Information Retrieval Journal 20.6 (2017): 606 – 634

DOI: http://dx.doi.org/10.1007/s10791-017-9312-z

Statistical biases in Information Retrieval Metrics for Recommender Systems

Alejandro Bellogín, Pablo Castells, Iván Cantador
Universidad Autónoma de Madrid
{alejandro.bellogin, pablo.castells, ivan.cantador}@uam.es

Abstract. There is an increasing consensus in the Recommender Systems community that the dominant error-based evaluation metrics are insufficient, and mostly inadequate, to properly assess the practical effectiveness of recommendations. Seeking to evaluate recommendation rankings – which largely determine the effective accuracy in matching user needs – rather than predicted rating values, Information Retrieval metrics have started to be applied for the evaluation of recommender systems. In this paper we analyse the main issues and potential divergences in the application of Information Retrieval methodologies to recommender system evaluation, and provide a systematic characterisation of experimental design alternatives for this adaptation. We lay out an experimental configuration framework upon which we identify and analyse specific statistical biases arising in the adaptation of Information Retrieval metrics to recommendation tasks, namely sparsity and popularity biases. These biases considerably distort the empirical measurements, hindering the interpretation and comparison of results across experiments. We develop a formal characterisation and analysis of the biases upon which we analyse their causes and main factors, as well as their impact on evaluation metrics under different experimental configurations, illustrating the theoretical findings with empirical evidence. We propose two experimental design approaches that effectively neutralise such biases to a large extent. We report experiments validating our proposed experimental variants, and comparing them to alternative approaches and metrics that have been defined in the literature with similar or related purposes.

1. Introduction

There is a raising awareness in the Recommender Systems (RS) community that important – or even central – open questions remain to be addressed concerning the evaluation of these systems. The error in predicting held-out user ratings has been by far the dominant offline evaluation methodology in the RS literature (Breese et al., 1998; Herlocker et al., 2004). The limitations of this approach are increasingly evident and have been extensively pointed out (Cremonesi et al., 2010). The prediction error has been found to be far from enough or even adequate to assess the practical effectiveness of a recommender system in matching user needs. The end users of recommendations receive lists of items rather than rating values, whereby recommendation accuracy metrics – as surrogates of the evaluated task – should target the quality of the item selection and ranking, rather than the system numeric scores that determine that selection.

For this reason, researchers are turning towards metrics and methodologies from the Information Retrieval (IR) field (Barbieri et al., 2011; Cremonesi et al., 2010; Herlocker et al., 2004), where ranking evaluation has been studied and standardised for decades. Yet, gaps remain between the methodological formalisation of tasks in both fields, which result in divergences in the adoption of IR methodologies, hindering the interpretation and comparability of empirical observations by different authors.

The use of offline IR evaluation techniques involves the adoption of the Cranfield paradigm (Voorhees and Harman, 2005), and common metrics such as precision, mean average precision (MAP), and normalised Discounted Cumulative Gain (nDCG) (Baeza-Yates and Ribeiro-Neto, 2011). Given the natural fit of top-n recommendations in an IR task scheme, the adoption of IR methodologies would be straightforward. However, recommendation tasks, settings, and available datasets for offline evaluation

involve subtle differences with respect to the common IR settings and experimental assumptions, which result in substantial biases to the effectiveness measurements that may distort the empirical observations and hinder comparison across systems and experiments.

Taking up from prior studies on the matter (Cremonesi et al., 2010; Herlocker et al., 2004; Shani and Gunawardana, 2011; Steck, 2011), we revisit the methodological assumptions underlying IR metrics, and analyse the differences between Recommender Systems and Information Retrieval evaluation and its implications. Upon this, we identify two sources of bias in IR metrics on recommender systems: data sparsity and item popularity. We characterise and study the effect of these two factors both analytically and empirically. We show that the value range of common IR metrics is determined by the density of the available user preference information, to such an extent that the measured values per se are not meaningful, except for the purpose of comparison within a specific experiment. Furthermore, we show that the distribution of ratings among items has a drastic effect on how different recommendation algorithms compare to each other. In this context, we propose and analyse two approaches to mitigate popularity bias on the measured ranking quality, providing theoretical and empirical evidence of their effectiveness.

Our study focuses on the offline side of recommender system evaluation. Online evaluation, typically AB testing, is the dominant and ultimately most reliable evaluation methodology in commercial settings. However offline evaluation is still a prevailing, affordable alternative in recommender systems research (where an online environment is not always readily available), and an important tool in the industrial context as well, typically as an economic and safe early-stage filter for algorithm selection and upgrade in commercial applications. On the other hand, inasmuch as online data may suffer from similar biases as offline evaluation does, our findings may generalise to online conditions as well.

The reminder of the paper follows by revisiting the principles and assumptions underlying the IR evaluation methodology: the Cranfield paradigm (Section 2). After that, in Section 3 we elaborate a formal synthesis of the main approaches to the application of IR metrics to recommendation. In Sections 4 and 5 we analyse, respectively, the sparsity and popularity biases of IR metrics on recommendation tasks. We present and evaluate two approaches to avoid these biases in Section 6, some works related to our paper are presented in Section 7, and end with some conclusions in Section 8.

2. Applying Information Retrieval Methodologies to the Evaluation of Recommender Systems

Offline Information Retrieval evaluation methodologies have been designed, studied and refined over the years under the so-called *Cranfield paradigm* (van Rijsbergen, 1989; Voorhees, 2001) for offline evaluation. In the Cranfield paradigm, as e.g. typically applied in the TREC campaigns (Voorhees and Harman, 2005), information retrieval systems are evaluated on a dataset comprising a set of documents, a set of queries – referred to as *topics*[1] and consisting of a description or representation of user information needs –, and a set of *relevance judgments* by human assessors, which play the role of a gold standard or ground truth (as understood e.g. in machine learning) against which the evaluated system's output accuracy is assessed and benchmarked. Judgments are typically provided by editorial assessors though in some cases end-users can play this role. The assessors manually inspect queries and documents, and decide whether each document is relevant or not for a query. Theoretically, each query-document pair should be assessed for relevance, which, for thousands or millions of documents, is obviously unfeasible. Therefore, a so-called *pooling* approximation is applied, in which the assessors actually inspect and judge

[1] In fact topics are a more general notion than queries, but for the purpose of our discussion and task comparison we will use queries as a most familiar notion for a broad audience.

Task element	TREC ad-hoc retrieval task	Recommendation task
Information need expression	Topic (query and description)	User profile
Candidate answers	All documents in the collection	Target item set
	Same for all queries	One or more per user, commonly different among users
Document data available as system input	Document content	Training ratings, item features
Relevance	Commonly topical, objective	Inherently personalised, fully subjective
Ground truth	Relevance judgments	Test ratings
Relevance assessment	Editorial assessors	End users
Relevance knowledge coverage (sparsity bias)	Reasonably complete (pooling)	Highly incomplete (inherently to task)
Relevance distribution of answers (popularity bias)	Not long-tailed	Long-tailed

Table 1. Offline recommender system evaluation setup: fitting the recommendation task in the Cranfield IR evaluation paradigm.

just a subset of the document collection, consisting of the union of the top-n documents returned by a set of systems for each query. These systems are commonly the ones to be evaluated and compared, and n is called the *pooling depth*, typically around 100 documents. While this procedure obviously misses some relevant documents, it has been observed that the degree of incompleteness is reasonably small, and the missing relevance does not alter the empirical observations significantly, at least up to some ratio between the pooling depth and the collection size (Buckley et al., 2007).

Whereas in a search system users may enter multiple queries, the recommendation task – in its classic formulation – typically considers a single "user need" per user, that is, a user has a set of cohesive preferences that defines her main interests. In this view, a natural fit of recommendation in the Cranfield paradigm would take users – as an abstract construct – as the equivalent of queries in ad-hoc retrieval (the user need to be satisfied), and items as equivalent to documents (the objects to be retrieved and ranked), as summarised in Table 1. A first obvious difference is that queries are explicit representations of specific information needs, whereas in a recommendation setting, user profile records are a global and implicit representation of what the user may need or like. Still, the query-user mapping is valid, inasmuch as user profiles may rightfully fit in the IR scheme as "vague queries."

The equivalent to Cranfield relevance judgments is less straightforward. User ratings for items, as available in common recommendation datasets, are indeed relevance judgments of items for user needs. However, many recommendation algorithms (mainly collaborative filtering methods) require these "relevance judgments" as input to compute recommendations. The rating data withholding evaluation approach, pervasive in RS research, naturally fits here: some test ratings can be held out as ground truth, and the others can be left as training input for the systems. Differently from TREC, here the "queries" and the relevance assessments, by (the task) definition, can only be entered by the same people: the end-users, and they only need to reflect their own subjectivity or whim, whereas in a search task, relevance assessors may want to capture more objective descriptions of relevance. Furthermore, how much data are taken for training and for ground truth is left open to the experiment designers, thus adding a free variable to be watched over as it significantly impacts the measurements.

Symbol			Meaning
$\mathcal{U}$	$\mathcal{I}$	$\mathcal{C}$	Set of all users | all items | candidate items
R	R_{test}	R_{train}	Set of all | test | training ratings
PR	PR_{test}	PR_{train}	Set of all | test | training positive ratings (likes)
$R(u)$	$R_{test}(u)$	$R_{train}(u)$	Set of all | test | training items rated by u
$PR(u)$	$PR_{test}(u)$	$PR_{train}(u)$	Set of all | test | training items liked by u
$R(i)$	$PR(i)$	…	Set of users who rated | like | … item i
T_u	T_u^r		Set of target items for u in AR | 1R
N_u	N_u^r		Non-relevant items added to build T_u | T_u^r
$P_s^u@n(T_u)$			$P@n$ of item set T_u as ranked by s for u
$top_s^u(T_u, n)$			Top n items in T_u as ranked by s for u
$\tau_s^u(i, S)$			Position of i in $S \ni i$ as ranked by s for u
$i_k^{u,s}$	$i_k^{u,r,s}$		The item ranked k-th in T_u | T_u^r by s for u
σ			Split ratio: $\lvert R_{test}\rvert/\lvert R\rvert$
ρ_u	ρ		$\rho_u = \lvert T_u \cap PR_{test}(u)\rvert/\lvert T_u\rvert, \rho = avg_u\, \rho_u$
t			"Average" target set size: $1/avg_u(1/\lvert T_u\rvert)$
δ			Relevance density in target sets: $\lvert PR\rvert/(t\lvert\mathcal{U}\rvert)$

Table 2. Notation summary. As a general convention, u is used for users and i for items. We use the symbol R for sets of ratings and PR for sets of positive ratings (a rating in this mathematical sense is essentially a user-item pair). As soon as we add a user or an item in parenthesis next to these symbols – as in $R(u)$ – they denote sets of items (the ones rated by u) or users (the ones who rated an item in question). And we add the subscript "train" or "test" to refer to the corresponding subsets, relative to a given split of the set of all ratings.

Furthermore, whereas in the IR setting all the documents in the collection are candidate answers for all queries, the set of target items on which recommender systems are tested is not necessarily the same for each user. In general, at least all the items with a test rating are included in the candidate set for all the corresponding raters, though not necessarily in a single run (Cremonesi et al., 2010). Moreover, it is common to select further non-rated target items, but not necessarily all the items (Bellogín et al., 2011), and the items rated by a user in the training set are generally excluded from the recommendation to this user. The way these options are configured has a drastic effect on the resulting measurements, with variations in orders of magnitude (Bellogín et al., 2011; Jannach et al., 2015).

In addition to the previous issues, the coverage of user ratings (density or lack of *sparsity*) is inherently much smaller in RS datasets compared to TREC collections. The amount of unknown relevance is pervasive in recommendation settings (it is in fact intrinsic for the task to make sense), to a point where some assumptions of the IR methodology may not hold, and the gap between measured and real metric values becomes so significant that a metric absolute magnitude may just lose any meaning. Still, such measurements may support comparative assessments between systems, as far as the bias is system-independent (sparsity bias). The missing relevance knowledge in TREC has been found to have a negligible effect on system comparison (Buckley et al, 2007), though this assumption may have to be revised as very large collections are becoming commonplace in IR evaluation. In this context, specific metrics, such as notably bpref (Buckley and Voorhees, 2004) and infAP (Yilmaz and Aslam, 2006), have been proposed in the IR field with the aim of better dealing with missing relevance information, distinguishing between the absence of judgment and explicit non-relevance judgments.

Design settings		Alternatives	
Base candidate items		**AI**	$\mathcal{C} = \mathcal{I}$
		TI	$\mathcal{C} = \bigcup_{u \in \mathcal{U}} R_{test}(u)$
Item selection	Relevant	**AR**	$T_u \supset PR_{test}(u)$
		1R	$\lvert T_u^r \cap PR_{test}(u) \rvert = 1$
	Non-relevant	**AN**	$N_u = \mathcal{C} \setminus PR_{test}(u) \setminus R_{train}(u)$
		NN	Fixed $\lvert N_u \rvert$, random sampling

Table 3. Design alternatives in target item set formation.

On the other hand, the TREC methodology makes the underlying assumption that missing relevance judgments should be uniformly distributed over documents. This is of course not quite true, as there are indeed biases in the pooling process: the ones introduced by the pooled systems. But it has been found that it is not unreasonable to work with that simplifying assumption. In contrast, the distribution of relevance in the retrieval space for recommender systems displays massive popularity skewness patterns that are not found with any comparable strength in IR datasets. The number of users who like each item is very variable (typically long-tailed) in recommendation datasets, and so is the amount of observed relevance conveyed as ratings, whereas in TREC collections very few documents are relevant for more than one query. We shall show that this phenomenon has a very strong effect not only on metric values, but more importantly on how systems compare to each other (popularity bias).

In order to provide a formal basis for our study we start by elaborating a systematic characterisation of design alternatives for the adaptation of IR metrics to recommender systems, taking into account prior approaches described in the literature. This formal framework will help us to analyse and describe the sparsity and popularity biases in the application of IR metrics to recommender systems, and study new approaches to mitigate them.

3. Characterisation of Design Alternatives in IR Methodologies for Recommender Systems

The application of Information Retrieval metrics to recommender systems evaluation has been studied by several authors in the field (Barbieri et al., 2011; Breese et al., 1998; Cremonesi et al., 2010; Herlocker et al., 2004; Shani and Gunawardana, 2011). We elaborate here an experimental design framework that aims to synthesise commonalities and differences between studies, encompassing prior approaches and supporting new variants upon a common methodological grounding.

We start by introducing some concepts and notation; Table 2 summarises all the notation that we shall introduce and use along the paper. Given a rating set split into training and test rating sets, we say an item $i \in \mathcal{I}$ is relevant for a user $u \in \mathcal{U}$ if u rated i positively, and its corresponding rating falls in the test set. By positive rating we mean a value above some threshold that is domain and design-dependent. All other items (non-positively rated or non-rated) are considered as non-relevant. In an evaluation setting, recommender systems are requested to rank a set of target items T_u for each user. Such sets do not need to be the same for each user, and can be formed in different ways. In all configurations, T_u contains a combination of relevant and non-relevant items, and the different approaches are characterised by how these are selected, as we describe next.

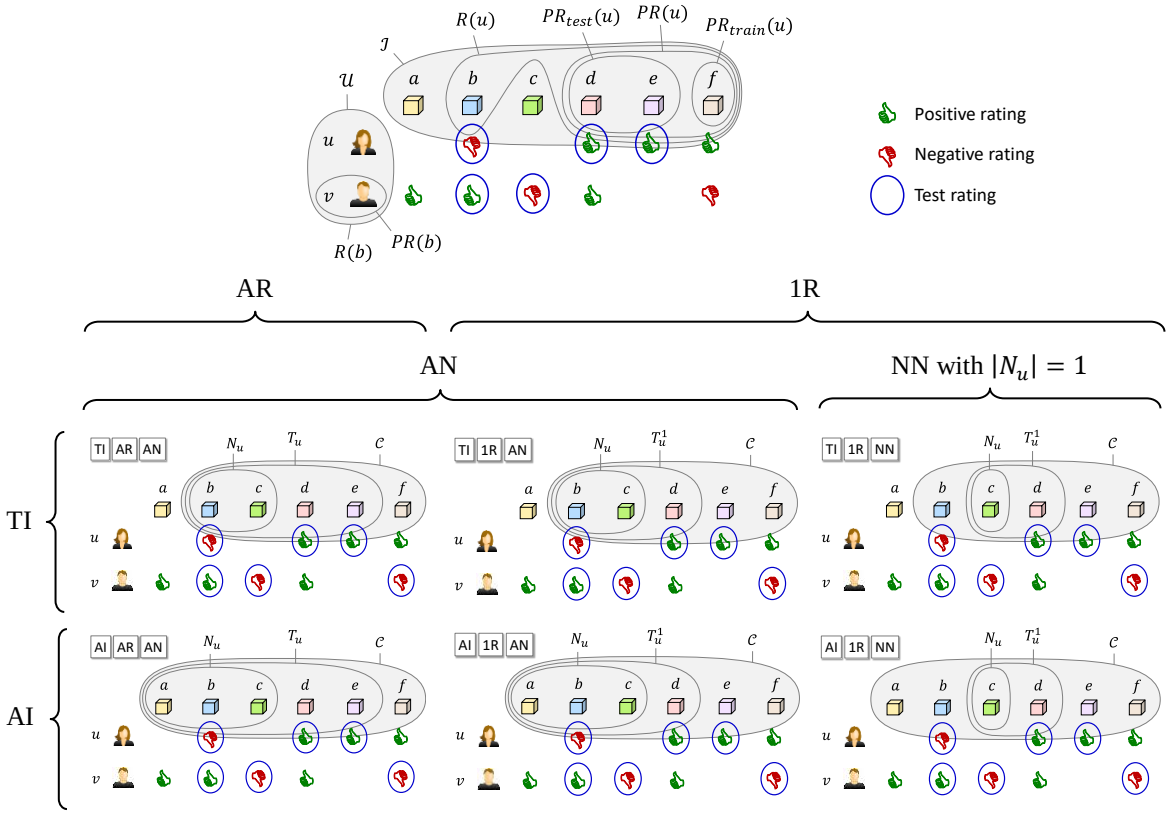

Figure 1. Sampling approach alternatives on a toy example. Each subfigure represents a rating matrix with relevant ("positive"), non-relevant ("negative"), and unobserved values, partitioned into training and test (circled) ratings. We show how the item subsets for user u are taken in different experimental configurations as listed in Table 3. The set of illustrated combinations are non-exhaustive to avoid further cluttering: in particular, we omit the AR-NN option; at the top of the figure we omit some of the sets defined in Table 2; and overall we just show sets for one user u. In the 1R option, we only show one of the two possible choices of relevant items in the T_u^r set (we show picking d in T_u^1, and another set T_u^2 would include e instead). In the NN option, the single ($|N_u| = 1$) non-relevant item is arbitrarily chosen (we pick c, but it could as well be b or, in the AI option, a). The reader can easily figure this out and the remaining options from the illustrated cases.

3.1. Target Item Sampling

We identify three significant design axes in the formation of the target item sets: candidate item selection, irrelevant item sampling, and relevant item selection. We consider two alternatives for each of these axes, summarised in Table 3, illustrated in Figure 1, and described next.

We shall use $R(u)$ and $PR(u)$ to denote the set of all and positively rated items by user u, respectively, and $|R(u)|$, $|PR(u)|$ to denote the respective size of those sets. With the subscripts "test" and "train" we shall denote the part of such sets (or their sizes) contained on the corresponding side of a data split. An equivalent notation $R(i)$, $PR(i)$, and so on, will be used for the ratings of an item, and when no user or item is indicated, the total number of ratings is denoted, i.e., R, PR.

For the candidate item selection alternatives, let $N_u = T_u \setminus PR_{\text{test}}(u)$ be the non-relevant target items for u. As a general rule, we assume non-relevant items are randomly sampled from a subset of candidate

items $\mathcal{C} \subset \mathcal{I}$, the choice of which is a design option. We mainly find two significant alternatives for this choice: $\mathcal{C} = \mathcal{I}$ (e.g. Shani and Gunawardana, 2011) and $\mathcal{C} = \bigcup_{u \in \mathcal{U}} R_{test}(u)$ (e.g. Bellogín et al., 2011; Vargas and Castells, 2011). The first one, which we denote as **AI** for "all items", matches the typical IR evaluation setting, where the evaluated systems take the whole collection as the candidate answers. The second, to which we shall refer as **TI** ("test items") is an advisable condition to avoid certain biases in the evaluation of RS, as we shall see.

Once $\mathcal{C}$ is set, let us decide the irrelevant item sampling strategy. For each user we select a set $N_u \subset \mathcal{C} \setminus PR_{\text{test}}(u) \setminus R_{\text{train}}(u)$, N_u can be sampled randomly for a fixed size $|N_u|$ (we call this option **NN** for "N non-relevant"), or all candidate items can be included in the target set, $N_u = \mathcal{C} \setminus PR_{\text{test}}(u) \setminus R_{\text{train}}(u)$ (we refer to this as **AN** for "all non-relevant"). Some authors have even used $T_u = R_{\text{test}}(u)$ (Basu et al., 1998; Jambor and Wang, 2010a; Jambor and Wang, 2010b), but we discard this option as it results in a highly overestimated precision (Bellogín et al., 2011). The size of N_u is thus a configuration parameter of the experimental design. For instance, in (Cremonesi et al., 2010) the authors propose $|N_u|$ = 1,000, whereas in (Bellogín et al., 2011) the authors consider $N_u = \bigcup_{v \in \mathcal{U}} R_{\text{test}}(v) \setminus R_{\text{train}}(u) \setminus PR_{\text{test}}(u)$, among other alternatives. To the best of our knowledge, the criteria for setting this parameter have not been analysed in detail in the literature, leaving it to common sense and/or trial and error. It is worth noting nonetheless that in general $|N_u|$ determines the number of calls to the recommendation algorithms, whereby this parameter provides a handle for adjustment of the cost of the experiments.Regarding the relevant item selection, two main options are reported in the literature, to which we shall refer as **AR** for "all relevant", and **1R** for "one relevant." In the AR approach all relevant items are included in the target set, i.e., $T_u \supset PR_{test}(u)$ (Bellogín et al., 2011). In the 1R approach, for user u, several target item sets T_u^r are formed, each including a single relevant item (Cremonesi et al., 2010). This approach may be more sensitive to the lack of recommendation coverage, as we shall observe later on. The choice between an AR or a 1R design involves a difference in the way the ranking quality metrics are computed, as we shall discuss in the next section.

3.2. AR vs. 1R Precision

Essentially, the way metrics are defined in AR and 1R differs in how they are averaged. In AR, the metrics are computed on each target set T_u in the standard way as in IR, and then averaged over users (as if they were queries). As a representative and simple to analyse metric, we shall use $P@n$ henceforth. Though more complex metrics such as MAP and nDCG are not as tractable formally, the analysis and findings we derive for precision are also observed empirically – with particular differences in some cases – with such metrics, as we shall point out. The mean AR precision of a recommender system s can be expressed as:

$$P_s@n = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{1}{n} |top_s^u(T_u, n) \cap PR_{test}(u)|$$

where $top_s^u(T_u, n)$ denotes the top n items in T_u ranked by s for u.

In the 1R design, drawing from (Cremonesi et al., 2010), we compute and average the metrics over the T_u^r sets, as follows:

$$1RP_s@n = precision(n) = \frac{1}{|PR_{test}|} \sum_{u \in \mathcal{U}} \sum_{r=1}^{|PR_{test}(u)|} P_s^u@n(T_u^r) \quad (1)$$

where $P_s^u@n(T_u^r)$ is the standard precision of T_u^r for u. This form to express the metric is equivalent to the original formulation in (Cremonesi et al., 2010), but allows a straightforward generalisation to any

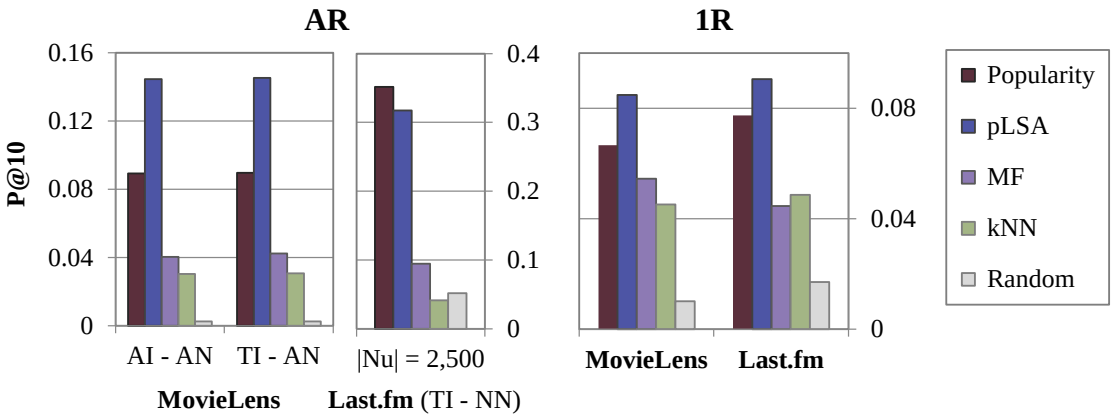

Figure 2. Precision of different recommendation algorithms on MovieLens 1M and Last.fm using AR and 1R configurations.

other IR metric such as MAP and nDCG, by just using them in place of $P_s^u@n$ in Equation (1). We shall intentionally use the same symbol P to refer both to 1R and AR precisions when there is no ambiguity. Whenever there may be confusion, or we wish to stress the distinction, we shall use 1RP to explicitly denote 1R precision.

AR precision basically corresponds to the standard precision as defined in IR, whereas 1R precision, while following essentially the same principle, departs from it in the formation of runs, and the way to average values. Additionally, note that the maximum value of 1RP@n is $1/n$ as we shall see in the next section, mainly since each run has only one relevant item. Besides, in Section 4 we shall establish a formal relation between both ways to compute precision.

3.3. Preliminary Experimental Comparison

In order to illustrate the effects of the design alternatives presented before, we show their results on three common collaborative filtering algorithms, based respectively on probabilistic Latent Semantic Analysis (pLSA) (Hofmann, 2004), matrix factorisation (MF) (Koren et al., 2009), and user-based nearest-neighbours (kNN) (Cremonesi et al., 2010). As additional baselines, we include recommendation by popularity and random recommendation. We use two datasets: the 1M version of MovieLens, and an extract from Last.fm published by Ò. Celma (Celma and Herrera, 2008). A 5-fold cross-validation is applied on 80-20% training-test splits of rating data (as is common practice, we loop through the set of all ratings and "flip a coin" – with 0.8 probability for training – to decide whether the rating is assigned to the training or test set, in such a way that the expected size of these sets is 80% and 20% respectively). The Last.fm data consist of music track playcounts for nearly 1,000 users, which we aggregate by artist (amounting to $|\mathcal{I}|$ = 176,892), and map into ratings as in (Celma and Herrera, 2008). We use a single temporal split of user music track *scrobbles*[2] in this dataset, with the same 80-20% ratio of training-test data.

Figure 2 shows the P@10 results with AR and 1R configurations. For 1R we shall always use TI-NN, with $|T_u|$ = 100. This is a significantly lower value than $|T_u|$ = 1,001 reported in (Cremonesi et al., 2010), but we have found it sufficient to ensure statistical significance (e.g. Wilcoxon $p \ll 0.001$ for all

[2] A scrobble is a play log record in Last.fm jargon, i.e. a user-track-timestamp triplet recorded each time a user plays a music track in the system.

pairwise differences between the recommenders in Figure 2), at a considerably reduced execution cost. We adopt the TI policy in 1R to avoid biases that we shall describe later. In the AR configuration we show TI-AN (meaning "TI followed by AN") and AI-AN for MovieLens, though we shall generally stick to TI-AN in the rest of the paper. In Last.fm we use only TI-NN and a temporal split, with $|N_u|$ = 2,500 for efficiency reasons, since $|\mathcal{I}|$ = 176,948 is considerably large in this dataset. We set the positive relevance rating threshold to 5 in MovieLens, as in (Cremonesi et al., 2010), whereas in Last.fm, we take any number above 2 playcounts as a sign of positive preference. We have experimented with other thresholds for positive ratings, obtaining equivalent results to all the ones that are reported here – the only difference is discussed in Section 6.

It can be seen that pLSA consistently performs best in most experimental configurations, closely followed by popularity, which is the best approach in Last.fm with AR, and that MF is generally superior to kNN. Some aspects strike our attention. First, even though P@10 is supposed to measure the same thing in all cases, the range of the metric varies considerably across configurations and datasets, and even the comparison is not always consistent. For instance, in AR popularity ranges from 0.08 on MovieLens to 0.35 on Last.fm; and AR vs. 1R produces some disagreeing comparisons on Last.fm. It may also be surprising that popularity, a non-personalised method, fares so well compared to other algorithms. This effect was already found in (Cremonesi et al., 2010) and (Steck, 2011), and in (Pradel et al., 2012). We also see that TI and AI produce almost the same results. This is because $\bigcup_{u\in\mathcal{U}} R_{test}(u) \sim \mathcal{I}$ in MovieLens; differences become noticeable in configurations where $\bigcup_{u\in\mathcal{U}} R_{test}(u)$ is significantly smaller than $\mathcal{I}$, as we shall see in Section 6.2. As mentioned before, note that in this case the upper bound of P@10 for the 1R methodology is 0.10. Other metrics such as nDCG, MAP, or Mean Reciprocal Rank (MRR) show similar trends as with P@10.

Some of this variability may reflect actual strengths and weaknesses of the algorithms for different datasets, but we shall show that a significant part of the observed variations is due to statistical biases arising in the adaptation of the Cranfield methodology to recommendation data, and are therefore meaningless with respect to the assessment of the recommenders accuracy. Specifically, we have found that the metrics are strongly biased to test data sparsity and item popularity. We shall analyse this in detail in Sections 4 and 5, but before that we establish a relation between AR and 1R precision that will help in this analysis.

3.4. Analytical Relation between AR and 1R Precision

We have seen that AR and 1R precisions produce in general quite different values, and we shall show they display different dependencies over certain factors. We find nonetheless a direct relation between the two metrics. Specifically, 1R precision is bound linearly by NN-AR precision, that is, $1RP_s@n = \Theta(P_s@n)$, as we show next.

Lemma. Let us assume the irrelevant item sampling in 1R is done only once for all the test ratings of a user, that is, we select the same set of non-relevant items $N_u^r = N_u$ in the T_u^r target sets. If we denote $T_u = N_u \cup PR_{test}(u)$ – in other words, $T_u = \bigcup_r T_u^r$ –, we have:

$$\frac{|\mathcal{U}|P_s@n}{|PR_{test}|} \leq 1RP_s@n \leq \frac{\sum_{u\in\mathcal{U}} m_u P_s^u@m_u(T_u)}{n\,|PR_{test}|} \tag{2}$$

with $m_u = n + |PR_{test}(u)| - 1$, where $P_s@n$ is the NN-AR precision computed with the target sets $\{T_u\}$.[3]

[3] Note than m_u is a function of n, but we omit an explicit indication of this to avoid further burdening the notation.

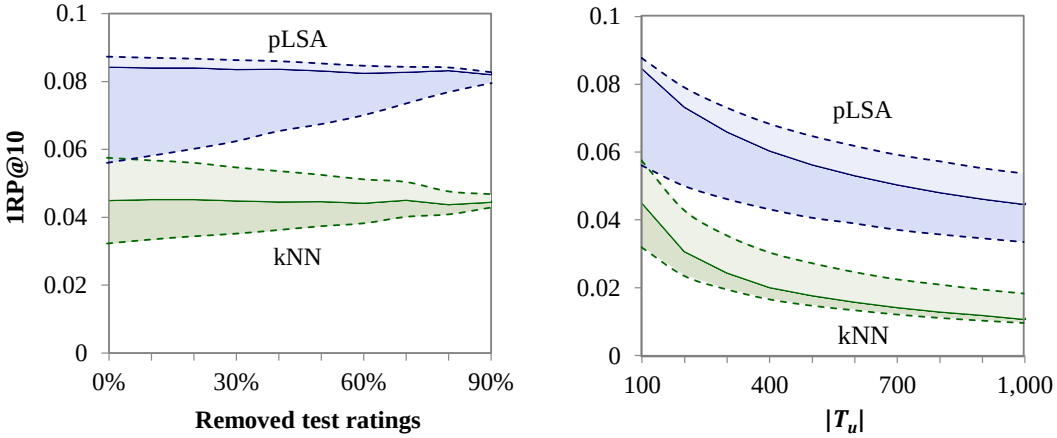

Figure 3. Empirical illustration of Equation (2). The curves show 1RP@10 and its bounds, for pLSA and kNN over MovieLens 1M. The light and dark shades mark the distance to the upper and lower bounds, respectively. The left side shows the evolution when progressively removing test ratings, and the right side displays the variation with $|T_u|$ ranging from 100 to 1,000.

Proof. Let i_u^r be the relevant item included in T_u^r, and let $\tau_s^u(i, S)$ denote the ranking position assigned to i by s for u within a set S, where $i \in S$. Since $T_u^r \subset T_u$, we have that $\tau_s^u(i_u^r, T_u^r) \leq \tau_s^u(i_u^r, T_u)$. This means that if i_u^r is ranked above n in T_u, then it is also above n in its target set T_u^r. Hence $\sum_{r=1}^{np_{test}(u)} |top_s^u(T_u^r, n) \cap PR_{test}(u)| \geq |top_s^u(T_u, n) \cap PR_{test}(u)|$. Summing on u, and dividing by n and $|PR_{test}|$ we prove the first inequality of Equation (2).

On the other hand, it is easy to see that $\tau_s^u(i_u^r, T_u^k) \geq \tau_s^u(i_u^r, T_u) + |PR_{test}(u)| - 1$. Thus, if i_u^r is ranked above n in T_u^k, then it is above $m_u = n + |PR_{test}(u)| - 1$ in T_u. Thus $\sum_{r=1}^{np_{test}(u)} |top_s^u(T_u^r, n) \cap PR_{test}(u)| \leq |top_s^u(T_u, m_u) \cap PR_{test}(u)| = m_u \mathrm{P}_s@m_u(T_u)$. And the second inequality of Equation (2) follows again by summing on u, and dividing by n and $|PR_{test}|$. □

Note that the assumption $N_u^r = N_u$ in the lemma is mild, inasmuch as the statistical advantage in taking different N_u^r for each r is unclear. Even in that case, $\mathrm{P}_s@n$ and $\mathrm{avg}_{u \in \mathcal{U}}\left(m_u \mathrm{P}_s^u@m_u(T_u)\right)$ should be reasonably stable with respect to the random sampling of N_u^r, and thus Equation (2) tends to hold.

We thus have a formal proof that AR and 1R precision are quite directly related, however the lemma does not establish how tight is the bound, and how close are the extremes from each other. Figure 3 illustrates and provides an example quantification of the relation between the AR bounds and the 1R values. The empirical observation suggests they provide similar while not fully redundant assessments. We also see that the bounding interval reduces progressively as $|T_u|$ is increased (right), and even faster with test data sparsity (left) – in sum, the metric converges to its bounds as $|T_u| \gg \mathrm{avg}_{u \in \mathcal{U}} |PR_{test}(u)| = |PR_{test}|/|\mathcal{U}|$.

4. Sparsity Bias

As mentioned earlier, we identify two strong biases in precision metrics when applied to recommendation. The first one is a sensitivity to the ratio of the test ratings vs. the added non-relevant items. We study this effect by an analysis of the expected precision for non-personalised and random recommendations in the AR and 1R settings.

4.1. Measuring the Expected Precision

Let $i_k^{u,s} \in T_u$ be the item ranked at position k in the recommendation output for u by a recommender system s, and let σ be the ratio of test data in the training-test data split. In an AR setup the expected precision at n (over the sampling space of data splits with ratio σ, the sampling of N_u, and any potential non-deterministic aspect of the recommender system – as, e.g., in a random recommender) is:

$$E[P_s@n] = \underset{u \in \mathcal{U}}{avg}\left(\frac{1}{n}\sum_{k=1}^{n} p(rel|i_k^{u,s}, u, T_u)\right)$$

where $p(rel|i,u)$ denotes the probability that item i is relevant for user u, i.e., the probability that $i \in PR_{test}(u)$. Now we may write $p(rel|i_k^{u,s}, u, T_u) \stackrel{\text{def}}{=} p(rel, T_u|i_k^{s,u}, u)/p(T_u|i_k^{s,u}, u)$, where we have $p(T_u|i_k^{s,u}, u) = p(i_k^{s,u} \in T_u) = |T_u|/|\mathcal{C}|$. On the other hand, $p(rel, T_u|i_k^{s,u}, u) \stackrel{\text{def}}{=} p(i_k^{s,u} \in PR_{test}(u) \cap T_u) = p\left(i_k^{s,u} \in PR_{test}(u)\right) = p(rel|i_k^{s,u}, u)$, since $PR_{test}(u) \subset T_u$ in the AR methodology. If s is a non-personalised recommender then $i_k^{u,s}$ and u are mutually independent, and it can be seen that $avg_{u \in \mathcal{U}}\, p(rel|i_k^{u,s}, u) = avg_{u \in \mathcal{U}}\, p(rel|i_k^{u,s})$. All this gives:

$$E[P_s@n] = \frac{|\mathcal{C}|}{n\, t}\sum_{k=1}^{n} \underset{u \in \mathcal{U}}{avg}\, p(rel|i_k^{u,s})$$

where $1/t = \text{avg}_{u \in \mathcal{U}}(1/|T_u|)$ – if T_u have all the same size, then $t = |T_u|$. As all relevant items for each user are included in her target set, we have $p(rel|i_k^{u,s}) = E\left[\left|PR_{test}(i_k^{u,s})\right|\right]/|\mathcal{U}|$. If ratings are split at random into test and training, this is equal to $\sigma\left|PR(i_k^{u,s})\right|/|\mathcal{U}|$. Hence, we have:

$$E[P_s@n] = \frac{\sigma|\mathcal{C}|}{n\, t\, |\mathcal{U}|}\sum_{k=1}^{n} \underset{u \in \mathcal{U}}{avg}\left|PR(i_k^{u,s})\right| \tag{3}$$

Now, if items were recommended at random, we would have $E\left[\left|PR\left(i_k^{u,RND}\right)\right|\right] = |PR|/|\mathcal{I}|$, and therefore:

$$E[P_{RND}@n] = E[P_{RND}] \sim \frac{\sigma\,|PR|}{t\,|\mathcal{U}|} = \sigma\,\delta \tag{4}$$

where δ is the average density of known relevance – which depends on how many preferences for items the users have conveyed, and the size of the target test item sets.

On the other hand, in a 1R evaluation setup, we have:

$$E[1RP_s@n] = \frac{1}{n\,|PR_{test}|}\sum_{u \in \mathcal{U}}\sum_{r=1}^{|PR_{test}(u)|}\sum_{k=1}^{n} p(rel|i_k^{u,r,s}, u, T_u^r)$$

where $i_k^{u,r,s} \in T_u^r$ denotes the item ranked at position k in T_u^r. For random recommendation, we have $p\left(rel|i_k^{u,r,RND}, u, T_u^r\right) = 1/|T_u^r| = 1/t$ since all target sets have the same size, whereby we have:

$$E[1RP_{RND}@n] = E[1RP_{RND}] = 1/t \tag{5}$$

4.2. Testing the Sparsity Bias

The above results for the expected random precision provide a formal insight on strong metric biases to characteristics of the data and the experimental configuration. In both Equations (4) for AR and (5) for 1R, we may express the expected random precision as $E[P_{RND}@n] = \text{avg}_{u \in \mathcal{U}}\,\rho_u = \rho$, where ρ_u is the ratio of positively rated items by u in T_u (or T_u^r, for that matter), and $\rho \sim \sigma\,\delta$, or $\rho = 1/t$, depending on

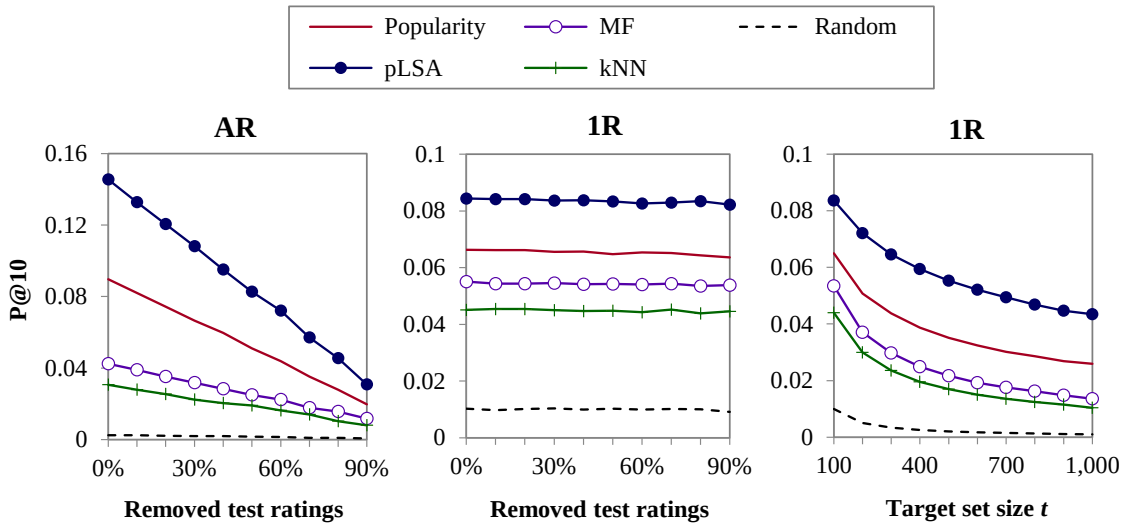

Figure 4. Evolution of the precision of different recommendation algorithms on MovieLens 1M, for different degrees of test sparsity. The x axis of the left and centre graphs shows different amounts of removed test ratings. The x axis in the right graph is the size of the target item sets.

the experimental approach. In the AR approach the density δ, and thus the ρ ratio, are also inversely proportional to t. Precision in this methodology is therefore sensitive to (grows linearly with) σ and $|PR|$, and is inversely proportional to t, whereas 1R is only sensitive (inversely proportional) to t. The expected precision of random recommendation naturally provides a lower bound for any acceptable recommender. Note that in any configuration of AR and 1R, the total precision of any system is $P_s = P_{RND} = \rho = E[P_{RND}@n]$, since as all systems are required to return (recommend) all items in the target sets T_u (or T_u^r), that is, the total precision does not depend on the ranking. At lower cutoffs, we expect to have $P_s@n > E[P_{RND}@n] = \rho$. In other words, the lower bound – and so the expected range – for the $P@n$ of recommender algorithms grows with the average ratio of relevant items per target item set.

The ρ ratio – hence the random precision – thus depends on several aspects of the experimental setup (the experimental approach, the split ratio σ, the number of non-relevant items in the target sets), and the test collection (the number of ratings, the number of users). Therefore, since ρ and the random precision can be adjusted arbitrarily by how the test sets are split, constructed, etc., we may conclude that **the specific value of the metric has a use for comparative purposes, but has no particular meaning by itself, unless accompanied by the corresponding average relevance ratio** ρ of the target test sets. This is naturally in high contrast to common IR datasets, where both the document collection and the relevance information are fixed and not split or broken down into subsets. In fact, the metric values reported in the TREC campaigns have stayed within a roughly stable range over the years (Armstrong et al., 2009a; Armstrong et al., 2009b). Note also that the sparsity bias we analyse here is different from the impact of training data sparsity in the performance of collaborative filtering systems. What we describe is a statistical bias caused by the sparsity of test data (as a function of overall data sparsity and/or test data sampling), and its effect does not reflect any actual variation whatsoever in the true recommendation accuracy.

The sparsity bias explains the precision range variations observed earlier in Figure 2. The empirically obtained values of random precision match quite exactly the theoretically expected ones. To what extent the random recommendation analysis generalises to other algorithms can be further analysed empirically. Figure 4 illustrates the bias trends over rating density and target set size, using the

experimental setup of Section 3.3 (with TI-AN in AR, and TI-NN in 1R). We show only the results in MovieLens – they display a similar effect on Last.fm. In the left and centre graphics, we simulate test sparsity by removing test ratings. In the right graphic we vary $t = |T_u|$ in a 1R configuration. We observe that the empirical trends confirm the theoretical analysis: precision decreases linearly with density in the AR methodology (left graphic, confirming a linear dependence on δ), whereas precision is independent from the amount of test ratings in the 1R approach (centre), and shows inverse proportionality to t (right). It can furthermore be seen that the biased behaviour analytically described for random recommendation is very similarly displayed by the other recommenders (only differing in linear constants). This would confirm the explanatory power of the statistical trend analysis of random recommendation, as a good reference for similar biases in other recommenders. On the other hand, even though the precision values change drastically in magnitude, it would seem that the comparison between recommenders is not distorted by test sparsity.

We should also underline the property that **the sparsity bias is controlled by design in the 1R approach**, as far as the metric value range is concerned, by just setting the number of sampled non-relevant candidate items: for a fixed t, Figure 4 shows the metric range remains much the same when varying the test density level. This means 1R precision values could be made to some extent comparable across datasets. If we keep reducing the rating density towards zero, eventually more and more users would not even have a single test rating for the 1R approach to be put in place, and the metric value rather than becoming unreliable, would become undefined. The anomaly might be less evident in the AR approach, which does not necessarily check for the availability of test ratings for all users: the metric comparison between systems would keep shrinking until loosing statistical significance and ultimately becoming random for lack of data.

We omit further graphs for other metrics besides precision, but we briefly comment their behaviour here. The sparsity bias is similarly displayed by metrics such as MRR and nDCG@n, for which we have observed an analogous empirical behaviour to the trends in Figure 4. Normalised metrics such as recall, nDCG and MAP are however not this directly affected by the sparsity bias, particularly in the usual low density situations (at extremely high densities even these metrics start saturating and converging towards the ratio of positive vs. negative ratings, obviously). Whereas recall seems quite unaffected by sparsity below e.g. the MovieLens 1M density level (i.e. in the range of the x axis of Figure 4 left), MAP and nDCG (without cutoff) are still slightly sensitive to the sparsity bias, possibly because of the interaction between rank-sensitivity and averaging over users: as density increases, the probability to rank relevant items at higher positions increases as well. Possibly because the rank discount in these metrics is not linear, this results in a slight increase in the average metric values. The sensitivity is even clearer in nDCG@n, especially for short values of n: the normalisation does not quite neutralise the bias, since IDCG@n reaches its maximum when a user has more than n relevant items while DCG@n keeps growing with density. Overall the slope of the dependency is slightly flatter for nDCG@n than for P@n, but just as steady.

In all the cases discussed so far, the sparsity effects thus mainly affect the range of the metric values, but not – except in sparsity extremes – the comparison between systems. We find however other biases in precision measurements which do directly affect the comparison of recommenders even in most ordinary circumstances, as we study in the next section.

5. Popularity Bias

Sparsity is not the only bias the metric measurements are affected by. The high observed values for a non-personalised method such as recommendation by popularity raise the question of whether this really

reflects a virtue of the recommender, or some other bias in the metric. We seek to shed some light on the question by a closer study.

5.1. Popularity-Driven Recommendation

Even though they contradict the personalisation principle, the good results of popularity recommendation can be given an intuitive explanation. By averaging over all users, precision metrics measure the overall satisfaction of the user population. A method that gets to satisfy a majority of users is very likely to perform well under such metrics. In other words, average precision metrics tend to favour the satisfaction of majorities, regardless of the dissatisfaction of minorities, whereby algorithms that target majority tastes will expectably yield good results on such metrics. This implicitly relies on the fact that on a random item split, the number of test ratings for an item correlates with its number of training ratings, and its number of positive ratings correlates with the total number of ratings. More formally, the advantage of popularity-oriented recommendation comes from the fact that in a random rating split, $E[|PR_{test}(i)|] \propto |PR(i)| \propto E[|PR_{train}(i)|] \propto |R_{train}(i)|$, which means that the items with many training ratings will tend to have many positive test ratings, that is, they will be liked by many users according to the test data. We analyse this next, more formally and in more detail.

In a popularity recommender $i_k^{u,POP}$ is the k-th item in the target set with most ratings in the training set – i.e., the system ranks items by decreasing order of $\left|R_{train}\left(i_k^{u,POP}\right)\right|$. This ranking is almost user-independent (except for those, statistically negligible, user items already in training which are excluded from the ranking) and therefore, for an AR experimental design, Equation (3) applies. Since we have $\sum_{k=1}^{n}\left|PR\left(i_k^{u,POP}\right)\right| = max_s \sum_{k=1}^{n}\left|PR(i_k^{u,s})\right|$ (as far as $E[|PR_{test}(i)|] \propto |R_{train}(i)|$ for a random training-test split), the popularity recommendation is the best possible non-personalised system, maximising $E[P_s@n]$. Popularity thus achieves a considerably high precision value, just for statistical reasons.

For a 1R experimental design, using Equation (2) (lemma) we have:

$$\frac{|\mathcal{U}|E[P_s@n]}{|PR_{test}|} \leq E[1RP_s@n] \leq \frac{\sum_u E[m_u P_s^u@m_u(T_u)]}{n\,|PR_{test}|}$$

Now, since $P_s@n$ and $P_s^u@m_u$ above are computed by AR, we may elaborate from Equation (3) for a non-personalised recommender, and we get:

$$\frac{|\mathcal{I}|}{n\;t\;|PR|}\;\underset{u\in\mathcal{U}}{avg}\sum_{k=1}^{n}\left|PR(i_k^{u,s})\right| \leq E[1RP_s@n] \leq \frac{|\mathcal{C}|}{n\;t\;|PR|}\;\underset{u\in\mathcal{U}}{avg}\sum_{k=1}^{m_u}\left|PR(i_k^{u,s})\right|$$

This experimental approach is thus equally biased to popular items, since the latter optimise $\sum_{k=1}^{n}\left|PR(i_k^{u,s})\right|$.

Note that the advantage of popularity over other recommenders is highly dependent on the skewness in the distribution of ratings over items: if all items were equally popular, the popularity recommender would degrade to random recommendation – in fact slightly worse, as $|PR_{test}(i)| \propto |R_{test}(i)| = |R|/|\mathcal{I}| - |R_{train}(i)|$, so popular items would have fewer positive test ratings. On the other extreme, if a few items (less than n) are liked by most users, and the rest are liked by very few, then popularity approaches the maximum precision possible.

5.2. Effect of Popularity Distribution on the Popularity Bias

In order to illustrate how the dependence between the popularity precision and the background popularity distribution evolves, we simulate different degrees of skewness in rating distributions. As a simulated distribution pattern we use a shifted power law $|R(i_k)| = c_1 + \beta(c_2 + k)^{-\alpha}$, where α determines the

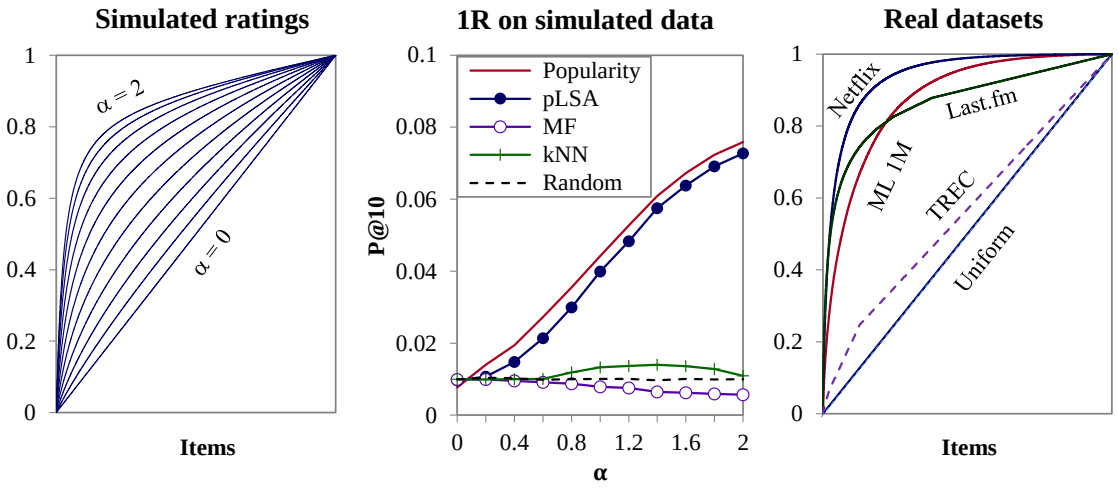

Figure 5. Effect of popularity distribution skewness on the popularity bias. The left graph shows the cumulated popularity distribution of artificial datasets with simulated ratings, with skewness ranging from α = 0 to 2. The x axis represents items by popularity rank, and the y axis displays the cumulative ratio of ratings. The central graph shows the precision of different recommendation algorithms on each of these simulated datasets. The right graph shows the cumulative distribution of positive ratings in real datasets.

skewness (e.g. $\alpha \sim 1.4$ for MovieLens 1M). Figure 5 (left) shows the shape of generated distributions ranging from uniform (α = 0) to a very steep long-tailed popularity distribution (α = 2), and (centre) how the measured precision evolves in this range. The artificial data are created with the same number of users, items, and ratings (therefore the same rating density) as in MovieLens 1M, setting c_1 and c_2 by a fit to this dataset, and enforcing these constraints by adjusting β. The rating values are assigned randomly on a 1-5 scale, also based on the prior distribution of rating values in MovieLens.

The results in Figure 5 (centre) evidence the fact that the precision of popularity-based recommendation is heavily determined by the skewness of the distribution. It benefits from steep distributions, and degrades to slightly below random (0.0077 vs. 0.0100) when popularity is uniform. This slightly below-random performance of popularity recommendation at α = 0 is explained by the fact that $E[|PR_{test}(i)|] \propto E[|R_{test}(i)|] = |R(i)| - E[|R_{train}(i)|]$ is inverse to the popularity ranking by $|R_{train}(i)|$ when $|R(i)|$ is uniform, as predicted at the end of the previous section. kNN and MF stay essentially around random recommendation. This is because the data are devoid of any consistent preference pattern (as collaborative filtering techniques would assume) in this experiment, since the ratings are artificially assigned at random, and the results just show the "pure" statistical dependency to the popularity distribution. pLSA does seem to take advantage of item popularity, as it closely matches the effectiveness of popularity recommendation. We show only the 1R design, but the effect is the same in AR. The popularity bias is quite similarly displayed empirically by other metrics we have tested, including recall, nDCG, MAP and MRR, with and without cutoff.

This observation also explains the difference between datasets from IR and those from recommendation with regards to the popularity bias. Figure 5 (right) shows the cumulative distribution of positive user interaction data per item in three datasets: Netflix, MovieLens, and Last.fm. The shapes of the curves are typical of long-tailed distributions, where a few popular items accumulate most of the preference data (Celma, 2010; Celma and Cano, 2008). This contrasts with the distribution of positive relevance judgments over documents in TREC data (same figure) – where we have aggregated the

judgments (as if, so to speak, we appended the relevance judgment files) from 30 tracks, as made available in the TREC relevance judgment list,[4] obtaining a set of 703 queries, 129,277 documents, and 149,811 positive judgments. The TREC distribution is considerably flatter, not far from uniform: 87.2% of documents are relevant to just one query, and the maximum number of positive assessments per document is 25 (3.6% of queries), whereas the top popular item in Netflix, MovieLens, and Last.fm, is liked by 20.1%, 32.7% and 73% of users, respectively. If we took, for instance, a full Web search log from a commercial engine, we might find that the number of queries for which documents are relevant might no longer have such a flat distribution, since we may expect certain documents to be much more often the object of search than others. However, we use this aggregated view here to illustrate what the IR researcher typically has in hands in common offline evaluation practice – it would in fact be interesting to explore to what extent and how our research might apply to large-scale search scenarios. We may expect to find a lower limit to the bias strength in search data though: a document cannot be relevant to *any* query, whereas an item (a song, a movie, a book), could be potentially liked by any person.

5.3. Sources of Bias: IR vs. RS

Several reasons account for this difference between retrieval and recommender datasets. First, in IR queries are selected by design, intending to provide a somewhat varied testbed to compare retrieval systems. Hence, including similar queries with overlapping relevance would not make much sense. Second, queries in natural search scenarios are generally more specific and narrower than global user tastes for recommendation, whereby the corresponding relevant sets have much less intersection. Furthermore, the TREC statistics we report are obtained by aggregating the data of many tracks, in order to seek any perceptible popularity slant. The typical TREC experiments are actually run on separate tracks comprising typically 50 queries, where very few documents, if any, are relevant to more than one query. Note also that even though we have filtered out over 0.7 million non-relevant plus nearly 5 million unlabelled documents in the TREC statistics, the non-relevant documents actually remain as input to the systems, contrarily to experiments in the recommender domain, thus making up an even flatter relevance distribution. Moreover, in the usual IR evaluation setting, the systems have no access to the relevance data – thus, they have no means to take a direct bias towards documents with many judgments –, whereas in recommendation, this is the primary input the systems (particularly collaborative filtering recommenders) build upon. The popularity phenomenon has therefore never been an issue in IR evaluation, and neither the metrics nor the methodologies have had to even consider this problem, which arises now when bringing them to the recommendation setting – where the overlap between user preferences is not only common, but actually needed by collaborative filtering algorithms.

In contrast, the data on which recommender systems are evaluated are collected through natural interaction with users, where the item selection for rating is subject to several kinds of natural biases. For instance, the MovieLens datasets, probably the most popular in recommender systems research, include movie ratings and other data collected from real users in the MovieLens website. The 1M subset (for which we reported results in previous sections) includes all the ratings entered from April 2000 to February 2003 by all users (with a minimum of 20 ratings each) who registered into the system during the year 2000 (Harper and Konstan, 2016). Users reached items in the MovieLens application through several means including searching and browsing at the user's initiative, as well as system's recommendation and front pages (e.g. top hits, recent releases). Item selection – and the resulting popularity distribution – was therefore exposed to both user and system biases. User bias factors include the probability that a random user has seen a movie (an assumed precondition for entering a rating), and preference biases in the will to

[4] http://trec.nist.gov/data/qrels_eng

enter a rating (e.g. people are typically more prone to rate movies they like than ones they do not, and the opposite can be true in other, e.g. complaint-driven domains). The system may add its own biases in the selection of front page items, and the overall biases may be subject to reinforcement by a feedback loop with the recommendation algorithm itself (Fleder and Hossanagar, 2017). Similarly, the Last.fm dataset includes the complete music play history of nearly 1,000 users from February 2005 to May 2009 (Celma and Herrera, 2008). Again, how users reach music tracks and decide to listen to them is the result of a complex combination of factors where some items have a higher prior probability to be reached than others. No further data sampling bias was there beyond that in neither of these two datasets other than selecting a period of time of observation.

We tend to believe that the popularity biases observed in these datasets have for the most part a source outside the system: one would think the biases of culture, marketing, commercial distribution channels, or mouth-to-mouth, should dominate over the potential deviations introduced by the system, as far as the latter can be expected not to strongly take the user away from his natural, personal view of the world. The direction we research here is nonetheless independent from that concern. Understanding and dissecting the factors that come into play in giving rise to popularity biases is beyond the reach of our current research (see e.g. Cañamares and Castells, 2014 for preliminary steps in that direction). In our present work, we aim to prove that such biases interfere in the outcome of offline experiments in potentially drastic ways: the offline evaluation procedure rewards the recommendation of items for which many observations are available, regardless of whether the wealth – or scarcity – of observations corresponds to an actual abundance – or lack – of positive appreciation by the user. In light of this realisation, we seek to provide principled procedures for removing the popularity-bias variable from the experiment as far as possible, as we propose in the next section.

6. Overcoming the Popularity Bias

After analysing the effects of popularity in precision metrics, the issue remains: to what extent do the good results of popularity recommendation reflect only a spurious bias in a metric, or any degree of actual recommendation quality? The same question should be raised for pLSA, which seems to follow the popularity trends quite closely. We address the question by proposing and examining alternative experimental configurations, where the statistical role of popularity gets reduced.

6.1. Percentile-Based Approach (P1R)

We propose a first approach to neutralise the popularity bias, which consists in partitioning the set of items into m popularity percentiles $\mathcal{I}_k \subset \mathcal{I}$, breaking down the computation of accuracy by such percentiles, and averaging the m obtained values. By doing so, in a common long-tailed popularity distribution, the margin for the popularity bias is considerably reduced, as the difference Δ_k in the number of positive test ratings per item between the most and least popular items of each percentile is not that high. The popularity recommender is forced to recommend as many unpopular as popular items, thus levelling the statistical advantage to a significant extent. It remains the optimal non-personalised algorithm, but the difference – and thus the bias – is considerably reduced. The technique is illustrated in Figure 6a.

A limitation of this approach is that it restricts the size of the target sets by $|T_u| \leq |\mathcal{I}|/m$. For instance, for m = 10 in MovieLens 1M, this imposes a limit of $|T_u| \leq \sim 370$, which seems acceptable for 1R. The restriction can be more limiting in the AR approach, e.g. the TI and AI options cannot be applied

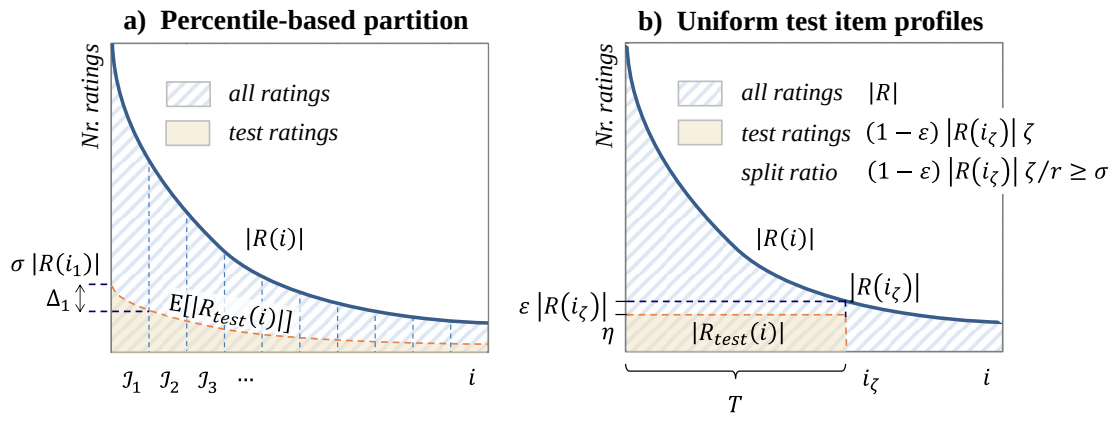

Figure 6. Rating splits by a) a popularity percentile partition (left), and b) a uniform number of test ratings per item (right). On the left, the red dashed split curve represents the expectation $E[|R_{test}(i)|]$ – i.e., the random split ratio needs not be applied on a per-item basis – whereas on the right it represents the actual exact value of $|R_{test}(i)|$.

(except within the percentiles). For this reason, we will only apply the percentile technique in the 1R design, a configuration to which we shall refer as **P1R**.

6.2. Uniform Test Item Profiles (UAR, U1R)

We now propose a second technique consisting of the formation of data splits where all items have the same amount of test ratings. The assumption is that the items with a high number of training ratings will no longer have a statistical advantage by having more positive test ratings. That is, the relation $E[|PR_{test}(i)|] \propto |R_{train}(i)|$ described in Section 5.1 breaks up. The approach consists of splitting the data by picking a set T of candidate items, and a number η of test ratings per item so that $|T|\eta/|R| = \sigma$. For this to be possible, it is necessary that $(1-\varepsilon)\,|R(i)| \geq \eta, \forall i \in T$, where ε is a minimum ratio of training ratings per item we consider appropriate. In particular, in order to allow for n-fold cross-validation, we should have $\varepsilon \geq 1/n$. The selection of T can be done in several ways. We propose to do so in a way that it maximises $|T|$, i.e., to use as many different target test items as possible, avoiding a biased selection towards popular items. If we sort $i_k \in \mathcal{I}$ by popularity rank, it can be seen that this is achieved by picking $T = \{i_k \in \mathcal{I} | k \leq \zeta\}$ with $\zeta = max\ \{k | (1-\varepsilon)\,|R(i_k)|\,k/|R| \geq \sigma\}$, so that $\eta = (1-\varepsilon)\,\left|R(i_\zeta)\right|$. Figure 6b illustrates this procedure.

The expected effect of this approach is that the statistical relation $E[|PR_{test}(i)|] \propto |PR(i)|$ no longer holds, and neither should hold now, as a consequence, the rationale described in Section 5.1 for popularity being the optimum non-personalised recommender. In fact, since $E[|PR_{test}(i)|] = \eta \cdot |PR(i)|/|R(i)|$ for any $i \in T$, and $\eta = \sigma \cdot |R|/|T|$, it can be seen that if $\mathcal{C} = T$ (TI policy) Equation (3) for AR yields:

$$E[P_s@n] = \frac{\sigma\,|R|}{n\,t\,|\mathcal{U}|} \sum_{k=1}^{n} \underset{u \in \mathcal{U}}{avg} \frac{\left|PR(i_k^{u,s})\right|}{\left|R(i_k^{u,s})\right|}$$

for any non-personalised recommender. If the ratio $\left|PR(i_k^{u,s})\right|/\left|R(i_k^{u,s})\right|$ of positive ratings does not depend on k, we have $E[P_s@n] = E[P_{RND}@n] = \sigma\,\delta$. This means that popularity recommendation may get some advantage over other recommenders only if – and to the extent that – popular items have a

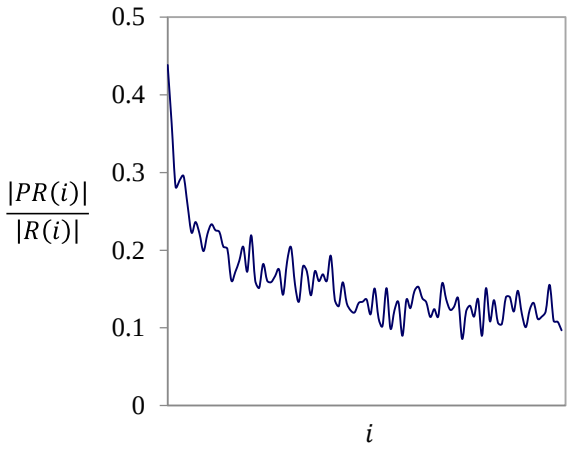

Figure 7. Positive ratings ratio vs. popularity rank of each item i in MovieLens 1M. The graph plots $|PR(i)|/|R(i)|$, where items are ordered by decreasing popularity. We display averaged values for 100 popularity segments, for a smoothed trend view.

higher ratio of positive ratings than unpopular items, and popularity recommendation will degrade to random precision otherwise. On the other hand, it can be seen that if $\mathcal{C} \supsetneq T$ (i.e., the TI policy is not adhered to), then $E[P_{RND}@n]$ would get reduced by a factor of $|T|/|\mathcal{C}|$.

For a non-personalised recommender in a 1R design, elaborating from Equations (2) and (3) we get:

$$\frac{|R|}{n\ t\ |PR|}\ \underset{u\in\mathcal{U}}{avg}\sum_{k=1}^{n}\frac{\left|PR(i_k^{u,s})\right|}{\left|R(i_k^{u,s})\right|} \le E[1RP_s@n] \le \frac{|R|}{n\ t\ |PR|}\ \underset{u\in\mathcal{U}}{avg}\sum_{k=1}^{m_u}\frac{\left|PR(i_k^{u,s})\right|}{\left|R(i_k^{u,s})\right|},$$

an equivalent situation where the measured precision of popularity recommendation is bound by the potential dependence between the ratio of positive ratings and popularity.

Figure 7 shows this ratio as $|PR(i)|/|R(i)|$ with respect to the item popularity rank in MovieLens 1M. It can be seen that indeed the ratio grows with popularity in this dataset, which does lend an advantage for popularity recommendation. Even so, we may expect the bias to be moderate – but this has to be tested empirically, as it depends on the dataset. Note also that in applications where all ratings are positive (as, e.g., in our Last.fm setup), popularity – and any non-personalised recommender – would drop exactly to random precision ($E[P_s@n] = \sigma\ \delta$ in AR and $1/t$ in 1R).

A limitation of this approach is that the formation of T may impose limits on the value of σ, and/or the size of T. If the popularity distribution is very steep, T may turn out small and therefore biased to a few popular items. Moreover, there is in general a solution for T only up to some value of σ – it is easy to see (formally, or just visually in Figure 6b) that as $\sigma \to 1$ there is no item for which $(1-\varepsilon)k|R(i_k)|/|R| \ge \sigma$, unless the popularity distribution was uniform, which is never the case in practice. We have however not found these limitations to be problematic in practice, and common configurations turn out to be feasible without particular difficulty. For instance, in MovieLens 1M we get $|T|$ = 1,703 for $\sigma = 0.2$ with ε = 0.2 (allowing for a 5-fold cross-validation), resulting in η = 118 test ratings per item.

This method can be used, as noted, in both the AR and 1R approaches. We shall refer to these combinations as **UAR** and **U1R** respectively, where 'U' stands for the "uniform" number of item test ratings. In U1R it is important to set $\mathcal{C} = T$ in order to sample non-relevant items within T (i.e., $N_u \subset T$, for the TI policy). Otherwise, popularity would have a statistical advantage over other recommenders, as it would systematically rank irrelevant items in $N_u \setminus T$ below any relevant item in T, whereas other

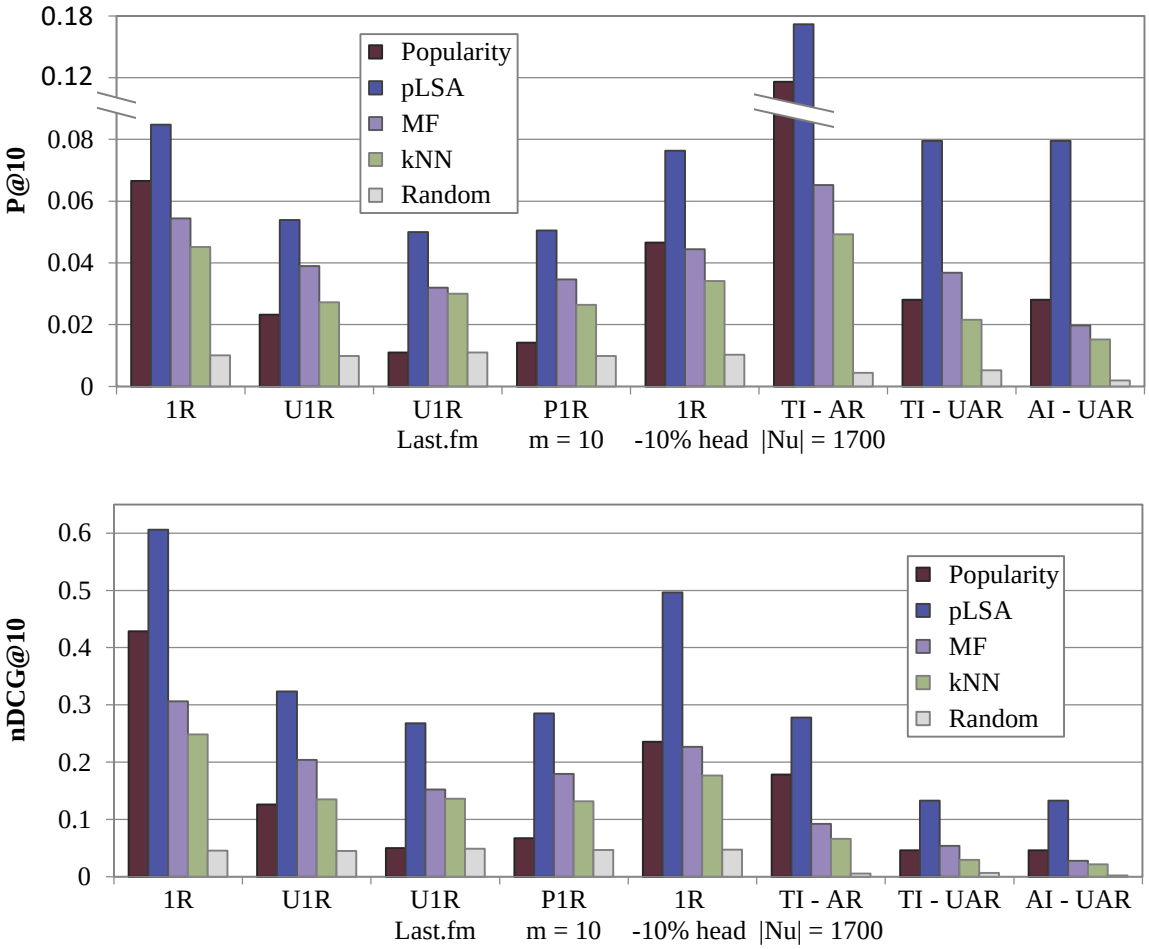

Figure 8. Precision and nDCG of recommendation algorithms on MovieLens 1M (and Last.fm only where indicated) using the 1R, U1R, P1R (m = 10 percentiles), AR, and UAR methodologies. The "-10% head" bars show the effect of removing the 10% most popular items from the test data (Cremonesi et al., 2010).

algorithms might not. The same can be considered in UAR, unless the experimental setup requires $|T_u| > |T|$, as, e.g., in the AI design. In that case a slight popularity bias would arise, as we shall see next.

6.3. Experimental Results

Figure 8 compares the results measured by 1R, AR and their corresponding popularity-neutralising variants. The setup is the same as in previous sections, except that for AR, we take TI-NN with $|N_u|$ = 1,700, to level with UAR in random precision. All the results correspond to MovieLens 1M except Last.fm where indicated. It can be seen that P1R, U1R and UAR effectively limit the popularity bias. The techniques seem to be more effective on 1R than AR: U1R and (even more) P1R actually place the popularity algorithm by the level of random recommendation, whereas the measured popularity precision decreases in UAR, but remains above kNN. The advantage of popularity over randomness in U1R and P1R is explained by the bias in the ratio of positive ratings in popular items (Figure 7). This ratio is constant in Last.fm, whereby popularity drops to random in U1R, as predicted by our analysis in the previous section, proving that the popularity bias remaining in the uniform-test approach is caused by this

factor. This residual bias is higher in U1R than P1R, because in the former, N_u is sampled over a larger popularity interval ($|T|$ = 1,703 vs. $|\mathcal{I}|$ / 10 = 370 items), giving a higher range for advantage by popularity, which also explains why the latter still overcomes kNN in UAR. We may observe the importance of using the TI policy in UAR, without which (in AI-UAR) a higher bias remains. We also show the effect of removing the 10% most popular head items from the test data (and also from $\mathcal{C}$, i.e., they are excluded from N_u sampling) in 1R, as a simple strategy to reduce the popularity bias (Cremonesi et al., 2010). We see that this technique reduces the measured precision of popularity, but it is not quite as effective as the proposed approaches.

It is finally worth emphasising how **the percentile and uniform-test approaches discriminate between pure popularity-based recommendation and an algorithm like pLSA**, which does seem to take popularity as one of its signals, but not the only one. The proposed approaches allow uncovering the difference, neutralising popularity but not pLSA, which remains the best algorithm in all configurations.

As we mentioned in Section 3, we have taken precision as a simple and common metric for our study, but all the proposed alternatives can be used straightforwardly with other standard IR metrics, such as MAP, nDCG, or MRR. Their application is direct in the AR setting; and they can be applied in 1R by simply introducing them in place of precision in the internal summation of Equation (1). Figure 8 shows results for nDCG, where we see that the analysed patterns hold just the same (we use the rating values above the relevance threshold as relevance grades for the computation of this metric). Results in terms of recall, MAP and MRR show quite the same trends. The AR approach provides room for a slightly wider metric variety than 1R, in the sense that some metrics reduce to each other in 1R. For instance, for a single relevant item, MAP is equivalent to Mean Reciprocal Rank (MRR $= 1/k$ where k is the rank of the first relevant item). And nDCG becomes insensitive to relevance grades in 1R (the grade of the single relevant item cancels out by the metric normalisation), whereas grades do make a difference in AR.

7. Related Work

To the best of our knowledge, the sparsity bias in recommender system evaluation has not been explicitly studied and analysed formally before. The development of algorithms that are able to cope with the sparsity of data is indeed a prominent research problem addressed in the literature, in the context of cold-start situations (being able to recommend new items and/or work with new users, see e.g. Kluver and Konstan, 2014). However, as we have presented in the paper, the data sparsity also affects the behaviour of the evaluation metrics, mainly as a statistical bias that does not reflect any actual variation of the true recommendation accuracy, but a perturbation on the range of the evaluation metric values. This perspective, as far as we know, has not been explored in depth in the field.

The popularity bias and its effects on recommendations started to be reported in recent years as an explicit issue, aside from the novelty/diversity problem or the existence of niche items (Jannach et al., 2015; Fleder and Hosanagar, 2007; Celma and Cano, 2008; Levy and Bosteels, 2010). In (Cremonesi et al., 2010), the authors acknowledged the fact that recommending popular items is trivial; hence, to successfully assess the accuracy of the recommendation algorithms presented in that paper, the authors partitioned the test data in two subsets: one only including very popular items (short-head) and another with the rest of the items (long-tail). For the second subset, the very popular items were excluded from the test set, a strategy followed by other authors to discriminate between pure popularity-based and personalised recommendations (Shi et al., 2011; Shi et al., 2012; Chen and Pan, 2013). As we have presented in Figure 8, this method is not as effective as the approaches we propose while, at the same time, our methods allow a more realistic setting where all the available information is used.

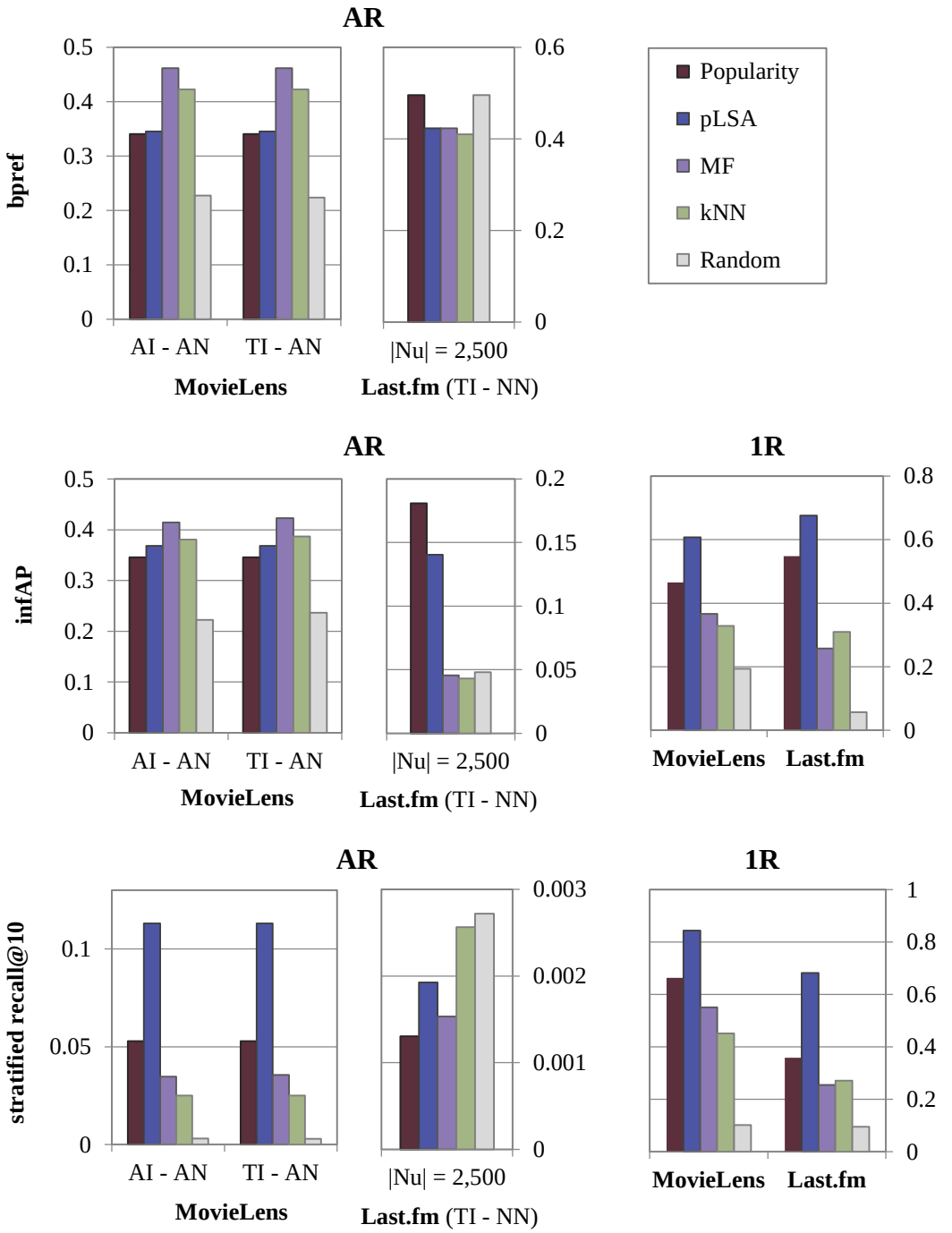

Figure 9. bpref (top), infAP (middle) and stratified recall with $\gamma = 1$ (bottom) of recommendation algorithms on MovieLens 1M and Last.fm using AR and 1R configurations. The displayed configurations are the same as in Figure 2 for P@10, for comparison.

A different perspective was studied by Jannach et al., 2015), who explored several variations of recommendation algorithms and analysed how they affect the popularity bias and, as an obvious trade-off in offline experiments, their final performance in terms of accuracy. They observed that less popular

items could be recommended with only small compromises on accuracy, although this may result in overfitted models, and hence, less generic and adjustable to any context. Upon a probabilistic reconstruction of memory-based collaborative filtering, Cañamares and Castells (2017) found deeper, structural reasons why neighbour-based algorithms are biased towards recommending popular items. Such findings thus confirm the fact that collaborative filtering algorithms are indeed biased towards popular items, and some techniques are proposed (by Jannach et al.) to somewhat straighten the bias. Research in this strand therefore addresses the problem at the algorithmic level. Our work is thus complementary in the perspective that we focus instead on the metrics and methodology that evaluate the algorithms.

In the IR field, metrics have been proposed with the specific purpose of better dealing with situations where relevance judgments are far from complete, such as bpref (Buckley and Voorhees, 2004) and infAP (Yilmaz and Aslam, 2006). The former can be thought of as the inverse of the fraction of judged irrelevant documents that are retrieved before relevant ones, whereas the latter is based on subsampling from the judgment pool and estimating the average precision from the subsample directly.

$$\text{bpref}\,(u) = \underset{i \in PR_{test}(u)}{avg}\left(1 - \frac{|\{j \in NR(u) \mid \tau_s^u(j) < \tau_s^u(i)\}|}{min(|PR_{test}(u)|, |NR(u)|)}\right)$$

$$\text{infAP}\,(u) = \underset{i \in PR_{test}(u)}{avg}\ \frac{1}{\tau_s^u(i)} + \frac{\tau_s^u(i) - 1}{\tau_s^u(i)}\left(\frac{|d100_u|}{\tau_s^u(i) - 1} \cdot \frac{rel\left(u, \tau_s^u(i)\right)}{rel\left(u, \tau_s^u(i)\right) + nrel\left(u, \tau_s^u(i)\right)}\right)$$

where we recall that $\tau_s^u(i)$ is the rank position of item i by system s for user u, $NR(u) = R_{test}(u) \setminus PR(u)$ represents the set of judged non-relevant items for a specific user u, and $rel(u,k) = |\{j \in PR_{test}(u) |\ \tau_s^u(j) \leq k\}|$ and $nrel(u,k) = |\{j \in NR(u) |\ \tau_s^u(j) < k\}|$ denote the number of relevant and not-relevant items presented above position k in the ranking.

For the sake of comparison, we present in Figure 9 the behaviour of these metrics for the same configurations we showed in Figure 2. If we compare the two figures, we observe that bpref and infAP do counter the popularity bias in the AR option for MovieLens. To this extent, these metrics could be seen as a good alternative to our proposed approaches. However, bpref and infAP seem to drag down pLSA along with popularity, whereas our methods clearly distinguish pLSA from popularity, ranking the former above all systems. Moreover, contrary to our approaches, popularity is still ranked first by bpref and infAP in Last.fm (AR option) Furthermore, the metrics seem to overestimate the performance of the random recommender in AR, to the point that random recommendation is the top system in Last.fm according to bpref, tied with popularity (and the third system by infAP, above MF and kNN), which we might regard as a shortcoming of these metrics..

We should also note that bpref does not make sense in the 1R configuration as the non-relevance judgments are removed in this option, and bpref returns 1 for all systems because a single relevant item is sampled per run (we therefore omit 1R for bpref in the figure). In fact, so-called implicit feedback datasets (the most common for recommender systems in real applications) such as Last.fm do not consider negative relevance at all, and bpref mostly loses meaning and becomes rather a measure of coverage (how many relevant items the system is able to compute a score for, related to the fact that the ranking function of most collaborative filtering algorithms is undefined for some user-item pairs), which explains the high metric value for random recommendation and popularity in Last.fm (as they reach full coverage). On 1R, infAP seems to display a similar bias for popularity as P@10 was shown to have in Figure 2, contrary to our proposed approaches, which are more consistent across all configurations.

Some bias-aware metrics have been proposed in the RS field as well. A first example can be found in (Shani et al., 2008), where a modified exponential decay score (a metric similar to nDCG proposed in Breese et al., 1998) was introduced so that higher scores are given to less popular items, by introducing an

item factor in the metric that accounts for the inverse item popularity. A more principled variation was proposed in (Steck and Xin, 2010) and (Steck, 2011), where the authors defined a popularity-stratification weight for the recall metric to account for the fact that ratings are missing not at random. The authors modify the recall metric by introducing weights based on the probability of observing a relevant rating for an item, these probabilities being a smooth function of the items' probabilities (more specifically, a polynomial relationship is assumed by the authors, allowing to consider different degrees depending on the missing data mechanism). In terms of our own notation used along the paper, the metric has the following definition:

$$\text{stratified-recall}(u;s)@n = \sum_{i \in top_S^u(T_u,n) \cap PR_{test}(u)} |PR(i)|^{-\frac{\gamma}{\gamma+1}} \Big/ \sum_{i \in PR_{test}(u)} |PR(i)|^{-\frac{\gamma}{\gamma+1}}$$

The parameter γ sets how aggressive the popularity compensation should be, ranging from $\gamma = 0$, for which stratified recall is equal to plain recall, to $\gamma = \infty$, for which the popularity penalization is linear to the number of positive ratings. Stratified recall can be seen as a reversed equivalent of relevance-aware long-tail novelty metrics proposed by Vargas and Castells (2011), which consider the option to weight novelty metrics (such as item "unpopularity") by the item relevance – whereas stratified recall weights relevance by unpopularity.

Steck (2011) reports experimental results for $\gamma = 1$ and ∞. We show in Figure 9 (bottom) the results with this metric for $\gamma = 1$ as an intermediate value, on the same configuration as in Figure 2 for P@10. We can see that except in AR for Last.fm, stratified recall does not differ much from precision as far as the popularity bias is concerned: popularity remains the second best algorithm according to the metric. In AR on Last.fm, stratified recall highly overemphasises the performance of random recommendation to the point of ranking it as the top system. It thus seems that, in these tests, the metric either falls short in penalising popularity, or overdoes it. A more balanced middle ground might likely be found from some appropriate, domain and configuration-dependent, value of γ. Besides the burden of tuning, choosing the appropriate parameter value would seem like a somewhat arbitrary decision for which we do not find straightforward criteria.

8. Conclusions

The application of Information Retrieval methodologies to the evaluation of recommender systems is not necessarily as straightforward as it may seem. Hence, it deserves close analysis and attention to the differences in the experimental conditions, and their implications on the explicit and implicit principles and assumptions on which the metrics build. We have proposed a systematic characterisation of design alternatives in the adaptation of the Cranfield paradigm to recommendation tasks, aiming to contribute to the convergence of evaluation approaches. We have identified assumptions and conditions underlying the Cranfield paradigm which are not granted in usual recommendation experiments. We have detected and examined resulting statistical biases, namely test sparsity and item popularity, which do not arise in common test collections from IR, but do interfere in recommendation experiments. Sparsity is clearly a noisy variable that is meaningless with respect to the value of a recommendation. Whether popularity is in the same case is less obvious; we propose experimental approaches that neutralise this bias, leaving way to an unbiased observation of recommendation accuracy, isolated from this factor. With a view to their practical application, we have identified and described the pros and cons of the array of configuration alternatives and variants analysed in this study.

In general we have found that evaluation metrics computed in AR and 1R approaches differ in how they are averaged. This means, more specifically, that precision obtained by approaches following a 1R

design is bound linearly by precision of AR approaches. Moreover, we have observed that a percentile-based evaluation considerably reduces the margin for the popularity bias, although the main limitation of this approach is that it specifies a constraint on the size of the possible target sets. Additionally, a uniform-test approach removes any statistical advantage provided by having more positive test ratings. Furthermore, we have found that both approaches discriminate between pure popularity-based recommendation and an algorithm like pLSA.

Popularity effects have started to be reported recently in recommender systems research and practice. Our research complements such findings by seeking principled theoretical and empirical explanations for the biases, and providing solutions within the frame of IR evaluation metrics and methodology – complementarily to the potential definition of new special-purpose metrics (Steck, 2011). The extent to which popularity is a noisy signal may be further analysed by developing more complete metric schemes incorporating gain and cost dimensions, where popular items would expectably score lower. Such metrics may, e.g., account for the benefits (to both recommendation consumers and providers) drawn from novel items in typical situations (Vargas and Castells, 2011), as a complement to plain accuracy. Online tests with real users should also be valuable for a comparative assessment of offline observations, and the validation of experimental alternatives. Finally, we think further central questions could be properly addressed and answered by the construction of new collections under appropriate conditions, e.g. devoid of certain biases in the data. Such an endeavour is most possibly out of the reach of a single research group, and certainly requires solving new practical problems, but we think it is feasible. In such an initiative it should be wise to tap into the knowledge, experience and lessons learned over more than two decades around the TREC community.

Acknowledgments

This work was partially supported by the national Spanish Government (grants nr. TIN2013-47090-C3-2 and TIN2016-80630-P). We wish to express our gratitude to the anonymous reviewers whose insightful and generous feedback guided us in producing an enhanced version of the paper beyond the amendment of flaws and shortcomings.

References

Armstrong, T. G., Moffat, A., Webber, W., and Zobel, J. (2009a). Has ad-hoc retrieval improved since 1994? In *Proceedings of the 32nd ACM conference on Research and development in Information Retrieval, SIGIR '09*, pages 692–693. ACM.

Armstrong, T. G., Moffat, A., Webber, W., and Zobel, J. (2009b). Improvements that don't add up: Ad-hoc retrieval results since 1998. In *Proceedings of the 18th ACM conference on Information and knowledge management, CIKM '09*, pages 601–610. ACM.

Baeza-Yates, R. and Ribeiro-Neto, B. (2011). *Modern Information Retrieval: The Concepts and Technology behind Search (2nd Edition) (ACM Press Books)*. Addison-Wesley Professional.

Barbieri, N., Costa, G., Manco, G., and Ortale, R. (2011). Modeling item selection and relevance for accurate recommendations: a bayesian approach. In *Proceedings of the 5th ACM conference on Recommender systems, RecSys '11*, pages 21–28. ACM.

Basu, C., Hirsh, H., and Cohen, W. W. (1998). Recommendation as Classification: Using Social and Content-Based Information in Recommendation. In *Proceedings of AAAI/IAAI '98*, pages 714–720.

Bellogín, A., Castells, P., and Cantador, I. (2011). Precision-oriented evaluation of recommender systems: an algorithmic comparison. *In Proceedings of the 5th ACM conference on Recommender systems, RecSys '11*, pages 333–336. ACM.

Breese, J. S., Heckerman, D., and Kadie, C. (1998). Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence, UAI '98*, pages 43–52.

Buckley, C., Dimmick, D., Soboroff, I., and Voorhees, E.M. (2007). Bias and the limits of pooling for large collections. *Information Retrieval* 10(6): 491–508.

Buckley, C. and Voorhees, E.M. (2004). Retrieval evaluation with incomplete information. In *Proceedings of the 27th ACM conference on Research and development in Information Retrieval, SIGIR '04*, pages 25–32. ACM.

Cañamares, R. and Castells, P. (2014). Exploring social network effects on popularity biases in recommender systems. In *Proceedings of the 6th Workshop on Recommender Systems and the Social Web, RSWeb '14, at the 8th ACM Conference on Recommender Systems, RecSys '14*.

Cañamares, R. and Castells, P. (2017). A Probabilistic Reformulation of Memory-Based Collaborative Filtering – Implications on Popularity Biases. In *Proceedings of the 40th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'17*. ACM.

Celma, O. (2010). *Music Recommendation and Discovery: The Long Tail, Long Fail, and Long Play in the Digital Music Space* (1st edition). Springer.

Celma, O. and Cano, P. (2008). From hits to niches?: Or how popular artists can bias music recommendation and discovery. In *NETFLIX '08: Proceedings of the 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition*, pages 1–8. ACM.

Celma, O. and Herrera, P. (2008). A new approach to evaluating novel recommendations. In *Proceedings of the 2nd ACM conference on Recommender systems, RecSys '08*, pages 179–186. ACM.

Chen, L. and Pan, W. (2013). Cofiset: Collaborative filtering via learning pairwise preferences over item-sets. In *Proceedings of the 13th SIAM International Conference on Data Mining*, pages 180–188.

Cremonesi, P., Koren, Y., and Turrin, R. (2010). Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the 4th ACM conference on Recommender systems, RecSys '10*, pages 39–46. ACM.

Fleder, D. M. and Hosanagar, K. (2007). Recommender systems and their impact on sales diversity. In *Proceedings 8th ACM Conference on Electronic Commerce (EC '07)*, pages 192–199.

Harper, F. M. and Konstan, J. A. (2016). The MovieLens Datasets: History and Context. *ACM Transactions on Interactive Intelligent Systems* 5(4): art. No. 19.

Herlocker, J. L., Konstan, J. A., Terveen, L. G., and Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems* 22(1): 5–53.

Hofmann, T. (2004). Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems* 22(1): 89–115.

Jambor, T. and Wang, J. (2010a). Goal-Driven Collaborative Filtering – A Directional Error Based Approach. In Gurrin, C., He, Y., Kazai, G., Kruschwitz, U., Little, S., Roelleke, T., Rüger, S., and Rijsbergen, K. (eds.), *Advances in Information Retrieval*, vol. 5993, chapter 36, pages 407–419. Springer.

Jambor, T. and Wang, J. (2010b). Optimizing multiple objectives in collaborative filtering. In *Proceedings of the 4th ACM conference on Recommender systems, RecSys '10*, pages 55–62. ACM.

Jannach, D., Lerche, L., Kamehkhosh, I., and Jugovac, M. (2015). What recommenders recommend: an analysis of recommendation biases and possible countermeasures. *User Modeling and User-Adapted Interaction* 25(5): 427–491.

Kluver, D. and Konstan, J. A. (2014). Evaluating recommender behavior for new users. In *Proceedings of the 8th ACM Conference on Recommender Systems, RecSys '14*, pages 121–128. ACM.

Koren, Y., Bell, R., and Volinsky, C. (2009). Matrix Factorization Techniques for Recommender Systems. *Computer* 42(8): 30–37.

Levy, M. and Bosteels, K. (2010). Music recommendation and the long tail. In *1st Workshop On Music Recommendation And Discovery, WOMRAD '10, at the 4th ACM Conference on Recommender Systems, RecSys '10*.

Pradel, B., Usunier, N., and Gallinari, P. (2012). Ranking with non-random missing ratings: Influence of popularity and positivity on evaluation metrics. In *Proceedings of the 6th ACM Conference on Recommender Systems, RecSys '12*, pages 147–154. ACM.

Shani, G., Chickering, D. M., and Meek, C. (2008). Mining recommendations from the web. In *Proceedings of the 2nd ACM Conference on Recommender Systems, RecSys '08*, pages 35–42. ACM.

Shani, G. and Gunawardana, A. (2011). Evaluating Recommendation Systems. In Ricci, F., Rokach, L., Shapira, B., and Kantor, P. B. (eds.), *Recommender Systems Handbook*, chapter 8, pages 257–297. Springer.

Shi, Y., Karatzoglou, A., Baltrunas, L., Larson, M., Oliver, N., and Hanjalic, A. (2012). Climf: Learning to maximize reciprocal rank with collaborative less-is-more filtering. In *Proceedings of the 6th ACM Conference on Recommender Systems, RecSys '12*, pages 139–146. ACM.

Shi, Y., Serdyukov, P., Hanjalic, A., and Larson, M. (2011). Personalized landmark recommendation based on geotags from photo sharing sites. In *Proceedings of the 5th International Conference on Weblogs and Social Media*.

Steck, H. (2011). Item popularity and recommendation accuracy. In *Proceedings of the 5th ACM conference on Recommender systems, RecSys '11*, pages 125–132. ACM.

Steck, H. and Xin, Y. (2010). A generalized probabilistic framework and its variants for training top-*k* recommender system. In *Proceedings of the Workshop on the Practical Use of Recommender Systems, Algorithms and Technologies, PRSAT '10*, pages 35–42.

van Rijsbergen, C. J. (1989). Towards an information logic. *SIGIR Forum* 23(SI): 77–86.

Vargas, S. and Castells, P. (2011). Rank and relevance in novelty and diversity metrics for recommender systems. In *Proceedings of the 5th ACM conference on Recommender systems*, *RecSys '11*, pages 109–116. ACM.

Voorhees, E. M. (2001). The philosophy of information retrieval evaluation. In *Evaluation of Cross-Language Information Retrieval Systems, 2nd Workshop of the Cross-Language Evaluation Forum, CLEF '01, Revised Papers*, pages 355–370.

Voorhees, E. M. and Harman, D. K., editors (2005). *TREC: Experiment and Evaluation in Information Retrieval*. MIT Press.

Yilmaz, E. and Aslam, J.A. (2006). Estimating average precision with incomplete and imperfect judgments. In *Proceedings of the 15th ACM conference on Information and knowledge management*, *CIKM '06*, pages 102–111. ACM.