the relation between selection-wise and recommendation-wise diversity is not quite linear (see last rows of Figs. 17 and 18). Therefore, it seems important to observe both metrics to determine the effect of RS on phenomena such as filter bubbles.

To conclude, variants of FuzzDA—although not directly optimized for diversity—provided relatively high values of all evaluated diversity metrics with the exception of selection-based collaborative diversity on ML1M dataset. We suspect that in this case, low diversities may be caused by the biased feedback acquirement process inherent for ML1M. In other words, if most of the selected items are similar w.r.t. CF, then elevating CTR would result in decreased CF diversity because there is simply not enough diversity in the test set data. INF strategies as well as Roulette-based selections can usually further increase the diversity of FuzzDA approaches especially for larger R. However, for Roulette-based selections this comes with the price of lower CTR. Thanks to the high values of CTR, iterative novelty and per-user diversities, FuzzDA+INF variants have good preconditions to maintain the long-term relevance of recommendations.

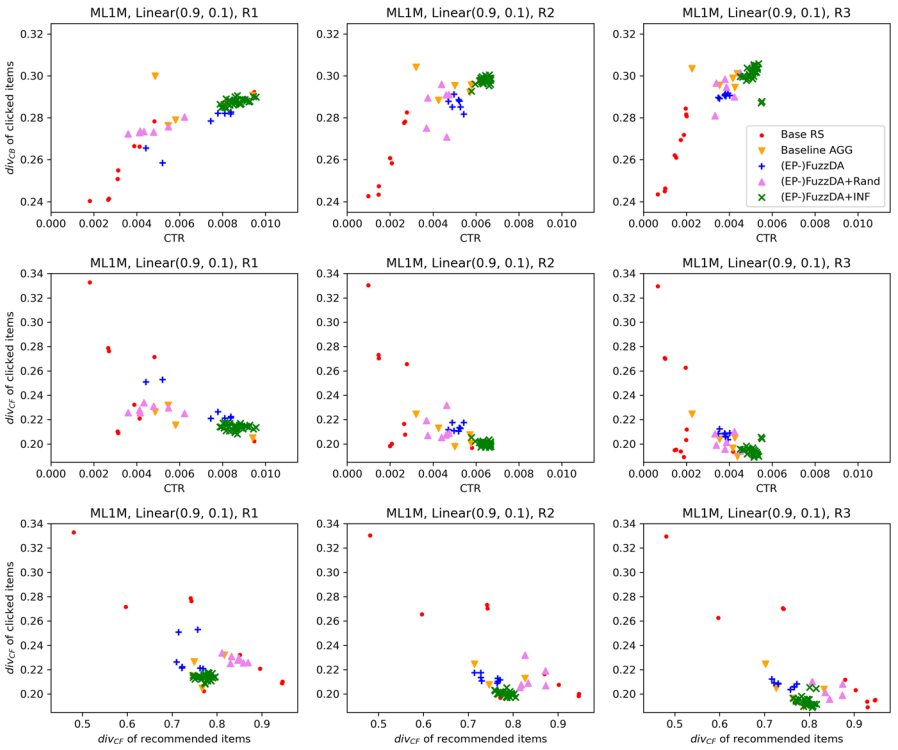

Fig. 18 Diversity of user's selection vs. CTR and diversity of recommendations on ML1M dataset. Only the results of *static* user model are depicted

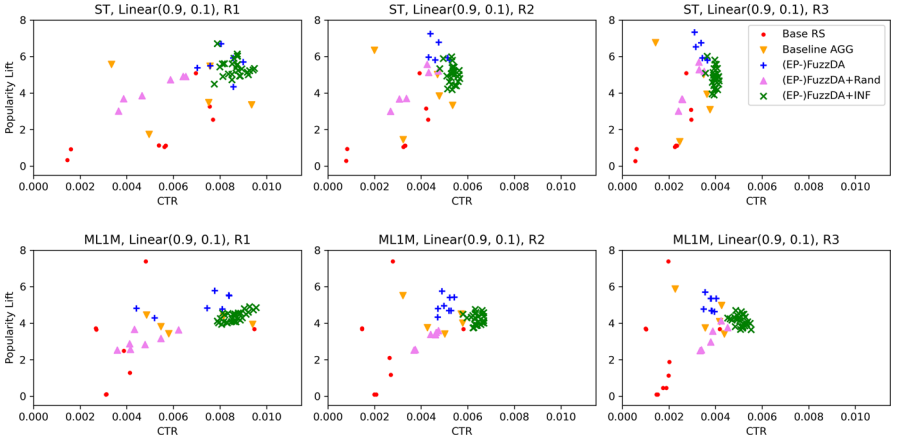

Fig. 19 Popularity Lift vs. CTR on ST and ML1M dataset. Only the results of *linear* user model are depicted

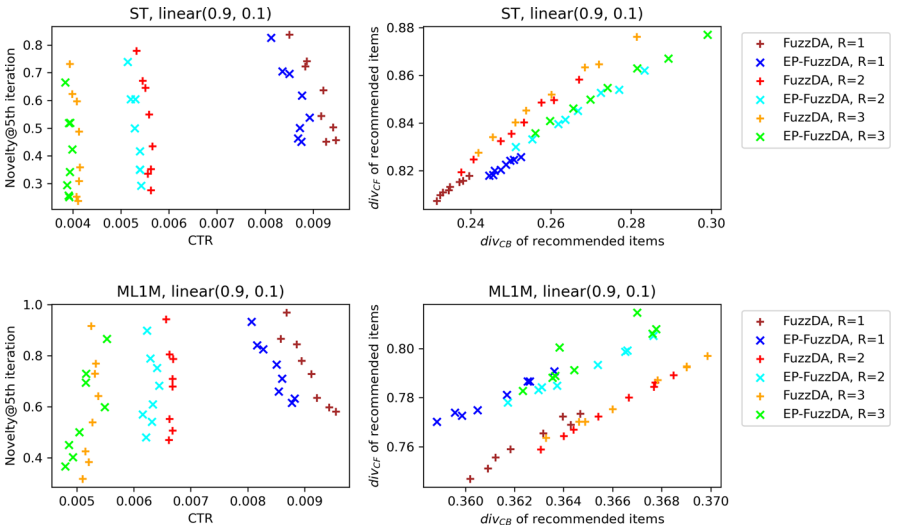

Fig. 20 Comparison of FuzzDA and EP-FuzzDA algorithm variants with INF penalties. Novelty vs. CTR trade-off is depicted in the left, while the trade-off between CF and CB diversities is in right

5.1.4 Popularity bias

Finally, we evaluated the approaches w.r.t. popularity bias. Figure 19 depicts results of Popularity lift compared with CTR. FuzzDA variants obtained relatively high values similarly as other RS aggregators. Yet still, the popularity lift values of FuzzDA or, e.g., BEER(TS,SB) were considerably lower than those of the worst base RS. As such, decreasing popularity bias of some (relevant but highly biased) RS via its aggregation with other, less biased RS is a plausible strategy to some extent. Nonetheless, if this direction is pursued, FuzzDA variants does not provide any special advantage w.r.t. popularity lift results over other aggregation strategies.

By utilizing INF penalization and especially Roulette-based items selection, the popularity lift can be decreased to some extent (especially for $R = 2$ and $R = 3$), but several base RS and baseline aggregators still received lower popularity lift scores (at the cost of lower CTR). In general, high values of CTR were almost exclusively achieved by approaches with high Popularity lift. It seems that in order to decrease the popularity bias of RS, some sacrifices to the overall recommendation relevance have to be made.

5.1.5 Comparing variants of FuzzDA algorithm

We also focused on a more detailed evaluation of individual variants of the FuzzDA framework. We found considerable differences between the performance of FuzzDA and EP-FuzzDA aggregation algorithms with TS and no INF penalties. Specifically, on ML1M dataset EP-FuzzDA received higher iterative novelty (0.275 vs. 0.202, paired t-test p-value: 2.3e–6), higher CF diversity (0.763 vs. 0.724, p-value: 1.2e–10), higher CB diversity (0.355 vs. 0.353, p-value: 0.022) and lower popularity lift

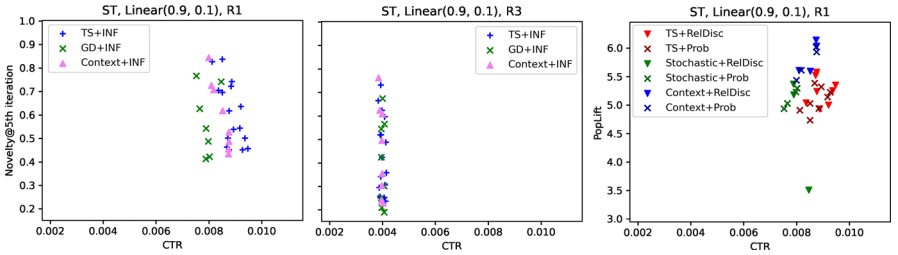

Fig. 21 Comparison of gradient descent (GD), fuzzy Thompson sampling (TS) and contextualized votes assignment strategies. Variants with INF penalization are displayed

(4.76 vs. 5.498, p-value: 2.3e–11) at the cost of slightly lower CTR (0.00552 vs. 0.00563, p-value: 0.04). Similar results (except for the popularity lift) were obtained also for ST dataset. While comparing FuzzDA vs. EP-FuzzDA with INF strategies (see Fig. 20), results slightly differ. Notably, FuzzDA outperforms EP-FuzzDA w.r.t. CTR and incremental novelty for all R values. On the other hand, EP-FuzzDA outperforms FuzzDA w.r.t. both CB and CT diversity of recommendations in most cases. (FuzzDA was slightly better w.r.t. div_{CB} on some variants of ML1M evaluations.) Therefore, the final selection of aggregation algorithm should be done based on the relative importance of these metrics in particular scenarios.

As for the votes assignment strategies, without INF penalization, all three variants performed similarly w.r.t. all diversity metrics. GD variant achieved considerably lower iterative novelty in ML1M dataset: in average 0.127 for GD compared to 0.239 for TS (paired t-test p-value: 0.002) and 0.228 for CX (p-value: 0.003). On ST dataset, CX and GD achieved similar novelty, while TS dominating both of them. CX variant received higher average popularity lift score in both datasets. On ML1M dataset, popularity lift for CX was 5.28, while for it TS was 5.13 (p-value: 0.0003) and for GD it was 4.79 (p-value: 3.4e–5). On ST dataset, popularity lift for CX was 6.622, while for TS it was 5.848 (p-value: 3.7e–5) and for GD it was 5.673 (p-value: 4.3e–5). These results seems unaffected by the introduction of INF penalties as can be seen in Fig. 21 (right). While evaluating variants with INF penalties, we observed rather larger differences w.r.t. CTR for $R = 1$ scenarios (Fig. 21, left). The best results were obtained by TS followed by CX and GD approaches. Nonetheless, the performance difference tends to vanish for larger R values (Fig. 21, center).

Finally, we focused on the variants of INF penalization. Figure 22 depicts dependencies between various diversity metrics and INF penalization strategies on ST dataset. It can be seen that in general, probabilistic penalization schemes achieve higher values of all evaluated diversity metrics. Similar results were obtained for iterative novelty as well (average nov_{ID}@5 of 0.536 for $Prob$, while 0.469 for $RelDist$ on ST dataset, paired t-test p-value: 8.7e–17). Probabilistic penalization also reduced the popularity lift more (5.017 vs. 5.217 on ST dataset, p-value: 1.0e–9) and provided higher values of CF diversity (0.848 vs. 0.839 on ST dataset, p-value: 6.1e–32) and CB diversity (0.261 vs. 0.253 on ST dataset, p-value: 3.9e–28). Again, those improvements come with a price of a slight decrease in CTR (0.00548 vs 0.00553 on ST dataset, p-value: 0.03). Given the favorable trade-off between decreased CTR and increased beyond

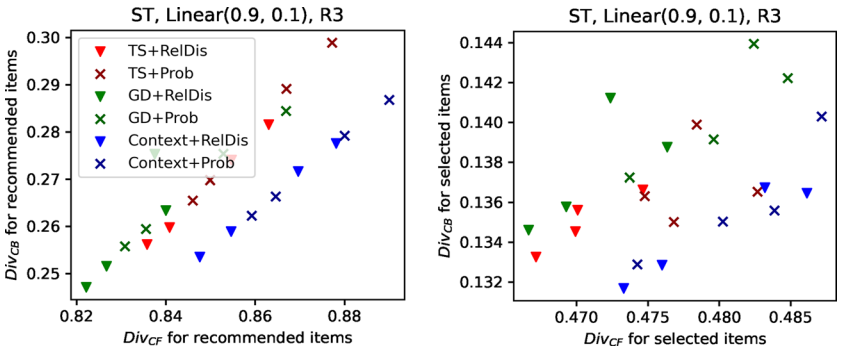

Fig. 22 Comparison Relevance discount (RelDis) and Probabilistic (Prob) penalization strategies on ST dataset. Variants of votes assignment strategies are displayed as well

accuracy metrics, we can recommend usage of $Prob$ strategy instead of $RelDist$ for most scenarios.

To conclude the offline evaluation section, variants of FuzzDA framework in general provided exceptional performance w.r.t. CTR and incremental novelty, especially in scenarios with larger R values. They also provided favorable results for a range of per-user diversity metrics. Results of popularity bias were rather mediocre for variants of FuzzDA as compared with other baseline aggregators. Therefore, if the per-item fairness is a key requirement of some particular domain (e.g., in multiple vendor scenarios), additional strategies mitigating popularity bias should be employed (Abdollahpouri 2019). Other than this, FuzzDA variants provide a favorable mixture of features for maintaining good long-term RS performance. Selection of a particular variant of FuzzDA should be considered w.r.t. the estimated features of the target domain. Selection of INF penalization strategy has a determining effect in situations with larger R values, while other components affect especially $R = 1$ results.

5.2 Online results

In order to verify the observations of offline simulations, we also evaluated some of the proposed methods in an online setting. Specifically, we conducted a between-subject A/B-testing, i.e., each user was assigned one RS variant at random.

In total, we evaluated five recommending strategies. As a baseline, we selected the best-performing base RS from offline scenario (a variant of Word2vec). We also included two methods evaluated in the preliminary study (Peška and Balcar 2019): BEER(TS,SB) and FuzzDA with GD-based votes assignments and no INF adjustments. Finally, two variants of EP-FuzzDA were evaluated: one with TS-based and the other with CX-based votes assignments. Both EP-FuzzDA variants also employed a probabilistic penalization with linear models for $noticed(k)$ and $relevant(t)$ functions. For $noticed(k)$, the max and min values were set at 0.5 and 0.0, respectively, and for $relevant(t)$ the max and min values were set at 1.0 and 0.5, respectively. The penalization values were selected at rather lower scale ($mps = 0.19$) due to the fact that recommendations are usually displayed on the edge of initially visible area in ST website.

Table 5 Results of online evaluations on ST website

Algorithm	CTR	(p-val)	PopLift	$nov_{ID}@5$	div_{CF}	div_{CB}
Word2vec	0.00171	0.0001	**1.081**	0.15*	0.312*	0.103*
BEER(TS,SB)	0.00231	0.12	1.839*	**0.549**	0.515*	0.17*
FuzzDA; GD, no INF	0.00244	0.28	10.858*	0.107*	0.823*	0.288*
EP-FuzzDA; TS, Prob	0.00228	0.13	6.401*	0.391*	0.829*	0.299*
EP-FuzzDA; CX, Prob	**0.00283**		6.691*	0.356*	**0.831**	**0.302**

$Nov_{ID}@5$ stands for iterative novelty at 5th user's iteration. GD, TS and CX stand for gradient descent, fuzzy Thompson sampling and contextualized votes assignments, respectively. Prob denotes probabilistic penalty model for INF. Best results are depicted in bold. For CTR, p-values for significance comparison to the best approach w.r.t. χ^2 test are depicted in a separate column. For other metrics, "*" denote significant differences (p-value < 0.01) between the best and other approaches w.r.t. t-test

The experiment was conducted for a period of approx. one and half month starting at mid-January, 2021. In total, over 250K recommendations were displayed to users during the experiment. Table 5 contains results of the online evaluations. The best results w.r.t. CTR were obtained by EP-FuzzDA with contextualized votes assignments. Nonetheless, the difference of individual approaches w.r.t. CTR was mostly above the usually considered significance levels (see the second column of Table 5). Notably, we cannot corroborate that EP-FuzzDA with contextualized votes outperformed FuzzDA with GD and no INF (p-value: 0.28) w.r.t. CTR. On the other hand, plain FuzzDA was considerably inferior w.r.t. several beyond-accuracy metrics, which lets us to hypothesize that its performance may degrade in the long term. Nonetheless, further evaluations are needed to corroborate this. The results also corroborated several findings of the offline simulations. INF penalization considerably decreased the popularity bias for FuzzDA variants. Base RS and FuzzDA algorithms tend to receive relatively low iterative novelty scores, which was improved by the usage of INF penalization as well. All three variants of FuzzDA algorithm provided highly diverse results (both CF and CB) and considerably outperform their counterparts w.r.t. these metrics.

Rather surprising was a relatively large difference in the CTR between both EP-FuzzDA variants as they only differ in the votes assignment strategy.[26] Note that both methods performed highly similar w.r.t. average novelty, diversity and popularity bias metrics. However, by analyzing the votes assignment logs, we found that both algorithms tend to prefer rather different base RS. Most popular and BPR MF recommendations were often utilized by EP-FuzzDA+TS. The votes assignment of EP-FuzzDA+CX were quite diverse even in the later stages of the experiment (which is natural, given the contextual dependence of votes), but variants of Cosine CB received higher votes quite often—see Fig. 23. In our logs, higher values for cosine CB correlated with lower values for BPR MF variants and Most Popular algorithm. Seemingly, EP-FuzzDA+CX created at least two voting profiles based on the received context. It is possible that although the overall parameters of both EP-FuzzDA+CX and EP-

[26] In the terms of absolute difference. In the following text, we assume that the p-value of 0.13 is sufficient to derive tentative conclusions.

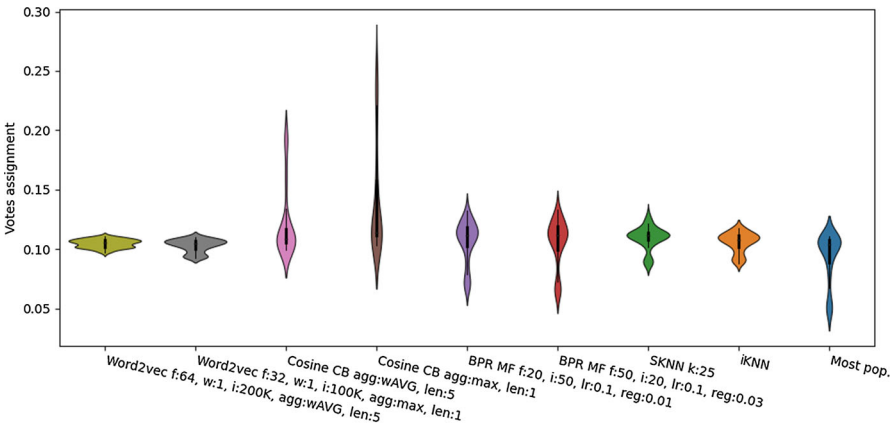

Fig. 23 Assigned votes distribution for EP-FuzzDA+CX during online evaluation. Note that first 20% of records were removed as votes tend to fluctuate a lot during the early learning stages

FuzzDA+TS were similar, the capability of context-aware partitioning was the decisive factor for higher conversions rate of the CX variant. Similar patterns were also observed in the offline simulation logs of several CX variants, but with lower intensity.

6 Conclusions and outlook

In this paper, we proposed several variants of FuzzDA framework for proportional aggregation of recommender system's results under the constraints of dynamic RS. The framework consists from an aggregator, a votes assignment strategy and a model for negative implicit feedback incorporation. We proposed several options for each of framework's components.

The FuzzDA framework was extensively evaluated both by offline simulations and in online A/B testing. In offline simulations, we focused on the role of repeated requests for recommendation, varying noticeability of objects and on an interplay between various evaluation metrics. Variants of the proposed framework secured a reasonable combination of evaluated metrics, often placed on or close to the Pareto front. In the online evaluation, a variant of EP-FuzzDA with contextualized votes assignment strategy received the highest CTR scores as well as highest per-user diversities.

There are several topics that we left for the future work. Some tasks are rather minor, such as exploring the potential of ranked discounts for EP-rel-sum criterion or determining the effect of user's preference stability on the recommendation results. Somewhat more challenging are adjustments of the user noticeability models. For example, by monitoring additional feedback (scrolling, mouse movements) and page layout, we can improve the estimation of whether the objects were actually noticed by the user. While this is doable in the online settings (Peska and Vojtas 2017), models of noticeability capable to process such information have to be proposed. Also, it is currently unknown how to transfer this knowledge into the offline simulations.

Another direction would be to estimate the ability of individual objects to "catch" user's attention.

Another rather complex topic is centered around the question of how to perform offline evaluation correctly. While studying the online results, we were surprised by the differences in the performance of two EP-FuzzDA variants, which only differ in the applied votes assignment strategy. We tracked the probable cause to the capability of one variant to adjust for the user's current context, especially the type of page the user is currently visiting. This is a feature we are currently unable to fully reproduce in offline simulations as it also incorporates a different model of user's selection behavior. Being able to automatically adjust the mixture of base RS according to the current context may be an important step towards unified self-adjusting recommending ecosystems. Nonetheless, inability to properly evaluate such systems offline may lead the research astray. Therefore, bridging this gap may be one of the most important challenges for our future work.

Acknowledgements Source codes of FuzzDA framework are available from https://github.com/sbalcar/HeterRecomPortfolio.

References

Abdollahpouri, H.: Popularity bias in ranking and recommendation. In: Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society, AIES '19, pp. 529–530. Association for Computing Machinery, New York (2019)

Abdollahpouri, H., Burke, R.: Reducing popularity bias in recommendation over time. arXiv:1906.11711 (2019)

Abdollahpouri, H., Mansoury, M., Burke, R., Mobasher, B.: The connection between popularity bias, calibration, and fairness in recommendation. In: Fourteenth ACM Conference on Recommender Systems. RecSys '20, pp. 726–731. Association for Computing Machinery, New York (2020)

Anelli, V.W., Bellini, V., Di Noia, T., La Bruna, W., Tomeo, P., Di Sciascio, E.: An analysis on time- and session-aware diversification in recommender systems. In: Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization, UMAP '17, pp. 270–274. Association for Computing Machinery, New York (2017)

Balcar, S., Peska, L.: Personalized implicit negative feedback enhancements for fuzzy d'hondt's recommendation aggregations. In: Proceedings of the 22nd International Conference on Information Integration and Web-Based Applications and Services, iiWAS '20, pp. 210–215. Association for Computing Machinery, New York (2020)

Barkan, O., Koenigstein, N.: Item2vec: neural item embedding for collaborative filtering. arXiv:1603.04259 (2016)

Beliakov, G., Calvo, T., James, S.: Aggregation Functions for Recommender Systems, pp. 777–808. Springer, Boston (2015)

Bertani, R.M., Bianchi, A.C.R., Costa, A.H.R.: Combining novelty and popularity on personalised recommendations via user profile learning. Expert Syst. Appl. **146**, 113149 (2020)

Brodén, B., Hammar, M., Nilsson, B.J., Paraschakis, D.: Ensemble recommendations via thompson sampling: an experimental study within e-commerce. In: IUI '18, pp. 19–29. ACM (2018)
Cao, H., Chen, E., Yang, J., Xiong, H.: Enhancing recommender systems under volatile user interest drifts. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM '09, pp. 1257–1266. Association for Computing Machinery, New York (2009)
Carbonell, J., Goldstein, J.: The use of mmr, diversity-based reranking for reordering documents and producing summaries. In: SIGIR '98, pp. 335–336. ACM, New York (1998)
Dang, V., Croft, W.B.: Diversity by proportionality: An election-based approach to search result diversification. In: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '12, pp. 65–74. Association for Computing Machinery, New York (2010). https://doi.org/10.1145/2348283.2348296
D'Hondt, V.: Système pratique et raisonné de représentation proportionnelle. C. Muquardt (1882)
Elahi, M., Abdollahpouri, H., Mansoury, M., Torkamaan, H.: Beyond algorithmic fairness in recommender systems. In: Adjunct Proceedings of the 29th ACM Conference on User Modeling, Adaptation and Personalization, UMAP '21, pp. 41–46. Association for Computing Machinery, New York (2021). https://doi.org/10.1145/3450614.3461685
Eskandanian, F., Mobasher, B.: Detecting changes in user preferences using hidden markov models for sequential recommendation tasks. arXiv:1810.00272 (2018)
Esmeli, R., Bader-El-Den, M., Abdullahi, H.: Improving session based recommendation by diversity awareness. In: Advances in Computational Intelligence Systems, pp. 319–330. Springer, Cham (2020)
Felfernig, A., Boratto, L., Stettinger, M., Tkalčič, M.: Group recommender systems: an introduction. Springer (2018). https://doi.org/10.1007/978-3-319-75067-5
Fleder, D., Hosanagar, K.: Blockbuster culture's next rise or fall: the impact of recommender systems on sales diversity. Manage. Sci. **55**(5), 697–712 (2009)
Frigó, E., Pálovics, R., Kelen, D., Kocsis, L., Benczúr, A.: Online ranking prediction in non-stationary environments. In: 1st Workshop on Temporal Reasoning in Recommender Systems, RecTemp 2017, pp. 28–34. CEUR-WS.org (2017)
Garcin, F., Faltings, B., Jurca, R., Joswig, N.: Rating aggregation in collaborative filtering systems. In: Proceedings of the Third ACM Conference on Recommender Systems, RecSys '09, pp. 349–352. Association for Computing Machinery, New York (2009)
Ge, Y., Zhao, S., Zhou, H., Pei, C., Sun, F., Ou, W., Zhang, Y.: Understanding Echo Chambers in E-Commerce Recommender Systems, pp. 2261–2270. SIGIR '20. Association for Computing Machinery, New York (2020)
Granka, L., Feusner, M., Lorigo, L.: Eye Monitoring in Online Search, pp. 347–372. Springer, Berlin (2008). https://doi.org/10.1007/978-3-540-75412-1_16
Harper, F.M., Konstan, J.A.: The movielens datasets: history and context. ACM Trans. Interact. Intell. Syst. **5**(4), 1–9 (2015)
Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D.: Session-based recommendations with recurrent neural networks. In: 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2–4, 2016, Conference Track Proceedings. arXiv:1511.06939 (2016)
Hosanagar, K., Fleder, D., Lee, D., Buja, A.: Will the global village fracture into tribes? Recommender systems and their effects on consumer fragmentation. Manag. Sci. **60**(4), 805–823 (2014)
Jannach, D., Ludewig, M., Lerche, L.: Session-based item recommendation in e-commerce: on short-term intents, reminders, trends and discounts. User Model. User-Adapt. Inter. **27**(3), 351–392 (2017)
Joachims, T., Granka, L., Pan, B., Hembrooke, H., Radlinski, F., Gay, G.: Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. ACM Trans. Inf. Syst. (2007). https://doi.org/10.1145/1229179.1229181
Kaminskas, M., Bridge, D.: Diversity, serendipity, novelty, and coverage: a survey and empirical analysis of beyond-accuracy objectives in recommender systems. ACM Trans. Interact. Intell. Syst. **7**(1), 1–42 (2016)
Kaya, M., Bridge, D.: A comparison of calibrated and intent-aware recommendations. In: Proceedings of the 13th ACM Conference on Recommender Systems, RecSys '19, pp. 151–159. Association for Computing Machinery, New York (2019)
Kaya, M., Bridge, D., Tintarev, N.: Ensuring fairness in group recommendations by rank-sensitive balancing of relevance. In: Fourteenth ACM Conference on Recommender Systems. RecSys '20, pp. 101–110. Association for Computing Machinery, New York (2020)

Koren, Y.: Collaborative filtering with temporal dynamics. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '09, pp. 447–456. Association for Computing Machinery, New York (2009)
Kotkov, D., Wang, S., Veijalainen, J.: A survey of serendipity in recommender systems. Knowl.-Based Syst. **111**, 180–192 (2016)
Lathia, N., Hailes, S., Capra, L., Amatriain, X.: Temporal diversity in recommender systems. In: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '10, pp. 210–217. Association for Computing Machinery, New York (2010)
Lerche, L., Jannach, D., Ludewig, M.: On the value of reminders within e-commerce recommendations. In: Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization, UMAP '16, pp. 27–35. Association for Computing Machinery, New York (2016)
Li, L., Chu, W., Langford, J., Schapire, R.E.: A contextual-bandit approach to personalized news article recommendation. In: Proceedings of the 19th International Conference on World Wide Web, WWW '10, pp. 661–670. Association for Computing Machinery, New York (2010)
Lin, K., Sonboli, N., Mobasher, B., Burke, R.: Calibration in collaborative filtering recommender systems: a user-centered analysis. In: Proceedings of the 31st ACM Conference on Hypertext and Social Media, HT '20, pp. 197–206. Association for Computing Machinery, New York (2020)
Ludewig, M., Jannach, D.: Evaluation of session-based recommendation algorithms. User Model. User-Adapt. Interact. **28**(4–5), 331–390 (2018)
Lunardi, G.M., Machado, G.M., Maran, V., de Oliveira, J.P.M.: A metric for filter bubble measurement in recommender algorithms considering the news domain. Appl. Soft Comput. **97**, 106771 (2020)
Malecek, L., Peska, L.: Fairness-preserving group recommendations with user weighting. In: Adjunct Proceedings of the 29th ACM Conference on User Modeling, Adaptation and Personalization, UMAP '21, pp. 4–9. Association for Computing Machinery, New York (2021). https://doi.org/10.1145/3450614.3461679
Mansoury, M.: Fairness-aware recommendation in multi-sided platforms. In: Proceedings of the 14th ACM International Conference on Web Search and Data Mining, WSDM '21, p. 1117–1118. Association for Computing Machinery, New York (2021). https://doi.org/10.1145/3437963.3441672
Masthoff, J.: Group Recommender Systems: Combining Individual Models, pp. 677–702. Springer, Boston (2011)
Medzihorsky, J.: Rethinking the d'hondt method. Polit. Res. Exchange **1**(1), 1–15 (2019)
Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: NIPS '13, pp. 3111–3119. Curran Associates Inc. (2013)
Neve, J., Palomares, I.: Aggregation strategies in user-to-user reciprocal recommender systems. In: 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC), pp. 4031–4036 (2019)
Nguyen, J., Sánchez-Hernández, G., Agell, N., Rovira, X., Angulo, C.: A decision support tool using order weighted averaging for conference review assignment. Pattern Recogn. Lett. **105**, 114–120 (2018)
Nguyen, T.T., Hui, P.M., Harper, F.M., Terveen, L., Konstan, J.A.: Exploring the filter bubble: the effect of using recommender systems on content diversity. In: Proceedings of the 23rd International Conference on World Wide Web, WWW '14, pp. 677–686. Association for Computing Machinery, New York (2014)
Oliveira, S., Diniz, V., Lacerda, A., Pappa, G.L.: Multi-objective evolutionary rank aggregation for recommender systems. In: 2018 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8 (2018)
Pan, W., Chen, L.: Gbpr: Group preference based bayesian personalized ranking for one-class collaborative filtering. In: Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence, IJCAI '13, pp. 2691–2697. AAAI Press (2013)
Peska, L.: Hybrid recommendations by content-aligned bayesian personalized ranking. New Rev. Hypermedia Multimed. **24**(2), 88–109 (2018)
Peska, L., Vojtas, P.: Using implicit preference relations to improve recommender systems. J. Data Semant. **6**(1), 15–30 (2017)
Peska, L., Vojtas, P.: Off-line vs. on-line evaluation of recommender systems in small e-commerce. In: Proceedings of the 31st ACM Conference on Hypertext and Social Media, HT '20, pp. 291–300. Association for Computing Machinery, New York (2020)
Peška, L., Balcar, S.: Fuzzy D'Hondt's algorithm for on-line recommendations aggregation. In: ORSUM '19, vol. 109, pp. 2–11. PMLR (2019)
Quadrana, M., Cremonesi, P., Jannach, D.: Sequence-aware recommender systems. ACM Comput. Surv. **51**(4), 1–36 (2018)

Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: Bpr: Bayesian personalized ranking from implicit feedback. In: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI '09, pp. 452–461. AUAI Press, Arlington, Virginia (2009)

Sacharidis, D.: Top-n group recommendations with fairness. In: Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, SAC '19, pp. 1663–1670. Association for Computing Machinery, New York (2019)

Sainte-Laguë, A.: La représentation proportionnelle et la méthode des moindres carrés. In: Annales scientifiques de l'école Normale Supérieure, vol. 27, pp. 529–542 (1910)

Santos, R.L., Macdonald, C., Ounis, I.: Exploiting query reformulations for web search result diversification. In: Proceedings of the 19th International Conference on World Wide Web, WWW '10, pp. 881–890. Association for Computing Machinery, New York (2010). https://doi.org/10.1145/1772690.1772780

Schedl, M., Zamani, H., Chen, C.W., Deldjoo, Y., Elahi, M.: Current challenges and visions in music recommender systems research. Int. J. Multimed. Inf. Retrieval **7**(2), 95–116 (2018). https://doi.org/10.1007/s13735-018-0154-2

Serbos, D., Qi, S., Mamoulis, N., Pitoura, E., Tsaparas, P.: Fairness in package-to-group recommendations. In: Proceedings of the 26th International Conference on World Wide Web, WWW '17, pp. 371–379. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE (2017)

Sinha, A., Gleich, D.F., Ramani, K.: Deconvolving feedback loops in recommender systems. In: Proceedings of the 30th International Conference on Neural Information Processing Systems, NIPS'16, pp. 3251–3259. Curran Associates Inc., Red Hook (2016)

Starychfojtu, J., Peska, L.: Smartrecepies: Towards cooking and food shopping integration via mobile recipes recommender system. In: Proceedings of the 22nd International Conference on Information Integration and Web-Based Applications and Services, iiWAS '20, pp. 144–148. Association for Computing Machinery, New York (2020)

Steck, H.: Calibrated recommendations. In: RecSys '18, pp. 154–162. ACM (2018)

Symeonidis, P., Coba, L., Zanker, M.: Counteracting the filter bubble in recommender systems: novelty-aware matrix factorization. Intelligenza Artificiale **13**, 37–47 (2019)

Vinagre, J., Jorge, A.M., Gama, J.: Fast incremental matrix factorization for recommendation with positive-only feedback. In: User Modeling. Adaptation, and Personalization, pp. 459–470. Springer, Cham (2014)

Xiao, L., Min, Z., Yongfeng, Z., Zhaoquan, G., Yiqun, L., Shaoping, M.: Fairness-aware group recommendation with pareto-efficiency. In: Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys '17, pp. 107–115. Association for Computing Machinery, New York (2017)

Zhao, Q., Chang, S., Harper, F.M., Konstan, J.A.: Gaze prediction for recommender systems. In: Proceedings of the 10th ACM Conference on Recommender Systems, RecSys '16, pp. 131–138. Association for Computing Machinery, New York (2016). https://doi.org/10.1145/2959100.2959150

Stepan Balcar is a PhD student at Department of Software Engineering, Charles University, Prague, Czech Republic. His main research areas are recommender systems, user preferences, heterogenity of methods and application of naturally inspired algorithms. He also focuses on computational artificial intelligence and high-performance computing.

Vit Skrhak obtained his Bachelor degree in 2020 from Charles University, Prague, Czech Republic and currently frequents *Software and data engineering* master program at the same institution. He is a member of the SIRET research group, and his research activities mainly relate to Video retrieval systems.

Ladislav Peska obtained his PhD in 2016 from Charles University, Prague, Czech Republic. He is currently an assistant professor at Department of Software Engineering, Charles University, Prague. He is a member of the SIRET research group, where his main research interests are recommender systems and multimedia retrieval. L. Peska has published over 50 papers in journals and conference proceedings, with 400+ citations and H-index 11 (GS).