Evaluating recommendation and search in the labor market

Michael Reusens[a], Wilfried Lemahieu[a], Bart Baesens[a,b], Luc Sels[a]

[a] *Faculty of Economics and Business, KU Leuven, Naamsestraat 69, 3000 Leuven, Belgium*
[b] *Southampton Business School, University of Southampton, United Kingdom,*
Email: {michael.reusens, wilfried.lemahieu, bart.baesens, luc.sels}@kuleuven.be

---

Abstract

This study evaluates the most popular recommender system algorithms for use on both sides of the labor market: job recommendation and job seeker recommendation. Recent research shows the drawbacks of focusing solely on predictive power when evaluating recommender systems, which become especially prominent in job- and job seeker recommendation, where aspects such as reciprocity and item spread are two other vital performance metrics for the quality of recommendations. Besides evaluating using these extra metrics, we compare recommendation with search using free text search engines. We measure what is gained, and what is lost when consuming items (jobs and job seekers) retrieved using search versus items presented via a recommender system. Based on insights in date recommendation literature, we propose changes to rating matrix construction aimed at mitigating the drawbacks of recommendation in the labor market. Our results, obtained from extensive experimentation on three datasets gathered from the Flemish public employment services, show that popular recommender algorithms perform significantly worse than user search in terms of reciprocity. Furthermore, we show that by swapping the rating matrices between two sides of a reciprocal recommender context, we can outperform user search in terms of reciprocity with limited trade off in predictive power. The insights from this research can help actors in the labor market to better understand the positioning of recommendation versus search, and to provide better job recommendations and job seeker recommendations.

Keywords: recommender systems, reciprocal recommendation, job recommendation, job

1. Introduction

Recommender systems are an established solution for the information overload caused by various information systems [1, 2]. Rather than requiring users of an information system to plow through all available information, they aim to filter out only those pieces of information relevant to the user. More and more, they are being evaluated for use in the labor market, where they are mainly used for job- but also for job seeker recommendation [3, 4, 5].

Defining what makes a "good" job recommender system is not straightforward. The labor market is a complex system in which multiple parties have different objectives. Job seekers wish to find a job that is interesting for their personal situation, which can differ based on competences, location, family situation and many other possible attributes. Job providers open up vacancies, which they want to allocate to appropriate job seekers. Attributes such as experience, competences and personality are just a handful of potential key factors in deciding if a person matches well with a vacancy. A third party in this reciprocal context are public employment services (PES). Their goal is to help the two aforementioned parties find each other. PESs aim at maximizing the number of good matches between job seekers and job providers. Clearly, both job seekers and job providers can benefit from systems that help them find relevant vacancies and job seekers: job seeker recommendation and job recommendation respectively. Based on interviews with the Flemish PES, and on earlier studies in the reciprocal context of online dating [6], we define the following goals for both a job recommender system and a job seeker recommender system: **1) The recommended items should be in line with the user's previously shown interests.** Not only is this a popular way of evaluating general recommender system quality, but this generally also leads to reduced recommendation risk [7]. Users that recognize their own interest in the recommended items will less likely be offended because they are more likely

to understand why certain items are recommended and complain about recommendation quality, which would be bad for the PESs public image. This goal is usually evaluated by measuring a recommender system's predictive power, using metrics such as root mean squared error, accuracy, recall, etc. In recent years, more and more voices in literature are raising the issue that evaluating recommender systems purely by looking at predictive power has serious drawbacks [8]. To address this concern, we introduce two extra goals. **2) Recommender systems that recommend items that reciprocate interest are better.** Even if a recommended job (or job seeker) is interesting for a user (measured by goal 1), if this interest is not reciprocated there will be almost no chance of a successful job - job seeker match. **3) Recommender systems that recommend a wider set of items are preferred over a recommender that recommends the same items over and over again.** This aligns with the need of treating everyone fairly being a public institution. It is undesirable for one job or job-seeker to be recommended often, while others are never recommended [9]. Furthermore, a more diverse set of recommendations can help users think out of the box, potentially discovering serendipitous opportunities. We leave some recommender specific goals, such as cold start solutions, out of scope because they have been extensively studied in earlier recommender system comparisons [10, 11]. We also limit our study to implicit-feedback based data: item clicks and item saves. Content-based recommender systems are out of scope. In previous work we showed that for job recommendation, explicit feedback can often be a poor indicator of user interest, compared to implicit feedback [3].

Free text search engines are a popular alternative to recommender systems for information filtering. Search engines work by taking a user-provided query, such as free text (e.g. "CEO position in Brussels") and return a ranking of the available items in order of best- to worst match with the search query. Note that there are differences between search and recommendation. Search is considered active information retrieval: the user triggers the retrieval by consciously providing a search query. Recommendation is considered pas-

sive information retrieval: items are automatically retrieved, without conscious user input. Both search engines and recommender systems are extremely popular, however a comparison between both with regard to how well they achieve their goals (as defined by goals 1, 2 and 3) is lacking in literature. As part of this research we will evaluate what is won, and what is lost when consuming items coming from recommendation versus items a user searched for themselves. This evaluation is of key importance for organizations that want to better understand the impact of introducing recommender systems as an extra information channel to its users. Furthermore, organizations are facing an information overload themselves, as a plethora of recommender system algorithms exist, and are continuously being developed. These algorithms range from naive baselines (E.g. random or most-popular recommendation, which are very easy to implement) to the most well known personalized algorithms (E.g. User-user or item-item collaborative filtering, which are readily available in almost every machine learning package) to specialized solutions tailored for reciprocal recommendation contexts (which are shown to outperform the standard algorithms, but likely take more time to implement and deploy). This study aims at helping with this information overload problem by analyzing the performance of these families of available recommender systems compared to each other, and to user search.

Compared to previous work, discussed in Section 2, we make the following contributions.

1. We show that vacancies consumed by traditional unidirectional recommender systems have much lower reciprocity than items consumed via traditional user search. To the best of our knowledge, we are the first to make such a comparison.
2. We show that the reciprocity of recommender items can be increased beyond the reciprocity of user search, with limited trade-off in predictive power by varying the rating matrices of two sides of a reciprocal recommender setting.
3. We benchmark the most popular recommender system algorithms for use in both sides of the labor market on three real-life datasets, not only focusing on recall but also reciprocity and item spread. This way we provide the recommender system com-

munity with extra data on how well recommender systems can be expected to work in the context of the labor market, where little publicly available data is available. Especially on job seeker recommendation, very few experiments have been published.

The remainder of this paper is structured as follows. First, we discuss the related work on which this study builds (Section 2), followed by introducing the algorithms and rating matrix variations used in this study (Section 3). Next, we present the experimental set-up, evaluation metrics, results and discussion (Section 4). We conclude by summarizing our key findings and presenting interesting topic for further research (Section 5).

2. Related Work

An increasing body of literature on job recommenders as unidirectional recommendation exists. One of the earliest job recommendation studies is published in [12]. They perform unidirectional collaborative filtering job recommendation and showed how interest profiles can be constructed using multiple types of implicit user feedback on an e-recruitment platform. In recent years, other studies have applied other recommendation algorithms in a unidirectional way for job recommendation. Next to collaborative filtering, studies have been done with case-based reasoning [13], cluster-based techniques [14], etc. For a comprehensive overview of job recommendation work, we refer to Al-Otaibi & Ykhlef [4] and Siting, Wenxing, Ning, & Fan [5]. However, most existing job recommendation approaches do not analyze the reciprocal nature of this recommendation task systems and only focus on the prediction of user interest [15]. [16] and [17] present a reciprocal job-job seeker matching approach based on a latent-factor model of explicit job and job seeker profiles. They only evaluate the recommender system based on accuracy and do not compare how a reciprocal job recommender compares to other (unidirectional) recommender algorithms. Furthermore, they are based on explicit user profiles rather than implicit interest data such as clicks, which is the focus of this study.

Another, richer, body of work in reciprocal recommendation comes from experiments in

the online dating context [6, 18, 19, 20, 21]. Online dating recommendation is in many ways similar to job recommendation, and job seeker recommendation. In both recommendation contexts, a recommended item is usually only a good recommendation in case interest is reciprocated. Furthermore, both systems recommend people, causing extra constraints with regards to time availability: A movie can be watched a million times, but a person will have a hard time going on a million dates or job interviews. Besides the similarities there are also prominent differences between the two recommendation domains, making it hard to generalize results in one context to the other one. In job recommendation or job seeker recommendation, users and items are disjoint and heterogeneous: a vacancy is described differently than a person, and its recommendation needs might be different. In online dating, users and items are not disjoint and not heterogeneous. In existing online dating studies, implicit feedback based algorithms have been shown to outperform systems based on explicit feedback [19, 22]. This further motivated our choice to limit our analyses to implicit feedback based recommender algorithms. They also compare reciprocal with unidirectional recommendation contexts, which we used as basis for the goals outlined in Section 1. Although mentioned by several of these authors that techniques developed for recommending dates could be applicable to the labor market, none of the proposed techniques have been evaluated on data from this context. Our proposed rating matrix transformations are based on ideas from this work and aim to bridge this gap in literature.

Comparing recommender systems with attention to a specific property has been frequently done in research. Recommender systems have been compared on behavior for new users [10], predictive power in online date recommendation [23] and others [24]. We follow the methodology used in these papers, especially with regards to algorithm - and evaluation metric selection.

3. Rating matrix and Recommender systems

In this Section, we present the algorithms used in the experiments: user-user collaborative filtering (UUCF), item-item collaborative filtering (IICF) , singular value decomposition recommendation (SVD) and three non-personalized baselines (random-, popularity- and inverse popularity recommendation). The algorithm selection is made based on a recent recommender comparison study [10], to which we added the inverse popularity baseline. We add this extra baseline algorithm because of goal 3: maximize item spread. We also propose methods of constructing the rating matrix in reciprocal settings, which we will combine with the aforementioned algorithms.

3.1. Reversing and combining rating matrices

In the remainder of this text, we will denote an arbitrary user as u, the set of all users as U, the x-th user as u_x, an arbitrary item as i, the set of all items as I and the y-th item as i_y. A rating of u on i is denoted as R_{ui}. R is also called the rating matrix used by a recommender system. Note that in reciprocal contexts, users are also items and, vice-versa: in job recommendation, the job seeker is considered the user and the job the item. In job seeker recommendation, the job (or job provider) is considered the user, and the job seeker the item.

Unidirectional recommender systems base themselves on interests shown by users for items. In a reciprocal context, people are both the user and the item, making the reverse interest a key factor in addressing our reciprocity goal. As part of our experiments we propose the variations of the rating matrix calculation used for recommendation presented in Table 1.

Table 1: Rating matrix configurations

Rating Matrix	**Description**
Standard Unidirectional	$R_{u,i} = 1$ if there was an interest indication from user u to item i

Popularity weighted Unidirectional	Weighted interests for each item i so that the $\sum_{u \in U} R_{u,i} = 1$
Reversed Unidirectional	$R_{u,i} = 1$ if there was an interest indication from item i to user u
Reciprocal	$R_{u,i} = 1$ if there was an interest indication from user u to item i OR from item i to user u
User-favored Reciprocal	$R_{u,i} = 1 + \alpha$ if there was an interest indication from user u to item i OR from item i in user u, with α nonzero in the first case

The intuition behind the reversed unidirectional and (user-favored) reciprocal setups is that in a reciprocal recommendation context the other party's interest is crucial and it makes sense to take that into account. We will experiment with the combinations of these rating matrices and the recommender algorithms discussed in Section 3.2. The goal is to evaluate how these popular algorithms will behave when using other than standard unidirectional interests as input. The Reciprocal, and User-favored Reciprocal rating matrices are based on earlier reciprocal work [20]. Note that we do not have a configuration in which we $R_{u,i} = 1$ if there was an interest indication from user u to item i AND from item i to user u. While it would make sense to include this in our benchmark, this matrix configuration resulted in too few ones. The rating matrix sparsity levels were too high making it impossible to generate recommendations given our experimental set-up. For this reason we left this configuration out of analysis. All rating matrix configurations in Table 1 will be used as input for the standard algorithms discussed in Section 3.2.

3.2. Recommendation algorithms

We use three established personalized recommender systems and three unpersonalized baseline recommenders in our study. The selection of these algorithms is based on a recent recommender system comparison study [10]. In this section we will give readers unfamiliar to these algorithms a brief high-level overview on how they work. We refer to the cited literature next to each algorithm for an in-depth description.

3.2.1. User-User Collaborative Filtering [25]

UUCF is one of the earliest personalized recommender algorithms. It is described in Algorithm 1. It works by first looking for users with similar interests (U_{sim}). In our study we use the Jaccard similarity coefficient between users' ratings for this. We also impose a minimum overlap size which expresses the minimum similarity a user must have to our target user in order to be included in U_{sim} as well a shrinking factor β parameter [7] to dampen the impact of highly similar users in neighborhood. These design choices are standard in literature [7].

Algorithm 1 UUCF: Which N items to recommend to user u_x

Require: Historical interest data for U and I: $R_{U,I}$

1: Calculate $U_{sim} = \{$users with similar interests as $u_x\}$
2: Calculate $I_{sim} = \{$items that users in U_{sim} found interesting$\}$
3: Recommend the N items (not yet seen by u_x) from I_{sim} that had the highest average interest from U_{sim}

3.2.2. Item-Item Collaborative Filtering [26]

IICF differs from UUCF because it uses item-item similarity instead of user-user similarity. Algorithm 2 describes the high-level intuition behind IICF: first find which items are liked by the same people, and then use this similarity to find items similar to items the user has shown to like in the past. We also use the jaccard similarity coefficient, minimal overlap size (now for items rather than users) and a shrinking β factor here.

Algorithm 2 IICF: Which N items to recommend to user u_x

Require: Historical interest data for U and I: $R_{U,I}$

1: Calculate $sim_{i_a - i_b}$ = {Item-item similarities, based on how often two items are liked by the same person}

2: Recommend the N items (not yet seen by u_x) with highest average $sim_{i_x - i_y}$ to items already liked by u_x

3.2.3. Singular Value Decomposition Recommendation [27]

Algorithm 3 shows the high-level strategy of the SVD-based recommendation algorithm. SVD-based recommendation works by first decomposing the large rating matrix R in two (or sometimes three) smaller matrices. The first matrix of the decomposition represents the interests of the users in the matrix rows. The last matrix represents the ratings on each items in the columns. Rather than looking at each individual user-item rating, this technique significantly reduces the dimensionality of the rating information. SVD-based recommendation gained a lot popularity since its succesful use in the Netflix recommendation challenge [28]. It usually outperforms UUCF and IICF, especially for large numbers of users and items. In our experiments, we vary the value for k (the larger the value, the more information is retained, but the more computationally intensive the algorithm becomes), and calculate the decomposition using the svds function of the rARPACK package [29].

Algorithm 3 SVD: Which N items to recommend to user u_x

Require: Historical interest data for U and I: $R_{U,I}$

1: Decompose R (a $|U|$ x $|I|$ matrix) in 2 matrices S (a $|U|$ x k matrix) and V (a k x $|I|$ matrix) so that $S * V \approx R$. Using V and S, R_{u_x,i_y} can be predicted by calculating $S[x,] * V[,y]$

2: Recommend the N items (not yet seen by u_x) with the highest predicted rating.

3.2.4. Naive baseline recommender systems

We also include a set of baseline algorithms to our study. These benchmarks show us the minimal performance a recommender system should have for it to be considered of any significant value to a user.

- Random: Non-personalized recommender that recommends random items. Each user gets a random sample of the available items as recommendations. This can be considered a true bottom line benchmark, however it is still included in many studies [10].

- Popularity: Non-personalized recommender that recommends the N most popular items. Popularity is measured in average interests per day. All users get the same recommendations. Recommending the N most popular items is a common recommendation strategy in many recommendation contexts, such as movie-recommendation and e-commerce.

- Inverse Popularity: Non-Personalized recommender that recommends the N random least popular items. Each user will get a random sample of all available items that are all equally (and most) unpopular. From a job- and job seeker recommendation perspective this one is interesting, since it aims for a more balanced spread of recommendations per item, which is in line with goal 3 presented earlier.

- Oracle: This algorithm simply recommends the items from the test-set (see Section 4). This simulates user search as it contains the items a user would consume if we let him look for items autonomously. We of course cannot use this for comparison of predictive quality, but it does give insight in what level of reciprocity (goal 2) and item spread (goal 3) users achieve on their own using search.

4. Experimental Setup and Results

This section presents the research questions tackled in this paper, the three datasets used in the experiments, the evaluation metrics used, and the experiments performed in order to answer the research questions.

Concretely, this research aims to answer the following questions:

1. How does user search perform with regard to reciprocity and item spread?

2. How do popular recommender algorithms perform with regard to predicting user interest, reciprocity and item spread? Do they outperform user search?
3. Can ratings from the other side of a reciprocal context help improve recommender systems?

Figure 1 presents a schematic representation of our experimental design and shows where each research question fits in. We look at the impact of recommendation algorithm and rating matrix construction choices on ability to predict user interest, reciprocity and item spread. We also compare this to how well user search performs with regard to reciprocity and item spread.

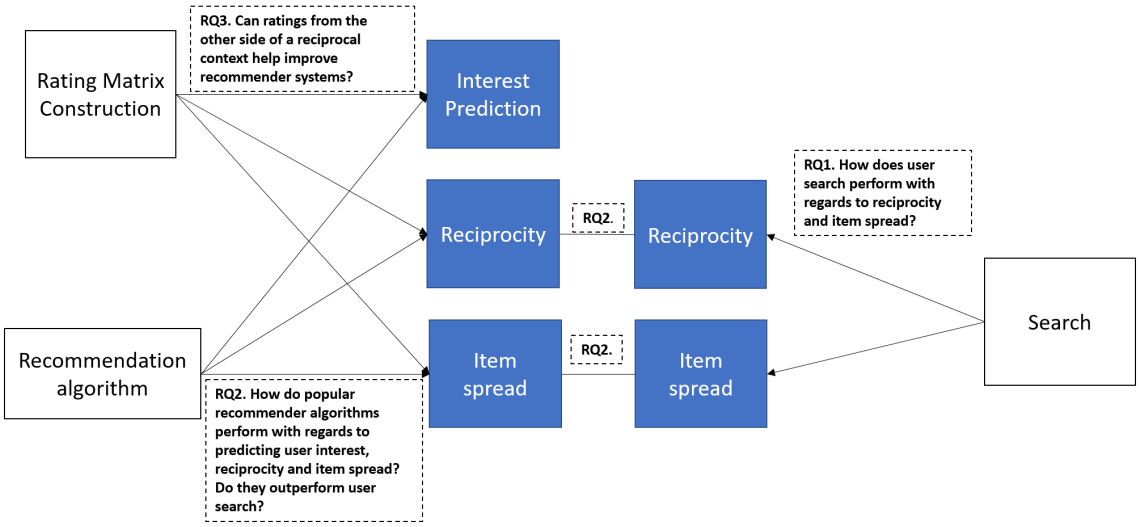

Figure 1: Schematic representation of experimental design

4.1. Data

The three datasets are presented in Table 2. All data was collected for the period June 22, 2015 - July 7, 2016, resulting in a span of 380 days. We collected click logs that are the result of job seeker searches on the 'Find a job' search tool available on the website of the Flemish PES (www.vdab.be). This is a simple keyword based search engine matching vacancies with a free text search query. This should mean that the data bias imposed by

Table 2: Interest datasets used

Dataset	**Rows**	**Unique users**	**Unique items**
Vacancies clicked	32,789,333	254,048	942,163
Vacancies saved	593,360	73,272	233,489
Job seekers saved	526,570	95,538	191,921

the search engine implementation by the PES is low, so that we can consider the data a clean indication of user interest. Multiple clicks can be linked to a single user by use of user name when users were logged in, or cookies if they had logged in earlier and received a cookie. Clicks of users that cleared their cookies and are not logged in could not be linked. As extra data from this side of the reciprocal context we also collected which vacancies were saved by which users for later inspection. We separate this data from the vacancy clicks because we had no control over the origin of the vacancy saves, meaning that the save could occur because an existing recommender system had recommended it to the user, thereby potentially biasing the dataset to its own believe of good recommendations. Note that we will not discuss how to best combine these 2 user interests. Work on how to combine different interest indicators has already been done in [30].

From the other side of the reciprocal spectrum, no click data was available that could be linked back to a single vacancy. We did collect client-save data, which is the result of job seekers being saved for a specific vacancy by job providers for later inspection.

All three data sources are unary: $R_{u,i}$ is 1 if u clicked on or saved i, and NA otherwise. The unary nature of the data is why the Jaccard index is used in UUCF and IICF rather than alternatives such as cosine distance of pearson correlation coefficient.

4.2. Evaluation metrics

In the experiments, we will measure how well we achieve the recommendation goals presented earlier using the metrics presented in Table 3. These metrics are based on previous recommender comparison studies [10].

Table 3: Evaluation metrics

Metric	**Explanation**
recall@25	The percentage of 25 left-out items that are found in top-25 recommendation. (This equals precision@25). Range: [0,1].
click reciprocity (only available for job seeker recommendation)	The percentage of items that clicked on the user. Range: [0,1].
save reciprocity	The percentage of items that saved the user to their profile. Range: [0,1].
item spread	The number of unique recommended items divided by the total number of recommended items. Range:]0,1].

We have chosen one metric for each recommendation objective. We use recall to measure how well the recommendations are in line with the user's interest. It is frequently used in literature [3] and usually correlates strongly with metrics that measure similar behavior, such as RMSE and NDCG [10]. Also note that since we recommend 25 items in a leave-25-out experiment, recall will result in the same value as precision. As with any recommender system study, it is important to keep in mind that there is a potential discrepancy between the relevancy measured using offline experiments, and the actual quality of a recommender system perceived by the user. In case of our experiments, we are quite confident that relevancy measured using recall is in line with relevancy for the the end user because earlier work on this data [3] showed that higher recall of a recommender system resulted in higher human appreciation of the recommendations.

4.3. Setup

The recommender systems were evaluated using a leave-N-out strategy in which the last N (N = 25) interest observations are held out as test set. The remaining earlier

shown interests are used to run the recommender system algorithm on. We employ top-25 recommendation, similar to other benchmarking studies [10], based on at least 5 interest signals. This limits us to users in our dataset with at least 30 clicks, or 30 saves performed. Our study leaves an analysis of cold-start behavior of algorithms out of scope, as other studies have already analyzed cold-start behavior of the algorithms used [10].

1,500 job seekers (once for the saved- and once for the clicked vacancies dataset) and 1,500 vacancies were randomly sampled to generate recommendations for. For each algorithm in our experiments, and varying parameter values for these algorithms, we generated top-25 recommendations for the 1,500 job seekers and 1,500 vacancies and evaluated their performance using the metrics presented in table 3.

The parameters MIN_OVERLAP, β (for UUCF and IICF) and k (for SVD) parameters were selected by optimizing for recall on a separate set of 1,500 randomly sampled users. In almost all cases optimal recall matched with optimal reciprocity and item spread. In the limited cases were it did not, the difference was small and the sub-optimality of reciprocity and item spread did not at all influence the conclusions we make from the results. The optimal value for MIN_OVERLAP was always 1, meaning that all users with at least 1 item interest as overlap are taken into account. Shrinking β and k had varying optimal parameters for the different experiments. Optimal β ranged from 0 to 25, optimal k ranged from 500 to 2,500. The full results of parameterizations are not included due to space constraints. All algorithms were implemented in R 3.2.2 [31].

4.4. Results

We will first discuss research questions 1 and 2, comparing user search with standard unidirectional recommendation. Next, we present the results of the rating matrix variations.

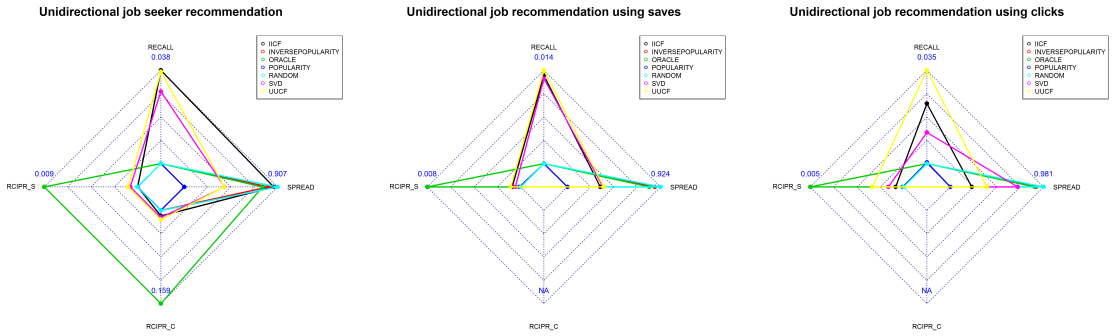

Figure 2: Standard job seeker recommendation versus search images

4.4.1. Search versus recommendation

Figure 2 shows the results for standard unidirectional job seeker recommendation compared to search. For each recommender algorithm only the best performing parameterization is shown. Note that the origin (value = 0) of each axis in the radar plots lies on the inner diamond, and not in the center of the image. The recall of user search is set to 0, while in reality it would be 1, for better image scaling. Since we only have save-based information from the employer's side, click-based reciprocity (RCIPR_C on the radar plots) is set to NA for job recommendation experiments.

As first observation on job seeker search, we see that job providers are often successful in terms of reciprocity in the job seekers they save for their vacancies. Of the job seekers saved by a job provider, 15.9% reciprocated the interest with a click and 0.9% by saving that vacancy.

User search significantly outperforms unidirectional IICF, UUCF and SVD in terms of (both click and save) reciprocity for all three datasets. There is no significant difference found[1] in the reciprocity metrics of unidirectional SVD, UUCF and IICF, so all three can be considered equally poor in this regard. This means that while these popular algorithms

[1]Significances mentioned between algorithms on the same set of users are measured using the Friedman test on a 95% confidence level.

help mitigate the information overflow problem by recommending items that are in line with a user's past interests [2], they do so at a cost: items looked at via standard unidirectional recommenders will less likely reciprocate potential interest.

The item spread of items observed using user search is in the neighborhood of the item spread resulting from the unidirectional recommender systems. No consistent differences between the item spread of user search and traditional unidirectional recommender systems can be observed.

For unidirectional job-seeker recommendation (Shown in Figure 2) IICF and UUCF had the best recall@25. There was no significant difference between their recall@25. SVD did perform significantly worse than the former two. In unidirectional save based job recommendation no significant recall difference could be found between IICF, UUCF and SVD. The recall scores were significantly lower than for job seeker recommendation and unidirectional click based job recommendation ($\pm 1.3\%$ compared to around $\pm 3\%$ for the other two algorithms). In click based job recommendation, UUCF was by far the better algorithm in terms of recall. It scored significantly better than IICF, which in turn scored better than SVD.

For item spread, we see conflicting results over the different datasets. In job seeker recommendation, IICF has a significantly higher item spread compared to UUCF and SVD between which no significant difference could be found. For save based job recommendation no significant differences could be found between UUCF, IICF and SVD. In click based recommendation we see almost the opposite of what we see for job seeker recommendation: SVD has a significantly higher item spread than UUCF, which has a higher item spread than IICF.

Because of the varying results, it is hard to make strong conclusions about which of the

[2]For the click based job recommendation, recalls between 3-4% on over 900,000 items to choose from are achievable, and all personalized algorithms strongly outperform the unpersonalized baselines.

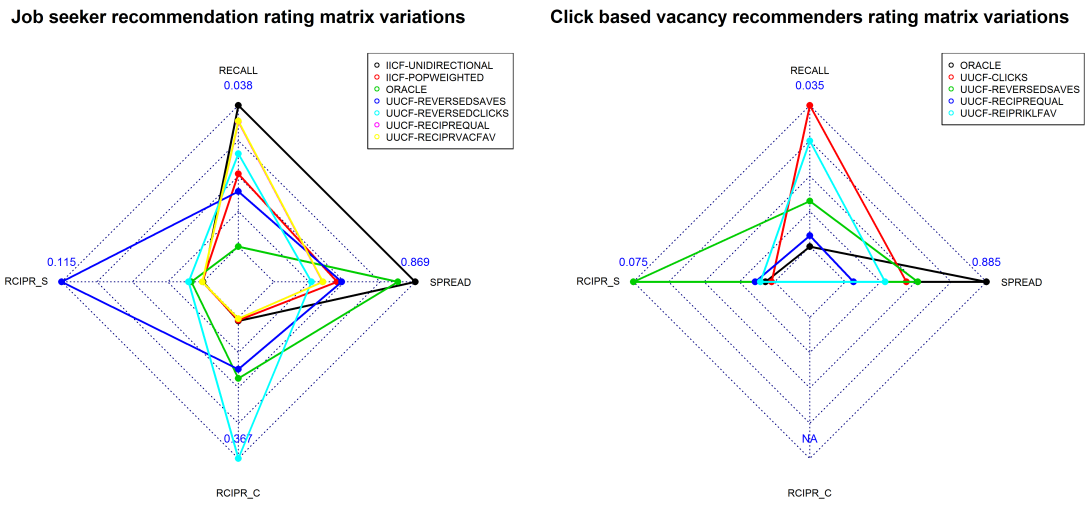

Figure 3: Rating matrix transformation approaches vs unidirectional recommendation

three unidirectional algorithms is most suited given the three goals put forward in Section 1. We can however conclude that they are severely lacking in reciprocity compared to the degree of reciprocity achieved by user search.

4.4.2. Impact of rating matrix construction

Figure 3 shows the main results for job seeker recommendation and click based job recommendation for the different rating matrix transformations, compared to user search, and compared to standard unidirectional recommendation. For each of the rating matrix transformations we show the top performing recommender algorithm. Unpersonalized algorithm baselines have been omitted from this figure given their poor performance. The conclusions for save based job recommendation are not shown in the figure, since they lead to the same conclusions.

When using a matrix transformation other than the unidirectional rating matrix, recall drops. This is to be expected, since recall measures how well we can predict values in the unidirectional matrix. By changing and rotating the rating matrix we are making

predictions out of domain, which is known to be more difficult. However, we argue that the drops in recall are not as strong given the benefits gained for some of the rating matrix transformations.

Looking at the reciprocity, the reversed unidirectional rating matrix set-up performs by far the strongest. More interestingly, this set-up even outperforms reciprocity obtained by user search.

There are mixed results in terms of item spread for the reversed unidirectional set-up. For job seeker recommendation it underperforms compared to the unidirectional set-up, but outperforms the unidirectional set-up for vacancy recommendation.

The (user-favored) reciprocal rating matrix approaches show underwhelming results. They lose in terms recall and item spread compared to unidirectional recommendation and gain not as much as the reversed unidirectional approach in terms of reciprocity. Again there are mixed results here: the reciprocal rating matrix outperforms user search in terms of reciprocity for vacancy recommendation, but underperforms in reciprocity for job seeker recommendation.

4.5. Discussion

We did not find consistent results when comparing recommender system algorithms (IICF, UUCF and SVD), so we cannot make a claim about which algorithm to best use for job recommendation or job seeker recommendation.

Unidirectional rating matrices result in the best recall, but they suffer in terms of reciprocity. Users consuming these vacancies would find a much lower level of reciprocated interest, than users looking at vacancies they search for themselves. This raises concerns about the dangers of losing certain qualitative aspects (in our case reciprocity) in the consumed information when depending on recommender systems. Our experiments clearly show that this concern is warranted. This result is not intuitive, as recommender systems base themselves on the search data. In future research, it is worthwhile to investigate

why the inference-step recommender systems make on the interest data leads to a drop in reciprocity.

By simply reversing the unidirectional rating matrix we achieve much higher levels of reciprocity in recommended items than with a standard unidirectional rating matrix. The level of reciprocity is even higher than that of the items searched for by the user himself/herself. This does however require a trade-off in terms of recall. Of the algorithm-rating matrix configurations presented, we believe reversed unidirectional results in the best compromise between goals 1, 2 and 3. The other non-unidirectional configurations did not result in favorable results compared to unidirectional recommendation or user search.

It is hard to generalize our conclusions to other domains, as it is not straightforward to pinpoint the causal factors behind our results. We do however have several future research ideas that could shed more light on this. When reversing the interests between the two sides of the reciprocal context, the recommendations are gravitating more from "here are vacancies you are likely interested in" towards "here are vacancies likely interested in you". When these two approaches have high overlap, the recall lost in the trade-off will be low, as shown in our experiments. We expect to find similar trade-off between reciprocity and recall for reciprocal recommendation systems in other reciprocal contexts where both sides of the reciprocal context tend to find each other equally well. In cases where both sides of the reciprocal context do not find each other well, we expect a steeper recall loss for increased reciprocity gains, compared to the trade-off in our experiments: When recommendations are generated based on which items are interested in a user, and the user is never interested in those items, the recall will likely be low. This does not mean that these low-recall recommendations are worthless, as in many reciprocal context it could be of interest who is (potentially) interested in you.

5. Conclusion

This study investigated the following research questions:

1. How does user search perform with regard to reciprocity and item spread?
2. How do popular recommender algorithms perform with regard to predicting user interest, reciprocity and item spread? Do they outperform user search?
3. Can ratings from the other side of a reciprocal context help improve recommender systems?

After an extensive comparison of multiple recommender system algorithms, with various rating matrix operations, on three datasets gathered form the Flemish public employment services, we were able to answer these questions.

How does user search perform with regard to reciprocity and item spread? Our results show that between 1% and 15% of items consumed by users via search reciprocate interest. Item spread of user search is around 76%-88% for the different datasets. This means that on average, each 100 items consumed by users will contain 82 unique items.

How do popular recommender algorithms perform according to recall, reciprocity and item spread? Do they outperform user search? We found that traditional recommender systems have good predictive power, but have strongly inferior reciprocal interest from recommended items, compared to items found by user search. In terms of item spread we found mixed results: For two datasets user search outperformed the best unidirectional recommender system and for one dataset unidirectional recommendation outperforms user search.

Can ratings from the other side of a reciprocal context help improve recommender systems? We propose several rating matrix transformations based on existing research in date recommendation, of which the simplest (reversed unidirectional) appeared to have the nicest properties: if the recommendation provider is willing to trade-off some predictive power strong increases in reciprocity that even outperform user search can be achieved. The item spread of recommender systems based on these rating matrix transformations is comparable to those using the standard unidirectional rating matrices.

Our results allow recommendation providers in the labor market (such as public employ-

ment services) to provide better job recommendations and job seeker recommendations, and to better position recommendation against user search.

5.1. Limitations and future work

We run our experiments on three datasets from the same PES. Running the same experiments on more reciprocal datasets, both from the labor market context and other reciprocal settings, such as online dating, would be valuable to further confirm and generalize our results beyond the labor market setting.

As already discussed in Section 4.2, no perfect offline evaluation strategy for recommendation systems exists. Given the dimensionality of our setup (in terms of number of algorithms, parameterizations, rating matrix variations,etc.), online confirmation of our results on real job seekers and job providers, in a live recommendation environment, proved to be infeasible. In previous experiments on the same data we saw a strong correlation between offline predictive power metrics and appreciation for the recommendations [3], but this strong correlation is shown to not always be the case [32], and our results should ideally be confirmed on real users.

We believe that the insights this work provides on the comparison between search and recommendation in terms of reciprocity, item spread and predictive power can lead to significant impact on the quality of the job (seeker) search process. However, a more longitudinal study is required to measure the exact impact of recommender systems with different levels of recall, reciprocity and item spread on the search process of job seekers and job providers. We are especially interested in the impact of reciprocity on job seeker motivation and user experience.

Acknowledgments: This research is supported by the Career Management Analytics research chair, sponsored by the Flemish PES: Vlaamse Dienst voor Arbeidsbemiddeling en Beroepsopleiding (VDAB).

References

[1] A. G. Schick, L. A. Gordon, S. Haka, Information overload: A temporal approach, Accounting, Organizations and Society 15 (3) (1990) 199–220.

[2] J. Bobadilla, F. Ortega, A. Hernando, A. Gutiérrez, Recommender systems survey, Knowl.-Based Syst. 46 (2013) 109–132. doi:10.1016/j.knosys.2013.03.012.
URL https://doi.org/10.1016/j.knosys.2013.03.012

[3] M. Reusens, W. Lemahieu, B. Baesens, L. Sels, A note on explicit versus implicit information for job recommendation, Decision Support Systems (2017) –doi:https://doi.org/10.1016/j.dss.2017.04.002.
URL http://www.sciencedirect.com/science/article/pii/S0167923617300611

[4] S. T. Al-Otaibi, M. Ykhlef, A survey of job recommender systems, International Journal of the Physical Sciences 7 (29) (2012) 5127–5142.

[5] Z. Siting, H. Wenxing, Z. Ning, Y. Fan, Job recommender systems: a survey, in: Computer Science & Education (ICCSE), 2012 7th International Conference on, IEEE, 2012, pp. 920–924.

[6] L. Pizzato, T. Rej, T. Chung, I. Koprinska, J. Kay, Recon: a reciprocal recommender for online dating, in: Proceedings of the fourth ACM conference on Recommender systems, ACM, 2010, pp. 207–214.

[7] F. Ricci, L. Rokach, B. Shapira, P. B. Kantor, Recommender Systems Handbook, 1st Edition, Springer-Verlag New York, Inc., New York, NY, USA, 2010.

[8] S. M. McNee, J. Riedl, J. A. Konstan, Being accurate is not enough: how accuracy metrics have hurt recommender systems, in: CHI'06 extended abstracts on Human factors in computing systems, ACM, 2006, pp. 1097–1101.

[9] M. Kunaver, T. Pozrl, Diversity in recommender systems - A survey, Knowl.-Based Syst. 123 (2017) 154–162. doi:10.1016/j.knosys.2017.02.009.
URL https://doi.org/10.1016/j.knosys.2017.02.009

[10] D. Kluver, J. A. Konstan, Evaluating recommender behavior for new users, in: Proceedings of the 8th ACM Conference on Recommender Systems, ACM, 2014, pp. 121–128.

[11] A. L. V. Pereira, E. R. Hruschka, Simultaneous co-clustering and learning to address the cold start problem in recommender systems, Knowl.-Based Syst. 82 (2015) 11–19. doi:10.1016/j.knosys.2015.02.016.
URL https://doi.org/10.1016/j.knosys.2015.02.016

[12] R. Rafter, B. Smyth, Passive profiling from server logs in an online recruitment environment, in: Workshop on Intelligent Techniques for Web Personalization at the the 17th International Joint Conference on Artificial Intelligence, Seattle, Washington, USA, August, 2001, 2001.

[13] K. Bradley, R. Rafter, B. Smyth, Case-based user profiling for content personalisation, in: Adaptive Hypermedia and Adaptive Web-Based Systems, Springer, 2000, pp. 62–72.

[14] W. Hong, S. Zheng, H. Wang, J. Shi, A job recommender system based on user clustering, Journal of Computers 8 (8) (2013) 1960–1967.

[15] W. Hong, L. Li, T. Li, W. Pan, ihr: an online recruiting system for xiamen talent service center, in: The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, IL, USA, August 11-14, 2013, 2013, pp. 1177–1185. doi:10.1145/2487575.2488199.
URL http://doi.acm.org/10.1145/2487575.2488199

[16] T. Keim, Extending the applicability of recommender systems: A multilayer framework for matching human resources, in: System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on, IEEE, 2007, pp. 169–169.

[17] J. Malinowski, T. Weitzel, T. Keim, Decision support for team staffing: An automated relational recommendation approach, Decision Support Systems 45 (3) (2008) 429–447.

[18] L. A. Pizzato, C. Silvestrini, Stochastic matching and collaborative filtering to recommend people to people, in: Proceedings of the fifth ACM conference on Recommender systems, ACM, 2011, pp. 341–344.

[19] P. Xia, B. Liu, Y. Sun, C. Chen, Reciprocal recommendation system for online dating, in: Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015, ACM, 2015, pp. 234–241.

[20] L. Pizzato, T. Rej, J. Akehurst, I. Koprinska, K. Yacef, J. Kay, Recommending people to people: the nature of reciprocal recommenders with a case study in online dating, User Modeling and User-Adapted Interaction 23 (5) (2013) 447–488.

[21] L. Li, T. Li, Meet: a generalized framework for reciprocal recommender systems, in: Proceedings of the 21st ACM international conference on Information and knowledge management, ACM, 2012, pp. 35–44.

[22] P. Xia, K. Tu, B. F. Ribeiro, H. Jiang, X. Wang, C. X. Chen, B. Liu, D. Towsley, Characterization of user online dating behavior and preference on a large online dating site, in: Social Network Analysis - Community Detection and Evolution, 2014, pp. 193–217. doi:10.1007/978-3-319-12188-8_9.
URL https://doi.org/10.1007/978-3-319-12188-8_9

[23] L. Brozovsky, V. Petricek, Recommender system for online dating service, arXiv preprint cs/0703042.

[24] A. Said, A. Bellogín, Comparative recommender system evaluation: benchmarking recommendation frameworks, in: Proceedings of the 8th ACM Conference on Recommender systems, ACM, 2014, pp. 129–136.

[25] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, J. Riedl, Grouplens: an open architecture for collaborative filtering of netnews, in: Proceedings of the 1994 ACM conference on Computer supported cooperative work, ACM, 1994, pp. 175–186.

[26] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in: Proceedings of the 10th international conference on World Wide Web, ACM, 2001, pp. 285–295.

[27] Y. Koren, Factorization meets the neighborhood: a multifaceted collaborative filtering model, in: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2008, pp. 426–434.

[28] R. M. Bell, Y. Koren, Lessons from the netflix prize challenge, SIGKDD Explorations 9 (2) (2007) 75–79. doi:10.1145/1345448.1345465.
URL http://doi.acm.org/10.1145/1345448.1345465

[29] Y. Qiu, J. Mei, authors of the ARPACK library. See file AUTHORS for details., rARPACK: Solvers for Large Scale Eigenvalue and SVD Problems, r package version 0.11-0 (2016).
URL http://CRAN.R-project.org/package=rARPACK

[30] L. Tang, B. Long, B.-C. Chen, D. Agarwal, An empirical study on recommendation with multiple types of feedback, in: Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, ACM, New York, NY, USA, 2016, pp. 283–292. doi:10.1145/2939672.2939690.
URL http://doi.acm.org/10.1145/2939672.2939690

[31] R Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria (2016).
URL https://www.R-project.org

[32] M. Rossetti, F. Stella, M. Zanker, Contrasting offline and online results when evaluating recommendation algorithms, in: Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016, 2016, pp. 31–34. doi:10.1145/2959100.2959176.
URL http://doi.acm.org/10.1145/2959100.2959176