ReCon: Reducing Congestion in Job Recommendation using Optimal Transport

Yoosof Mashayekhi[1], Bo Kang[1], Jefrey Lijffijt[1], and Tijl De Bie[1]

Ghent University, Ghent, Belgium
{yoosof.mashayekhi,bo.kang,jefrey.lijffijt,tijl.debie}@ugent.be

Abstract. Recommender systems may suffer from *congestion*, meaning that there is an unequal distribution of the items in how often they are recommended. Some items may be recommended much more than others. Recommenders are increasingly used in domains where items have limited availability, such as the job market, where congestion is especially problematic: Recommending a vacancy—for which typically only one person will be hired—to a large number of job seekers may lead to frustration for job seekers, as they may be applying for jobs where they are not hired. This may also leave vacancies unfilled and result in job market inefficiency. We propose a novel approach to job recommendation called `ReCon`, accounting for the congestion problem. Our approach is to use an optimal transport component to ensure a more equal spread of vacancies over job seekers, combined with a job recommendation model in a multi-objective optimization problem. We evaluated our approach on two real-world job market datasets. The evaluation results show that `ReCon` has good performance on both congestion-related (e.g., Congestion) and desirability (e.g., NDCG) measures.

Keywords: Job Recommendation · Congestion Reduction.

1 Introduction

The aim of job recommendation is to recommend jobs (items) that are suitable for job seekers (users) and, if applied correctly, can have a positive impact on their career paths. In a job market, there is competition between job seekers for jobs and also between jobs (or companies) for job seekers [7]. This competition may be aggravated by the presence of so-called *congestion*, meaning that some items are more visible and desirable than others [4]. The use of job recommender systems risks to create or amplify congestion, as desirable items are often recommended to many users. This risks frustrating job seekers, as they will be unlikely to be successful when all applying for the same jobs. It is therefore important that job recommender systems explicitly account for congestion and try to limit or reduce it. The aim of this paper is to address that need.

Problem 1. The problem we address in this paper is to reduce congestion in job recommendation, i.e., to improve congestion-related measures such as Congestion [1], while recommending desirable recommendations, i.e., to keep good performance of desirability measures such as NDCG.

Our approach is to spread jobs over job seekers more equally using the optimal transport theory. We optimize the spread of jobs over job seekers with a given job recommendation model as a multi-objective task. Since optimal transport aims to match two distributions, we minimize the optimal transport between job seekers and jobs, to enforce a high matching degree between each job and a few job seekers and vice versa. Hence, we expect that the congestion in the generated recommendations by the model is reduced.

The joint optimization of the base recommendation model and the optimal transport component leads to having a trained model that offers advantages over post-processing approaches in real-life scenarios that require incremental updates of the model and also consider cold-start users and items.

The main **contributions** of this paper are:

- We introduce a novel approach, `ReCon`, to reduce congestion in a given job recommendation model. We jointly optimize the job recommendation model objective function and optimal transport cost between job seekers and jobs as a multi-objective task. (Section 2)
- We discuss the requirements for employing the so-called entropic optimal transport using the Sinkhorn algorithm [5] to facilitiate the optimization in `ReCon`. (Section 2.2)
- We evaluate `ReCon` on two job recommendation datasets and compare it with the baselines in terms of three desirability measures (e.g., NDCG), and also three congestion-related measures (e.g., Congestion). (Section 3)

Related work and conclusions are summarized in Sections 4 and 5 respectively.

2 Proposed Method

In this section, we introduce our approach, `ReCon`, to address Problem 1, reducing congestion in job recommendation. We first explain the multi-objective approach in Section 2.1. We then introduce the requirements for the efficient optimization of `ReCon` in Section 2.2. Finally, we suggest a matching cost function and a similarity function for computing optimal transport between job seekers and jobs in Section 2.3 that meet those requirements.

We also give a brief introduction to optimal transport theory and its efficient optimization using the Sinkhorn algorithm in Appendix A.

2.1 ReCon

Our approach is to optimize the desirability objective function of a recommendation model and an optimal transport cost between job seekers and jobs as a multi-objective task. The optimal transport problem distributes job seekers more equally over all jobs and vice versa. Hence, we expect that the recommendations from `ReCon` have lower congestion.

For a given recommendation model M with an objective function O_M and a set of parameters θ, we represent the matching score between users and items

by $\boldsymbol{P} = [p_{ui}]$. The matching score between user u and item i is defined by $p_{ui} = f_\theta(u, i)$, where f is a function of the set of parameters θ. We define the multi-objective task as follows:

$$O_{ReCon} = O_M + \lambda O_C, \tag{1}$$

where O_C is the objective function of the optimal transport between job seekers and jobs that is defined using the recommendation scores $\boldsymbol{P}$. The goal of O_C is to reduce the congestion in the recommendations. The weight of O_C in the multi-objective task is also indicated by the hyper-parameter λ.

To recommend top-k items to user u, we have to compare the recommendation scores between u and all items. To account for this competition between different items for the same user, it is important not only to penalize the cost of matching user-item pairs that are part of the optimal transport solution in the optimal transport objective function but also to penalize the similarity of user-item pairs that are not part of the optimal transport solution. Hence, the optimal transport optimization problem in `ReCon` is:

$$\begin{aligned} \min_{\boldsymbol{F}} \quad & \sum_{u\in U}\sum_{i\in I} f_{ui}c(p_{ui}) + (1-f_{ui})s(p_{ui}), \quad \text{s.t.} \quad f_{ui} \geq 0 \quad \forall (u,i) \in U \times I, \\ & \sum_{i\in I} f_{ui} = w_u \quad \forall u \in U, \quad \sum_{u\in U} f_{ui} = w_i \quad \forall i \in I, \end{aligned} \tag{2}$$

where c is a function of $\boldsymbol{P}$ to indicate the cost of matching a user and an item and s is a function of $\boldsymbol{P}$ to indicate the similarity of a user and an item. We can write the objective function in Eq. (2) as $\sum_{u\in U}\sum_{i\in I} s(p_{ui}) + \sum_{u\in U}\sum_{i\in I} f_{ui}(c(p_{ui}) - s(p_{ui}))$. Since the first double summation does not depend on $\boldsymbol{F}$, it can be optimized outside the linear program in Eq. (2). Thus, the objective function in the linear program can be written as $\sum_{u\in U}\sum_{i\in I} f_{ui}(c(p_{ui}) - s(p_{ui}))$, which can be optimized efficiently using the Sinkhorn algorithm. We denote this objective function by O_{COT}. Hence, the multi-objective task can be written as follows:

$$O_{ReCon} = O_M + \lambda O_{COT} + \lambda \sum_{u\in U}\sum_{i\in I} s(p_{ui}). \tag{3}$$

2.2 Efficient optimization

For efficient optimization, all terms in Eq. (3) have to be differentiable w.r.t. θ. We first require $\boldsymbol{P}$ to be differentiable w.r.t. θ. Next, the Sinkhorn algorithm could be used for optimizing O_{COT}, which is differentiable w.r.t. $\boldsymbol{P}$ when c and s are differentiable w.r.t. $\boldsymbol{P}$. We also require O_M to be differentiable w.r.t. $\boldsymbol{P}$. Hence, the objective function O_{ReCon} in Eq. (3) would be differentiable w.r.t. θ and optimization algorithms such as gradient descent may be employed for the optimization.

In summary, the requirements for efficient optimization are:

1. Recommendation scores $\boldsymbol{P}$ must be differentiable w.r.t. θ.
2. The objective function of M, O_M, must be differentiable w.r.t. $\boldsymbol{P}$.
3. Functions c and s must be differentiable w.r.t. $\boldsymbol{P}$.

2.3 The matching cost and similarity functions

High recommendation score p_{ui} indicates that user u and item i may match well. Hence, the cost of matching them in optimal transport should be low and their similarity should be high. If $p_{ui} > 0$, we suggest using the following functions c and s as the cost of matching u with i and the similarity of u and i respectively:

$$c(p_{ui}) = -\ln(p_{ui}), \quad s(p_{ui}) = -\ln(1 - p_{ui}). \tag{4}$$

The intuition behind choosing this term is explained in Appendix B.

Functions c and s in Eq. (4) are differentiable w.r.t. the recommendation scores $\boldsymbol{P}$. Hence, functions c and s in Eq. (4) meet the third requirement in Section 2.2, allowing for efficient optimization.

3 Experimental Evaluation

In this section, we describe the experimental evaluation of `ReCon` to see how does `ReCon` perform in congestion-related measures and desirability measures compared to the base recommendation models and the baselines.

3.1 Experimental evaluation details

We use a subset of VDAB[1] and CareerBuilder[2] datasets for evaluation. Moreover, CNE [6] and a content-based neural network (NN) are used as the base recommendation models. We compare `ReCon` with two baselines, CAROT [1] and FAirRec [9]. More details about the datasets, the recommendation models, baselines, evaluation settings, details of hyper-parameters, more results for baseline comparison, execution time comparison, and hyper-parameter sensitivity analysis for λ are presented in Appendix C. The source code and the public dataset for repeating the experiments is available here[3].

3.2 Results

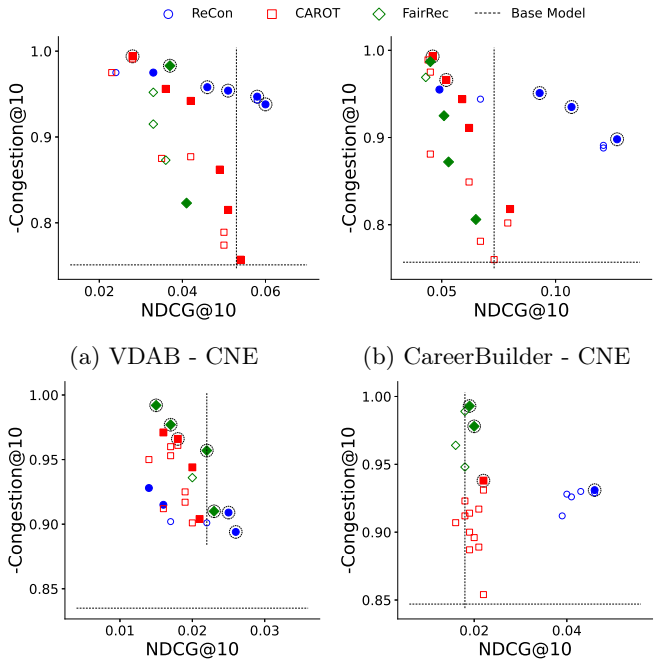
Here we compare the methods in terms of the desirability measures and congestion-related measures. Figure 1 shows desirability measures against congestion-related measures for all datasets. `ReCon` mostly improves congestion-related measures, while keeping good performance (or improving over the base recommendation

[1] `https://www.vdab.be/`
[2] `https://www.kaggle.com/c/job-recommendation`
[3] `https://anonymous.4open.science/r/ReCon-B83B/ReCon_source_code/`

model in some cases) of desirability measures for some selections of hyper-parameters. We can also observe that `ReCon` is Pareto optimal for some selections of hyper-parameters. While achieving almost the same or better desirability measure performance compared to the base recommendation models, `ReCon` can result in better congestion-related measures compared to the base recommendation models and the baselines.

The same conclusion could generally be established for all the desirability measures against all congestion-related measures, where for some selections of hyper-parameters, `ReCon` usually finds a good trade-off between both measures. More figures for different combinations of measures are available in Appendix C.

Compared to the baselines, ReCon has an additional advantage in that it trains a recommendation model, whereas the baselines simply generate a recommendation list. This makes ReCon particularly advantageous in real-world applications where the model requires incremental updates to account for new interactions or the model has to generate recommendations for cold-start users and cold-start items.



(a) VDAB - CNE (b) CareerBuilder - CNE

(c) VDAB - NN (d) CareerBuilder - NN

Fig. 1: Desirability versus congestion-related measures (higher values are better). Points represent different hyper-parameter combinations. Pareto optimal points per method are filled. Pareto optimal points across methods have a circle around.

4 Related Work

One important aspect of job recommendation models is that there is competition between job seekers/jobs for the same jobs/job seekers [7]. This competition amplifies the problem of congestion with recommendations in the job market.

In [3], a job application redistribution system was proposed for LinkedIn to prevent job postings from receiving an excessive or insufficient number of applications. To achieve this goal, the job recommendation scores were adjusted using a dynamic forecasting model that penalized or boosted scores based on the predicted number of applications. Chen et al. [4] use a decentralized economic model to learn the scoring functions and use an optimal transport approach for the final recommendations to reduce congestion in the recommendations.

To reduce congestion in job recommendation, two-step approaches (including CAROT) are proposed in [1,8]. The first step uses a model to predict the recommendation scores, and the second step employs optimal transport to better distribute jobs between job seekers and generate the final recommendations. In contrast, `ReCon` integrates the congestion reduction with recommendation into a single multi-objective task.

From a fairness perspective, two recent approaches (including FairRec) [9,2] develop greedy algorithms to ensure Envy-Free job recommendations for job seekers and to guarantee a minimum exposure for job vacancies.

5 Conclusion and Future Work

In this paper, we proposed a novel approach, `ReCon`, for reducing congestion in job recommendation systems using optimal transport theory. Our approach aimed to ensure a more equal spread of jobs over job seekers by combining optimal transport and a given job recommendation model in a multi-objective optimization problem. The evaluation results showed that our approach had good performance on both congestion-related and desirability measures. The proposed method can be a promising solution to the problem of congestion in job recommendation systems, and can potentially lead to more efficient and effective job matching.

For future work, we intend to evaluate `ReCon` using various recommendation models. Moreover, we aim to scale `ReCon` to be applicable to large-scale job market datasets by optimizing optimal transport in min-batches.

Acknowledgements The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) (ERC Grant Agreement no. 615517), and under the European Union's Horizon 2020 research and innovation programme (ERC Grant Agreement no. 963924), from the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" programme, and from the FWO (project no. G091017N, G0F9816N, 3G042220). Part of the experiments were conducted on pseudonimized HR data generously provided by VDAB.

References

1. Bied, G., Perennes, E., Naya, V., Caillou, P., Crépon, B., Gaillac, C., Sebag, M.: Congestion-avoiding job recommendation with optimal transport. In: FEAST workshop ECML-PKDD 2021 (2021)
2. Biswas, A., Patro, G.K., Ganguly, N., Gummadi, K.P., Chakraborty, A.: Toward fair recommendation in two-sided platforms. ACM Transactions on the Web (TWEB) **16**(2), 1–34 (2021)
3. Borisyuk, F., Zhang, L., Kenthapadi, K.: Lijar: A system for job application redistribution towards efficient career marketplace. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. p. 1397–1406. KDD '17, Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3097983.3098028, https://doi.org/10.1145/3097983.3098028
4. Chen, K.M., Hsieh, Y.W., Lin, M.J.: Prediction and congestion in two-sided markets: Economist versus machine matchmakers. SSRN Electronic Journal (2019)
5. Cuturi, M.: Sinkhorn distances: Lightspeed computation of optimal transport. In: Burges, C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K. (eds.) Advances in Neural Information Processing Systems. vol. 26. Curran Associates, Inc. (2013), https://proceedings.neurips.cc/paper_files/paper/2013/file/af21d0c97db2e27e13572cbf59eb343d-Paper.pdf
6. Kang, B., Lijffijt, J., De Bie, T.: Conditional network embeddings. In: 7th International Conference on Learning Representations, ICLR 2019 (2019)
7. Mashayekhi, Y., Li, N., Kang, B., Lijffijt, J., De Bie, T.: A challenge-based survey of e-recruitment recommendation systems. arXiv preprint arXiv:2209.05112 (2022)
8. Naya, V., Bied, G., Caillou, P., Crépon, B., Gaillac, C., Pérennes, E., Sebag, M.: Designing labor market recommender systems: the importance of job seeker preferences and competition. In: 4. IDSC of IZA Workshop: Matching Workers and Jobs Online-New Developments and Opportunities for Social Science and Practice (2021)
9. Patro, G.K., Biswas, A., Ganguly, N., Gummadi, K.P., Chakraborty, A.: Fairrec: Two-sided fairness for personalized recommendations in two-sided platforms. In: Proceedings of The Web Conference 2020. p. 1194–1204. WWW '20, Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3366423.3380196, https://doi.org/10.1145/3366423.3380196
10. Piketty, T.: The economics of inequality. Harvard University Press (2015)
11. Reimers, N., Gurevych, I.: Sentence-bert: Sentence embeddings using siamese bert-networks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics (11 2019)

Appendix A Optimal Transport

Optimal transport refers to the problem of finding the most efficient way to transport one given distribution of resources (e.g., masses, probabilities) to another. Mathematically, this can be represented as finding a transportation plan that minimizes the cost of transporting the resources from one place to another, subject to certain constraints. The cost of transporting the resources is often represented using a distance function. In the context of the present paper, we

are interested in the optimal transport between users U (job seekers) and items I (jobs), where $\boldsymbol{D} = [d_{ui}]$ denotes a cost of matching users with items. Hence, the optimal transport plan $\boldsymbol{F} = [f_{ui}]$ is given as:

$$\begin{aligned} \min_{\boldsymbol{F}} \quad & \sum_{u \in U} \sum_{i \in I} f_{ui} d_{ui}, \quad \text{s.t.} \quad f_{ui} \geq 0 \quad \forall (u,i) \in U \times I, \\ & \sum_{i \in I} f_{ui} = w_u \quad \forall u \in U, \quad \sum_{u \in U} f_{ui} = w_i \quad \forall i \in I. \end{aligned} \tag{5}$$

A tractable relaxation of the above optimization problem is proposed in [5], by regularization with an entropy term:

$$\begin{aligned} \min_{\boldsymbol{F}} \quad & \sum_{u \in U} \sum_{i \in I} f_{ui}(d_{ui} + \epsilon \log(f_{ui})), \\ & \text{s.t.} \quad \sum_{i \in I} f_{ui} = w_u \quad \forall u \in U, \quad \sum_{u \in U} f_{ui} = w_i \quad \forall i \in I, \end{aligned} \tag{6}$$

with ϵ the regularization weight. This relaxed optimization problem can be solved with the Sinkhorn algorithm [5], where its solution is differentiable w.r.t. the cost values $\boldsymbol{D}$, and suitable for various optimization tasks.

For simplicity, we set $w_u = \frac{1}{|U|}$ for all users $u \in U$ and $w_i = \frac{1}{|I|}$ for all items $i \in I$ in Eq. (5) and Eq. (6), so that the total mass for users and items is the same and all users, respectively items, are treated equally.

Appendix B The Matching Cost and Similarity Functions

In this section, we present our intuition for designing functions c and s as Eq.(4).

By choosing functions c and s by Eq.(4), the cost of optimal transport O_C in Eq. (1) is $-\sum_{u \in U} \sum_{i \in I} f_{ui} \ln(p_{ui}) + (1 - f_{ui}) ln(1 - p_{ui}) = -\sum_{u \in U} \sum_{i \in I} \ln(p_{ui}^{f_{ij}}) + \ln((1 - p_{ui})^{1 - f_{ij}}) = -\ln \prod_{u \in U} \prod_{i \in I} p_{ui}^{f_{ui}} (1 - p_{ui})^{1 - f_{ui}}$.

The intuition behind choosing this term is as follows. In the case where $|U| = |I|$ the optimal transport has an optimum where each user is mapped to exactly one item and vice versa, i.e., $f_{ij} \in \{0, 1\}$ in $\boldsymbol{F} = [f_{ij}]$. For this case, if $\boldsymbol{P}$ represents the matching probabilities, it is easy to see that computing the optimal transport would be the same as computing a matching between users and items with the highest probability accounting for the user-item pairs that are not part of the matching as well.

Appendix C Experimental Evaluation

In this section, we describe the datasets, the recommendation models, baselines, evaluation settings, details of hyper-parameters, more results for baseline comparison, execution time comparison, and hyper-parameter sensitivity analysis for λ.

C.1 Datasets

We evaluated the methods using two datasets:

VDAB[4]: VDAB is the employment service of Flanders in Belgium. The dataset contains a suitably anonymized sample of the applications made by job seekers to available job vacancies in the last ten days of 2018.

CareerBuilder[5]: CareerBuilder is an e-recruitment platform, which matches job seekers with jobs. We use the applications made by job seekers to available job vacancies in the last ten days of its public dataset[6] for the evaluation.

To better use collaborative filtering signals, we only keep job seekers and jobs with at least four interactions in both datasets. Table 1 shows the main statistics of the datasets.

Table 1: Main statistics of the datasets used for evaluation.

Dataset	Job seekers	Jobs	Train Interactions	Val. Interactions	Test Interactions
VDAB	1693	2931	9766	1428	2950
CareerBuilder	3876	4337	24316	1071	4557

C.2 Recommendation models

We evaluate methods using both collaborative filtering and content-based methods as the base recommendation models.

We use Conditional Network Embedding (CNE; [6]) as the base collaborative filtering recommendation model. CNE is a network embedding method that proposes a probability distribution for the network conditional on the embedding and finds the optimal embedding by maximum likelihood estimation. CNE satisfies the requirements in Section 2.2 for efficient optimization.

For the base content-based recommendation model (NN), we design a feed-forward neural network with a binary cross-entropy loss function. The features we use include numerical features, categorical features, and textual features. For the textual features, we use multi-lingual Sentence-BERT embeddings [11] and use PCA to reduce the dimensions to eight.

C.3 Baselines

We compare `ReCon` with the following baselines that both receive the recommendation scores from a trained model in their first step. Hence, both baselines have post-processing approaches.

[4] `https://www.vdab.be/`
[5] `https://www.careerbuilder.com/`
[6] `https://www.kaggle.com/c/job-recommendation`

CAROT [1]: CAROT employs optimal transport to redistribute jobs among job seekers. The final recommendation is according to the optimal transport solution. Since both methods presented in [1] and [8] have a similar approach, we only compare our proposed method with CAROT [1].

FairRec [9]: FairRec is a greedy method that ensures Envy-Free job recommendations for job seekers and guarantees a minimum exposure for job vacancies by converting the fair recommendation problem into a fair allocation problem. Since [9] and [2] have a similar approach and the modification in [2] is not focused on the congestion problem, we only compare our proposed method with FairRec [9]. As FairRec requires $|I| \leq k \cdot |U|$, we did not evaluate it for $k = 1$.

C.4 Settings

Both datasets include the interactions in a period of ten days. For each dataset, we use the interactions in the first six days for training, the seventh day for validation (for early stopping based on AUC for CNE and Congetion@1 for `ReCon`), and the last three days for the test set. We also use the source code provided by the authors for the baselines.

For Carot, we select ϵ from {1, 10, 100}. We also use four different functions to obtain the matching costs in the second step of the method (that solves the optimal transport problem) including linear (Id+), exponential (Exp+), rank-based (Rank), or NDCG-like (NDCG) functions as mentioned in CAROT paper [1]. For FairRec, we select α (the coefficient for the producer-side guarantee in their method) from {0.2, 0.4, 0.6, 0.8, 1.0}. Since FairRec only generates the recommendations and not the scores, we run the method for each value k in the top-k recommendation for evaluation. For `ReCon` with CNE, we select λ from {0.001, 0.005, 0.01, 0.05, 0.1} for both datasets. For `ReCon` with NN, we select λ from {0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05} for VDAB and from {1e-11, 5e-11, 1e-10, 5e-10, 1e-09, 5e-09} for CareerBuilder.

We evaluate the methods on the following desirability measures:

- NDCG: Normalized discounted cumulative gain, which is computed by summing the true scores ranked in the order induced by the predicted scores after applying a logarithmic discount, divided by the best possible score.
- Recall: The proportion of relevant items found in the top-k recommendations.
- Hit Rate: The fraction of users for which the correct answer is included in the top-k recommendations.

We also evaluate the methods on the following congestion-related measures:

- Congestion: Congestion is defined as the negative entropy of item market shares, where item market share $MS_l(i)$ of item i is defined as the fraction of users such that item i appears among their top-k recommendations [1]. As suggested by Bied et al. [1], Congestion is divided by $\log(|I|)$ (I represents the set of items) to map it to $[-1, 0]$. Congestion is minimized if the entropy of the market shares is minimized. Hence, -1 is the optimal value.

- Coverage: The fraction of items involved in top-k recommendation of at least one user [1].
- Gini Index: The Gini index is a statistical measure of the inequality of a distribution [10]. In the context of job recommendation, the Gini index is computed for the market share of items.

C.5 Details of hyper-parameters of the recommendation models

In this section, we describe the hyper-parameters used in the recommendation models.

For the base CNE model, we use the degree prior (for more information please see [6]) and use dot product as the distance function. We also perform hyper-parameter tuning based on validation AUC to select the hyper-parameters. For both datasets, the best embedding dimension is four and the best learning rate is 0.05. The best weight decay in AdamW optimizer for VDAB is 0.1 and for CareerBuilder is 0.01. The batch size is set to 4096 for VDAB and 2048 for CareerBuilder.

For ReCon with CNE,we experimented with different combinations of hyper-parameters and monitored all desirability measures and congestion-related measures. For both datasets, the embedding dimension eight shows to have a good trade-off between the desirability measures and the congestion-related measures. For VDAB, we set the learning rate to 0.05 and weight decay to 0.01. For CareerBuilder, we set the learning rate to 0.1 and the weight decay to 0.001. We used the same batch size as the base CNE model. We also set ϵ to 10, and the number of Sinkhorn iterations to 100 for both datasets.

For the base content-based model NN, we perform hyper-parameter tuning based on validation AUC to select the hyper-parameters. For VDAB, we set the batch size to 512, embedding dimension for categorical features to 32, the learning rate to 0.05, and weight decay to 0.1. the network consists of four layers with 32, 16, 8, and 4 units. For CareerBuilder, we set the batch size to 256, the embedding dimension for categorical features to 16, the learning rate to 0.0005, and the weight decay to 0.0001. the network consists of four layers with 16, 16, 8, and 4 units.

For ReCon with NN, we experimented with different combinations of hyper-parameters, monitored all desirability measures and congestion-related measures, and selected a set of hyper-parameters that show a good trade-off between all measures. For VDAB, we set the batch size to 4096, the embedding dimension for categorical features to 16, the learning rate to 0.0005, and the weight decay to 0.1. the network consists of two layers with 64 and 32 units. For CareerBuilder, we set the batch size to 2048, the embedding dimension for categorical features to 4, the learning rate to 0.0005 and weight decay to 0.01. the network consists of two layers with 64 and 16 units. We also set ϵ to 10 for both datasets and set the number of Sinkhorn iterations to 5 for VDAB and to 10 for CareerBuilder.

C.6 Baseline comparison

Here we compare the methods in terms of the desirability measures and congestion-related measures. Figures 2-13 show the performance of all methods, where they all compare a desirability measure (NDCG, Recall, or Hit Rate) and a congestion-related measure (Congestion, Coverage, or Gini Index). We can observe that for some selections of hyper-parameters, `ReCon` usually finds a good trade-off between both measures.

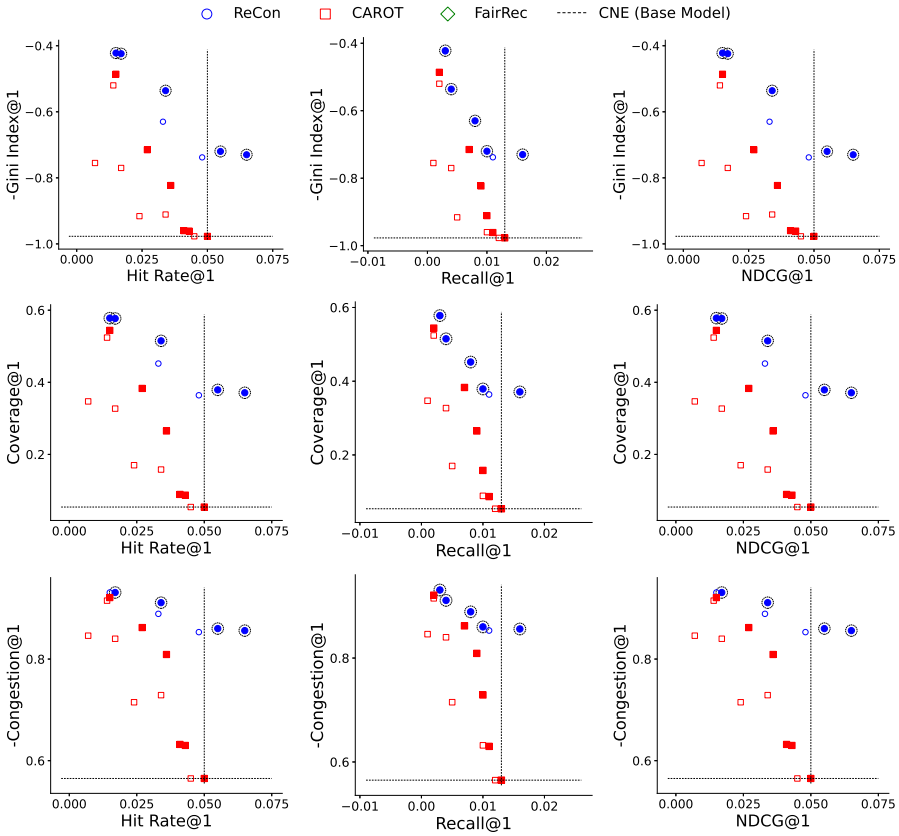


Fig. 2: Desirability versus congestion-related measures in VDAB dataset with CNE for top-1 recommendation (higher values are better). Points represent different hyper-parameter combinations. Pareto optimal points per method are filled. Pareto optimal points across methods have a circle around.

Execution time comparison In this experiment, we compare the methods in terms of the execution time.

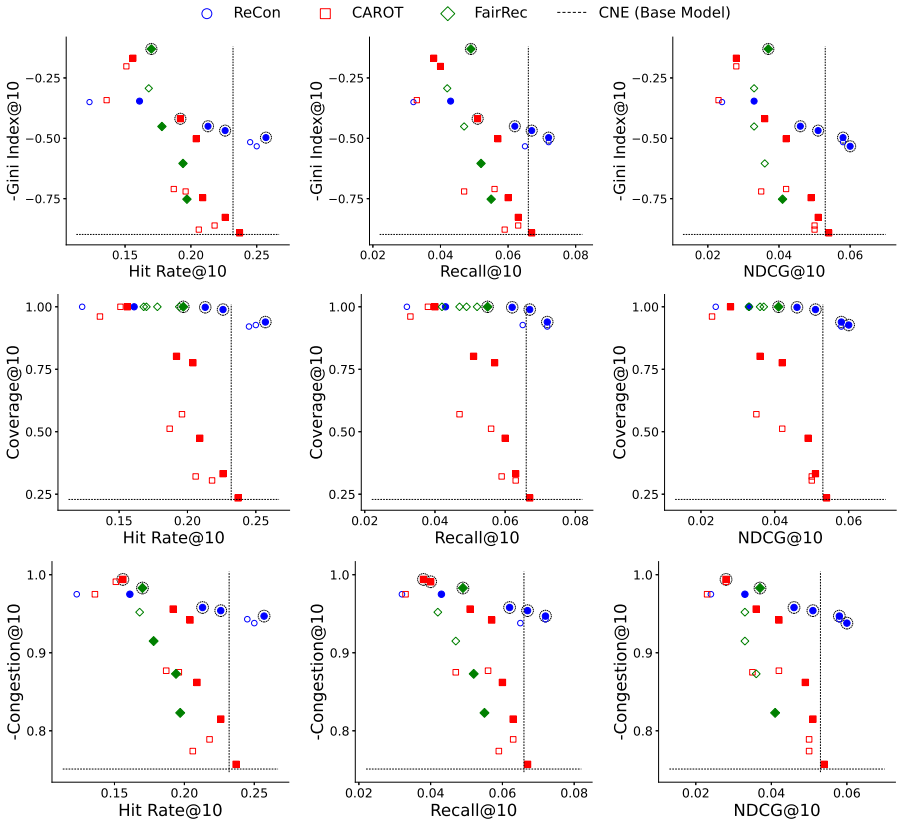


Fig. 3: Desirability versus congestion-related measures in VDAB dataset with CNE for top-10 recommendation (higher values are better). Points represent different hyper-parameter combinations. Pareto optimal points per method are filled. Pareto optimal points across methods have a circle around.

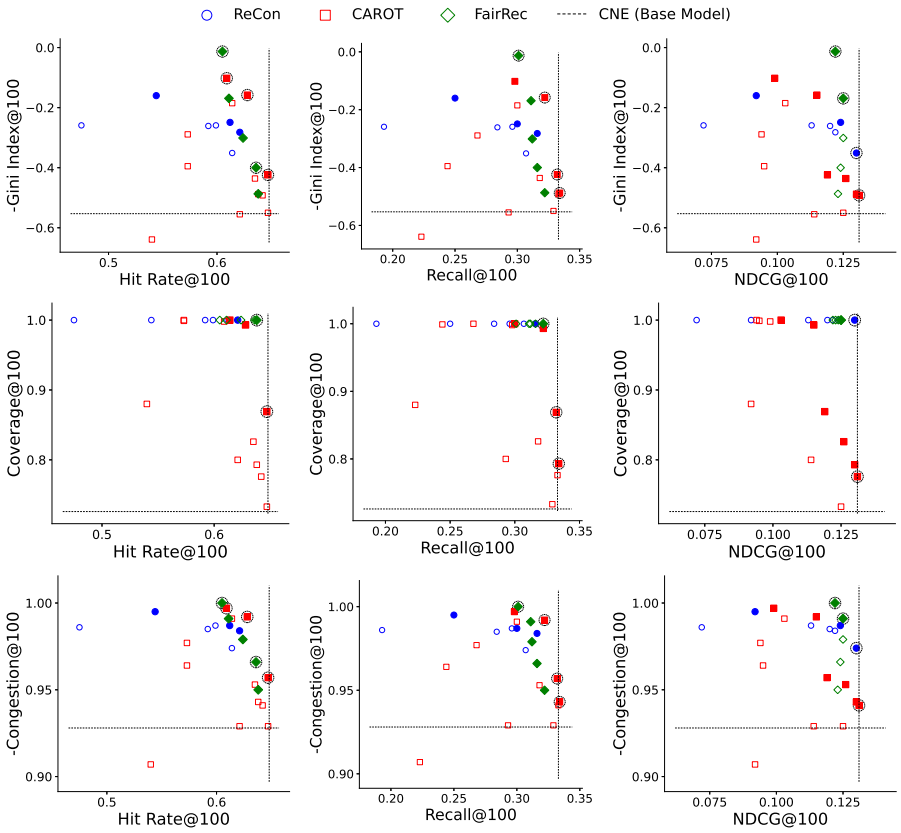


Fig. 4: Desirability versus congestion-related measures in VDAB dataset with CNE for top-100 recommendation (higher values are better). Points represent different hyper-parameter combinations. Pareto optimal points per method are filled. Pareto optimal points across methods have a circle around.

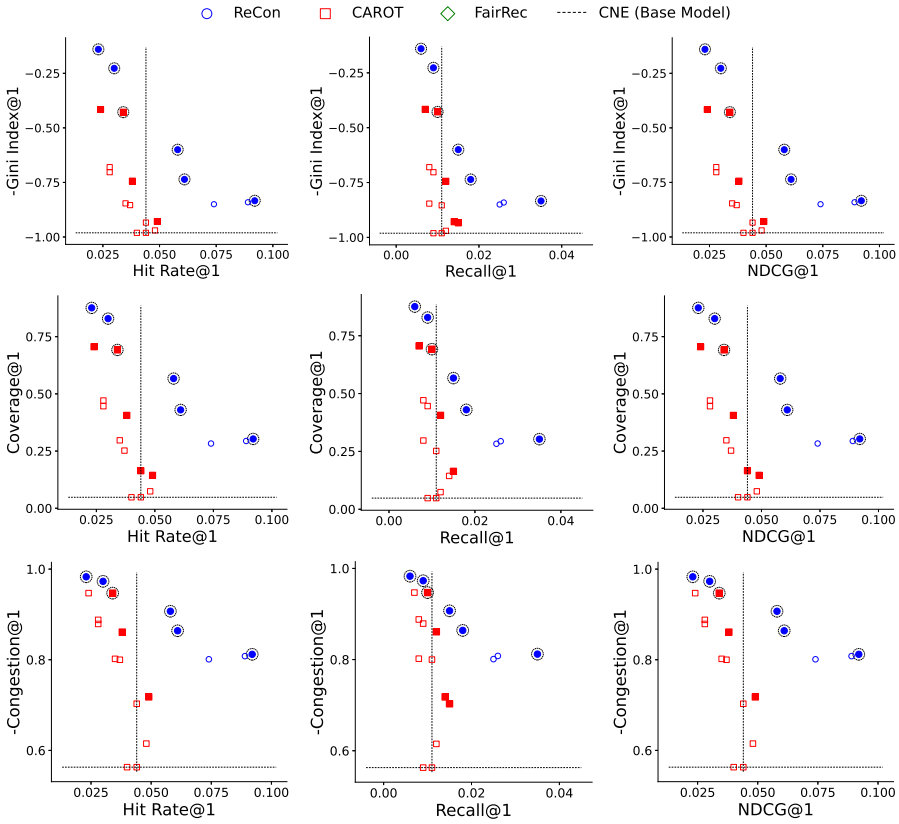


Fig. 5: Desirability versus congestion-related measures in CareerBuilder dataset with CNE for top-1 recommendation (higher values are better). Points represent different hyper-parameter combinations. Pareto optimal points per method are filled. Pareto optimal points across methods have a circle around.

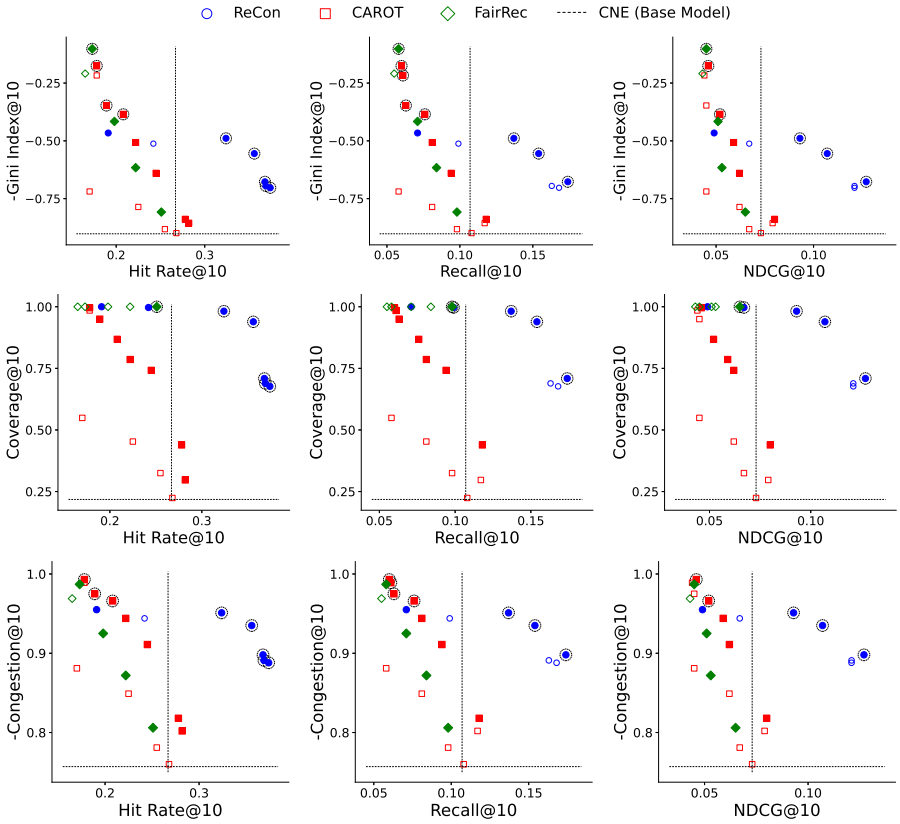


Fig. 6: Desirability versus congestion-related measures in CareerBuilder dataset with CNE for top-10 recommendation (higher values are better). Points represent different hyper-parameter combinations. Pareto optimal points per method are filled. Pareto optimal points across methods have a circle around.

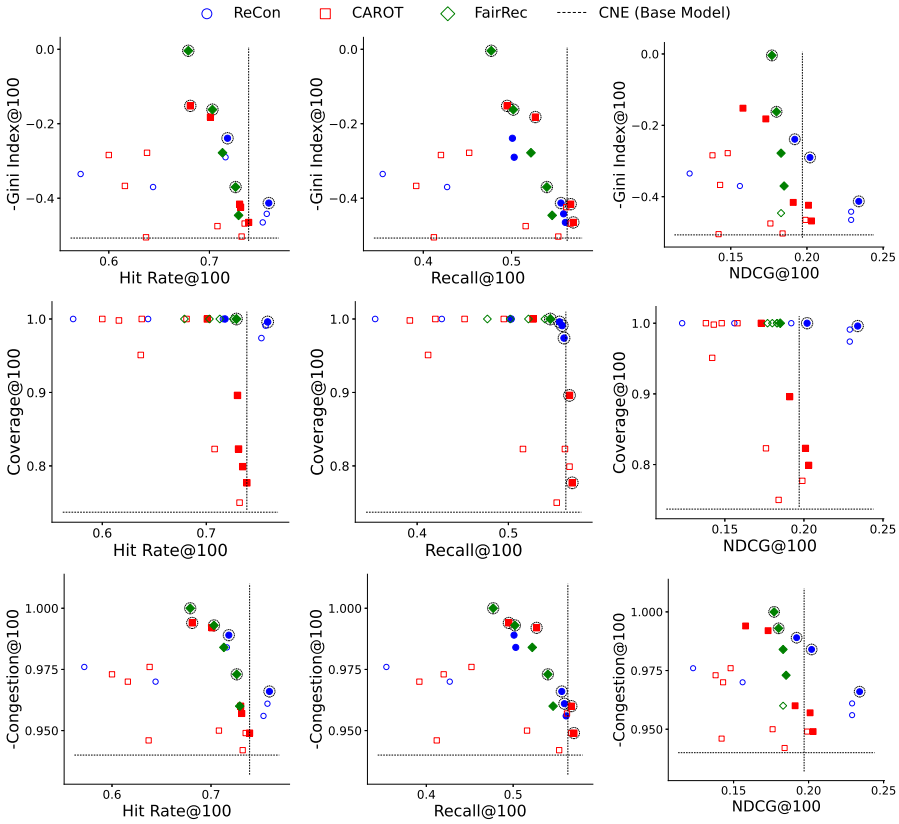


Fig. 7: Desirability versus congestion-related measures in CareerBuilder dataset with CNE for top-100 recommendation (higher values are better). Points represent different hyper-parameter combinations. Pareto optimal points per method are filled. Pareto optimal points across methods have a circle around.

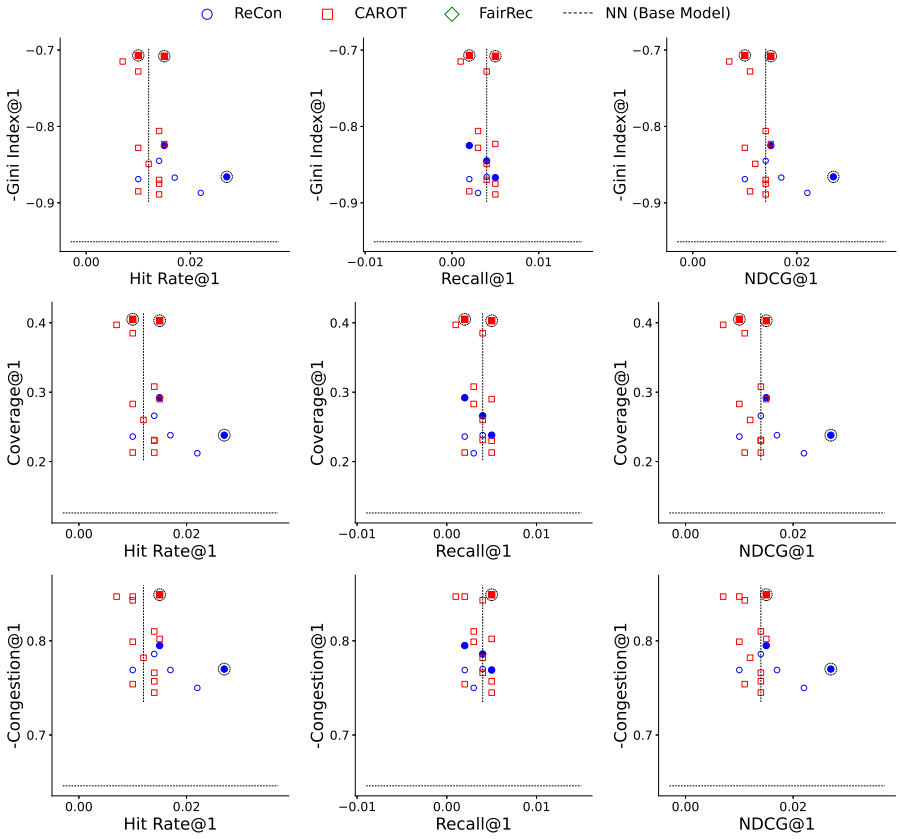


Fig. 8: Desirability versus congestion-related measures in VDAB dataset with NN for top-1 recommendation (higher values are better). Points represent different hyper-parameter combinations. Pareto optimal points per method are filled. Pareto optimal points across methods have a circle around.

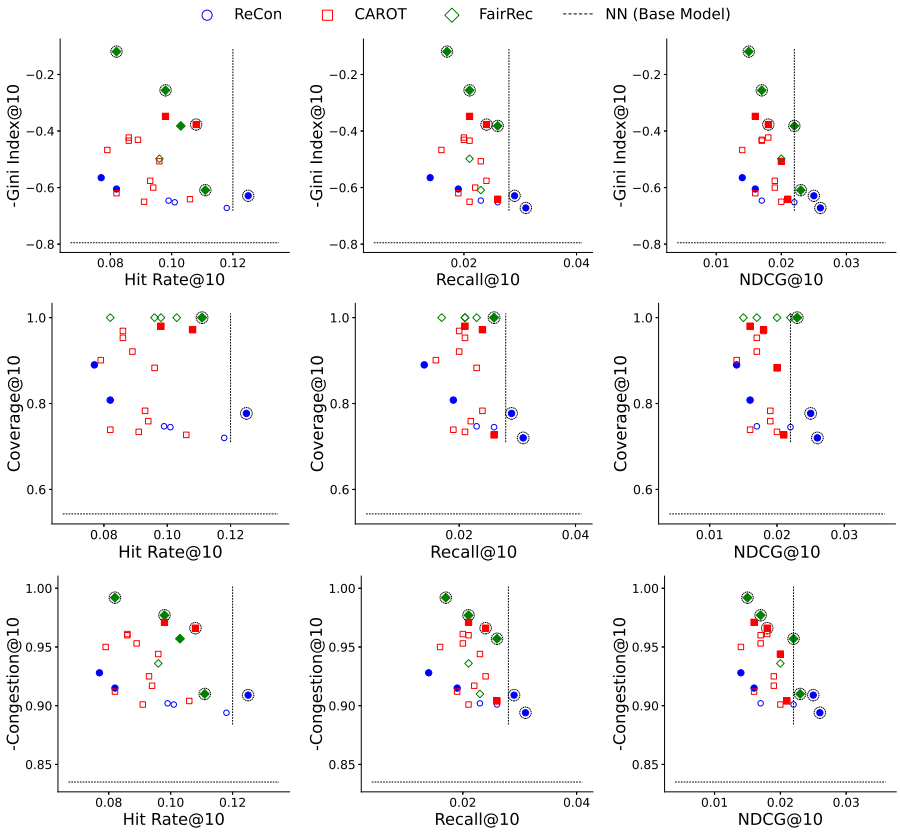


Fig. 9: Desirability versus congestion-related measures in VDAB dataset with NN for top-10 recommendation (higher values are better). Points represent different hyper-parameter combinations. Pareto optimal points per method are filled. Pareto optimal points across methods have a circle around.

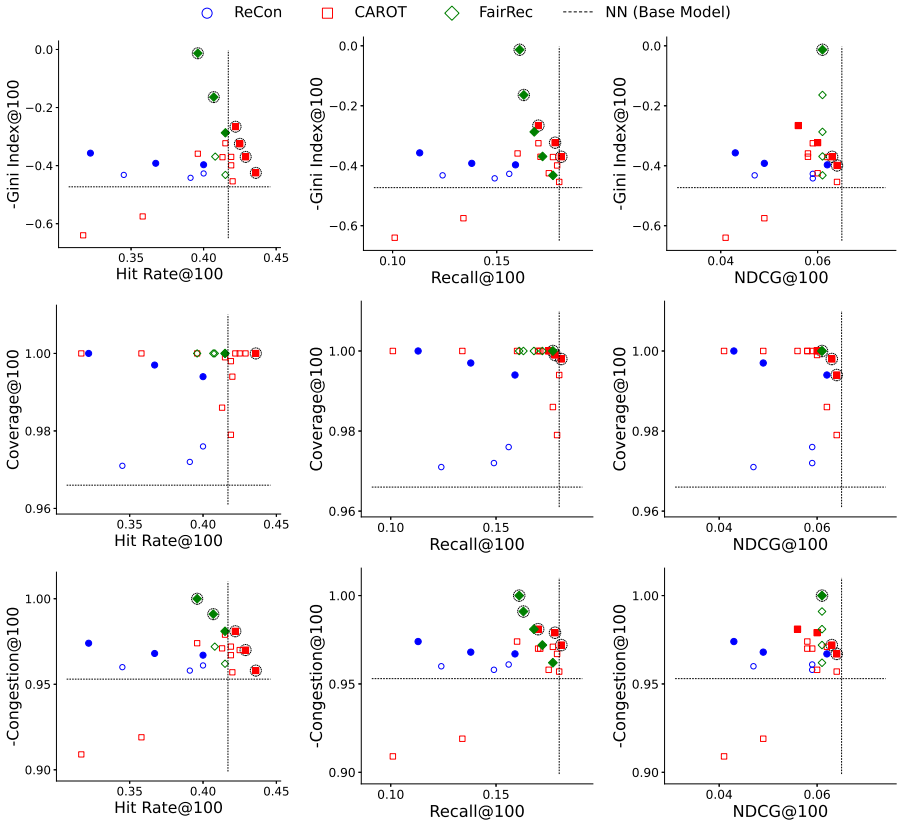


Fig. 10: Desirability versus congestion-related measures in VDAB dataset with NN for top-100 recommendation (higher values are better). Points represent different hyper-parameter combinations. Pareto optimal points per method are filled. Pareto optimal points across methods have a circle around.

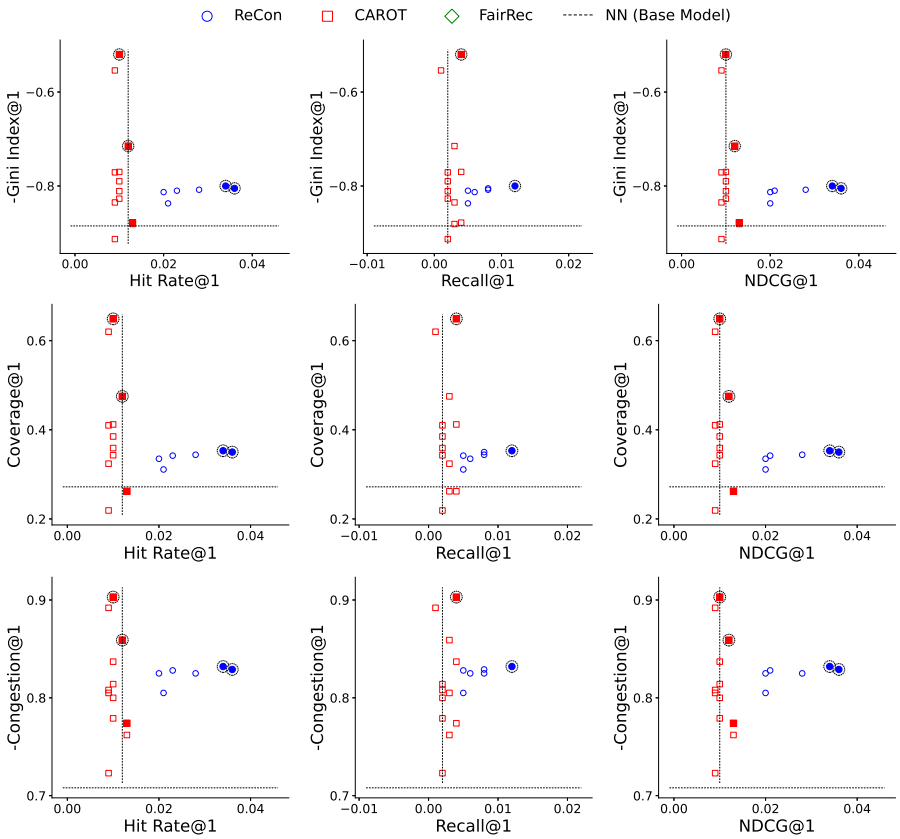


Fig. 11: Desirability versus congestion-related measures in CareerBuilder dataset with NN for top-1 recommendation (higher values are better). Points represent different hyper-parameter combinations. Pareto optimal points per method are filled. Pareto optimal points across methods have a circle around.

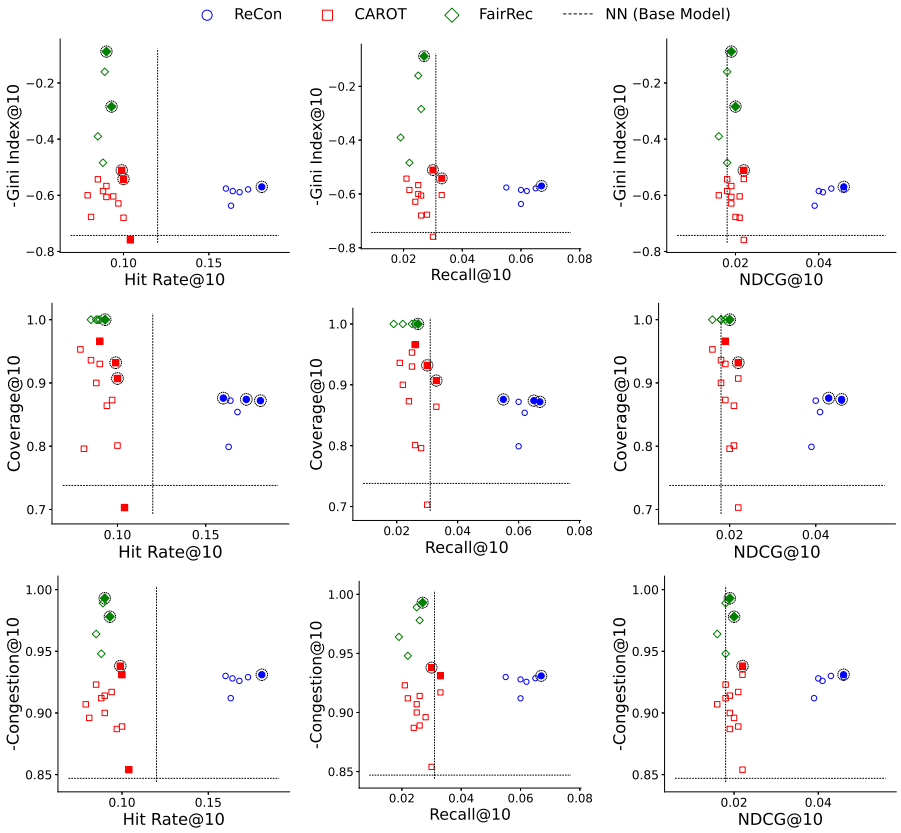


Fig. 12: Desirability versus congestion-related measures in CareerBuilder dataset with NN for top-10 recommendation (higher values are better). Points represent different hyper-parameter combinations. Pareto optimal points per method are filled. Pareto optimal points across methods have a circle around.

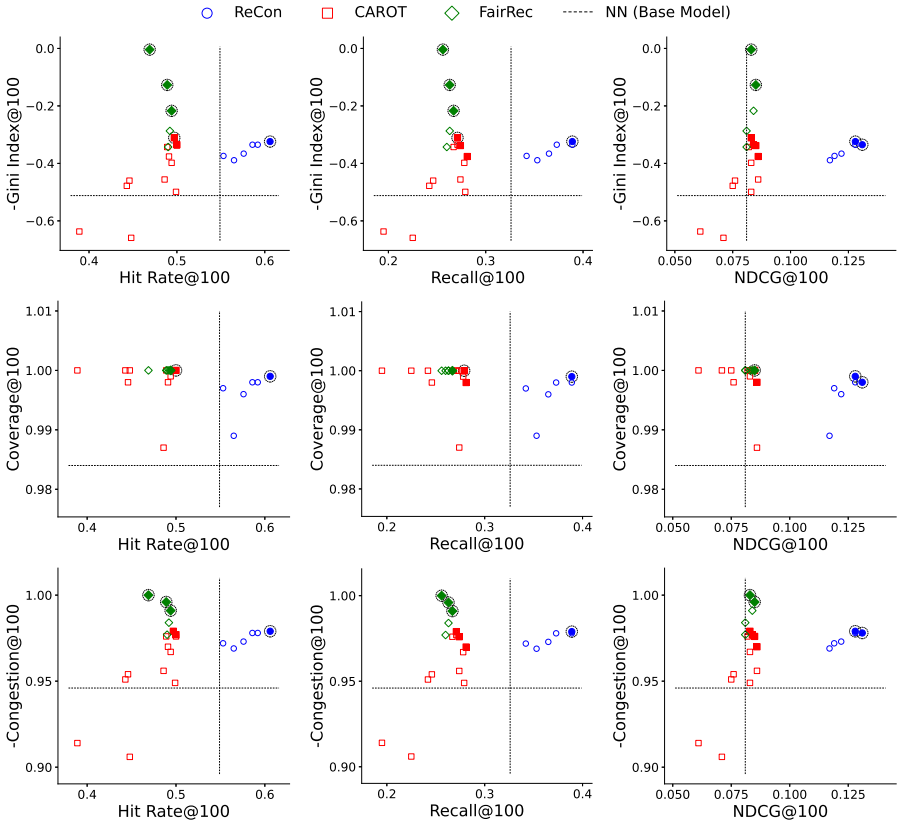


Fig. 13: Desirability versus congestion-related measures in CareerBuilder dataset with NN for top-100 recommendation (higher values are better). Points represent different hyper-parameter combinations. Pareto optimal points per method are filled. Pareto optimal points across methods have a circle around.

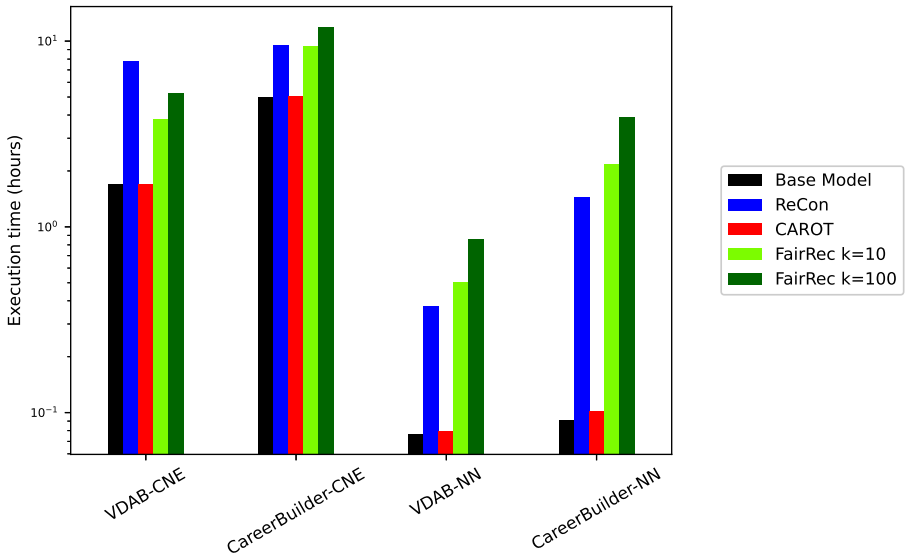


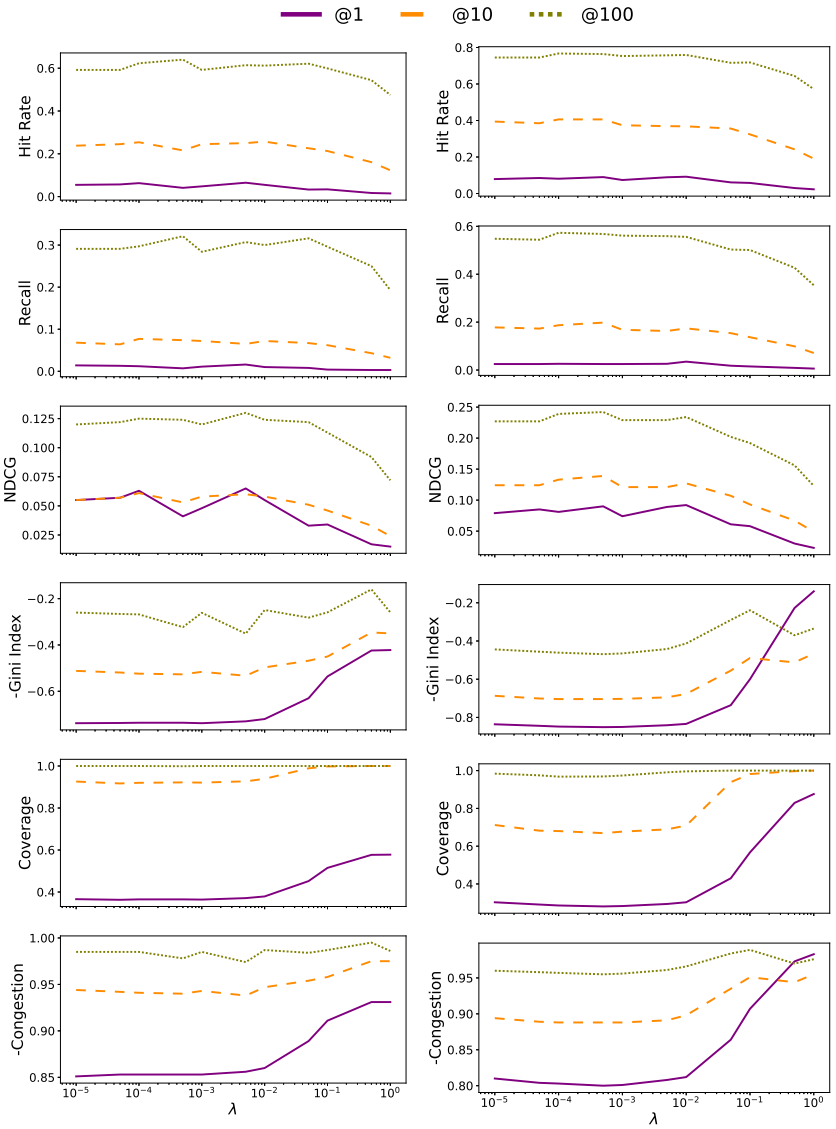
Fig. 14: Execution time (log-scale) for both datasets and both base recommendation models. The execution time of each baseline is averaged over different selections of hyper-parameters.

Figure 14 shows the execution time in hours for both datasets and both base recommendation models. The execution time of each baseline is averaged over different selections of hyper-parameters. The training time of CNE is added to the execution time of CAROT and FairRec, as they receive the matching scores from the base recommendation model. `ReCon` is slower than the base recommendation models and CAROT since it optimizes the optimal transport cost and the base recommendation model jointly.

The execution time of FairRec depends on the value of k in the top-k recommendation, where it is faster for smaller values of k. `ReCon` is faster than FairRec with NN and also for $k = 100$ in the CareerBuilder dataset with CNE.

C.7 Hyper-parameter sensitivity analysis for λ

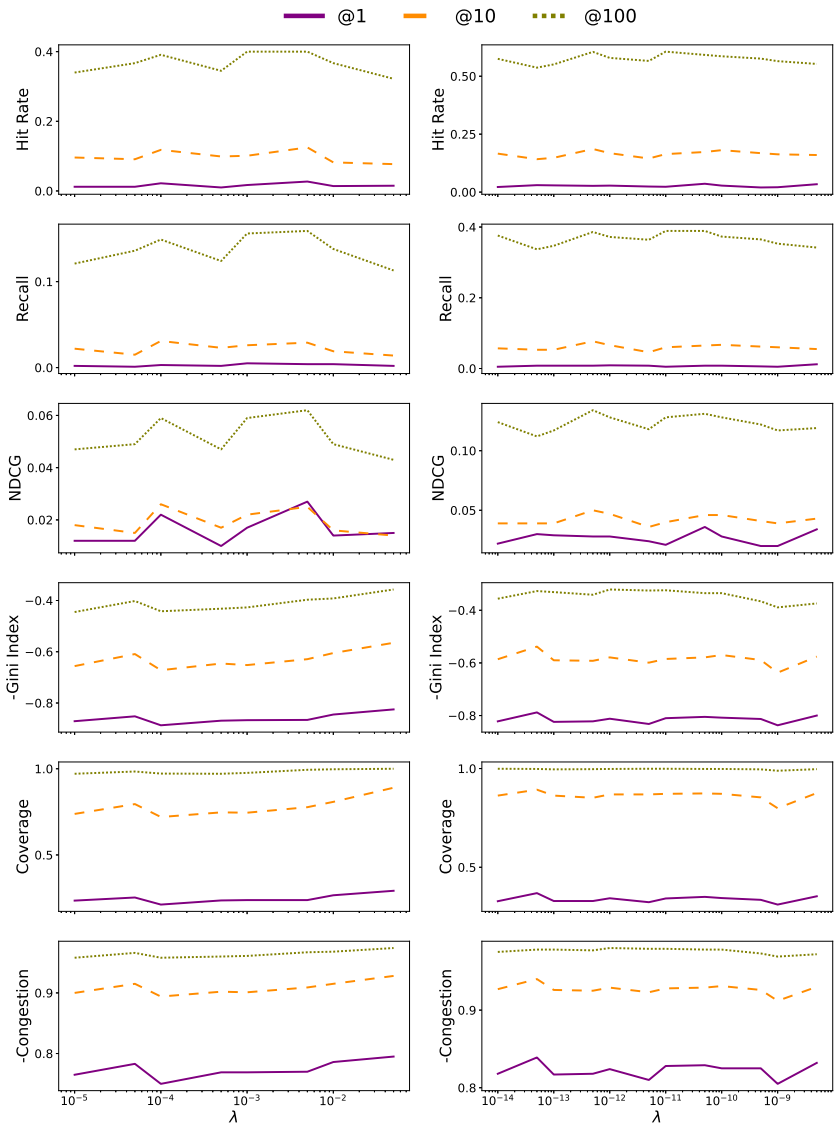
In this section, we analyze the performance of `ReCon` for different values of λ (the weight of the optimal transport in `ReCon`). Figure 15 and Figure 16 show different measures for different values of λ for both datasets with CNE and NN, respectively. As expected, we can observe that congestion-related measures mostly improve by increasing λ, but at the cost of desirability measures.



(a) VDAB

(b) CareerBuilder

Fig. 15: Performance of `ReCon` in both datasets with CNE for different values of λ (the weight of the optimal transport in `ReCon`).



(a) VDAB (b) CareerBuilder

Fig. 16: Performance of `ReCon` in both datasets with NN for different values of λ (the weight of the optimal transport in `ReCon`).