Fast Learning of MNL Model from General Partial Rankings with Application to Network Formation Modeling

Jiaqi Ma[*]
jiaqima@umich.edu
School of Information
University of Michigan
Ann Arbor, Michigan, USA

Xingjian Zhang[*]
jimmyzxj@umich.edu
Department of EECS
University of Michigan
Ann Arbor, Michigan, USA

Qiaozhu Mei
qmei@umich.edu
School of Information and
Department of EECS
University of Michigan
Ann Arbor, Michigan, USA

ABSTRACT

Multinomial Logit (MNL) is one of the most popular discrete choice models and has been widely used to model ranking data. However, there is a long-standing technical challenge of learning MNL from many real-world ranking data: exact calculation of the MNL likelihood of *partial rankings* is generally intractable. In this work, we develop a scalable method for approximating the MNL likelihood of general partial rankings in polynomial time complexity. We also extend the proposed method to learn mixture of MNL. We demonstrate that the proposed methods are particularly helpful for applications to choice-based network formation modeling, where the formation of new edges in a network is viewed as individuals making choices of their friends over a candidate set. The problem of learning mixture of MNL models from partial rankings naturally arises in such applications. And the proposed methods can be used to learn MNL models from network data without the strong assumption that temporal orders of all the edge formation are available. We conduct experiments on both synthetic and real-world network data to demonstrate that the proposed methods achieve more accurate parameter estimation and better fitness of data compared to conventional methods[1].

CCS CONCEPTS

• **Information systems** → **Learning to rank**; **Probabilistic retrieval models**; *Web searching and information discovery*.

KEYWORDS

Learning to rank; multinomial logit model; Plackett-Luce model; partial ranking; network formation modeling

---

[*]Both authors contributed equally to this research.

[1]Code available at https://github.com/xingjian-zhang/Fast-Partial-Ranking-MNL.

---

1 INTRODUCTION

Discrete choice models [49] concern how individuals make choices from a candidate set of alternatives, and have wide applications in many areas, such as recommendation [35], information retrieval [25], economics [31], etc. With the increasing availability of abundant real-world data, more and more often we not only have the record of people's single choice over the candidate set, but also relative rankings of multiple items. Learning discrete choice models from ranking data has thus attracted much research attention [15, 24, 25]. In this paper, we focus on the problem of learning Multinomial logit (MNL) model, which is one of the most popular discrete choice models and also known as Plackett-Luce model [27, 42].

One long-standing technical challenge for learning MNL model from real-world ranking data is that the exact calculation of the MNL likelihood of *partial rankings* is generally intractable when the number of candidate choices is large. This computational challenge has limited the application of MNL model to some special types of ranking data, such as *Top-K ranking*, where the exact order of the top K candidates is required to be known. There are two lines of research that aim to approximate the MNL likelihood of general partial rankings. The first direction is to use *rank breaking* [20, 21, 46], which usually extracts pairwise comparisons from the general partial rankings and treats the pairs as independent observations. To mitigate the information loss in the extraction of pairwise comparisons, Khetan and Oh [21] further proposed *generalized rank breaking*, which extracts maximal ordered partitions from the general partial rankings to preserve more order information. The second direction is to approximate the likelihood by Markov Chain Monte Carlo (MCMC) sampling [24]. However, both generalized rank breaking and sampling-based methods suffer from an exponential time complexity in terms of the number of candidate choices, which are not suitable for large-scale social network modeling. Recently, Ma et al. [28] demonstrated that the likelihood of *Partitioned-Preference rankings*, which is a special type of partial rankings but is more general than Top-K rankings, can be efficiently approximated through a numerical integral approach.

In this paper, our first contribution is the development of a scalable method for learning MNL from general partial rankings, by combining the idea of generalized rank breaking [21] and the numerial integral approach [28]. We also extend the method to learn the mixture of MNL models by proposing an Expectation-Maximization (EM) algorithm with a novel initialization method to facilitate better

convergence. Through simulation study, we demonstrate that the proposed methods are able to achieve significantly lower computational cost while having similar estimation accuracy compared to state-of-the-art sampling-based baseline methods.

We further demonstrate that the proposed methods are particularly useful on applications to choice-based network formation modeling, which is a new and exciting application domain of the MNL model recently introduced by Overgoor et al. [39]. Concretely, we can view the social network formation process as people making choices of their friends. The formation of each new edge in the network can be viewed as a node choosing to connect another node, over all the candidate nodes available. Most prior choice-based network modeling methods assume the temporal orders of all the edge formation are available and interpret the sequentially established edges as top-K rankings [39, 40]. However, such assumptions may be unrealistic in practice. On one hand, there are a lot of network data where the temporal formation information is partially missing or even totally unavailable. On the other hand, the temporal information does not necessarily imply the level of preference in the friend choices. Our proposed methods, instead, do not rely on such strong assumptions when being used to learn the MNL models.

We apply the proposed methods to large-scale network formation modeling on both synthetic and real-world networks. We generate synthetic network through mixture of preferential attachment and uniform attachment models. Our empirical results demonstrate that the proposed methods are able to faithfully recover the ground truth parameters of the generative models, while prior choice-based network formation modeling methods fail when the temporal information is missing. Experiments on two real-world networks show that the proposed methods are able to achieve better link predication accuracy than prior baselines, suggesting that the proposed methods are able to achieve a better fitness of the data.

2 RELATED WORK

2.1 Learning MNL from Partial Rankings

Many algorithms have been proposed to learn MNL model from ranking data [1, 5, 15, 21, 28, 30, 36] and its mixture [10, 24, 34, 37, 54]. However, few of them is able to learn mixture of MNL models from general partial rankings in a tractable time. Among the existing methods to learn MNL models, some are designed for full rankings [1, 54]; some are designed for special cases of partial rankings [15, 21, 28, 30, 34, 36], e.g., Top-K rankings or Partitioned-Preference rankings. But only a few algorithms are proposed to learn MNL models from the most general partial rankings. Moreover, existing methods that can process general partial rankings all suffer from the intractable time complexity. Liu et al. [24] proposed an algorithm that samples full rankings from partial rankings to learn the MNL model, but this method is not scalable for large dataset due to its exponential time complexity with respect to the number of items. The algorithm by Khetan and Oh [21] breaks the general partial rankings into maximal-ordered partitions but has to discard a part of hard-to-evaluate data to confine the time complexity. Though the method proposed by Ma et al. [28] is able to learn MNL models in polynomial time, it can handle partitioned preferences at best, which is still not the most general case. Our proposed method combines the algorithms of Khetan and Oh [21]

and Ma et al. [28] to tackle this problem using an efficient numerical integral estimation that can be evaluated in polynomial time.

2.2 Network Formation Modeling with Discrete Choice Models

Modeling the formation and growth of network is essential to explore the structure of networks [8, 13, 16]. Overgoor et al. [39] recently proposed a framework to model the growth of network as discrete choice of incoming and existing nodes. This framework is general enough to include many existing growth patterns, e.g. preferential attachment. Inspired by this framework, a vast number of researches utilize the temporal information of social networks to better understand the dynamics of the networks [23, 43, 48, 51, 53]. Many researchers apply this framework in various domains related to networks such as sociology, physics, medical science and etc [9, 14, 18, 19, 23, 29, 38, 44, 50]. Moreover, many extensions have been made from this framework [3, 11, 40, 48].

In complement of the existing studies, this work applies our proposed MNL learning methods to handle the situations where temporal information of the edge formation is not fully available or where the temporal information does not directly translate to the preference of nodes. In such situations, learning from partial rankings naturally arises.

3 FAST LEARNING OF MNL FROM GENERAL PARTIAL RANKINGS

In this section, we investigate how to efficiently learn an MNL model from general partial rankings. We develop a scalable method by combining the ideas from two recent studies [21, 28]. We also extend the proposed method to efficiently learn a mixture of MNL model with an EM algorithm.

3.1 Problem Formulation and Notations

We start by introducing the formal definition of the MNL model, and the intractability problem of estimating the MNL likelihood of general partial rankings.

The MNL model. An MNL model is a popular discrete choice model that characterizes how one makes choices from a group of items. In particular, we assume each individual has an underlying utility score for each of the item, and the observed rankings of the items given by the individuals are noisy version of the utility score order. Now suppose there are N different items and denote the set of items, $\{1, \ldots, N\}$, by $\lfloor N \rfloor$. Also denote the set of all possible permutations of $\lfloor N \rfloor$ as $2^{\lfloor N \rfloor}$. The MNL model is defined below.

DEFINITION 1 (MULTINOMIAL LOGIT (MNL) MODEL). *For an individual with $\boldsymbol{w} = [w_1, \ldots, w_N]^T$ as the underlying utility scores of the N items, under an* MNL *model, the probability of observing a certain ranking of these items, $(i_1, i_2, \ldots, i_N) \in 2^{\lfloor N \rfloor}$, is defined as*

$$p\left((i_1, i_2, \ldots, i_N); \boldsymbol{w}\right) = \prod_{j=1}^{N} \frac{\exp\left(w_{i_j}\right)}{\sum_{l=j}^{N} \exp\left(w_{i_l}\right)}. \quad (1)$$

Partial rankings. In practice, we often do not observe clear full rankings of the N items. For example, we may know someone's favorite top 5 movies or friends, but rarely their full preferences over
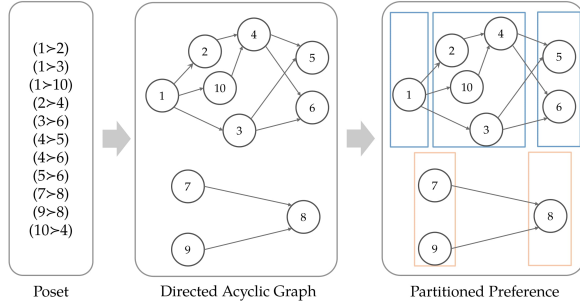
Figure 1: An illustrative example of the conversion from a poset to a DAG, and the extraction of the Partitioned-Preference rankings from a DAG.

all the movies or all the friends. Mathematically, we can represent such partially observed rankings as partially ordered sets (posets). For example, a poset, $\{(3 > 2), (3 > 5), (4 > 1)\}$, which is defined on the set of items $\{1, 2, 3, 4, 5\}$ and indicates that the item 3 is preferred over the item 2 and the item 5; the item 4 is ranked higher than the item 1; but the relative ranking between, e.g., the item 3 and the item 1 is unknown.

There are two well-known properties of posets, which we will utilize in this paper. First, assuming that there is no tied ranking, each poset corresponds to a *directed acyclic graph* (DAG). See Figure 1 as an example. Second, we can define *extensions* and *linear extensions* of a poset, where the latter can be used to characterize the full rankings that are consistent with a partial ranking.

DEFINITION 2 (EXTENSION OF POSET). *A poset $\mathcal{G}_2$ is an* extension *of another poset $\mathcal{G}_1$ if both posets are defined on the same set of items X, and for all elements $x, y \in X$, if $(x > y) \in \mathcal{G}_1$, then $(x > y) \in \mathcal{G}_2$. Moreover, we call $\mathcal{G}_2$ a* linear extension *of $\mathcal{G}_1$ if $\mathcal{G}_2$ is additionally a full order of X. And we denote the set of all linear extensions of $\mathcal{G}_1$ on X as $\Omega(\mathcal{G}_1; X)$.*

Intractability of the MNL likelihood on partial rankings. Under an MNL model parameterized by utility scores $\boldsymbol{w}$, the probability of observing a partial ranking $\mathcal{G}$ is the total probability of full rankings that are consistent with the observed $\mathcal{G}$, which is given by

$$P(\mathcal{G}; \boldsymbol{w}) = \sum_{(i_1, \cdots, i_N) \in \Omega(\mathcal{G}; \lfloor N \rfloor)} \prod_{l=1}^{N} \frac{\exp(w_{i_l})}{\sum_{r=l}^{N} \exp(w_{i_r})}. \quad (2)$$

For general partial rankings, the number of linear extensions usually grows exponentially with the number of items N. Therefore, the summation over $\Omega(\mathcal{G}; \lfloor N \rfloor)$ in Eq. (2) makes the likelihood intractable to calculate. Furthermore, except for a few special types of partial rankings, there is no easy way to simplify Eq. (2) into a tractable closed-form formula.

Partitioned-Preference rankings. In Section 3.3, we propose a polynomial-time algorithm that is able to efficiently approximate the MNL likelihood of partial rankings for large N. The proposed method is built on top of Ma et al. [28], which proposed a numerical approach for fast calculation of the MNL likelihood of a special

type of partial rankings, *Partitioned-Preference rankings* [22, 26, 28], as formally defined below.

DEFINITION 3 (PARTITIONED-PREFERENCE RANKINGS). *Given a subset $S \subset \lfloor N \rfloor$, an ordered list of M disjoint partitions of S, $(S_1, S_2, \ldots, S_M)$ is called a* Partitioned-Preference ranking *on S if (a) $\cup_{m=1}^{M} = S$ and $S_m \cap S_{m'} = \emptyset$ for any $m \neq m'$; (b) $S_1 > S_2 > \cdots > S_M$, where $S_m > S_{m'}$ indicates that any item in the partition S_m has a higher rank than items in the partition $S_{m'}$; (c) the relative ranking of items within the same partition is unknown.*

It is easy to verify that Partitioned-Preference rankings are a strict subset of partial rankings through a counter example below.

LEMMA 1. *The partial ranking $\{(3 > 2), (3 > 5), (4 > 1)\}$ is not a Partitioned-Preference ranking on $\{1, 2, 3, 4, 5\}$.*

PROOF. See Appendix A.1. □

3.2 Preliminary: A Numerical Approach for Partitioned-Preference Rankings

As a preliminary, we briefly review the numerical approach proposed by Ma et al. [28]. At the core of this approach is an alternative formulation of the MNL likelihood of a Partitioned-Preference ranking. It is shown that when the partial ranking takes the form of $S_1 > \cdots S_M$, the likelihood in Eq. (2) can be rewritten as

$$P(S_1 > \cdots > S_M; \boldsymbol{w}) = \prod_{m=1}^{M-1} \int_{u=0}^{1} \prod_{i \in S_m} \left(1 - u^{\exp(w_i - w_{R_{m+1}})}\right) du, \quad (3)$$

where $R_{m+1} = \cup_{l=m+1}^{M} S_l$ and $w_{R_{m+1}} = \log \sum_{j \in R_{m+1}} \exp(w_j)$.

Compared to Eq. (2), Eq. (3) does not involve the intractable summation over all the linear extensions. While the exact calculation of Eq. (3) is still not tractable due to the integrals involved in the formula, these integrals are one-dimensional and can be efficiently approximated by numerical integration. Furthermore, the gradients of the likelihood with respect to the utility score $\boldsymbol{w}$ can also be approximated by numerical integration. It is also shown that, to achieve any given numerical precision ε for the likelihood and the gradients, the time complexity of the numerical approach is at most $O(N + \frac{1}{\varepsilon} \left| \cup_{m=1}^{M-1} S_m \right|^3)$, which is much more efficient than the exponential complexity if using Eq. (2).

3.3 A Polynomial-Time Algorithm for MNL Likelihood of General Partial Rankings

As we have seen at the end of Section 3.1, Partitioned-Preference rankings are a strict subset of partial rankings. Therefore, the numerical approach using Eq. (3) cannot be directly applied to approximate the MNL likelihood of more general partial rankings. However, inspired by Ma et al. [28], we can utilize the idea of *Generalized Rank Breaking* (GRB) [21] to extend the numerical approach to derive a polynomial-time algorithm approximating the MNL likelihood of general partial rankings.

Specifically, Khetan and Oh [21] demonstrate that, for each general partial ranking, extracting maximal-ordered partitions (which correspond to Partitioned-Pereference rankings) of the corresponding DAG and calculating likelihood of the extracted Partitioned-Preference rankings can be a good proxy of the original likelihood.

Therefore, a natural idea extending the numerical approach [28] to general partial rankings is to first extract (multiple) Partitioned-Preference rankings from the corresponding DAG, and then use the numerical approach to estimate the likelihood of each Partitioned-Preference ranking.

However, a remaining question is how to calculate the joint likelihood of *multiple* Partitioned-Preference rankings. Fortunately, we show in the following Proposition 1 that the joint likelihood of multiple Partitioned-Preference rankings can be simply decomposed as the product of individual Partitioned-Preference ranking's.

PROPOSITION 1. *For two sets of Partitioned-Preference rankings* $S_1 > \cdots > S_M$ *and* $T_1 > \cdots > T_{M'}$, *where* $\cup_{m=1}^{M} S_m = S$ *and* $\cup_{m=1}^{M'} = T$, *and* $S, T \subseteq \lfloor N \rfloor$. *To ease the notation, we use* $\mathcal{G}_S$ *to denote the partial ranking* $S_1 > \cdots > S_M$ *and similarly define* $\mathcal{G}_T$. *If* $S \cap T = \emptyset$, *then*

$$P(\mathcal{G}_S, \mathcal{G}_T; \boldsymbol{w}) = P(\mathcal{G}_S; \boldsymbol{w})P(\mathcal{G}_T; \boldsymbol{w}). \quad (4)$$

PROOF. See Appendix A.2. □

We summarize the proposed method for efficiently approximating the MNL likelihood of a general partial ranking $\mathcal{G}$ in Algorithm 1, which we call it **Numerical GRB** (NumGRB for short).

Algorithm 1: Numerical GRB (NumGRB)

input : A partial ranking $\mathcal{G}$.
Output : Approximate log-likelihood $l(\mathcal{G})$ for $\mathcal{G}$.

1 $l(\mathcal{G}) \leftarrow 0$;
2 $C \leftarrow$ Strongly connected components of $\mathcal{G}$;
3 **for** $\mathcal{G}' \in C$ **do**
4     $S \leftarrow [\,]$;
5   **while** $|Vertices(\mathcal{G}')| > 0$ **do**
6        $L \leftarrow$ Lowest common ancestors of all sink nodes of $\mathcal{G}'$;
7        $S \leftarrow [Vertices(\mathcal{G}') \backslash L,] + S$;
8        $\mathcal{G}' \leftarrow$ subgraph of $\mathcal{G}'$ induced by L;
9     $l(\mathcal{G}) \leftarrow l(\mathcal{G}) + P(S; \boldsymbol{w})$ where $P(S; \boldsymbol{w})$ is computed by Eq. (3);
10 **return** $l(\mathcal{G})$

Model learning. For any model parameterizing the utility scores $\boldsymbol{w}$, the model parameters can be learned by minimizing the negative log-likelihood through gradient descent methods. The gradients of the likelihood for each Partitioned-Preference ranking can also be efficiently calculated through numerical integral. In the rest of this paper, we also call this model learning method as NumGRB.

Time complexity of Algorithm 1. The major sub-procedures of Algorithm 1 are (1) finding the strongly connected components (SCCs) of the DAG, (2) extracting the maximal-ordered partition for each SCC by recursively finding the common ancestor algorithm, and (3) calculating the log-likelihood (and its gradients) of each Partitioned-Preference ranking through the numerical approach. Specifically, finding the SCCs have time complexity $O(N+E)$, where E is the number of edges of the DAG. Extracting the Partitioned-Preference rankings overall takes a time complexity of $O(N^{3.6})$ [21]

(as common ancestors of all the sink nodes of a DAG can be found in time complexity $O(N^{2.6})$ [5]). Finally, calculating the log-likelihood (and the gradients) using the numerical approach for each SCC has time complexity at most $O(N^3)$. Therefore, the overall time complexity for Algorithm 1 is polynomial-time[2].

3.4 Learning Mixture of MNL with EM

We further extend the proposed method to learn mixture of MNL models, which often leads to better fitness of data [55] and is of particular interest for the application to network formation modeling [39, 40]. Formally, a mixture of MNL model is defined below.

DEFINITION 4 (MIXTURE OF MNL). *Given an integer* $k \geq 1$, *a mixture of MNL model contains two parts of parameters:* $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_k)$, *where* $\pi_r \geq 0$ *for* $1 \leq r \leq k$, *and* $\sum_{1 \leq r \leq k} \pi_r = 1$; $(\boldsymbol{w}^{(1)}, \ldots, \boldsymbol{w}^{(k)})$, *where* $\boldsymbol{w}^{(r)} \in \mathbb{R}^N$ *is the parameter of the r-th MNL component. The probability of observing a partial ranking* $\mathcal{G}$ *is defined as*

$$P\left(\mathcal{G}; \boldsymbol{\pi}, (\boldsymbol{w}^{(1)}, \ldots, \boldsymbol{w}^{(k)})\right) = \sum_{1 \leq r \leq k} \pi_r P(\mathcal{G}; \boldsymbol{w}^{(r)}). \quad (5)$$

We propose to learn the mixture of MNL with an expectation-maximization (EM) algorithm, where the likelihood of each MNL component is estimated by Algorithm 1. In practice, however, standard EM algorithm with random initialization often converges to bad local optima, which is also observed in the learning of other types of mixture models [17]. As a remedy, we further propose a clustering-based initialization method that greatly stabilizes the convergence of EM.

Clustering-based initialization. It is common to initialize an EM algorithm with clustering methods such as K-means [32]. In our case, however, applying K-means requires us to measure the distance between partial rankings, which is rarely studied. Following the idea that the same item is likely to have similar relative ranks in two different rankings if the two ranking are generated from the same MNL component, we propose a novel ranking distance to assist the clustering of ranking data. Formally, we define the relative rank of an item in a Partitioned-Preference ranking below.

DEFINITION 5 (RELATIVE RANK). *Consider a Partitioned-Preference ranking* $\mathcal{G}_S = (S_1 > \cdots > S_M)$, *we define the relative rank* $r_{\mathcal{G}_S}(i)$ *as*

$$r_{\mathcal{G}_S}(i) = \frac{m(i) - 1}{M - 1} \quad (6)$$

where $1 \leq m(i) \leq M$ *and the* $m(i)$*-th partition* $S_{m(i)}$ *contains item* i.

Intuitively, $r_{\mathcal{G}_S}(i)$ measures the relative rank of an item by the rank of its partition. Then we can define the ranking distance.

DEFINITION 6 (RANKING DISTANCE). *Consider two Partitioned-Preference rankings* $\mathcal{G}_S = (S_1 > \cdots > S_M)$ *with* $S = \cup_{m=1}^{M} S_m$, *and* $\mathcal{G}_T = (T_1 > \cdots > T_{M'})$ *with* $T = \cup_{m'=1}^{M'} T_{m'}$, *we define the* ranking distance d *between* $\mathcal{G}_S$ *and* $\mathcal{G}_T$ *as*

$$d(\mathcal{G}_S, \mathcal{G}_T) = \sqrt{\frac{\sum_{i \in S \cap T} \left(r_{\mathcal{G}_S}(i) - r_{\mathcal{G}_T}(i)\right)^2}{|S \cap T|}}.$$

---

[2]In practice, we find it is usually significantly faster than this worst-case bound.

With the ranking distance defined in Eq. (7), we can now use K-means to cluster all the observed partial rankings based on the extracted Partitioned-Preference rankings. To initialize the EM algorithm, we learn a single MNL model on each cluster and then initialize an MNL component with the learned parameters. The full EM algorithm (named **NumGRB-EM**) is summarized in Algorithm 2.

Algorithm 2: NumGRB-EM

---

input : A list of n observed partial rankings
$G = (G_1, \ldots, G_n)$, number of components k,
number of iterations B.

Output : Parameters of the mixture of MNL:
$\pi, (w^{(1)}, \ldots, w^{(k)})$.

1 Apply clustering-based initialization to $w^{(1)}, \ldots, w^{(k)}$;

2 Initialize $\pi_1 = \cdots = \pi_k = 1/k$;

3 **for** $b = 1, \ldots, B$ **do**

4    $\quad \forall 1 \leq j \leq n, 1 \leq r \leq k$, compute $P(G_j; w^{(r)})$ by
   Algorithm 1;

5    $\quad \forall 1 \leq j \leq n, 1 \leq r \leq k$, compute γ_{jr} by

$$\gamma_{jr} \leftarrow \frac{P(G_j; w^{(r)})\pi_r}{\sum_{s=1}^{k} P(G_j; w^{(s)})\pi_s}$$

$\quad \forall 1 \leq r \leq k$, update π_r by

$$\pi_r \leftarrow \frac{\sum_j \gamma_{jr}}{n}$$

$\quad$ Update $w^{(r)}$ for each r, weighing each data point G_j
   with its class responsibility γ_{jr}, i.e.,

$$w^{(r)} \leftarrow \arg\max_{w} \sum_{j=1}^{n} \gamma_{jr} \log P(G_j; w)$$

6 **return** $\pi, w^{(1)}, \ldots, w^{(k)}$.

---

4 SIMULATION STUDY

In this section, we empirically verify the estimation accuracy and computation cost of the proposed methods through simulation.

4.1 Experiment Setup

We compare the proposed methods, **NumGRB** for single MNL and **NumGRB-EM** for mixture of MNL, with a state-of-the-art baseline **ELSR-Gibbs** [24], which is a sampling-based method that can be applied to learn both single and mixture of MNL.

We closely follow the simulation setup by Liu et al. [24] for our experiments. We conduct experiments on synthetic data generated by both single and mixture of MNL.

We treat the the utility scores w of the (mixture of) MNL as free parameters, and respectively apply the proposed methods and ELSR-Gibbs to learn w. We implement the proposed methods with PyTorch [41] and use AdaGrad optimizer [6] with an initial learning rate of 0.5. For the baseline ELSR-Gibbs, we use the official implementation[3] released by the authors of Liu et al. [24]. We run

---

[3]https://github.com/zhaozb08/MixPL-SPO

our experiments on a GeForce RTX 2080 Ti GPU and an Intel(R) Xeon(R) Gold 6230 CPU @ 2.10GHz CPU.

We evaluate the estimation accuracy of each method by the mean squared error (MSE) between the learned parameters and ground truth parameters, both normalized with a softmax function. For the mixture of MNL, the average MSE on all components is reported. We also report the running time of each method to evaluate the computation cost.

4.2 Experiments on Single MNL

Synthetic data generated by single MNL. Given the number of items N, we first generate the ground truth utility scores $w = (w_1, \ldots, w_N)$ uniformly on $[-2, 2]$. Then we draw n samples of full rankings from an MNL model parameterized by w following Eq. (1). To obtain partial rankings, we sample from all the $\frac{N(N-1)}{2}$ pairwise comparisons determined by the full rankings, and keep each pairwise comparison with probability p independently.
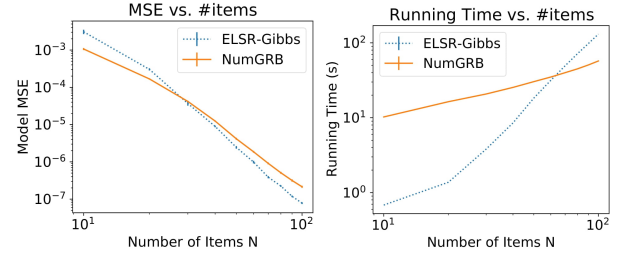

Figure 2: MSE and running time of NumGRB and ELSR-Gibbs with varying number of items N. The sample size n is fixed as 5,000. The sample rate p is fixed as 0.25. Both axes of the plots are in the logarithmic scale with base 10. The results are averaged over 50 different random seeds and error bars (barely visible) indicate the standard error of the mean.

Results with varying number of items N. Figure 2 shows the MSE and running time of the proposed NumGRB and the baseline ELSR-Gibbs with N varying from 10 to 100.

In terms of MSE, the proposed NumGRB achieves similar estimation accuracy on most configurations of N. The trend that MSE is decreasing with N is due to the artifact that the softmax scores of parameters sum to 1 and larger N implies smaller normalized scores.

In terms of running time, while the proposed NumGRB has a heavier overhead when N is small, the computational cost of ELSR-Gibbs quickly increases and surpasses NumGRB when N becomes larger. It is worth noting that the figure is a log-log plot. The linear curve of NumGRB indicates that it has a polynomial-time complexity, while the curve of ELSR-Gibbs apparently grows faster than linear, as ELSR-Gibbs does not have a polynomial-time guarantee. We are not able to provide experiment comparison for N significantly larger than 100 because the baseline ELSR-Gibbs suffers from numerical errors for large N. In contrast, the proposed NumGRB is still numerically stable for at least $N = 1000$.

Results with varying number of samples n. Figure 3 shows the MSE and running time of the proposed NumGRB and the baseline ELSR-Gibbs with n varying from 1,000 to 10,000. In terms of MSE,
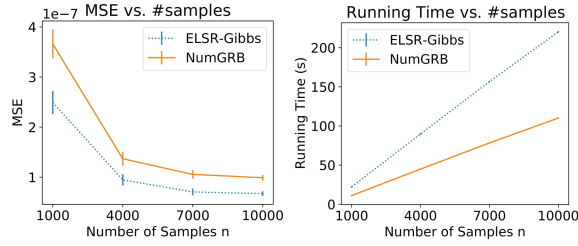
Figure 3: MSE and running time of NumGRB and ELSR-Gibbs with varying number of samples n. The number of items N is fixed as 60. The sample rate p is fixed as 0.5. Note both axes of the plots are in linear scale. The results are averaged over 10 different random seeds and error bars indicate the standard error of the mean.
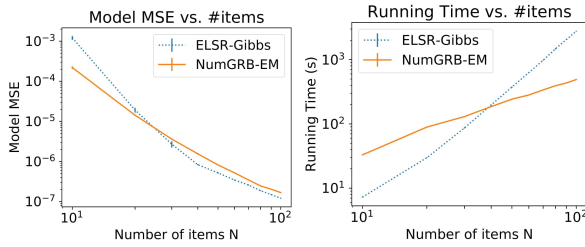

Figure 4: MSE (averaged over three MNL components) and running time of NumGRB-EM and ELSR-Gibbs with varying number of items N. The sample size n is fixed as 5,000. The sample rate p is fixed as 0.5. Both axes are in the logarithmic scale with base 10. The results are averaged over 50 different random seeds with the the worst 10 results discarded.

while ELSR-Gibbs appears to have better sample efficiency, the difference is not very large (note the y-axis is now in linear scale to show the difference). In terms of running time, both methods appear to increase linearly with n, but the computation cost of ELSR-Gibbs increases faster than the proposed NumGRB.

4.3 Experiments on Mixture of MNL

Synthetic data generated by mixture of MNL. Following Liu et al. [24], we generate synthetic data from a 3-mixture of MNL model. We assign equal weights to the 3 mixture components, i.e., $\pi_r = 1/3$ for $r = 1, 2, 3$. The configuration for each MNL component is the same as the single MNL described in Section 4.2. When drawing each of the n samples of full rankings, we first randomly select a mixture component according to π, then draw a full ranking from this MNL component. After we get the n full rankings, we sample partial rankings similarly as for the single MNL.

Results. Figure 4 shows the MSE and running time of the proposed NumGRB-EM and the baseline ELSR-Gibbs with N varying from 10 to 100. We first note that, as the negative log-likelihood of the mixture model is non-convex, both the proposed NumGRB-EM and the baseline ELSR-Gibbs occasionally converge to bad local optima where some of the mixture components are completely missed by the model. In such a case, the performance is of orders of magnitude worse compared to the performance when all 3 MNL components

are recovered. Such bad local optima dominate the average MSE over different random trials, making the comparison of average MSE meaningless. For a fair and meaningful comparison, we discard the worst 10 trials among the 50 random trials for each method with each N, in order to make sure trials that do not recover all three components are removed. We also summarize the number of trials that converge to bad local optima for each method. For the baseline ELSR-Gibbs, 82 out of the 500 training (50 random trials on 10 different setting of N) fail to recover all three components. For the proposed NumGRB-EM, only 46 out of the 500 training fail to recover all three components.

In terms of the average MSE, we again observe that the proposed NumGRB-EM achieves similar estimation accuracy as the baseline ELSR-Gibbs on most configurations of N. In terms of the running time, the proposed NumGRB-EM obtains even larger advantage over ELSR-Gibbs, compared to the experiments on single MNL.

5 APPLICATION TO NETWORK FORMATION MODELING

We further apply the proposed methods to network formation modeling on both synthetic and real network data.

5.1 Network Formation as Discrete Choice

We first review the basic discrete-choice-based network formation modeling framework proposed by Overgoor et al. [39]. The central idea is to view the formation of a directed edge from node i to node j as the process that i chooses j over a set of alternative nodes, which can then be modeled by a single MNL or a mixture of MNL. One merit of this framework is that it elegantly unifies many existing network formation models, such as preferential attachment [2], uniform attachment [4], and latent space model [12]. More importantly, this framework allows one to combine node features with different network formation mechanisms into a single model and estimate their relative importance.

Next, we review two concrete examples, preferential attachment model and uniform attachment model.

DEFINITION 7 (PREFERENTIAL ATTACHMENT). *If the formation of new edges in a directed graph follows* preferential attachment*, the new arriving node connects to an existing node j with a probability proportional to a power of their degree d_j, i.e.,*

$$P(j; V) = \frac{d_j^\alpha}{\sum_{l \in V} d_l^\alpha} = \frac{\exp(\alpha \log d_j)}{\sum_{l \in V} \exp(\alpha \log d_l)}, \qquad (7)$$

where V denotes the candidate set.

DEFINITION 8 (UNIFORM ATTACHMENT). *If the formation of new edges in a directed graph follows uniform attachment, the new arriving node connects to any existing node j in the candidate set V with the same probability, i.e.,*

$$P(j; V) = \frac{1}{|V|} = \frac{\exp(1)}{\sum_{l \in V} \exp(1)}. \qquad (8)$$

As can be seen in their definitions, the preferential attachment and uniform attachment models can be viewed as MNL models with utility score for j being $\alpha \log d_j$ and 1 respectively.

More generally, under the preferential attachment model, the probability for a node sequentially choosing $(j_1, j_2, \ldots, j_K)$ to connect can be written as

$$P((j_1, j_2, \ldots, j_K); V) = \prod_{k=1}^{K} \frac{\exp(\alpha \log d_{j_k})}{\sum_{l \in V \setminus \{j_1, \ldots, j_{k-1}\}} \exp(\alpha \log d_l)}.$$

However, one limitation of most existing methods under this framework [39, 40] is that learning the model from an observed network requires the knowledge of full rankings for everyone's all existing edges, which is unrealistic in many real-world scenarios. The proposed methods in this work, NumGRB and NumGRB-EM, can be applied to learn from more general partial rankings of existing edges in a network.

5.2 Experiments on Synthetic Network Data

We first apply the proposed NumGRB-EM on synthetic network data generated by mixture of 4 variants of network formation models. Through our experiments, we demonstrate that (1) the proposed principled method for learning from partial rankings outperforms naive approximation methods; (2) correctly specifying the mixture components not only affects the learning of mixture weights but also improves the model parameter estimation of each mixture component.

Synthetic network data generation. We simulate the growth of a directed network with a synthetic (r, p)-model following Overgoor et al. [39]. When a new edge is formed, with probability p, it is formed by uniform attachment, and with probability $1 - p$, it is formed by preferential attachment with $\alpha = 1$. After choosing the attachment pattern, we choose the candidate set V to fully determine the mixture component: with probability r, the candidate set is all nodes in the network that have not been connected by the source node, while with probability $1 - r$, the candidate set is restricted to the friends-of-friends (FoF) of the source node. Therefore, the synthetic data is generated by a mixture of 4 edge-choice distributions, which we denote as UA (uniform attachment, mixture weight pr), PA (preferential attachment, mixture weight $(1 - p)r$), UA-FoF (uniform attachment restricted to FoF, mixture weight $p(1 - r)$), PA-FoF (preferential attachment restricted to FoF, mixture weight $(1 - p)(1 - r)$).

We generate synthetic network data with varying values of (r, p) pairs. Due to the space limit, the detailed data generation procedure is provided in Appendix A.3.

Experiment setup. The learning task of this experiment is to learn 5 parameters of the mixture model: the mixture weights for the 4 components as well as a shared model parameter α for PA and PA-FoF. And we apply the proposed NumGRB-EM method to learn the parameters from the data.

For the baseline, we note that ELSR-Gibbs is not scalable enough for this experiment setup. We instead implement a naive baseline method (named as **Naive**) that treats each new edge formation as an independent top-one ranking, such that the calculation of likelihood becomes feasible (but with information lost). For both NumGRB-EM and Naive, we further apply them to learn a 2-mixture (PA and PA-FoF) and a single model (PA only), in order to investigate their performance under mis-specified settings.

Para. (True Val.)	NumGRB-EM			Naive		
	4-Mix	2-Mix	Single	4-Mix	2-Mix	Single
r=0.2 p=0.2 UA (0.04)	0.055 (0.004)			0.127 (0.018)		
UA-FoF (0.16)	0.179 (0.024)			0.317 (0.018)		
PA (0.16)	0.174 (0.007)	0.239 (0.009)	1.000 (0.000)	0.091 (0.013)	0.213 (0.002)	1.000 (0.000)
PA-FoF (0.64)	0.593 (0.025)	0.761 (0.009)		0.466 (0.023)	0.787 (0.002)	
α (1)	0.989 (0.019)	0.877 (0.008)	0.814 (0.006)	1.518 (0.098)	0.945 (0.021)	0.829 (0.006)
r=0.5 p=0.5 UA (0.25)	0.262 (0.008)			0.256 (0.013)		
UA-FoF (0.25)	0.249 (0.013)			0.252 (0.010)		
PA (0.25)	0.249 (0.008)	0.502 (0.004)	1.000 (0.000)	0.247 (0.015)	0.516 (0.014)	1.000 (0.000)
PA-FoF (0.25)	0.240 (0.017)	0.498 (0.004)		0.246 (0.005)	0.484 (0.014)	
α (1)	0.949 (0.028)	0.557 (0.033)	0.632 (0.011)	1.199 (0.042)	0.589 (0.021)	0.644 (0.009)
r=0.8 p=0.8 UA (0.64)	0.521 (0.052)			0.476 (0.037)		
UA-FoF (0.16)	0.128 (0.002)			0.106 (0.009)		
PA (0.16)	0.308 (0.052)	0.817 (0.005)	1.000 (0.000)	0.344 (0.039)	0.792 (0.005)	1.000 (0.000)
PA-FoF (0.04)	0.043 (0.008)	0.183 (0.005)		0.074 (0.006)	0.208 (0.005)	
α (1)	0.742 (0.124)	0.234 (0.023)	0.353 (0.006)	0.773 (0.099)	0.275 (0.014)	0.323 (0.005)
r=1 p=0 UA (0)	0.020 (0.010)			0.050 (0.009)		
UA-FoF (0)	0.000 (0.000)			0.000 (0.000)		
PA (1)	0.980 (0.010)	1.000 (0.000)	1.000 (0.000)	0.950 (0.009)	1.000 (0.000)	1.000 (0.000)
PA-FoF (0)	0.000 (0.000)	0.000 (0.000)		0.000 (0.000)	0.000 (0.000)	
α (1)	0.994 (0.016)	0.993 (0.014)	0.998 (0.012)	1.045 (0.008)	0.996 (0.013)	0.989 (0.006)

Table 1: Estimated model parameters by NumGRB-EM and Naive on the synthetic network data. The results are averaged over 5 random trials and the numbers in brackets indicate the standard error of the mean. The synthetic network data are always generated by 4-mixtures of UA, UA-FoF, PA, and PA-FoF with ground truth $\alpha = 1$. We generate data with 4 settings for different values of r and p. The columns with heads "4-Mix", "2-Mix", and "Single" refer to the learned models being specified with 4-mixtures, 2-mixtures (PA and PA-FoF), and a single model (PA), respectively. The estimated parameters include the mixture weights for the four components, as well as the parameter α shared by PA and PA-FoF. There are blank entries for "2-Mix" and "Single" as they are not specified.

Results. The experiment results are summarized in Table 1. Comparing NumGRB-EM and Naive, the proposed NumGRB significantly outperforms Naive in terms of both the estimation of mixture weights and the distribution parameter α in most cases. This verifies that a principled method for calculating the likelihood of partial rankings is critical for the model parameter estimation accuracy.

Regarding the specification of the mixture models, we start with the setting $r = 1$ and $p = 0$, which is a special case where the ground truth model is actually a single model following PA. It is not surprising that all models fit well in this case because they

all contain the PA component. It is worth noting that applying NumGRB-EM to learn 4-mixtures or 2-mixtures is able to faithfully recover the single model case. In more general settings where $0 < r, p < 1$, we can see that, the 4-mixture models trained by the proposed NumGRB-EM method outperform their mis-specified counterparts not only in terms of the mixture weights, but also in terms of the distribution parameter α.

Together, the experiments suggest that both the proposed partial ranking likelihood estimation and the extension to mixture models are helpful for learning choice-based network formation models.

5.3 Experiments on Real-World Network Data

We further apply the proposed method on two real-world network datasets, Flickr [33] and Microsoft Academic Graph[4], and compare with the method by Overgoor et al. [39]. For both datasets, we closely follow most of the data processing steps by Overgoor et al. [39]. The key difference between our method for fitting the data and Overgoor et al. [39]'s lies in how we interpret the ranking of target nodes given the observed network.

For Flickr, Overgoor et al. [39] use the temporal order of the edge formation in the network to construct the full ranking of the target nodes connected by the source node, with the underlying assumption that earlier formed edges correspond to more preferred target nodes. We think, however, this assumption might be too strong, as it might be hard to tell the relative ranking of two target nodes with edges formed within a small time window. We instead use the temporal window of the edge formation to construct Partitioned-Preference rankings.

For Microsoft Academic Graph, there is even no temporal order available for the edge formation, as the citation links from a source node (a new paper) are almost always formed simultaneously (at its publication), and therefore the ranking of target nodes (exiting papers) is naturally a Partitioned-Preference ranking with 2 partitions (cited or not cited). Overgoor et al. [39] fit the data with the naive method as we described in Section 5.2, which treats each edge as an independent top-one ranking.

For the experiments, we parameterize the utility scores with linear models on node features used by Overgoor et al. [39]. We report both the learned linear coefficients and the precision@k metrics for link prediction. Due to space limit, more detailed experiment setups are provided in Appendix A.4.

Results. The experiment results on Flickr and Microsoft Academic Graph are provided in Table 2 and Table 3 respectively.

In terms of the precision@k metrics for link prediction, we observe that treating the rankings of target nodes as Partitioned-Preference rankings and train the models with the proposed Num-GRB outperforms the method by Overgoor et al. [39] when there are more features included in the model. The difference is smaller when there are very few features, which is not surprising as the model is not able to learn sophisticated patterns in these cases.

In terms of the estimated linear coefficients, qualitatively the explanations by models trained with both methods have similar trend. Quantitatively, however, some of the estimated coefficients by the two methods differ by a large amount. For example, comparing Model #3 and #3* in Table 2, the relative strengths of the

	Overgoor et al. [39]				NumGRB			
	#1	#2	#3	#4	#1*	#2*	#3*	#4*
log Followers	1.149		0.7150	0.536	0.806		0.4810	0.3470
Has Degree	−0.580		−0.631	−1.745	−3.556		−3.527	−3.335
Reciprocal	8.419	8.347	8.197	7.903	5.854	4.491	5.614	5.171
Is FoF		6.120	3.955			3.918	2.888	
2 Hops				6.290				3.778
3 Hops				2.851				1.348
4 Hops				0.583				−0.877
5 Hops				−0.585				−1.92
≥ 6 Hops				−1.122				−2.264
Precision@1	0.7606	0.7898	0.8238	0.8262	0.7622	0.7898	0.8414	**0.8448**
Precision@3	0.6215	0.6385	0.6737	0.6756	0.6227	0.6398	0.6796	**0.6846**
Precision@5	0.5444	0.5573	0.5880	0.5895	0.5464	0.5592	0.5926	**0.5948**

Table 2: Estimated model parameters and precision@k for link prediction on Flickr. Model #1-4 are trained by the baseline method by Overgoor et al. [39]. Model #1-4* are trained by the proposed NumGRB. For the precision@k metrics, the bold markers denote the best performance and the underline markers denote the better performance between the corresponding baseline and proposed methods.

	Overgoor et al. [39]				NumGRB			
	#1	#2	#3	#4	#1*	#2*	#3*	#4*
log Citations	0.717	0.794	1.052	1.044	0.649	0.611	0.932	0.922
Has Degree	1.684	1.677	1.862	1.830	0.280	0.250	1.495	1.462
Has Same Author		6.523	5.928	5.913		4.870	4.430	4.424
log Age			−1.096	−1.069			−1.458	−1.443
log Max Papers by Authors				0.029				0.030
Precision@1	0.2495	0.4745	0.4960	0.4970	0.2490	0.4835	0.4960	**0.5020**
Precision@3	0.1955	0.4128	0.4213	0.4235	0.1957	0.4132	0.4293	**0.4295**
Precision@5	0.1628	0.3558	0.3654	0.3664	0.1629	0.3545	**0.3735**	0.3731
Precision@10	0.1238	0.2609	0.2683	0.2690	0.1238	0.2587	0.2756	**0.2758**

Table 3: Estimated model parameters and precision@k for link prediction on Microsoft Academic Graph. The markers are the same as in Table 2.

coefficients between "Has Degree" and "Reciprocal" are very different. Researchers facing the results given by Model #3 may interpret that the impact of having zero followers is marginal. However, the results by Model #3* tell a largely different story. And the latter is probably a better fit of the data (as suggested by the link prediction accuracy).

6 CONCLUSION

In this paper, we propose novel methods for fast learning of the MNL model and its mixture from general partial rankings. We verify both the effectiveness and efficiency of the proposed methods through simulation studies. We also demonstrate that the proposed methods can be very useful on applications to choice-based network formation modeling, where the formation of edges is viewed as discrete choices from (mixture of) MNL models. Through experiments on both synthetic networks and real-world networks, we show that efficient learning (mixture of) MNL models from partial rankings is necessary for proper estimation of the model parameters, which are critical for downstream network analysis tasks in scientific research.

Acknowledgements

REFERENCES

[1] Hossein Azari Soufiani, William Ziwei Chen, David C Parkes, and Lirong Xia. 2013. Generalized method-of-moments for rank aggregation. (2013).

[2] Albert-László Barabási and Réka Albert. 1999. Emergence of scaling in random networks. *science* 286, 5439 (1999), 509–512.

[3] Federico Battiston, Giulia Cencetti, Iacopo Iacopini, Vito Latora, Maxime Lucas, Alice Patania, Jean-Gabriel Young, and Giovanni Petri. 2020. Networks beyond pairwise interactions: structure and dynamics. *Physics Reports* 874 (2020), 1–92.

[4] Duncan S Callaway, John E Hopcroft, Jon M Kleinberg, Mark EJ Newman, and Steven H Strogatz. 2001. Are randomly grown graphs really random? *Physical Review E* 64, 4 (2001), 041902.

[5] Artur Czumaj, Mirosław Kowaluk, and Andrzej Lingas. 2007. Faster algorithms for finding lowest common ancestors in directed acyclic graphs. *Theoretical Computer Science* 380, 1-2 (2007), 37–46.

[6] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research* 12, 7 (2011).

[7] Paul Erdos, Alfréd Rényi, et al. 1960. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci* 5, 1 (1960), 17–60.

[8] Fred Feinberg, Elizabeth Bruch, Michael Braun, Brett Hemenway Falk, Nina Fefferman, Elea McDonnell Feit, John Helveston, Daniel Larremore, Blakeley B McShane, Alice Patania, et al. 2020. Choices in networks: a research framework. *Marketing Letters* 31, 4 (2020), 349–359.

[9] Xiang Fu, Shangdi Yu, and Austin R Benson. 2020. Modelling and analysis of tagging networks in Stack Exchange communities. *Journal of Complex Networks* 8, 5 (2020), cnz045.

[10] Isobel Claire Gormley, Thomas Brendan Murphy, et al. 2008. A mixture of experts model for rank data with applications in election studies. *Annals of Applied Statistics* 2, 4 (2008), 1452–1477.

[11] Harsh Gupta and Mason A Porter. 2020. Mixed logit models and network formation. *arXiv preprint arXiv:2006.16516* (2020).

[12] Peter D Hoff, Adrian E Raftery, and Mark S Handcock. 2002. Latent space approaches to social network analysis. *Journal of the american Statistical association* 97, 460 (2002), 1090–1098.

[13] Petter Holme. 2019. Rare and everywhere: Perspectives on scale-free networks. *Nature communications* 10, 1 (2019), 1–3.

[14] Petter Holme and Juan C Rocha. 2021. Networks of climate change: Connecting causes and consequences. *arXiv preprint arXiv:2105.12537* (2021).

[15] David R Hunter and Kenneth Lange. 2004. A tutorial on MM algorithms. *The American Statistician* 58, 1 (2004), 30–37.

[16] Gerardo Iñiguez, Hang-Hyun Jo, and Kimmo Kaski. 2019. Special Issue "Computational Social Science".

[17] Chi Jin, Yuchen Zhang, Sivaraman Balakrishnan, Martin J Wainwright, and Michael I Jordan. 2016. Local maxima in the likelihood of gaussian mixture models: Structural results and algorithmic consequences. *Advances in neural information processing systems* 29 (2016), 4116–4124.

[18] Anshul Kanakia, Kuansan Wang, Yuxiao Dong, Boya Xie, Kyle Lo, Zhihong Shen, Lucy Lu Wang, Chiyuan Huang, Darrin Eide, Sebastian Kohlmeier, et al. 2020. Mitigating biases in cord-19 for analyzing covid-19 literature. *Frontiers in research metrics and analytics* 5 (2020), 14.

[19] Mari Kawakatsu, Philip S Chodrow, Nicole Eikmeier, and Daniel B Larremore. 2021. Emergence of hierarchy in networked endorsement dynamics. *Proceedings of the National Academy of Sciences* 118, 16 (2021).

[20] Ashish Khetan and Sewoong Oh. 2016. Data-driven rank breaking for efficient rank aggregation. In *International Conference on Machine Learning*. PMLR, 89–98.

[21] Ashish Khetan and Sewoong Oh. 2018. Generalized rank-breaking: computational and statistical tradeoffs. *The Journal of Machine Learning Research* 19, 1 (2018), 983–1024.

[22] Guy Lebanon and Yi Mao. 2008. Non-parametric modeling of partially ranked data. *Journal of Machine Learning Research* 9, Oct (2008), 2401–2429.

[23] Eun Lee, Aaron Clauset, and Daniel B Larremore. 2021. The Dynamics of Faculty Hiring Networks. *arXiv preprint arXiv:2105.02949* (2021).

[24] Ao Liu, Zhibing Zhao, Chao Liao, Pinyan Lu, and Lirong Xia. 2019. Learning Plackett-Luce Mixtures from Partial Preferences. *Proceedings of the AAAI Conference on Artificial Intelligence* 33, 01 (Jul. 2019), 4328–4335. https://doi.org/10.1609/aaai.v33i01.33014328

[25] Tie-Yan Liu. 2011. Learning to rank for information retrieval. (2011).

[26] Tyler Lu and Craig Boutilier. 2014. Effective sampling and learning for mallows models with pairwise-preference data. *The Journal of Machine Learning Research* 15, 1 (2014), 3783–3829.

[27] R Duncan Luce. 1959. *Individual choice behavior: A theoretical analysis*. Wiley.

[28] Jiaqi Ma, Xinyang Yi, Weijing Tang, Zhe Zhao, Lichan Hong, Ed H. Chi, and Qiaozhu Mei. 2021. Learning-to-Rank with Partitioned Preference: Fast Estimation for the Plackett-Luce Model. arXiv:2006.05067 [cs.LG]

[29] Carolina Mattsson. 2020. *Financial Transaction Networks to Describe and Model Economic Systems*. Ph. D. Dissertation. Northeastern University.

[30] Lucas Maystre and Matthias Grossglauser. 2015. *Fast and accurate inference of Plackett-Luce models*. Technical Report.

[31] Daniel McFadden. 1977. Modelling the choice of residential location. (1977).

[32] Volodymyr Melnykov and Igor Melnykov. 2012. Initializing the EM algorithm in Gaussian mixture models with an unknown number of components. *Computational Statistics & Data Analysis* 56, 6 (2012), 1381–1395.

[33] Alan Mislove, Hema Swetha Koppula, Krishna P Gummadi, Peter Druschel, and Bobby Bhattacharjee. 2008. Growth of the flickr social network. In *Proceedings of the first workshop on Online social networks*. 25–30.

[34] Cristina Mollica and Luca Tardella. 2017. Bayesian Plackett–Luce mixture models for partially ranked data. *Psychometrika* 82, 2 (2017), 442–458.

[35] Alejandro Mottini, Alix Lhéritier, Rodrigo Acuna-Agost, and Maria A Zuluaga. 2018. Understanding Customer Choices to Improve Recommendations in the Air Travel Industry.. In *RecTour@ RecSys*. 28–32.

[36] Sahand Negahban, Sewoong Oh, and Devavrat Shah. 2017. Rank centrality: Ranking from pairwise comparisons. *Operations Research* 65, 1 (2017), 266–287.

[37] Sewoong Oh and Devavrat Shah. 2014. Learning mixed multinomial logit model from ordinal data. *arXiv preprint arXiv:1411.0073* (2014).

[38] Jan Overgoor, Lada A Adamic, et al. 2020. The Structure of US College Networks on Facebook. In *Proceedings of the International AAAI Conference on Web and Social Media*, Vol. 14. 499–510.

[39] Jan Overgoor, Austin R. Benson, and Johan Ugander. 2020. Choosing to Grow a Graph: Modeling Network Formation as Discrete Choice. arXiv:1811.05008 [cs.SI]

[40] Jan Overgoor, George Pakapol Supaniratisai, and Johan Ugander. 2020. Scaling choice models of relational social data. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1990–1998.

[41] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* 32 (2019), 8026–8037.

[42] Robin L Plackett. 1975. The analysis of permutations. *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 24, 2 (1975), 193–202.

[43] Thomas Reeves, Anil Damle, and Austin R Benson. 2020. Network interpolation. *SIAM Journal on Mathematics of Data Science* 2, 2 (2020), 505–528.

[44] Nir Rosenfeld, Kojin Oshiba, and Yaron Singer. 2020. Predicting choice with set-dependent aggregation. In *International Conference on Machine Learning*. PMLR, 8220–8229.

[45] Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-June Hsu, and Kuansan Wang. 2015. An overview of microsoft academic service (mas) and applications. In *Proceedings of the 24th international conference on world wide web*. 243–246.

[46] Hossein Azari Soufiani, David Parkes, and Lirong Xia. 2014. Computing parametric ranking models via rank-breaking. In *International Conference on Machine Learning*. PMLR, 360–368.

[47] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 990–998.

[48] Kiran Tomlinson and Austin R Benson. 2020. Learning Interpretable Feature Context Effects in Discrete Choice. *arXiv preprint arXiv:2009.03417* (2020).

[49] Kenneth E Train. 2009. *Discrete choice methods with simulation*. Cambridge university press.

[50] Enrico Ubaldi, Raffaella Burioni, Vittorio Loreto, and Francesca Tria. 2021. Emergence and evolution of social networks through exploration of the Adjacent Possible space. *Communications Physics* 4, 1 (2021), 1–12.

[51] Rūta Užupytė and Ernst C Wit. 2020. Test for triadic closure and triadic protection in temporal relational event data. *Social Network Analysis and Mining* 10, 1 (2020), 1–12.

[52] John I Yellott Jr. 1977. The relationship between Luce's choice axiom, Thurstone's theory of comparative judgment, and the double exponential distribution. *Journal of Mathematical Psychology* 15, 2 (1977), 109–144.

[53] Jean-Gabriel Young, Guillaume St-Onge, Edward Laurence, Charles Murphy, Laurent Hébert-Dufresne, and Patrick Desrosiers. 2019. Phase transition in the recoverability of network history. *Physical Review X* 9, 4 (2019), 041056.

[54] Zhibing Zhao, Peter Piech, and Lirong Xia. 2016. Learning mixtures of Plackett-Luce models. (2016), 2906–2914.

[55] Zhibing Zhao, Tristan Villamil, and Lirong Xia. 2018. Learning mixtures of random utility models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

A APPENDIX

A.1 Proof of Lemma 1

Proof of Lemma 1. For the sake of contradiction, assume $\{(3 \succ 2), (3 \succ 5), (4 \succ 1)\}$ is a Partitioned-Preference ranking on $\{1, 2, 3, 4, 5\}$.

As we do not know the relative ranking between 3 and 4, so according to (a) and (b) in Definition 3, 3 and 4 must be in the same partition. Similarly, 3 and 1 must be in the same partition. However, we also know $4 \succ 1$, which contradicts with (c) in Definition 3. □

A.2 Proof of Proposition 1

We first introduce a well-known connection between the MNL model and Gumbel distribution [52]. For an MNL model parameterized by $\mathbf{w}$, let $g_{\mathbf{w}_1}, g_{\mathbf{w}_2}, \ldots, g_{\mathbf{w}_N}$ denote N independent Gumbel random variables following

$$\text{Gumbel}(\mathbf{w}_1), \text{Gumbel}(\mathbf{w}_2), \ldots, \text{Gumbel}(\mathbf{w}_N),$$

where $\text{Gumbel}(w)$ refers to the Gumbel distribution with location parameter w. Then the probability of observing a full ranking $(i_1, \ldots, i_N) \in 2^{\lfloor N \rfloor}$ under the MNL model is equal to the probability that $g_{\mathbf{w}_{i_1}} > g_{\mathbf{w}_{i_2}} > \cdots > g_{\mathbf{w}_{i_N}}$, i.e.,

$$P((i_1, \ldots, i_N); \mathbf{w}) = P(g_{\mathbf{w}_{i_1}} > g_{\mathbf{w}_{i_2}} > \cdots > g_{\mathbf{w}_{i_N}}).$$

So there is a bijective mapping between the full rankings and the orders of the Gumbel variables.

Proof of Proposition 1. Given a set of Gumbel variables,

$$g_{\mathbf{w}_1}, g_{\mathbf{w}_2}, \ldots, g_{\mathbf{w}_N},$$

for any $A \subseteq \lfloor N \rfloor$, define

$$\bar{m}(A) \triangleq \max_{i \in A} g_{\mathbf{w}_i}, \quad \underline{m}(A) \triangleq \min_{i \in A} g_{\mathbf{w}_i}.$$

Further denote $\tilde{S}_m = \cup_{l=m}^M S_l$ and $\tilde{T}_m = \cup_{l=m}^{M'} T_l$.

Based on the bijective mapping between the full rankings and the orderes of Gumbel variables, it is easy to verify that the event of $\Omega(\mathcal{G}_S; \lfloor N \rfloor)$ corresponds to the event of Gumbel variables satisfying

$$\underline{m}(S_1) > \bar{m}(\tilde{S}_2), \underline{m}(S_2) > \bar{m}(\tilde{S}_3), \ldots, \underline{m}(S_{M-1}) > \bar{m}(\tilde{S}_M).$$

While the event of $\Omega(\mathcal{G}_T; \lfloor N \rfloor)$ corresponds to the event of Gumbel variables satisfying

$$\underline{m}(T_1) > \bar{m}(\tilde{T}_2), \underline{m}(T_2) > \bar{m}(\tilde{T}_3), \ldots, \underline{m}(T_{M'-1}) > \bar{m}(\tilde{T}_{M'}).$$

Then we have

$$P(\mathcal{G}_S, \mathcal{G}_T; \mathbf{w})$$
$$= P(\underline{m}(S_1) > \bar{m}(\tilde{S}_2), \ldots, \underline{m}(S_{M-1}) > \bar{m}(\tilde{S}_M),$$
$$\underline{m}(T_1) > \bar{m}(\tilde{T}_2), \ldots, \underline{m}(T_{M'-1}) > \bar{m}(\tilde{T}_{M'})).$$

Further, since $S \cap T = \emptyset$, and the Gumbel variables are mutually independent, we have

$$P(\underline{m}(S_1) > \bar{m}(\tilde{S}_2), \ldots, \underline{m}(S_{M-1}) > \bar{m}(\tilde{S}_M),$$
$$\underline{m}(T_1) > \bar{m}(\tilde{T}_2), \ldots, \underline{m}(T_{M'-1}) > \bar{m}(\tilde{T}_{M'}))$$
$$= P\left(\underline{m}(S_1) > \bar{m}(\tilde{S}_2), \ldots, \underline{m}(S_{M-1}) > \bar{m}(\tilde{S}_M)\right)$$
$$\cdot P\left(\underline{m}(T_1) > \bar{m}(\tilde{T}_2), \ldots, \underline{m}(T_{M'-1}) > \bar{m}(\tilde{T}_{M'})\right)$$
$$= P(\mathcal{G}_S; \mathbf{w})P(\mathcal{G}_T; \mathbf{w}).$$

□

A.3 Detailed Synthetic Network Generation Procedure

Given a pair of (r, p), we generate the network with the following procedure. We first generate an initial network using Erdős-Rényi random graph [7] with 1000 nodes and probability 0.005, and we also randomly choose 20 nodes and increase their edges by 50 to 80 to increase the variety of node degrees. We then randomly select half of the nodes in the network as source nodes to form new edges. For each source node, we sample 1 out of the 4 mixture components, UA, PA, UA-FoF, and PA-FoF as its edge-choice distribution, and make each node form 5 new edges[5]. Throughout the new edge formation process, we use the node degrees in the initial network when calculating the choice distribution. We treat the new edge formation for each source node as a Partitioned-Preference ranking with 2 partitions: the target nodes pointed by the new edges are in the first partition while the nodes that are not connected by the source node are in the second partition. We vary the values of r and p to get multiple settings of synthetic data. For each setting, we repeat the data generation with 5 random seeds.

A.4 Detailed Experiment Setup for Experiments on Real-World Networks

Experiment setup for Flickr. We follow the data processing steps of Overgoor et al. [39] to extract the network data. In this network, we consider 4 types features: the log of number of the followers, whether the following edge is reciprocal, whether the follower and the followed user is friends-of-friends, and the path length (hop) from the new follower to the followed user before the new edge is made. It is possible that a user, has never been followed before. Thus, we use the censored log function, i.e., define $\log(0) = 0$, and add another feature "has degree" to distinguish the zero case. We sample 50000 users in a period (20 days) that have following events, and record all the new edges formed by them. Edges formed in the same 10 days are considered in one partition. Edges formed earlier are assumed to be more preferable than later ones. We then sample 100 edges uniformly at random from the dataset as the negative samples for each user. The testing set is sampled in the same way, with the sampling period later for 20 days. We train 4 linear models shown in Table. 2 with the proposed NumGRB method and evaluate them using precision@[1, 3, 5] on testing set.

Experiment setup for Microsoft Academic Graph. Microsoft Academic Graph contains the events of citations between publications. Our task is to predict the references of a new publication among a large candidate set. We continue to follow the steps of Overgoor et al. [39] to extract the citation data in the domain of climatology. We include 4 features of the candidate references: the log of number of citations at the time of citation, whether the paper shares authors with the candidate reference, the log age of the paper in years at the time of citation, and the maximum number of publications ever by any one of the authors at the time of publication. We still use the censored log function to avoid log of zero. Additionally, we introduce another feature, whether the candidate reference has degree, to distinguish if the number of citation is 0 or

---

[5]A few nodes may form less than 5 edges when their candidate set is FoF and the size of the candidate set is less than 5.

1. We sample 12,000 papers after 2010 as source nodes and record references for each paper. We sample 5000 negative samples for each sampled paper uniformly at random. The data is then split into training set and testing set by time. We train 4 linear models shown in Table. 3 using the proposed NumGRB method and report precision@[1, 3, 5, 10] on testing set.