Extended Abstract: Congestion-Avoiding Job Recommendation with Optimal Transport

Guillaume Bied[1,2], Elia Perennes[2], Victor Alfonso Naya[1], Philippe Caillou[1], Bruno Crépon[2], Christophe Gaillac[2,3], and Michele Sebag[1]

[1] Laboratoire Interdisciplinaire des Sciences du Numérique (LISN), Orsay, France
[2] Centre de Recherche en Economie et Statistique (CREST), Palaiseau, France
[3] Toulouse School of Economics (TSE), Toulouse, France

1 Introduction

This paper is motivated by the design of public policies in the job market domain, building upon the state of the art in recommender systems [23]. The specifics of the job market is that, while recommender systems aim at independently recommending to each user the most desirable item *for them*, it is inappropriate to recommend the same irresistible job ad to many job seekers: this would induce a congestion phenomenon at the population level and a poor eventual satisfaction at the individual level. More generally in domains such as job or matrimonial markets, referred to as *reciprocal recommendation settings* [17], an appropriate recommendation policy should globally take into account the populations of job seekers and job ads, and somehow connect both populations in a congestion-free way.

Taking inspiration from related works in recommender systems [15,16,5] and in econometrics [6,9], this paper investigates the coupling of optimal transport [8,20] with recommender systems. The presented approach, referred to as *Congestion avoiding recommendation with Optimal Transport* (CAROT), learns matchings between the users (job seekers) and the items (job ads) populations, aimed to maximize some trade-off between the interestingness of the recommended items, and their sufficient diversity at the population level (as opposed to recommendation serendipity [14], aimed at the recommendation diversity at the individual level). The scientific questions considered in the paper thus regard: i) how to define a new recommendation indicator, namely the congestion; ii) how to algorithmically bound this congestion; iii) how to assess the trade-off between the mainstream recommendation performance indicator, that is, recall, and congestion.

The contribution of the paper is threefold. Firstly, congestion avoidance is formalized within the optimal transport framework (Section 3). Secondly, the CAROT algorithm proposed to tackle this problem is agnostic regarding the data distribution (as opposed to the assumptions done in [15,16,9,6]), and is less computationally demanding than e.g. combinatorial optimization approaches [25]. Thirdly, the merits of CAROT are empirically demonstrated on a large scale proprietary dataset in the job market domain, kindly provided by the French National Employment Agency, *Pôle emploi*. The experimental results (Section 4) demonstrate the robustness of the approach and yield some unexpected lessons about the interactions of the recall and congestion indicators. The paper concludes with some research perspectives. Experiments on the public benchmark MAR, in the matrimonial market domain, are presented for the sake of comparative evaluation with [15] in supplementary material.

2 Formal background

This section defines the reciprocal recommendation problem, referring the reader to [17] for a comprehensive survey, and discusses some related work. The optimal transport setting is thereafter introduced for the sake of self-containedness [8,20].

Notations. Let n (respectively m) denote the number of users (resp. items), with x_i (resp. y_j) the description of the i-th user (resp. j-th item). The boolean collaborative matrix $M_{i,j}$ is such that $M_{i,j} = 1$ iff user i selected item j.

Position of the problem. A recommender system usually learns a scoring function ϕ such that the matrix defined from $\phi_{i,j} = \phi(x_i, y_j)$ maximizes the fit with the collaborative matrix M (expressed in terms of mean-square error or KL divergence), penalized with a regularization term [1]. With no loss of generality it is assumed in the remainder that the items recommended to the i-th user are ordered by increasing $\phi_{i,j}$.

In reciprocal recommendation [17], item j is subject to capacity constraint n_j: only the top n_j users selecting this item can be served. New optimization objectives and algorithms need be defined to accommodate such constraints.

Related works. An early approach facing reciprocal recommendation, [11] proceeds by learning scoring function ϕ as the solution of a constrained optimization problem. [25] casts reciprocal recommendation as a (NP-hard) multi-objective optimization problem, where the additional objective accounts for satisfying the capacity constraints; it is tackled using greedy optimization. In [2], the scoring function ϕ is used to estimate the global popularity of an item; the recommendation is repaired at the individual level, shifting upward or downward the items recommended to a given user depending on the item popularity. [5], inspired from decentralized economic models, considers the scoring functions reflecting the mutual utility of x_i w.r.t. y_j, and uses an optimal transport approach (see below) to construct a 1-to-1 recommendation, or *matching*.

Computational optimal transport. Optimal transport (OT) aims to map some (continuous or discrete) distribution μ onto another distribution ν. In the following, μ (respectively ν) stands for the uniform discrete distribution on the set of n users (resp. on the set of m items). Denoting $\mathit{\Gamma}(\mu, \nu)$ the set of measures such that their marginals with respect to 1st and 2nd arguments respectively are μ and ν, letting $C_{i,j}$ be the cost of mapping i onto j, the OT problem aims to find the joint distribution γ^* in $\mathit{\Gamma}(\mu, \nu)$ s.t.:

$$\gamma^*(C) = \arg \min_{\gamma \in \mathit{\Gamma}(\mu,\nu)} \sum_{i=1}^{n} \sum_{j=1}^{m} \gamma_{i,j} C_{i,j} \tag{1}$$

A tractable relaxation of the above optimization problem is proposed by Cuturi [8], by regularization with an entropic term:

$$\gamma^*(C, \varepsilon) = \arg \min_{\gamma \in \mathit{\Gamma}(\mu,\nu)} \sum_{i=1}^{n} \sum_{j=1}^{m} \gamma_{i,j} \left(C_{i,j} + \varepsilon \log(\gamma_{i,j})\right) \tag{2}$$

with ε the regularization weight. The optimal solution of Eq. (2) is of the form $\gamma_{i,j} = \alpha_i \exp\left(-C_{i,j}/\varepsilon\right)\beta_j$, where α and β enforce the constraints on the marginals of γ.

Discussion The main approaches in OT-based recommendation assume the observed collaborative matrix M to be the optimal solution of an OT plan based on some matching cost C [6,10,15,16]: they learn C from training data, and use the estimated cost model to build matchings on other data.

In the context of the job market, it is however debatable whether the actual matching, i.e. the observed collaborative matrix M, should be viewed as the solution of an optimal transport plan: by construction, M is the result of a decentralized process whereas the OT solution results from a centralized one. Accordingly, the proposed approach will be structured along two phases: learning the matching cost function C from M (without assuming M to be an OT solution), and using C within an OT process.

3 Overview of CAROT

Let $\phi_{i,j}$ in $\mathcal{R}$ denote the sought recommendation score of the j-item for the i-th user, and define boolean indicator $\mathbb{1}_{i\to j,k,\phi}$ as 1 iff j ranks among the k top recommendations for i. Subscript ϕ is omitted if clear from context.

Performance criteria Beside the standard Recall@k indicator, measuring the fraction of users for which the actually preferred item is ranked among the top-k recommendations,

$$\text{Recall@}k(\phi) = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{m} M_{i,j}.\mathbb{1}_{i\to j,k} \tag{3}$$

we define the notion of *item market share* $MS_\ell(j)$ of item j as follows, to measure the fraction of users i such that j is among the top ℓ items recommended to i (with $\ell < m$):

$$MS_\ell(j) = \frac{1}{n\times\ell}\sum_{i=1}^{n}\mathbb{1}_{i\to j,\ell}$$

Informally, the congestion is minimized if the entropy of the market shares is minimized (all the more so as ℓ goes to 1):

$$\text{Congestion@}\ell(\phi) := \sum_{j=1}^{m} MS_\ell(j)\log\left(MS_\ell(j)\right) \tag{4}$$

For the sake of fair comparisons, the congestion indicator is mapped to $[-1,0]$ by division by $\log(m)$; the perfect congestion avoidance is obtained for equal market shares of the items, with -1 as optimal value.

Enforcing congestion avoidance with OT. As said, optimal transport is applied based on a cost matrix $C_{i,j}$ depending on the learned recommendation score $\phi_{i,j}$. Leaving end-to-end learning of $C_{i,j}$ for further work, we consider $C_{i,j} = g(\phi_{i,j})$, with g a monotonous scalar function, hyper-parameter of the approach, such that the cost $C_{i,j}$ of transporting i toward j increases with $\phi_{i,j}$ (that is, as the relevance of matching i and j decreases). Additionally, $\phi_{i,j}$ is capped to the score of the 1,000-th item recommended to each i, noted $\phi_i^{(1000)}$ ($\phi_{i,j} \leftarrow \max(\phi_{i,j}, \phi_i^{(1000)})$ in the following).

Four g functions have been considered, respectively linear, or exponential functions of $\phi_{i,j}$, or rank-based, or NDCG-like [3].

For a fair comparison of the results obtained with same entropic regularization weight ε, $C_{i,j}$s are normalized s.t. $\sum_{i,j} C_{i,j} = 1$.

The CAROT *algorithm* Overall, CAROT is a 2-step process: i) learning a scoring function ϕ; ii) solving the optimal transport problem defined from $C_{ij} = g(\phi_{i,j})$.

CAROT*: 1. Learning ϕ.* The two considered recommendation learning approaches are XGBOOST and neural networks (NN). XGBOOST is a state of art recommender system based on gradient boosting [23], that can be efficiently trained by aggressively subsampling the negative pairs (i, j), at the expense of a lesser scalability in recommendation. NN is a neural net, whose architecture is tailored to the specifics of the domain (e.g., considering submodules devoted to geographic or skill-related informations). The descriptions of user x_i and item y_j are mapped onto latent spaces, respectively noted $z_{x,i}$ and $z_{y,j}$, and their adequacy $\phi_{i,j}$ is sought as $z_{x,i}^T A z_{y,j}$ with A a matrix [4]. The mappings and matrix A are learned in an end-to-end fashion using a triplet loss [24]. Formally, assuming that each user selects a single item, considering wlog (up to a permutation of the items) that user i selects item i, the learning goal is to

$$\text{Minimize} \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{m} (\phi_{i,i} - \phi_{i,j} + \eta)_+ \tag{5}$$

with $A_+ = \max(0, A)$ and $\eta > 0$ hyper-parameter of the approach. As for XGBOOST case, negative sampling is used to cope with the number of negative pairs. The hyper-parameters of XGBOOST and NN are detailed in Appendix C.

CAROT*: 2. Optimal transport.* Depending on the regularization weight ε and the g function (with $C_{ij} = g(\phi_{i,j})$), discrete distribution γ is trained by optimizing Eq. (2). Note that the extension of the approach to the general reciprocal recommendation case (*e.g.* where several positions are opened for the j-th job ad) is straightforward by making ν_j proportional to the capacity constraint of item j.

Eventually, the CAROT recommendation proceeds deterministically, ordering the j items recommended to user i in decreasing order w.r.t. $\gamma_{i,j}$.

4 Experimental validation

After detailing the goals of experiments and experimental setting, this section presents the empirical validation of the approach (detailed in Appendix D).

Experimental setting The first goal of experiments is to assess the efficiency of the proposed approach in terms of trade-off between recall and congestion. The second goal is to investigate how the results depend on the hyper-parameters of the approach (ϕ being learned using XGBOOST or NN; entropic regularization weight ε ranging in $10^{-2}, \ldots, 10^{2}$; transport cost C_{ij} defined as $g(\phi_{ij})$ with g ranging in {Id, Exp, Ndcg, Rank}).

With each hyper-parameter setting is associated seven performance indicators: recall@k with $k = 1,\ 10,\ 100$, congestion@k with $k = 1,\ 10$, and coverage@k with $k = 1,\ 10$, indicating the fraction of items involved in top-k recommendation of at least one user. Additional performance indicators are reported and discussed in Appendix D.

The performance is assessed on a proprietary benchmark dataset JOB from *Pôle emploi*. The training set includes circa 1,650,000 job seekers, 477,000 job ads and 43,000 matches (signed contracts) reported in *Ile de France* during the Feb.-Oct 2018 period. The description x_i (respectively y_j) of a job seeker (resp. job ad) is in $\mathcal{R}^{448}$ (resp $\mathcal{R}^{582}$). Function ϕ is learned on the training set; the optimal transport plan γ is computed on the test set, restricted to the job sector of logistics for scalability reasons, including 110,000 job seekers, 14,200 job ads and 450 matches in Nov. 2018. Complementary results on a public benchmark in the domain of the matrimonial market are discussed in Appendix A for the sake of comparative evaluation.

Table 1: Comparative Results on JOB: Recall, Coverage and Congestion.

	Algorithm	Recall (%)			Coverage (%)		Congestion	
		@1	@10	@100	@1	@10	@1	@10
	ϕRandom	0	0.21	0.65	99.95	100	-0.99	-0.99
CAROT-XGB	ϕ XGB	9.62	31.40	61.59	12.94	25.16	-0.62	-0.64
CAROT-XGB	γ^{XGB},$g = Id$+,$\varepsilon = 1.0$	4.81	21.99	57.87	21.61	31.76	-0.74	-0.75
CAROT-XGB	γ^{XGB},$g = Id$+,$\varepsilon = 0.1$	2.18	15.31	56.01	27.54	41.24	-0.78	-0.81
CAROT-XGB	γ^{XGB},$g = Id$+,$\varepsilon = 0.01$	4.37	20.45	43.21	46.75	57.61	-0.85	-0.79
CAROT-XGB	γ^{XGB},$g = ndcg$,$\varepsilon = 1.0$	9.62	31.61	62.36	12.96	26.14	-0.62	-0.67
CAROT-XGB	γ^{XGB},$g = ndcg$,$\varepsilon = 0.1$	8.97	25.38	46.06	14.69	30.84	-0.67	-0.74
CAROT-XGB	γ^{XGB},$g = ndcg$,$\varepsilon = 0.01$	5.03	14.00	18.81	36.81	57.52	-0.82	-0.81
CAROT-NN	ϕ NN	5.68	28.66	57.98	6.02	17.78	-0.46	-0.49
CAROT-NN	γ^{NN},$g = Id$+,$\varepsilon = 1.0$	6.78	26.14	60.39	11.99	26.30	-0.62	-0.65
CAROT-NN	γ^{NN},$g = Id$+,$\varepsilon = 0.1$	2.40	19.03	50.43	28.23	40.16	-0.80	-0.79
CAROT-NN	γ^{NN},$g = Id$+,$\varepsilon = 0.01$	3.93	16.30	27.89	53.38	62.35	-0.83	-0.70
CAROT-NN	γ^{NN},$g = ndcg$,$\varepsilon = 1.0$	5.68	27.46	59.08	6.02	19.75	-0.46	-0.55
CAROT-NN	γ^{NN},$g = ndcg$,$\varepsilon = 0.1$	5.25	23.3	49.01	8.85	26.40	-0.53	-0.65
CAROT-NN	γ^{NN},$g = ndcg$,$\varepsilon = 0.01$	1.53	12.36	24.28	35.41	51.56	-0.81	-0.81

Results Table 1 summarizes the results, with mainly three lessons. Firstly, NN is dominated by XGBOOST in terms of all three performance indicators: recall, coverage and congestion; furthermore, the lesser recall of NN (4% loss in recall@100) comes with a much lower coverage (7% loss in coverage@1). This counter-performance is blamed on the architecture of the neural net (ongoing experiments suggest that the refinement of specific modules accounting for geographic and skill-related aspects allows NN to catch up). Secondly, coverage (@1 and @10) monotonically increases, and recall (@1, @10, @100) monotonically decreases as ε decreases from 1 to .01, leaving little hope that one can combine a good coverage with a decent recall. More encouraging is the fact that the congestion@1 can be significantly improved (from -.62 to .78) at the expense of a moderate recall loss (recall@10 goes from 62% to 56%) for $g = Id, \varepsilon = .1$.

The option $g = ndcg$ has little (slightly positive) effects for $\varepsilon = 1$ and strongly detrimental effects for $\varepsilon = .1$ or .01.

Surprisingly, decreasing ε yields a better (lower) congestion at the expense of a worse recall. This effect was unexpected as the higher ε, the more uniform the transport plan γ (everything else being equal) hence the lower the congestion should be. The tentative interpretation for this fact (backed by complementary experiments reported in Appendix B) is that all performance indicators depend on the order induced by γ, as opposed to, the actual $\gamma_{i,j}$ values. While the variance of $\gamma_{i,j}$ does decrease as ε increases, the "winner take all" phenomenon persists, i.e. the top-1 recommendations over all users cover a meager 13% to 20% of the items.

5 Conclusion and Perspectives

Along the "AI for good" trend [13], this paper aims to prevent the undesirable effects of recommender systems in the domain of job market. Specifically, if some job ads are independently recommended to many job seekers, then a congestion phenomenon is observed at the global level, entailing a waste of time and other detrimental effects for both populations of job seekers and recruiters.

The proposed approach takes inspiration from optimal transport, with the idea of globally "transporting" the job seekers population onto the job ads population, enforcing a decent recall with low congestion. The key question thus becomes the definition of the transport cost. In this paper, the transport cost is based on a mainstream recommender score. The interesting and surprising lessons learned from the application of the approach on a real-world large scale dataset is that the transport cost and the transport regularization (used to enforce the OT scalability [8]) interact in subtle ways. Chiefly, a strong regularization (expectedly yielding a uniform transport plan) significantly degrades the recall *while it does not improve the congestion* to the desired extent.

This work opens two main perspectives. On the algorithmic side, future work will investigate the end-to-end learning of the recommendation plan, taking into account both the recall and the coverage. An intermediate goal is to learn the function g used to derive the transport costs from the scoring function.

More fundamentally, building a "fair job recommender system" should be viewed as learning a prescriptive model (*follow this policy to achieve the intended goals*), as opposed to a predictive model (*follow this policy as it accurately estimates the users preferences*) [12,19,22]. The merits and limits of such a prescriptive model will require experiments along the randomized controlled trials (RCT) methodology [18].

Acknowledgments

This research was supported by DATAIA convergence institute as part of the “Programme d’Investissement d’Avenir" (ANR-17-CONV-0003) operated by Institut Polytechnique de Paris and Université Paris-Saclay.

References

1. Aggarwal, C.C.: Recommender systems, vol. 1. Springer (2016)
2. Borisyuk, F., Zhang, L., Kenthapadi, K.: Lijar: A system for job application redistribution towards efficient career marketplace. In: Proceedings of the 23rd ACM SIGKDD. pp. 1397–1406. ACM (2017). https://doi.org/10.1145/3097983.3098028
3. Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., Hullender, G.: Learning to rank using gradient descent. In: Proceedings of the 22nd international conference on Machine learning. pp. 89–96 (2005)
4. Chechik, G., Shalit, U., Sharma, V., Bengio, S.: An online algorithm for large scale image similarity learning. Advances in Neural Information Processing Systems **22**, 306–314 (2009)
5. Chen, K.M., Hsieh, Y.W., Lin, M.J.: Prediction and congestion in two-sided markets: Economist versus machine matchmakers. SSRN Electronic Journal (2019). https://doi.org/10.2139/ssrn.3318778
6. Chiappori, P.A., Salanié, B.: The econometrics of matching models. Journal of Economic Literature **54**(3), 832–61 (2016)
7. Cremonesi, P., Koren, Y., Turrin, R.: Performance of recommender algorithms on top-n recommendation tasks. In: Proceedings of the Fourth ACM Conference on Recommender Systems. p. 39–46. RecSys ’10, Association for Computing Machinery, New York, NY, USA (2010). https://doi.org/10.1145/1864708.1864721, https://doi.org/10.1145/1864708.1864721
8. Cuturi, M.: Sinkhorn distances: Lightspeed computation of optimal transport. In: Advances in Neural Information Processing Systems. p. 2292–2300 (2013)
9. Galichon, A.: Optimal transport methods in economics. Princeton University Press (2018)
10. Galichon, A., Salanié, B.: Cupid’s invisible hand: Social surplus and identification in matching models. Available at SSRN 1804623 (2020)
11. Gualdi, S., Medo, M., Zhang, Y.: Crowd avoidance and diversity in socio-economic systems and recommendation. CoRR **abs/1301.1887** (2013), http://arxiv.org/abs/1301.1887
12. Imbens, G.W., Rubin, D.B.: Causal inference in statistics, social, and biomedical sciences. Cambridge University Press (2015)
13. Kleinberg, J., Ludwig, J., Mullainathan, S., Rambachan, A.: Algorithmic fairness. In: Aea papers and proceedings. vol. 108, pp. 22–27 (2018)
14. Kunaver, M., Požrl, T.: Diversity in recommender systems–a survey. Knowledge-Based Systems **123**, 154–162 (2017)
15. Li, R., Ye, X., Zhou, H., Zha, H.: Learning to match via inverse optimal transport. J. Mach. Learn. Res. **20**(80), 1–37 (2019)
16. Liu, R., Balsubramani, A., Zou, J.: Learning transport cost from subset correspondence. arXiv preprint arXiv:1909.13203 (2019)
17. Palomares, I., Porcel, C., Pizzato, L., Guy, I., Herrera-Viedma, E.: Reciprocal recommender systems: Analysis of state-of-art literature, challenges and opportunities towards social recommendation. arXiv preprint arXiv:2007.16120 (2020)
18. Pearce, W., Raman, S.: The new randomised controlled trials (RCT) movement in public policy: challenges of epistemic governance. Policy Sci **47**, 387–402 (2014)

19. Peters, J., Janzing, D., Schölkopf, B.: Elements of causal inference: foundations and learning algorithms. MIT press (2017)
20. Peyré, G., Cuturi, M.: Computational optimal transport. Foundations and Trends® in Machine Learning **11**(5-6), 355–607 (2019)
21. Schmitzer, B.: Stabilized sparse scaling algorithms for entropy regularized transport problems. arXiv preprint arXiv:1610.06519 (2019)
22. Vasile, F., Bonner, S.: Causal embeddings for recommendation: An extended abstract. In: IJCAI-19. pp. 6236–6240. International Joint Conferences on Artificial Intelligence Organization (7 2019). https://doi.org/10.24963/ijcai.2019/870, https://doi.org/10.24963/ijcai.2019/870
23. Volkovs, M., Yu, G.W., Poutanen, T.: Content-based neighbor models for cold start in recommender systems. In: Proceedings of the Recommender Systems Challenge 2017 - RecSys Challenge 17. ACM Press (2017). https://doi.org/10.1145/3124791.3124792
24. Weinberger, K.Q., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. J. Mach. Learn. Res. **10**(2) (2009)
25. Xia, B., Yin, J., Xu, J., Li, Y.: We-rec: A fairness-aware reciprocal recommendation based on walrasian equilibrium. Knowledge-Based Systems **182** (08 2019). https://doi.org/10.1016/j.knosys.2019.07.028

Supplementary material

Appendix A: results on a public matrimonial dataset

Dataset description A public benchmark in the domain of the matrimonial market (noted MAR), first introduced by [15], is used for the sake of comparison with the state of art. The data include 2,475 men (respectively women), partitioned in 50 clusters. Each individual is described with 11 mostly ordinal features. The 1-to-1 matching is described at the individual level and the data also include the $M_{c,c'}$ collaborative matrix, reporting the fraction of matches between men from cluster c and women from cluster c'.

Benchmarks The baseline results on MAR are those of RIOT [15], an SVD-based decomposition, and itemKNN [7]. At the cluster level, the performance indicators include the RMSE and the MAE between the collaborative matrix M at the cluster level and the estimated recommendation matrix, measured using a 5-cross fold validation. CAROT is also assessed at the individual level, using the performance indicators in section 4.

Results on MAR On the matrimonial benchmark, tables 2 and 3 respectively display the comparative results obtained at the cluster[4] and the individual level.

At the cluster level, γ^{XGB} slightly but statistically significantly improves on RIOT w.r.t. both RMSE (8.89$\pm$ 0.11 as compared to 8.98$\pm$ 0.17) and MAE (5.80$\pm$ 0.13 in contrast to 5.79$\pm$ 0.12). γ^{NN} also slightly improves on RIOT. Other benchmarks (random, PMF, SVD and itemKNN) are outperformed.

At the individual level, XGBOOST significantly outperforms NN in both terms of recall and congestion for all values of k.

γ^{XGB} is found to only improve the congestion at the expense of the recall: improving the congestion (from -.84 to -.98) is obtained by decreasing the recall@10 (from 28.4% to 23.7% at best, for $g = Id, \varepsilon = 10^{-2}$).

For γ^{NN}, the congestion can be significantly improved (from -.84 to -.98, for $g = Id, \varepsilon = 10^{-2}$) while preserving the recall@10 (circa 15.4%).

It is suggested however that the recall and the congestion indicators are actually *not* antagonistic in the MAR problem: by construction, the sought collaborative matrix is a permutation. The main difficulty for this recommendation problem thus seemingly comes from the small size of the dataset and the poor description of the individuals.

[4] The difference with [15] is explained as a bug was found (and corrected) in the publicly available code for RIOT and other baselines, dividing the error by the number of folds in each iteration. The performance order is not modified by (correcting) the bug.

Table 2: Comparative results on MAR at the cluster level; average and standard deviation of the RMSE and MAE w.r.t. the cluster matrix M, over 5-fold CV. Results for CAROT correspond to $g = Id+$,$\varepsilon = 1$.

	Random	PMF	SVD	itemKNN	RIOT	γ^{NN}	γ^{XGB}
RMSE	10.71± 0.13	446.6± 9.86	441.4± 11.2	9.36± 0.12	9.12± 0.12	8.98± 0.17	8.89± 0.11
MAE	7.22± 0.06	251.3± 6.00	249.2± 5.71	6.30± 0.03	5.98± 0.10	5.80± 0.13	5.79± 0.12

Table 3: Comparative Results on MAR at the individual level: Recall, Coverage and Congestion.

	Algorithm	Recall (%)		Coverage (%)		Congestion	
		@1	@10	@1	@10	@1	@10
	ϕ Random	0.16	2.27	63.32	100	-0.90	-0.98
CAROT-XGB	ϕ XGBoost	7.93	27.88	48.55	98.69	-0.84	-0.94
	γ^{XGB},$g = Id+$,$\varepsilon = 1.0$	8.05	28.41	49.77	99.18	-0.85	-0.95
	γ^{XGB},$g = Id+$,$\varepsilon = 0.1$	8.01	27.02	72.73	100	-0.93	-0.95
	γ^{XGB},$g = Id+$,$\varepsilon = 0.01$	6.47	23.77	96.05	100	-0.98	-0.84
	γ^{XGB},$g = ndcg$,$\varepsilon = 1.0$	7.93	28.2	48.55	99.02	-0.84	-0.95
	γ^{XGB},$g = ndcg$,$\varepsilon = 0.1$	8.10	25.72	59.42	100	-0.89	-0.93
	γ^{XGB},$g = ndcg$,$\varepsilon = 0.01$	6.06	19.49	94.26	100	-0.98	-0.73
CAROT-NN	ϕ NN	3.82	15.50	46.27	98.00	-0.83	-0.93
	γ^{NN},$g = Id+$,$\varepsilon = 1.0$	2.84	14.32	38.86	92.47	-0.80	-0.90
	γ^{NN},$g = Id+$,$\varepsilon = 0.1$	3.94	15.46	70.12	100	-0.92	-0.98
	γ^{NN},$g = Id+$,$\varepsilon = 0.01$	3.78	15.46	93.48	100	-0.98	-0.95
	γ^{NN},$g = ndcg$,$\varepsilon = 1.0$	3.82	15.63	46.27	98.73	-0.83	-0.94
	γ^{NN},$g = ndcg$,$\varepsilon = 0.1$	4.23	13.87	57.99	99.91	-0.88	-0.93
	γ^{NN},$g = ndcg$,$\varepsilon = 0.01$	2.89	11.60	93.44	100	-0.98	-0.72

Appendix B: Higher entropic regularization may not reduce congestion

After [20] (prop. 4.1), when the weight ε of the entropic regularization term goes to ∞, the solution γ of the regularized optimal transport problem tends to a uniform coupling. When $\epsilon \to 0$ instead the solution converges to the optimal transport plan with maximal entropy. Informally, increasing ϵ leads to solutions γ that are less sparse.

Unexpectedly however, the exploitation of γ through the sorting recommendation process is such that a more uniform γ does not necessarily lead to less congestion.

This phenomenon is investigated in simulation. 1,000 cost matrices C of size $n = 30, m = 10$ are independently generated, with $C_{ij} \sim \mathcal{U}(\frac{j}{m}, \frac{j}{m}+1)$ (items being ordered by increasing attractivity). Transport plans γ with uniform marginals w.r.t. users and items are then computed using Sinkhorn algorithm with entropic regularization weight $\varepsilon = 100$ and $\varepsilon = 0.01$. The average and standard deviation over the 1,000 runs of the congestion obtained after sorting these plans indicate that the congestion is significantly higher for the higher value of ε:

ϵ	Mean congestion@1	Std.
100	-0.940521	0.029445
0.01	-0.996059	0.003586

Figures 1, 2, 3 and 4 illustrate this phenomenon on a single representative run. γ_{ij} are more uniform when $\epsilon = 100$ than when $\epsilon = 0.01$, sorting each line leads to a more unequal distribution of recommendations towards the different offers.

Altogether, higher entropic regularization has an indeterminate impact on congestion, and may increase it in practice. The choice of the ϵ should thus be chosen based on a validation set, as well as on numerical criteria for the convergence of Sinkhorn's algorithm. One may note that taking extremely small values of ϵ using a naive implementation of Sinkhorn's algorithm may have adverse consequences on numerical stability as well as convergence speed, although alternatives have been developed, for instance in [21].

Appendix C: Hyperparameters

This appendix details the hyperparameters used to train XGBOOST and NN on both benchmarks.

XGBOOST

On MAR, XGBOOST is used with its default parameters, except for the number of boosting rounds, set to 200. A logistic loss is used and the negative sampling ratio is set to 50 (Table 4).

On JOB, XGBOOST is used with the hyper-parameters reported in Table 5. Other hyper-parameters are set to their default value.

Fig. 1: Raw costs

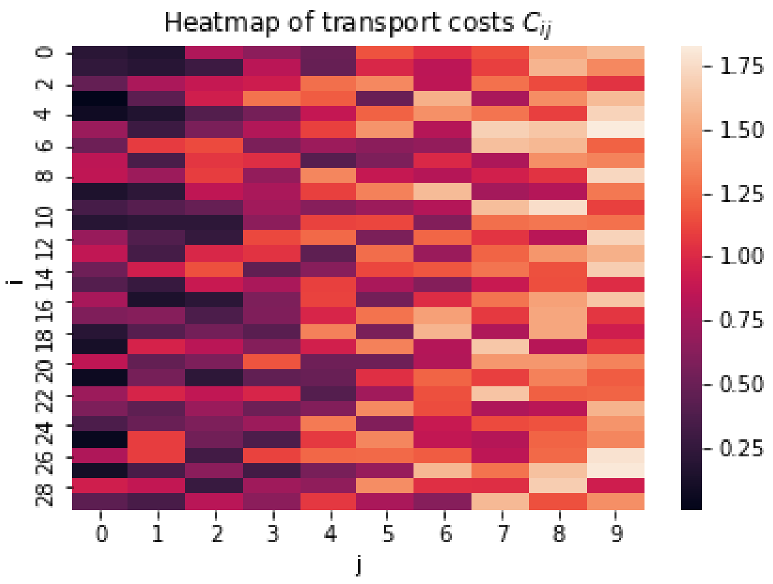


Fig. 2: $\epsilon = 100$

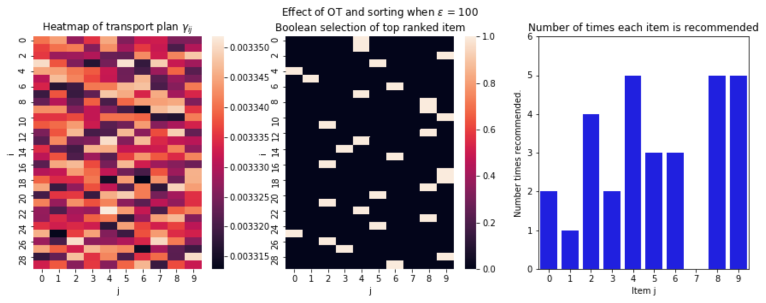


Fig. 3: $\epsilon = 0.1$

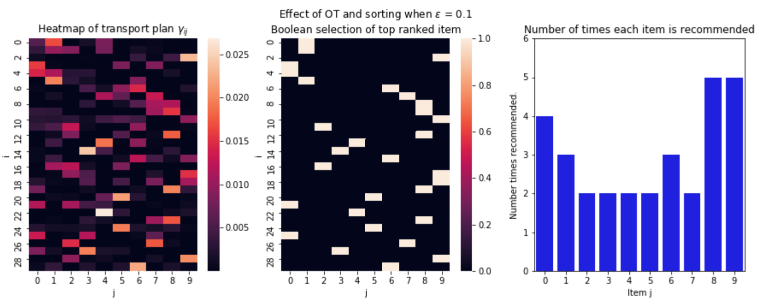


Fig. 4: $\epsilon = 0.01$

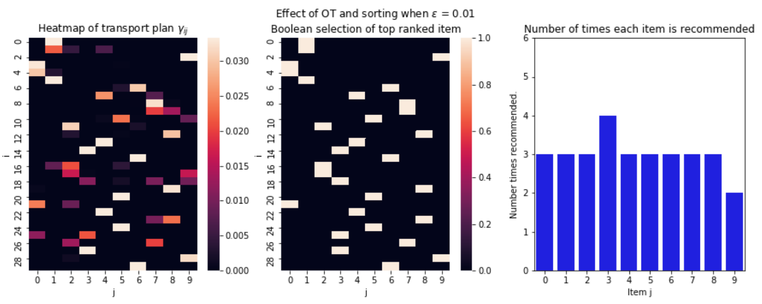


Table 4: XGBOOST Hyperparameters on MAR

num_boost_round	200
Loss	Logistic
Negative sampling ratio	50

Table 5: XGBOOST Hyperparameters on JOB

col_sample_bytree	0.6
eta	0.075
gamma	0.85
max_depth	12
min_child_weight	1
subsample	0.9
num_boost_round	400
Loss	Logistic
Negative sampling ratio	50

NN

The margin parameter η in the triplet loss is set to 1 in all experiments.

On MAR, NN is used with the hyper-parameters reported in Table 6. In each batch, 10 negative pairs are uniformly selected for each positive one.

Table 6: NN Hyperparameters on MAR

Layer 1	*tanh*, size = 300
Embedding	*tanh*, size = 300
Optimizer	Adam
Learning rate	0.001
Epochs	300
Batch size	64
Negative sampling ratio (per epoch)	10

On JOB, the neural architecture is adapted to account for the domain knowledge, involving four modules:

A "geographic" 2-100-100-50 module takes as input the (standard-scaled) latitude and longitude, with 2 hidden layers of size 100 and outputs a representation of the user/item location in dimension 50. All activation functions are tanh. This module is trained for 100 epochs (batch size 32) with Adam optimizer and base learning rate 10^{-4}. Negative sampling selects items farther than the actual positive one.

A "skill" 14,000-200-100 module takes as input the (standard-scaled) skills, with 1 hidden layer of size 200 (activation RELU) and outputs a representation of size 100 (activation function tanh). The module is trained for 100 epochs (batch size 32) with Adam optimizer and base learning rate 10^{-4}. The similarity matrix is diagonal.

An "other" d-500-200 module takes as input the other descriptive features, with $d = 448$ for users and $d = 582$ for items, with a hidden layer of size 500 (activation RELU) and outputs a representation of size 200 (activation function tanh). The module is trained for 100 epochs (batch size 32) with Adam optimizer and base learning rate 10^{-4}.

The overall architecture is warm-started using the preliminary training of the above three modules. The similarity matrix A is constrained to be block-wise diagonal. The module is trained for 35 epochs (batch size 256) with Adam optimizer and base learning rate 10^{-4}.

Except for the "geography" module, negative examples are sampled uniformly anew in each epoch, with a negative ratio of 50.

Other hyper-parameters are detailed in Table 5.

Table 7: Hyperparameters - NN (JOB)

Geography module	
Layer 1	*tanh*, size = 100
Layer 2	*tanh*, size = 100
Embedding	*tanh*, size = 50
Optimizer	Adam
Learning rate	0.0001
Epochs	100
Batch size	32
Skills module	
Layer 1	ReLu, size = 200
Embedding	*tanh*, size = 100
Optimizer	Adam
Learning rate	0.0001
Epochs	100
Batch size	32
Other module	
Layer 1	ReLu, size = 500
Embedding	*tanh*, size = 200
Optimizer	Adam
Learning rate	0.0001
Epochs	100
Batch size	256
Training from warm start	
Block-diagonal Structure	True
Epochs	10 / 25
Optimizer	Adam
Learning rates	0.0001 / 0.00001
Batch size	256

Appendix D: Full results

Table 8 give the full results obtained on the MAR dataset, including cluster-level evaluations.

Table 9 give the full results obtained on the JOB dataset. Figure 5 displays all method results in the 2D recall@10, congestion@10 plan. This display illustrates the trade-off between both indicators and shows the Pareto front of the non-dominated approaches.

Figure 6 displays so-called Lorenz curves plotting the percentage of overall job ads included in the top-t recommended items versus t (akin a Gini index).

Finally, table 10 displays computational costs. It shows a limited training time of respectively XGBOOST (circa 2 hours) and NN (circa 30 mn). The cost of optimal transport increases as ε decreases, up to circa 10mn for $\varepsilon = .01$ (see also [21]). The highest part of the cost comes from computing the recommendations with XGBOOST and γ^{XGB}, due to the fact that it requires to compute joint adequacy features for all (user, item) pairs.

Fig. 5: Pareto front Congestion (-Congestion@10) - Recommendation accuracy (Recall@10) tradeoff, JOB Dataset

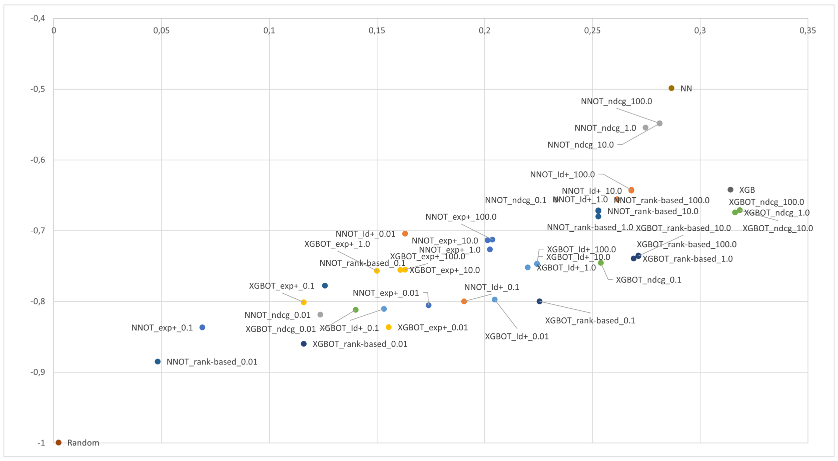


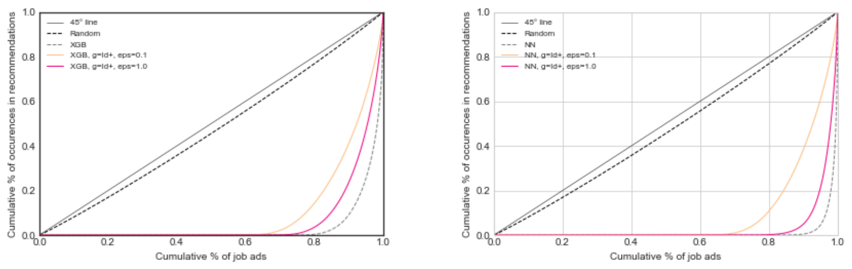
Table 8: Results - MAR Matrimonial dataset

Algorithm	Recall		Coverage		Congestion		Ind-Cluster		Cluster-Cluster	
	@1	@10	@1	@10	@1	@10	RSME	MAE	RSME	MAE
ϕ Random	0.16	2.27	63.32	100	-0.90	-0.98	12.68	6.186	nc	nc
ϕ XGBoost	7.93	27.88	48.55	98.69	-0.84	-0.94	12.60	5.619	nc	nc
ϕ NN	3.82	15.5	46.27	98	-0.83	-0.93	12.99	5.905	nc	nc
CAROT - XGBoost										
γ^{XGB},$g = exp+$,$\varepsilon = 100.0$	8.01	28.16	48.51	99.14	-0.84	-0.95	12.64	5.629	9.044	5.944
γ^{XGB},$g = exp+$,$\varepsilon = 10.0$	7.97	28.16	48.59	99.14	-0.84	-0.95	12.64	5.629	9.016	5.928
γ^{XGB},$g = exp+$,$\varepsilon = 1.0$	8.09	28.08	49.57	99.22	-0.85	-0.95	12.57	5.616	8.856	5.756
γ^{XGB},$g = exp+$,$\varepsilon = 0.1$	8.14	28.37	73.82	100	-0.93	-0.98	12.06	5.427	16.41	6.376
γ^{XGB},$g = exp+$,$\varepsilon = 0.01$	6.63	26.98	95.44	100	-0.98	-0.95	11.87	5.341	24.30	7.221
γ^{XGB},$g = Id+$,$\varepsilon = 100.0$	8.1	28.41	49.2	99.06	-0.84	-0.95	12.56	5.603	9.044	5.944
γ^{XGB},$g = Id+$,$\varepsilon = 10.0$	8.1	28.41	49.2	99.1	-0.84	-0.95	12.56	5.603	9.022	5.931
γ^{XGB},$g = Id+$,$\varepsilon = 1.0$	8.05	28.41	49.77	99.18	-0.85	-0.95	12.55	5.596	8.887	5.786
γ^{XGB},$g = Id+$,$\varepsilon = 0.1$	8.01	27.02	72.73	100	-0.93	-0.95	12.13	5.440	19.51	6.704
γ^{XGB},$g = Id+$,$\varepsilon = 0.01$	6.47	23.77	96.05	100	-0.98	-0.84	11.99	5.391	24.49	7.257
γ^{XGB},$g = ndcg$,$\varepsilon = 100.0$	7.93	28.2	48.55	98.98	-0.84	-0.95	12.60	5.619	nc	nc
γ^{XGB},$g = ndcg$,$\varepsilon = 10.0$	7.93	28.24	48.55	99.02	-0.84	-0.95	12.60	5.619	nc	nc
γ^{XGB},$g = ndcg$,$\varepsilon = 1.0$	7.93	28.2	48.55	99.02	-0.84	-0.95	12.60	5.619	nc	nc
γ^{XGB},$g = ndcg$,$\varepsilon = 0.1$	8.1	25.72	59.42	100	-0.89	-0.93	12.34	5.512	nc	nc
γ^{XGB},$g = ndcg$,$\varepsilon = 0.01$	6.06	19.49	94.26	100	-0.98	-0.73	12.02	5.433	nc	nc
γ^{XGB},$g = rank - based$,$\varepsilon = 100.0$	7.93	27.63	48.55	98.98	-0.84	-0.95	12.60	5.619	nc	nc
γ^{XGB},$g = rank - based$,$\varepsilon = 10.0$	7.93	27.63	48.55	98.98	-0.84	-0.95	12.60	5.619	nc	nc
γ^{XGB},$g = rank - based$,$\varepsilon = 1.0$	7.93	27.63	48.55	99.02	-0.84	-0.95	12.60	5.619	nc	nc
γ^{XGB},$g = rank - based$,$\varepsilon = 0.1$	7.53	25.76	62.43	99.95	-0.90	-0.93	12.25	5.473	nc	nc
γ^{XGB},$g = rank - based$,$\varepsilon = 0.01$	6.02	21.41	84.08	100	-0.96	-0.79	11.87	5.352	nc	nc
CAROT NN										
γ^{NN},$g = exp+$,$\varepsilon = 100.0$	1.5	9.32	13.75	51.4	-0.56	-0.65	16.10	6.824	9.045	5.945
γ^{NN},$g = exp+$,$\varepsilon = 10.0$	1.54	9.48	14.56	53.07	-0.58	-0.66	15.96	6.795	9.030	5.935
γ^{NN},$g = exp+$,$\varepsilon = 1.0$	1.95	11.35	20.38	65.76	-0.65	-0.75	15.17	6.655	8.976	5.839
γ^{NN},$g = exp+$,$\varepsilon = 0.1$	3.74	15.5	54.94	99.06	-0.87	-0.97	12.77	5.896	12.32	5.940
γ^{NN},$g = exp+$,$\varepsilon = 0.01$	3.78	15.67	88.15	100	-0.97	-0.97	12.03	5.543	23.14	7.164
γ^{NN},$g = Id+$,$\varepsilon = 100.0$	2.76	14	35	88.88	-0.77	-0.87	13.68	6.196	9.045	5.944
γ^{NN},$g = Id+$,$\varepsilon = 10.0$	2.72	14	35.4	89.58	-0.78	-0.88	13.64	6.179	9.024	5.931
γ^{NN},$g = Id+$,$\varepsilon = 1.0$	2.84	14.32	38.86	92.47	-0.80	-0.90	13.40	6.085	8.980	5.798
γ^{NN},$g = Id+$,$\varepsilon = 0.1$	3.94	15.46	70.12	100	-0.92	-0.98	12.36	5.668	17.08	6.512
γ^{NN},$g = Id+$,$\varepsilon = 0.01$	3.78	15.46	93.48	100	-0.98	-0.95	12.02	5.576	24.37	7.264
γ^{NN},$g = ndcg$,$\varepsilon = 100.0$	3.82	15.67	46.27	98.53	-0.83	-0.94	12.99	5.905	nc	nc
γ^{NN},$g = ndcg$,$\varepsilon = 10.0$	3.82	15.67	46.27	98.53	-0.83	-0.94	12.99	5.905	nc	nc
γ^{NN},$g = ndcg$,$\varepsilon = 1.0$	3.82	15.63	46.27	98.73	-0.83	-0.94	12.99	5.905	nc	nc
γ^{NN},$g = ndcg$,$\varepsilon = 0.1$	4.23	13.87	57.99	99.91	-0.88	-0.93	12.51	5.716	nc	nc
γ^{NN},$g = ndcg$,$\varepsilon = 0.01$	2.89	11.6	93.44	100	-0.98	-0.72	11.94	5.504	nc	nc
γ^{NN},$g = rank - based$,$\varepsilon = 100.0$	3.82	15.14	46.27	99.26	-0.83	-0.94	12.99	5.905	nc	nc
γ^{NN},$g = rank - based$,$\varepsilon = 10.0$	3.82	15.14	46.27	99.26	-0.83	-0.94	12.99	5.905	nc	nc
γ^{NN},$g = rank - based$,$\varepsilon = 1.0$	3.82	15.18	46.27	99.3	-0.83	-0.94	12.99	5.905	nc	nc
γ^{NN},$g = rank - based$,$\varepsilon = 0.1$	3.7	14.28	65.81	100	-0.91	-0.94	12.29	5.674	nc	nc
γ^{NN},$g = rank - based$,$\varepsilon = 0.01$	2.76	12.29	83.84	100	-0.96	-0.84	12.16	5.612	nc	nc

Table 9: Results - JOB Employment dataset

Algorithm	Recall			Coverage		Congestion		OT Comp. Time
	@1	@10	@100	@1	@10	@1	@10	sec.
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
ϕ Random	0	0.21	0.65	99.95	100	-0.99	-0.99	
ϕ XGB	9.62	31.4	61.59	12.94	25.16	-0.62	-0.64	
ϕ NN	5.68	28.66	57.98	6.02	17.78	-0.46	-0.49	
CAROT - XGBoost								
γ^{XGB},$g = exp+$,$\varepsilon = 1000.0$	3.93	16.3	52.18	20.98	34.25	-0.73	-0.75	35.93
γ^{XGB},$g = exp+$,$\varepsilon = 100.0$	3.93	16.3	52.18	21	34.26	-0.73	-0.75	40.04
γ^{XGB},$g = exp+$,$\varepsilon = 10.0$	3.93	15.86	52.18	21.03	34.33	-0.73	-0.75	49.91
γ^{XGB},$g = exp+$,$\varepsilon = 1.0$	3.71	14.98	50.76	20.93	34.8	-0.73	-0.75	45.84
γ^{XGB},$g = exp+$,$\varepsilon = 0.1$	1.53	11.59	49.67	27.23	44.7	-0.78	-0.80	55.20
γ^{XGB},$g = exp+$,$\varepsilon = 0.01$	3.06	15.97	52.29	48.88	59.05	-0.86	-0.83	514.1
γ^{XGB},$g = Id+$,$\varepsilon = 1000.0$	5.03	22.42	59.73	21.19	31.01	-0.74	-0.74	36.03
γ^{XGB},$g = Id+$,$\varepsilon = 100.0$	5.03	22.42	59.4	21.18	31.01	-0.74	-0.74	40.03
γ^{XGB},$g = Id+$,$\varepsilon = 10.0$	5.03	22.42	58.97	21.24	31.09	-0.74	-0.74	49.60
γ^{XGB},$g = Id+$,$\varepsilon = 1.0$	4.81	21.99	57.87	21.61	31.76	-0.74	-0.75	48.27
γ^{XGB},$g = Id+$,$\varepsilon = 0.1$	2.18	15.31	56.01	27.54	41.24	-0.78	-0.81	67.69
γ^{XGB},$g = Id+$,$\varepsilon = 0.01$	4.37	20.45	43.21	46.75	57.61	-0.85	-0.79	448.8
γ^{XGB},$g = ndcg$,$\varepsilon = 1000.0$	9.62	31.83	62.36	12.96	26.05	-0.62	-0.67	40.69
γ^{XGB},$g = ndcg$,$\varepsilon = 100.0$	9.62	31.83	62.36	12.96	26.05	-0.62	-0.67	37.50
γ^{XGB},$g = ndcg$,$\varepsilon = 10.0$	9.62	31.83	62.36	12.96	26.06	-0.62	-0.67	36.34
γ^{XGB},$g = ndcg$,$\varepsilon = 1.0$	9.62	31.61	62.36	12.96	26.14	-0.62	-0.67	42.03
γ^{XGB},$g = ndcg$,$\varepsilon = 0.1$	8.97	25.38	46.06	14.69	30.84	-0.67	-0.74	45.99
γ^{XGB},$g = ndcg$,$\varepsilon = 0.01$	5.03	14	18.81	36.81	57.52	-0.82	-0.81	478.0
γ^{XGB},$g = rank-based$,$\varepsilon = 1000.0$	9.4	27.13	60.5	15.82	37.2	-0.69	-0.73	36.36
γ^{XGB},$g = rank-based$,$\varepsilon = 100.0$	9.4	27.13	60.5	15.82	37.2	-0.69	-0.73	36.28
γ^{XGB},$g = rank-based$,$\varepsilon = 10.0$	9.4	27.13	60.28	15.85	37.2	-0.69	-0.73	39.69
γ^{XGB},$g = rank-based$,$\varepsilon = 1.0$	9.4	26.91	59.19	16.09	37.28	-0.69	-0.73	45.54
γ^{XGB},$g = rank-based$,$\varepsilon = 0.1$	7	22.53	44.42	24.06	38.74	-0.76	-0.79	49.69
γ^{XGB},$g = rank-based$,$\varepsilon = 0.01$	2.18	11.59	21	56.69	68.13	-0.87	-0.85	312.7
CAROT - NN								
γ^{NN},$g = exp+$,$\varepsilon = 1000.0$	5.25	20.35	51.2	19.7	32.96	-0.69	-0.71	36.46
γ^{NN},$g = exp+$,$\varepsilon = 100.0$	5.25	20.35	51.2	19.73	32.96	-0.69	-0.71	39.29
γ^{NN},$g = exp+$,$\varepsilon = 10.0$	5.25	20.13	50.98	19.83	33.1	-0.69	-0.71	49.46
γ^{NN},$g = exp+$,$\varepsilon = 1.0$	4.15	20.24	50	21.37	34.41	-0.71	-0.72	49.30
γ^{NN},$g = exp+$,$\varepsilon = 0.1$	0.65	6.89	42.23	35.04	50.43	-0.82	-0.83	58.90
γ^{NN},$g = exp+$,$\varepsilon = 0.01$	2.62	17.39	37.85	58.32	65.97	-0.87	-0.80	490.8
γ^{NN},$g = Id+$,$\varepsilon = 1000.0$	6.78	26.8	59.19	11.03	25.21	-0.60	-0.64	36.07
γ^{NN},$g = Id+$,$\varepsilon = 100.0$	6.78	26.8	59.19	11.05	25.21	-0.60	-0.64	36.08
γ^{NN},$g = Id+$,$\varepsilon = 10.0$	6.78	26.8	59.19	11.14	25.3	-0.60	-0.64	46.01
γ^{NN},$g = Id+$,$\varepsilon = 1.0$	6.78	26.14	60.39	11.99	26.3	-0.62	-0.65	49.23
γ^{NN},$g = Id+$,$\varepsilon = 0.1$	2.4	19.03	50.43	28.23	40.16	-0.80	-0.79	54.80
γ^{NN},$g = Id+$,$\varepsilon = 0.01$	3.93	16.3	27.89	53.38	62.35	-0.83	-0.70	518.9
γ^{NN},$g = ndcg$,$\varepsilon = 1000.0$	5.68	28.11	59.73	6.02	19.51	-0.46	-0.54	36.80
γ^{NN},$g = ndcg$,$\varepsilon = 100.0$	5.68	28.11	59.73	6.02	19.51	-0.46	-0.54	37.60
γ^{NN},$g = ndcg$,$\varepsilon = 10.0$	5.68	28.11	59.51	6.02	19.53	-0.46	-0.54	40.72
γ^{NN},$g = ndcg$,$\varepsilon = 1.0$	5.68	27.46	59.08	6.02	19.75	-0.46	-0.55	45.86
γ^{NN},$g = ndcg$,$\varepsilon = 0.1$	5.25	23.3	49.01	8.85	26.4	-0.53	-0.65	46.89
γ^{NN},$g = ndcg$,$\varepsilon = 0.01$	1.53	12.36	24.28	35.41	51.56	-0.81	-0.81	517.8
γ^{NN},$g = rank-based$,$\varepsilon = 1000.0$	5.68	25.27	52.95	10.74	38.47	-0.58	-0.67	37.20
γ^{NN},$g = rank-based$,$\varepsilon = 100.0$	5.68	25.27	52.95	10.76	38.47	-0.58	-0.67	36.78
γ^{NN},$g = rank-based$,$\varepsilon = 10.0$	5.68	25.27	52.95	10.79	38.5	-0.58	-0.67	40.34
γ^{NN},$g = rank-based$,$\varepsilon = 1.0$	5.68	25.27	52.29	11.24	38.71	-0.59	-0.67	46.50
γ^{NN},$g = rank-based$,$\varepsilon = 0.1$	3.06	12.58	40.7	26.55	42.65	-0.74	-0.77	49.85
γ^{NN},$g = rank-based$,$\varepsilon = 0.01$	0.65	4.81	25.6	61.91	73.03	-0.89	-0.88	502.9

Fig. 6: Lorenz curves computed on Top10 recommendations



(a) XGB on JOB dataset

(b) NN on JOB dataset

Table 10: Computational runtime in seconds on JOB (averaged over all g options). NN is trained on a server with 2 Intel Xeon Silver 4214 2,2GHz CPUs, 192Go RAM, and a Tesla T4 GPU. XGBOOST is trained on a DELL PowerEdge R640 server with 2X Intel Xeon Gold 6130 2.10GHz CPUs (2 $\times$ 16 cores) and 384Go RAM. The optimal transport plan is computed on the DELL with same resources as for XGBOOST.

Comp. Time	ϕ		γ^{XGB}			γ^{NN}		
	XGBoost	NN	$\epsilon = 0.01$	$\epsilon = 0.1$	$\epsilon = 1$	$\epsilon = 0.01$	$\epsilon = 0.1$	$\epsilon = 1$
Total	104,340	4,104	104,778	104,394	104,385	4,611	4,156	4,148
(inc. Learning/OT)	(7,454/−)	(2,039/−)	(7,454/438)	(7,454/54)	(7,454/45)	(2,039/507)	(2,039/52)	(2,039/44)