JoLA: Job Landscape Aware Job Recommendation

Solal Nathan[1,*], Guillaume Bied[2], Elia Perennes[3,4], Philippe Caillou[1], Bruno Crepon[4], Christophe Gaillac[5] and Michèle Sebag[1]

[1]LISN, INRIA–CNRS UMR 9015,Université Paris-Saclay

[2]Ghent University, IDLab, Department of Electronics and Information Systems

[3]France Travail

[4]Centre de Recherche en Économie et de Statistiques (CREST), CNRS, École polytechnique, GENES, ENSAE Paris, Institut Polytechnique de Paris, 91120 Palaiseau, France

[5]University of Geneva, GSEM

Abstract

Job recommendation (JR), among the most critical challenges of AI, aims to alleviate frictional unemployment with major potential impacts on society and economy at large. However, Job Recommender Systems (JRS) might become counter-productive and create a congestion phenomenon, if job seekers are mostly recommended the most popular job ads.

This paper proposes a novel perspective on JRS, observing that the job market tends to involve a number of so-called "orphan" job ads, that receive very few or no applications. The orphan-job phenomenon is detrimental to the job market as it mechanically decreases the number of jobs effectively considered, worsening the market imbalance and increasing the congestion; in the long term, it also tends to prevent companies from publishing other ads, *de facto* creating a sleeping job market that is not revealed to the job seekers.

This paper introduces new JRS losses, aimed to prevent both the congestion and the orphan-jobs phenomenon, based on a differentiable approximation of the market share attributed to a job ad. The resulting so-called *Job Landscape Aware* recommender system (JoLA) is experimentally assessed and compared with the state of the art on public datasets, showing new trade-offs that exist between standard recommendation metrics and congestion, while enforcing the desired exposure for most ads. The JoLA code is publicly available at https://codeberg.org/solal/jola.

Keywords

Job recommender system, Congestion avoidance, Exposure in ranking, Popularity bias, Matching, Labor market

1. Introduction

Job Recommender Systems (JRSs), at the forefront of AI for good, are viewed as a promise to reduce frictional unemployment [1]. In the perspective of job seekers, JRSs can potentially alleviate the burden of finding a job best suited to them, among the many job ads available through diverse canals, ranging from e.g. LinkedIn [2, 3] or CareerBuilder to Public Employment Services (PES) [4]. In the perspective of recruiters, JRSs can symmetrically help finding job seekers best suited to a job ad [5].

JRSs build upon the massive development of recommender systems [6, 7], prompted by the deployment of e-commerce and entertainment platforms since the mid 1990s [6]. Yet, unlocking the full potential of these systems in the context of job platforms or PES requires taking into account specificities of the labor market [4], the more so given the high-risk nature of this AI application [8, 9].

Among these specificities is the fact that job recommendation is a multi-stakeholder, reciprocal problem [10, 11, 12]. Enforcing an appropriate distribution of the recommendations, as detailed below, is achieved by intervening in an in-processing or post-processing manner on the recommendation policy [13] beyond the standard probability ranking principle.

This paper focuses on enforcing two desired properties of the recommendation distribution.[1] The first one is to avoid the so-called *congestion* phenomenon. As job ads are *rival goods* – an item (job ad) can most generally be attributed to a single user – recommending a small subset of popular job ads to most users can create severe congestions, where many job seekers compete for the same job ads. The second one is that many job ads – referred to as *orphan jobs* in the following – actually receive few or no applications, *de facto* worsening the job market imbalance, and discouraging the affected companies from publishing more job ads on job platforms or at the PES.

How to avoid congestion and resist the bias toward popular items has been extensively investigated in the general recommender systems literature [15] and in the context of Job Recommender systems [16] (more in Section 4). How to ensure a fair share of users' attention to every item has also been extensively considered in the literature [17], although much less so in the JRS context [18] to our best knowledge. Still, the point of attracting job seekers to every job ad appears a strategic issue, in both perspectives of job seekers and recruiters. On the one hand, recommending orphan jobs to job seekers contributes to alleviate the congestion and the competition on the job market. On the other hand, the fact that companies receive more (or at least some) applications on every job ad might incentivize them to publish more and possibly more diverse job ads. Actually, some practitioners suggest that a significant part of the jobs to be fulfilled are *sleeping ones*: they are not published on the PES platforms as the recruiters feel that they would receive no applications.[2]

This paper focuses on the design and assessment of a job recommender system addressing both issues of congestion and orphan jobs, with two main contributions. First, we propose using a differentiable approximation of the market

RecSys in HR'25: The 5th Workshop on Recommender Systems for Human Resources, in conjunction with the 19th ACM Conference on Recommender Systems, September 22–26, 2025, Prague, Czech Republic.

solal.nathan@inria.fr (S. Nathan)

0000-0002-5800-1271 (S. Nathan)

[1]Another most desired property concerns the fairness of the recommendation policy [5, 14], as the recommender system can amplify the biases and discriminations observed in the data. Fairness aspects are outside the scope of this paper.

[2]Indeed, part of the so-called sleeping job ads require skills that might be missing among the job seekers. However, the indication that some people are willing to apply to such job ads might still unlock the situation, e.g. by pointing at the need to offer specific training and the existing motivation to follow these.

share associated with each job ad (the number of applications received after recommending the top-k items corresponding to each job seeker) to design loss functions targeting the distribution of exposure among job ads. Such loss functions may be added to standard pointwise losses in JRSs to minimize the number of orphan jobs, or to reduce congestion, as an in-processing strategy. Secondly, a JRS trained using these compound training losses, called *Job Landscape Aware* recommender system (JoLA) is assessed on public job recommendation datasets, and compared with the state of the art, chiefly the ReCon approach [16, 4]. Most interestingly, in the considered settings the number of orphan jobs appears to be significantly more informative than the entropy of the market shares, classically used to train a congestion-avoiding JRS [16, 19].

The paper is organized as follows. Section 2 presents an overview of the proposed JoLA. The experimental setting used to comparatively assess JoLA, and the empirical results are respectively detailed in Sections 3 and 4. Section 5 discusses the position of the proposed approach with respect to related work. We last conclude and present the research perspectives opened by the proposed approach.

2. Overview of JoLA

This section presents the proposed JoLA approach and details its components.

Notations. The recommendation dataset consists of the interaction matrix $M(n, m)$ between n users (job seekers) and m items (job ads). Binary interactions are considered: $M[i, j] = 1$ iff the i-th user applies on the j-th job ad, and 0 otherwise. Following [4] the feature-based description of users and items is not considered, as the focus is on the recommendation policy in a warm-start recommendation setting [15].
It is worth noting however that the proposed approach extends to the feature-based recommendation setting in a straightforward manner.

2.1. Position of the warm-start recommendation problem

Recommender systems classically proceed by learning from the interaction matrix a real-valued score function $s(i, j)$. The recommendation policy π_s recommends to each i-th user a list of items, ordered by decreasing value of $s(i, j)$. Most generally, the set of recommendations to the i-th user is the set of top-k job ads after $s(i, \cdot)$ with k a positive integer ($1 \leq k \leq m$). Let us denote $j_{s,i,u}$ the u-th item recommended to the i-th user after policy π_s.

A scoring function is first assessed from its *Recall at k*, measuring the fraction of true interactions present in the top-k recommendations, averaged over all users:

$$Recall@k(s) = Average_{i\in[n]} \left\{ \frac{\sum_{u=1}^{k} M(i, j_{s,i,u})}{\sum_{u=1}^{m} M(i, j_{s,i,u})} \right\}$$

A more relaxed performance metrics is the *Hit-rate at k*, measuring the fraction of users for whom at least one interaction appears in the top-k recommendations:

$$HR@k(s) = Average_{i\in[n]} \left\{ 1_{\sum_{u=1}^{k} M(i,j_{s,i,u})} \right\}$$

where indicator function 1_A is 1 iff A is positive, and 0 otherwise.

Following the standard methodology for warm-start collaborative filtering [20], the scoring function is learned from the training part of the interaction matrix, and assessed from its performance on the rest of the interaction matrix.

2.2. A multi-stakeholder recommendation problem

As said, the domain of job recommendation involves rival goods, i.e. a given item can be attributed to at most one user. Two more performance metrics are thus considered to assess the global impact of the recommendation policy. The first one is the congestion score [16], defined as follows.

Let us first define the market share of an item, noted $MS(j)$, as the number of job seekers who are recommended this item according to π_s:

$$MS(j) = \#\{i \in [n] \; s.t. \; s(i, j) >= s(i, j_{s,i,k})\}$$

The normalized market share noted $MS_n(j)$ is $MS(j)$ divided by nk; the normalized market shares thus sum to 1. The congestion is then classically defined as the entropy of the normalized market shares:

$$Congestion(s) = -\sum_{j\in[m]} MS_n(j) \log MS_n(j)$$

Another performance metrics, noted $\mathcal{O}_{p,k}$ and generalizing the coverage [21], is also considered. $\mathcal{O}_{p,k}$ counts the number of job ads that are involved in less than p recommendations, with p a positive integer.[3] Formally, a job ad is said to be a (p, k)-orphan iff its number of applicants after π_s is less than p, when each user receives k recommendations. The proportion of (p, k)-orphans noted $\mathcal{O}_{p,k}$ is then defined as:

$$\mathcal{O}_{p,k}(s) = \frac{1}{m} \sum_{j\in[m]} 1_{MS(j)<p}$$

Note that for $p = 1$, the $\mathcal{O}_{p,k}$ metrics is but the complementary of the coverage metrics.

2.3. The JoLA algorithm

JoLA is built by training two neural embeddings respectively associated with users and items and referred to as ϕ and ψ. The recommendation score $s(i, j)$ associated to the pair made of the i-th user and j-th item is defined as:

$$s(i, j) = sigmoid(\langle \phi(i), \psi(j) \rangle + b)$$

The use of the sigmoid enforces $s(i, j)$ in $[0, 1]$ for a better training stability; constant b is trained and adjusts the range of recommendation scores amenable to optimization (with non vanishing gradients) [22].

Embeddings are learned by minimizing a compound loss. The first term of the learning loss classically is the binary

[3] Note that by design, the maximal value of p such that there exists no (p, k)-orphan is

$$p_{max} = \left\lfloor \frac{nk}{m} \right\rfloor \quad (1)$$

For $p > p_{max}$, the recommendation policy can but create (p, k)-orphans. Note that [18] minimizes the number of orphan jobs for $p = p_{max}$.

cross-entropy loss $L_{BCE}(s)$, meant to approximate the interaction matrix and defined as:

$$\mathscr{L}_{BCE}(s) = \sum_{i\in[n],j\in[m]} \begin{array}{l} M(i,j)\, log(s(i,j)) \\ + (1 - M(i,j))\, log(1 - s(i,j)) \end{array} \quad (2)$$

A first JoLA mode, referred to as JoLA-Congestion (JoLA-c), is trained by minimizing the loss defined as:

$$\mathscr{L}_{Congestion}(s) = \mathscr{L}_{BCE}(s) + \alpha Congestion(s) \quad (3)$$

with $\alpha > 0$ the weight of the congestion term; the congestion term relies on a differentiable approximation of the market shares, detailed below.

Likewise, a second JoLA mode, referred to as JoLA-Orphan (JoLA-o), is trained by minimizing:

$$\mathscr{L}_{Orphan}(s) = \mathscr{L}_{BCE}(s) + \alpha \mathscr{L}_{p,k}(s) \quad (4)$$

with $\alpha > 0$ the weight of the orphan loss defined as:

$$\mathscr{L}_{p,k}(s) = \frac{1}{m \cdot \mathscr{O}_{p,k}(s)} \sum_{j\ /\ MS(j)<p} (p - MS(j))^2 \quad (5)$$

with $\mathscr{O}_{p,k}(s)$ the proportion of (p,k)-orphans after policy π_s.

2.4. Differentiable Market Share

For simplicity and computational frugality, an approximate computation of the market share is used:[4]

$$MS(j) = \frac{1}{\mu} \sum_{i\in[n]} (s(i,j) - s(i,j_{s,i,k}))_+$$

where $A_+ = max(0, A)$ and μ set to the difference between $s(i,j)$ and $s(i,j_{s,i,k})$ when the difference is positive, averaged over all recommendations:

$$\mu = \frac{1}{nk} \sum_{i\in[n],j\in[m]} (s(i,j) - s(i,j_{s,i,k}))_+$$

Informally, $MS(j)$ considers all users i who will be recommended the j-th job ad after π_s (i.e., $s(i,j) >= s(i,j_{s,i,k})$). The sum of these scalar differences is approximated into a count by the division by μ.

2.5. The case of invisible job ads

A main weakness of the orphan loss (Eq. 5) is that its gradient does not involve any j-th job ad whose market share is 0, referred to as *invisible job ad*, as

$$\forall i \in [n], s(i,j) < s(i,j_{s,i,k})$$

For such invisible job ads i, the loss gradient can only flow through L_{BCE}, hampering the evolution of i toward any better market share.

Several options have been considered to take into account the case of invisible job ads. We eventually define a third JoLA mode, referred to as JoLA-Orphan-compound (JoLA-oc), where the recommendation score is trained by minimizing:

$$\mathscr{L}^c_{Orphan}(s) = \mathscr{L}_{BCE}(s) + \alpha \mathscr{L}_{p,k}(s) + \beta \mathscr{L}^c_{p,k}(s) \quad (6)$$

[4] An alternative left for further work is to use differentiable sorting and ranking [23, 24, 25]. We opted for the approximate estimation for the sake of computational frugality, after preliminary experiments showing a good accuracy of the approximation for $k \geq 10$.

where the third term with weight $\beta > 0$ aims to decrease the market share of non-orphan job ads, thereby increasing the market share of orphan job ads (including the invisible ones).

$$\mathscr{L}^c_{p,k}(s) = \frac{1}{m \cdot (1 - \mathscr{O}_{p,k}(s))} \sum_{j\in[m]/\ MS(j)>p} (MS(j) - p) \quad (7)$$

3. Experimental Setting

This section details the goals of experiments and the methodology followed to experimentally validate JoLA.

3.1. Goals of experiments

Our main goal is to comparatively assess the respective merits of the diverse JoLA modes in terms of the trade-off among the different performance indicators. The trade-off concerns the accuracy metrics (HR@10 and Recall@10) vs the user metrics (congestion) vs the recruiter metrics (number of (p,k)-orphans).

3.2. Baselines and Benchmarks

The first baseline is the policy based on the optimization of the only BCE loss (Eq. 2). The second baseline is the ReCon JRS [4], also aimed at alleviating the congestion through an in-processing method (more in Section 5); as said, ReCon is the approach most related to ours.

For the sake of comparison and reproducibility, we thus consider the same public datasets as those used to assess ReCon, i.e. the small and large CareerBuilder datasets described in Table 1, abbreviated as CB-S and CB-L. Both are extracted from the whole anonymized CareerBuilder dataset, released for the 2012 Kaggle competition.[5] CB-S and CB-L respectively retain interactions from the last 10 and 90 days. The training/validation/test split of the interaction matrix, the training interaction matrix noted M_{train}, the validation interaction matrix noted M_{val} and the batch sampling process are set after [4] for the sake of a fair comparison. M_{val} is used to determine the early stopping (see section 3.6) of the BCE phase.

3.3. Training/test split and batch sampling

Following ReCon's experimental setting, each batch $\mathscr{B}$ includes i) B positive interactions (i,j) uniformly sampled in the positive interactions in M_{train}; ii) negative interactions sampled as follows. For each positive interaction (i,j), two user indices $i', i"$ (respectively two item indices $j', j"$) are additionally uniformly sampled without replacement in $[n]$ (resp. $[m]$) such that $(i,j'), (i,j"), (i',j), (i",j)$ are negative interactions according to M_{train}.

The BCE loss is computed after Eq. 2, and divided by $5B$ for the sake of normalization.

Congestion and orphan losses are computed from the market shares estimated on the batch. Formally, all (unique) job ads involved in $\mathscr{B}$ are considered and the associated market share is computed by approximating how many (unique) users involved in $\mathscr{B}$ are recommended this job ad. The value of $\mathscr{O}(s)$ involved in the losses (Eqs. 5 - 6) corresponds to the number of (p,k)-orphans in $\mathscr{B}$.

[5] https://www.kaggle.com/competitions/job-recommendation/data.

Table 1
Benchmark datasets: CareerBuilder-Small (CB-S) and CareerBuilder-Large (CB-L)

Dataset	Job seekers	Jobs ads	Train Interactions	Validation Interactions	Test Interactions
CB-S	3876	4337	24316 (81%)	1071 (4%)	4557 (15%)
CB-L	42346	40542	450670 (96%)	9453 (2%)	9690 (2%)

3.4. Performance indicators

The reported HR@k and Recall@k metrics are computed on the test interaction matrix M_{test}. The fraction of orphan and invisible jobs are computed on the whole dataset ($M_{train} \cup M_{val} \cup M_{test}$), as their estimation on a small dataset (e.g. M_{test} only) is irrelevant (Fig. 1). For the sake of readability, all tables report the proportion of *non*-orphans (to be maximized) instead of the number of orphans denoted $o_{8,10}$. All performance metrics thus follow the "higher is better" principle.

3.5. Hyper-parameter configurations

Following ReCon again, the dimension d of the user embedding ϕ (resp. item embedding ψ) varies in [256, 512]. Each coordinate of ϕ and ψ is initialized to $\varepsilon - 4$ with ε randomly drawn after $\mathcal{N}(0, .01)$ in ReCon [16].

In early experiments, the initialization of ϕ and ψ in JoLA was based on the singular value decomposition (SVD) decomposition of M_{train}, significantly speeding up the optimization of the BCE loss term. In counterpart however, this initialization resulted in a large number of invisible job ads in the first optimization epochs, increasing from circa 0 with a random initialization to circa 20% with SVD initialization, all the more so with small batch sizes:

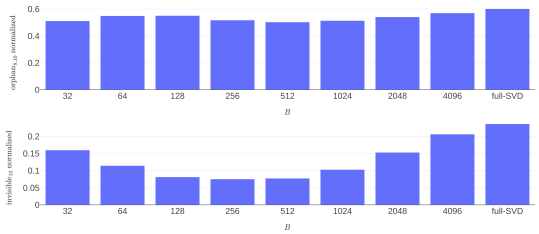

Figure 1: CareerBuilder-Small: Sensitivity w.r.t. the batch size B in [32, 4096], with $k = 10$, $p = 8$, when considering an SVD initialization of the user and item embeddings ϕ and ψ. Top: fraction of (p, k)-orphans according to policy π_{SVD}. Bottom: fraction of invisible job ads according to π_{SVD}. Rightmost column: fraction of (p, k)-orphans and invisible job ads in M_{train}.

The hyper-parameters of ReCon are taken from [16] (parameters of the Sinkhorn algorithm: $\gamma = 10.0$, $\varepsilon = 1 \times 10^{-2}$ and $max\_iters = 100$). The dimension d of embeddings ϕ and ψ ranges in [256, 512]; their initialization is random (see above).

The hyper-parameters of JoLA include: i) the type of loss (BCE only; JoLA-c; JoLA-o and JoLA-oc); ii) the loss weights α and β; iii) the size of the batch, controlled after the number B of positive interactions; iv) the initial learning rate adapted using AdamW [26]; v) the dimension d of the embeddings. In all experiments, $k = 10$ and the threshold p on the market shares is set to 8 (= p_{max}, Eq. 1). The selected hyper-parameter configuration is determined by optimizing loss BCE: the learning rate is 10^{-2}; the batch size is $B = 1024$ for CB-S and $B = 4096$ for CB-L; the embedding dimension $d = 512$. The value of hyper-parameter α (respectively β) varies in $\{10^{-\ell}, ..10^{+\ell}\}$ for $\ell = 2$.

3.6. Computational resources

The runtimes are measured on consumer grade GPU.

- CPU: 12th Gen Intel i3-12100 (8) @ 5.500GHz
- GPU: NVIDIA GeForce RTX 2070 SUPER [8 GiB VRAM]
- Memory: 32 GiB

All experiments consider 50 epochs, amounting to circa 10 minutes on CB-S and circa 4 hours and 20mn on CB-L.

The overall computational cost of JoLA is reduced by using a schedule. A first warm-start phase relies on the optimization of the BCE loss (Eq. 2) during the first epochs. The embeddings ϕ and ψ trained at the end of this first phase are further optimized by JoLA-c, JoLA-o and JoLA-oc during the last epochs, respectively considering Eq. 3, 4 and 6.

For CB-S, the first phase lasts for 25 epochs. For CB-L, the length of the first phase is set by early-stopping, based on the HR@10 score on M_{val} (30 epochs on average).

4. Experimental Validation

This section reports on the comparative experimental validation of JoLA on CB-S and CB-L. The JoLA code is publicly available at https://codeberg.org/solal/jola. In all tables and figures, $o_{8,10}$ corresponds to the number of non-(p, k)-orphans for $k = 10$, $p = 8$.

4.1. CareerBuilder-Small

Table 2 displays the comparative performances obtained on CB-S for $d = 512$ and batch size $B = 1,024$, selecting the non-dominated configurations. ReCon improves on BCE in terms of congestion (from .92 to .94) and (p, k)-orphans (from .35 to .38), with same HR@10, at the expense of a slight loss in Recall@10 (from .32 to .31).

JoLA-c achieves a quasi optimal congestion (.99) and an excellent performance in terms of non-(p, k)-orphans (.68) at the expense of a significant loss in HR@10 (from .55 to .45) and in Recall@10 (from .32 to .24). JoLA-o achieves a more balanced trade-off than JoLA-c: for a slightly lesser congestion (.98) and number of non-(p, k)-orphans (.64) it reaches a better HR@10 and Recall@10 (.49 and .28). Finally, JoLA-oc can be finely controlled using weight β to either favor the congestion and the number of non-(p, k)-orphans (.99 and .70) or the accuracy metrics (.49 and .27).

All in all, Table 2 presents a set of non-dominated configurations, enabling the designer to select their preferred trade-offs among the different performance metrics. Schematically, BCE is the best approach for Recall@10 and HR@10; JoLA-oc is the best one for congestion and non-(p, k)-orphans. ReCon is non-dominated w.r.t. congestion and HR@10.

It is noteworthy that the reduction of non-(p,k)-orphans goes with a better congestion: indeed, if all or most market shares are above p (all the more so as $p = p_{max}$, Eq. 1) then the market shares can only be balanced. In other words, the metrics suited to job seekers and to recruiters do not seem to be antagonistic.

Another remark is that the congestion value appears to be less finer-grained (varying from .92 to .99) than the number of non-(p,k)-orphans (varying from .35 to .70). More specifically, the number of orphans and the congestion value are informative for different modes of the market share distribution, after complementary experiments (not shown in the paper): for a high number of non-(p,k)-orphans ($\geq$ 80%), the congestion widely varies in [.4, .8]. Quite the contrary, for high values of congestion ($>$ .9), the number of non-(p,k)-orphans widely varies. In other words, the number of non-(p,k)-orphans appears to be more detailed in the interesting region, where the market shares are not too imbalanced.

Table 2
CareerBuilder-Small: Comparative performances of the baselines BCE and ReCon, versus JoLA-congestion, JoLA-orphan and JoLA-compound

	α	β	hr@10	r@10	$o_{8,10}$	cong@10
BCE			0.55	0.32	0.35	0.92
ReCon	10^{-5}		0.55	0.31	0.38	0.94
JoLA-c	10		0.48	0.27	0.72	0.99
JoLA-o	1		0.49	0.28	0.64	0.98
JoLA-oc	1	.1	0.49	0.27	0.67	0.98
JoLA-oc	1	1	0.45	0.25	0.70	0.99

4.2. CareerBuilder-Large

Table 3 displays the comparative performances obtained on CB-L for $d = 512$ and batch size $B = 4,096$ (except for ReCon which uses $B = 1,024$ due to higher VRAM usage limitations).

The difference among the CB-L and CB-S datasets is visible as the BCE performance in terms of HR@10 and Recall@10 is higher; in the meanwhile, no approach performs as well on CB-L in terms of congestion and number of non-(p,k)-orphans, as on CB-S. Both differences suggest that CB-L is less diversified than CB-S.

On CB-L, the best approach in terms of HR@10 and Recall@10 still is BCE (respectively .60 and .41), at the expense of a low congestion (.90) and number of non-(p,k)-orphans (.31). ReCon improves on BCE in both terms of congestion and non-(p,k)-orphans (respectively .97 and .49) with no loss in terms of HR@10 and a very slight loss in Recall@10 (from .41 to .40). JoLA-c significantly improves on ReCon in terms of congestion (.98) and non-(p,k)-orphans (.64) with a loss in HR@10 (from .60 to .55) and Recall@10 (from .40 to .38).

JoLA-o very slightly improves on JoLA-c in terms of HR and Recall, at the expense of a significant loss in terms of congestion (.98 to .95) and non-(p,k)-orphans (from .64 to .51).

Finally, JoLA-oc dominates JoLA-o in all respects; it is slightly dominated by JoLA-c in terms of congestion and non-(p,k)-orphans and slightly better in terms of HR@10 and Recall@10.

Table 3
CareerBuilder-Large: Comparative performances of the baselines BCE and ReCon, versus JoLA-congestion, JoLA-orphan and JoLA-compound

	α	β	hr@10	r@10	$o_{8,10}$	cong@10
BCE			0.60	0.41	0.31	0.90
ReCon	10^{-5}		0.60	0.40	0.49	0.97
JoLA-c	1		0.55	0.38	0.64	0.98
JoLA-o	.1		0.56	0.39	0.51	0.95
JoLA-oc	.1	.1	0.58	0.39	0.53	0.96
JoLA-oc	.1	1	0.57	0.39	0.57	0.97

4.3. Pareto fronts

Figures 2 and 3 display the Pareto front of the performance metrics on respectively CB-S and CB-L, specifically the trade-off between HR@10 and Congestion (top-left), HR@10 and number of non-(p,k)-orphans (top-right), Recall@10 and Congestion (bottom-left) and Recall@10 and number of non-(p,k)-orphans (bottom-right). As said, HR@10 and Recall@10 are computed on the test interaction matrix; the number of (p,k)-orphans and congestion are computed on the whole interaction matrix.

On CB-S, all four Pareto fronts present a similar structure: mostly ReCon appears on the right side (high accuracy metrics, Recall@10 or HR@10); JoLA-o and JoLA-oc appear in the center (trade-off between accuracy metrics and congestion/non-(p,k)-orphans); and JoLA-c dominates in the left (best congestion or non-(p,k)-orphans).

On CB-L, the four Pareto fronts are more diverse. A significant difference with CB-S is that JoLA-o is generally dominated. Except for this difference, we still have ReCon in the right, together with BCE (high accuracy metrics), JoLA-oc in the center, and JoLA-c in the left.

5. Position w.r.t. Related Work

The goals of preventing congestion and ensuring a fair distribution of the recommendations among items, users, or stakeholders, addressing both ethical and practical concerns have been extensively considered in the general literature devoted to recommender systems (RS) [27, 14, 28, 29, 13].

After [30], the main three methods to deal with popularity bias and exposure fairness in RS include i) pre-processing methods (e.g. based on data sampling or item exclusion [31]); ii) in-processing methods (based on regularization, constraints and/or weighting of the training loss) [32, 33, 34], and iii) post-processing methods, where the scoring function is perturbed using re-scaling, re-ranking and model aggregation [35, 36, 37].

In the domain of Job Recommender systems, the research has long been slowed down by the absence of public datasets, preventing the reproducible assessment of the algorithms. Indeed, the anonymization of HR-related datasets raises many privacy concerns, all the more severe as such datasets may involve vulnerable users. The RecSys challenge in 2017, based on the Xing benchmark [38], has been instrumental in the field [39, 40] (note that the Xing dataset does not involve precise indications as to the locations of the job ads and the job seekers).

As said, the most related approach to ours is ReCon [16, 4] (see [41] for a survey). ReCon is an in-processing method using entropy-based regularization similar to [42] (see also

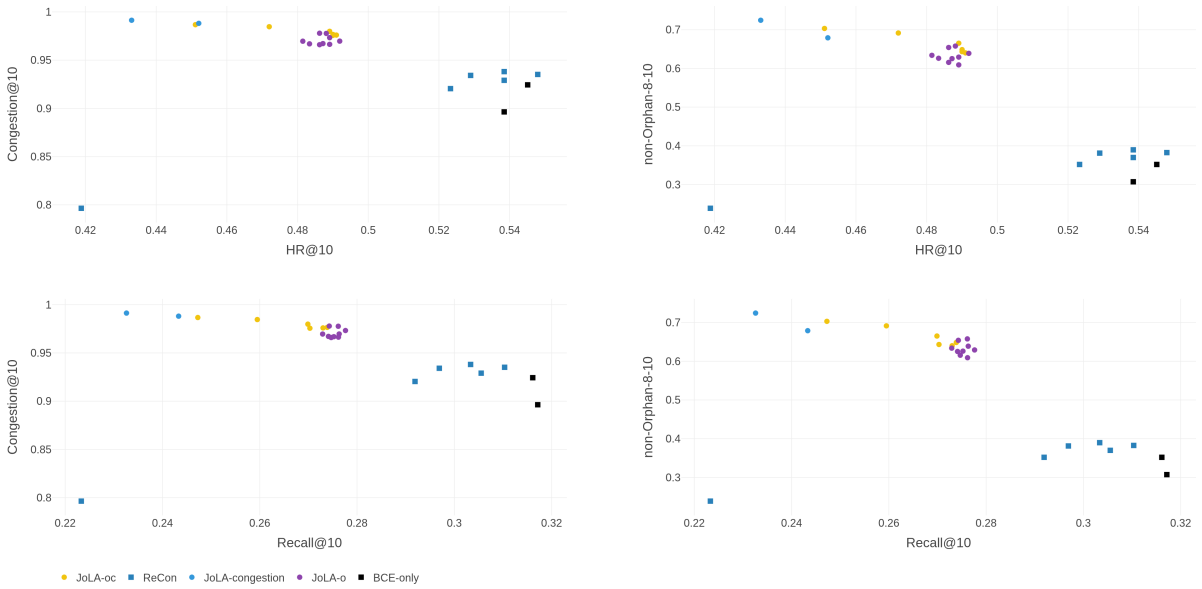

Figure 2: CareerBuilder-Small: Tradeoffs between the performance indicators for BCE, ReCon, JoLA-c, JoLA-o and JoLA-oc. Top-left: HR@10 vs Congestion. Top-right: HR@10 vs number of non-orphans. Bottom-left: Recall@10 vs Congestion. Bottom-right: Recall@10 vs number of non-orphans. (Better seen in color).

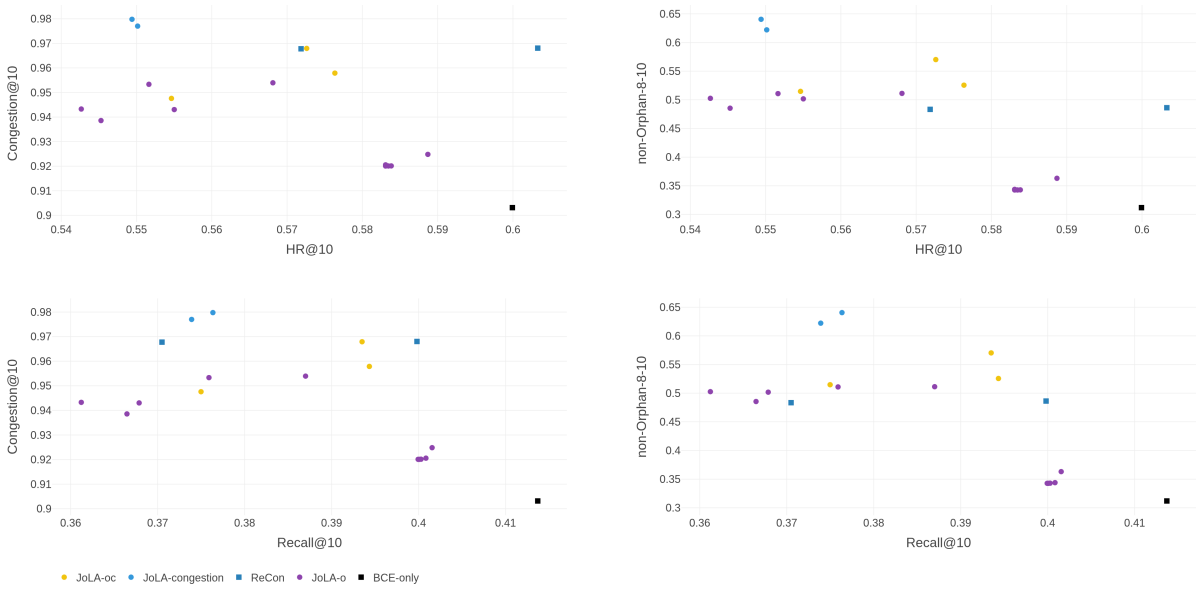

Figure 3: CareerBuilder-Large: Tradeoffs between the performance indicators for BCE, ReCon, JoLA-c, JoLA-o and JoLA-oc. Top-left: HR@10 vs Congestion. Top-right: HR@10 vs number of non-orphans. Bottom-left: Recall@10 vs Congestion. Bottom-right: Recall@10 vs number of non-orphans. (Better seen in color).

[43], where the regularization is based on the Gini index) or borrowing optimal transport ideas [44, 45] for [16]. More remotely related is [19] using Optimal Transport and [18] using envy-freeness and exposure, both in a post-processing manner.

The main originality of the proposed JoLA lies in the three proposed losses. All these losses aim to shape the distribution of the market shares, though in different ways. Specifically, the congestion metrics is optimal when market shares are equal, and it takes very similar values for sufficiently flat distributions. Quite the contrary, the orphan metrics mostly require that no market share be too small; the over-popularity of some job ads is only penalized in the orphan-compound loss (Eq. 6). The diverse approaches thus pave the Pareto front defined from the users and recruiters-related metrics, enabling the designers to select the proper loss depending on their context and priorities.

A potential weakness of JoLA is that it involves several hyper-parameters (type of loss, weight α and weight β in Eq. 6). A key limitation of the main orphan loss (Eq. 4) however is that the so-called invisible job ads are not taken into account in the back-propagation. The compound orphan loss aims to alleviate this limitation. Further work is concerned with reconsidering the learning schedule, and using

gradient clipping as an alternative to the use of weights α and β.

6. Conclusion

A main contribution of the presented JoLA approach is to explore several losses, aimed at controlling the distribution D_{MS} of the market shares associated to the considered job ads. The extensively studied congestion, that is the KL distance of D_{MS} and the uniform distribution, reaches its optimum when all market shares are equal. Note that this property is far from being satisfied in the whole interaction matrix, where the distribution of the market shares presents a majority of very unpopular job ads (with less than two applications) and circa 30% very popular job ads. An optimal recommendation policy in terms of congestion is thus focused on reducing the popularity of the most popular jobs.

In contrast, the orphan loss (Eq. 4) reaches its optimum when every market share is greater than some minimum p. This property is *very far* from being satisfied in the considered public datasets. Note that it is even less satisfied in PES datasets where a very significant fraction of the job ads are invisible (receiving no applications).

Complementary results show that: i) for a same HR value, the congestion and orphan metrics can significantly vary; ii) for a same congestion, with decent HR value, the orphan metrics can significantly vary. This observation is explained as the orphan metrics defines a finer-grained assessment of the market share distribution, than the c. Specifically, if the orphan metrics reaches a good value, then the congestion metrics reaches a good value too; but the converse does not hold true.

As said, the goal of decreasing the number of orphan or invisible job ads aims to two benefits: decreasing the competition in the short run; unlocking the sleeping job ad market, in the long run.

This approach opens several perspectives for further research. A short term perspective is to prevent or decrease the presence of invisible job seekers in the batch, through enforcing that each item participates in at least one positive interaction.. An on-going perspective is to confront an orphan-avoidance policy to the context of PES, and assess its robustness w.r.t. human and computational priorities, noting that the orphan phenomenon is significantly more severe in PES data then in the public CB-S and CB-L datasets, that are engineered to enforce the presence of sufficiently many interactions in the whole matrix.

Acknowledgements

The research leading to these results has received funding from the Special Research Fund (BOF) of Ghent University (BOF20/IBF/117), from the Flemish Government under the “Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen” programme, and from the FWO (project no. G073924N).

References

[1] P. Kircher, Job search in the 21st century, Journal of the European Economic Association (2022).

[2] J. Li, D. Arya, V. Ha-Thuc, S. Sinha, How to get them a dream job?: Entity-aware features for personalized job search ranking, Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2016).

[3] S. C. Geyik, Q. Guo, B. Hu, C. Ozcaglar, K. Thakkar, X. Wu, K. Kenthapadi, Talent search and recommendation systems at linkedin: Practical challenges and lessons learned, arXiv (2018). arXiv:1809.06481v1.

[4] Y. Mashayekhi, B. Kang, J. Lijffijt, T. De Bie, Scalable Job Recommendation With Lower Congestion Using Optimal Transport, IEEE Access 12 (2024) 55491–55505.

[5] T. Le Barbanchon, R. Rathelot, A. Roulet, Gender differences in job search: Trading off commute against wage, The Quarterly Journal of Economics 136 (2021) 381–426.

[6] J. Bennett, S. Lanning, The netflix prize, Proceedings of KDD Cup and Workshop 2007 (2007).

[7] Y. Koren, R. M. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (2009).

[8] European Parliament and Council of the European Union, Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act), Official Journal of the European Union, L 1689, 12 July 2024, 2024.

[9] C. O’Neil, Weapons of math destruction: How big data increases inequality and threatens democracy, Broadway Books, 2016.

[10] H. Abdollahpouri, M. Mansoury, Multi-sided Exposure Bias in Recommendation, arXiv.org (2020).

[11] I. Palomares, C. Porcel, L. Pizzato, I. Guy, E. Herrera-Viedma, Reciprocal recommender systems: Analysis of state-of-art literature, challenges and opportunities towards social recommendation, Information Fusion 69 (2021) 103–127. doi:https://doi.org/10.1016/j.inffus.2020.12.001.

[12] A. Banerjee, P. Banik, W. Wörndl, A review on individual and multistakeholder fairness in tourism recommender systems, Frontiers in Big Data 6 (2023). doi:10.3389/fdata.2023.1168692.

[13] G. Patro, L. Porcaro, L. Mitchell, Q. Zhang, M. Zehlike, N. Garg, Fair ranking: A critical review, challenges, and future directions, Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency (2022).

[14] Y. Deldjoo, D. Jannach, A. Bellogín, A. Difonzo, D. Zanzonelli, Fairness in recommender systems: Research landscape and future directions, User Modeling and User-Adapted Interaction (2022) 1–50.

[15] C. C. Aggarwal, Recommender Systems - The Textbook, Springer, 2016.

[16] Y. Mashayekhi, B. Kang, J. Lijffijt, T. De Bie, ReCon: Reducing Congestion in Job Recommendation using Optimal Transport, in: Proceedings of the 17th ACM Conference on Recommender Systems, ACM, 2023, pp. 696–701. doi:10.1145/3604915.3608817.

[17] Y. Wang, W. Ma, M. Zhang, Y. Liu, S. Ma, A survey on the fairness of recommender systems, ACM Transactions on Information Systems 41 (2023).

[18] G. K. Patro, A. Biswas, N. Ganguly, K. P. Gummadi, A. Chakraborty, FairRec: Two-Sided Fairness for Personalized Recommendations in Two-Sided Platforms, in: Proceedings of The Web Conference 2020, 2020, pp. 1194–1204.

[19] G. Bied, E. Perennes, V. Naya, P. Caillou, B. Crépon,

C. Gaillac, M. Sebag, Congestion-avoiding job recommendation with optimal transport, in: FEAST workshop@ECML-PKDD, 2023.
[20] G. Takács, I. Pilászy, B. Németh, D. Tikk, Scalable collaborative filtering approaches for large recommender systems, J. Mach. Learn. Res. 10 (2009) 623–656.
[21] M. Kaminskas, D. Bridge, Diversity, serendipity, novelty, and coverage: a survey and empirical analysis of beyond-accuracy objectives in recommender systems, ACM Transactions on Interactive Intelligent Systems (TiiS) 7 (2016) 1–42.
[22] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, T.-S. Chua, Neural collaborative filtering, Proceedings of the 26th International Conference on World Wide Web (2017).
[23] M. Cuturi, O. Teboul, J.-P. Vert, Differentiable Ranks and Sorting using Optimal Transport, 2019. URL: http://arxiv.org/abs/1905.11885. doi:10.48550/arXiv.1905.11885, arXiv:1905.11885 [cs, stat].
[24] F. Petersen, H. Kuehne, C. Borgelt, O. Deussen, Differentiable Top-k Classification Learning, 2022. URL: http://arxiv.org/abs/2206.07290. doi:10.48550/arXiv.2206.07290, arXiv:2206.07290 [cs].
[25] F. Petersen, C. Borgelt, H. Kuehne, O. Deussen, Monotonic Differentiable Sorting Networks, 2022. URL: http://arxiv.org/abs/2203.09630. doi:10.48550/arXiv.2203.09630, arXiv:2203.09630 [cs, stat].
[26] I. Loshchilov, F. Hutter, Decoupled weight decay regularization, in: International Conference on Learning Representations, 2017.
[27] Y. Wang, Weizhi, M. Zhang, Y. Liu, Shaoping, A Survey on the Fairness of Recommender Systems, ACM Transactions on Information Systems 41 (2022) 1–43.
[28] Y. Wu, J. Cao, G. Xu, Fairness in Recommender Systems: Evaluation Approaches and Assurance Strategies, ACM Transactions on Knowledge Discovery from Data 18 (2023) 1–37.
[29] J. Smith, L. Beattie, H. Cramer, Scoping Fairness Objectives and Identifying Fairness Metrics for Recommender Systems: The Practitioners' Perspective , Proceedings of the ACM Web Conference 2023 (2023).
[30] A. Klimashevskaia, D. Jannach, M. Elahi, C. Trattner, A survey on popularity bias in recommender systems, User Modeling and User-Adapted Interaction 34 (2024) 1777–1834. doi:10.1007/s11257-024-09406-0.
[31] P. Cremonesi, F. Garzotto, R. Pagano, M. Quadrana, Recommending without short head, in: Proceedings of the 23rd International Conference on World Wide Web, WWW '14 Companion, Association for Computing Machinery, 2014, p. 245–246. doi:https://doi.org/10.1145/2567948.2577286.
[32] S. Seymen, H. Abdollahpouri, E. C. Malthouse, A unified optimization toolbox for solving popularity bias, fairness, and diversity in recommender systems., in: MORS@ RecSys, 2021.
[33] H. Abdollahpouri, R. Burke, B. Mobasher, Controlling popularity bias in learning-to-rank recommendation, in: Proceedings of the eleventh ACM conference on recommender systems, 2017, pp. 42–46.
[34] X. Wang, W. H. Wang, Providing item-side individual fairness for deep recommender systems, in: Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency, 2022, pp. 117–127.
[35] Z. Zhu, Y. He, X. Zhao, Y. Zhang, J. Wang, J. Caverlee, Popularity-opportunity bias in collaborative filtering, in: Proceedings of the 14th ACM international conference on web search and data mining, 2021, pp. 85–93.
[36] Z. Zhu, Y. He, X. Zhao, J. Caverlee, Popularity bias in dynamic recommendation, in: Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining, 2021, pp. 2439–2449.
[37] H. Abdollahpouri, M. Mansoury, R. Burke, B. Mobasher, E. Malthouse, User-centered evaluation of popularity bias in recommender systems, in: Proceedings of the 29th ACM conference on user modeling, adaptation and personalization, 2021, pp. 119–129.
[38] J. Lian, F. Zhang, M. Hou, H. Wang, X. Xie, G. Sun, Practical lessons for job recommendations in the cold-start scenario, in: Proceedings of the Recommender Systems Challenge 2017, RecSys Challenge '17, Association for Computing Machinery, New York, NY, USA, 2017. doi:10.1145/3124791.3124794.
[39] M. Volkovs, H. Rai, Z. Cheng, G. Wu, Y. Lu, S. Sanner, Two-stage model for automatic playlist continuation at scale, in: Proceedings of the ACM Recommender Systems Challenge, RecSys Challenge, ACM, 2018, pp. 9:1–9:6.
[40] M. Volkovs, G. Yu, T. Poutanen, Dropoutnet: Addressing cold start in recommender systems, in: Advances in Neural Information Processing Systems, 2017, pp. 4957–4966.
[41] Y. Mashayekhi, N. Li, B. Kang, J. Lijffijt, T. De Bie, A challenge-based survey of e-recruitment recommendation systems, ACM Computing Surveys 56 (2023) 1–33.
[42] M. Elahi, D. K. Kholgh, M. S. Kiarostami, S. Saghari, S. P. Rad, M. Tkalčič, Investigating the impact of recommender systems on user-based and item -based popularity bias, Information Processing & Management 58 (2021) 102655. doi:10.1016/j.ipm.2021.102655.
[43] X. Ren, L. Lü, R.-R. Liu, J. Zhang, Avoiding congestion in recommender systems, New Journal of Physics 16 (2014). doi:10.1088/1367-2630/16/6/063057.
[44] M. Cuturi, Sinkhorn distances: Lightspeed computation of optimal transport, in: C. Burges, L. Bottou, M. Welling, Z. Ghahramani, K. Weinberger (Eds.), Advances in Neural Information Processing Systems, volume 26, Curran Associates, Inc., 2013.
[45] G. Peyré, M. Cuturi, Computational optimal transport, volume 11, Now Publishers, Inc., 2019.

A. Appendix

A.1. Details on the datasets

CareerBuilder-Small

```
Dataset Statistics:
Total unique users: 3876
Total unique items: 4337
Total interactions: 29944
Train interactions: 24316
Validation interactions: 1071
Test interactions: 4557
Combined Test+Val interactions: 5628
Unique users in train: 3876
Unique items in train: 4337
Unique users in validation: 247
Unique items in validation: 867
```

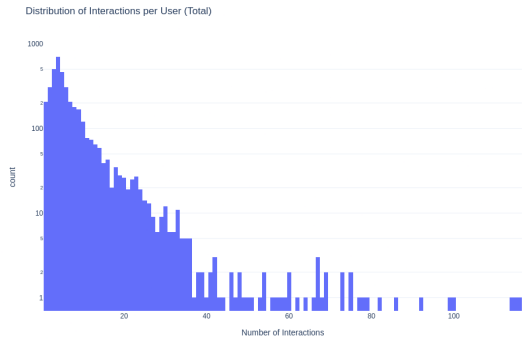

Figure 4: Histogram of number of interactions (degree) of users in CB-S

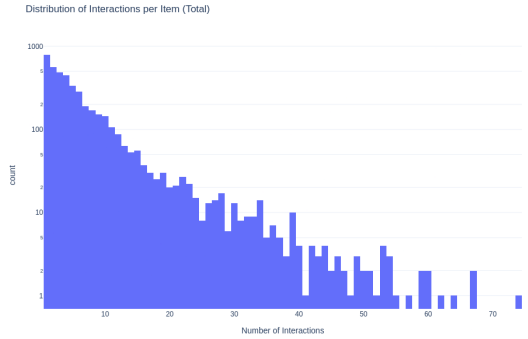

Figure 5: Histogram of number of interactions (degree) of items in CB-S

```
Average interactions per user (total): 7.73
Train interactions: 24316
Average interactions per user (train): 6.27
Validation interactions: 1071
Average interactions per user (val): 4.34
Test interactions: 4557
Average interactions per user (test): 4.33
Combined Test+Val interactions: 5628
Average interactions per user (test+val): 4.64
Unique users in train: 3876 (100.00%)
Unique items in train: 4337 (100.00%)
Unique users in validation: 247 (6.37%)
Unique items in validation: 867 (19.99%)
Unique users in test: 1053 (27.17%)
Unique items in test: 2480 (57.18%)
Overlapping users train+val: 247 (6.37%)
Overlapping items train+val: 867 (19.99%)
Overlapping users train+test: 1053 (27.17%)
Overlapping items train+test: 2480 (57.18%)
```

CareerBuilder-Large

```
Dataset Statistics:
Total unique users: 42346
Total unique items: 40542
Total interactions: 469813
Train interactions: 450670
Validation interactions: 9453
Test interactions: 9690
Combined Test+Val interactions: 19143
```

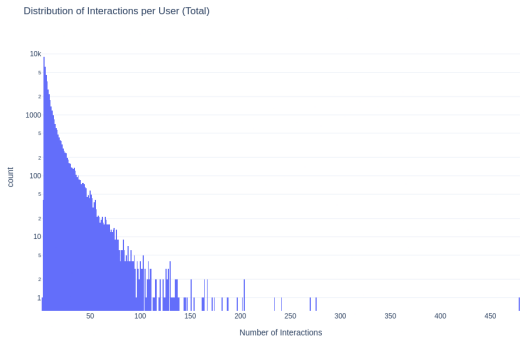

Figure 6: Histogram of number of interactions (degree) of users in CB-L

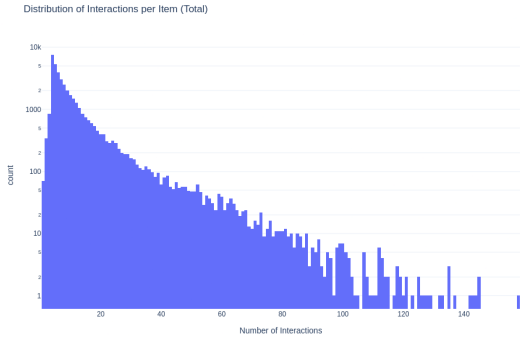

Figure 7: Histogram of number of interactions (degree) of items in CB-L

```
Unique users in train: 42346
Unique items in train: 40542
Unique users in validation: 2701
Unique items in validation: 5227
Average interactions per user (total): 11.09
Train interactions: 450670
Average interactions per user (train): 10.64
Validation interactions: 9453
Average interactions per user (validation): 3.50
Test interactions: 9690
Average interactions per user (test): 3.63
Combined Test+Val interactions: 19143
Average interactions per user (test+val): 4.02
Unique users in train: 42346 (100.00%)
Unique items in train: 40542 (100.00%)
Unique users in validation: 2701 (6.38%)
Unique items in validation: 5227 (12.89%)
Unique users in test: 2672 (6.31%)
Unique items in test: 4847 (11.96%)
Overlapping users train+validation: 2701 (6.38%)
Overlapping items train+validation: 5227 (12.89%)
Overlapping users train+test: 2672 (6.31%)
Overlapping items train+test: 4847 (11.96%)
Overlapping users train+combined test+val: 4758
```