Making Group Decisions from Natural Language-Based Preferences

Farhad Mohsin, Lei Luo, Wufei Ma, Inwon Kang
Zhibing Zhao, Ao Liu, Rohit Vaish, Lirong Xia

Abstract

Traditional approaches for group decision-making—in particular, voting—typically assume that agents explicitly submit well-defined ordinal preferences (e.g., rankings) over the alternatives. However, in real-world settings such as online discussion forums, preferences are often expressed implicitly in natural language. We propose a framework for making group decisions from natural language-based preferences. Our approach combines ideas from random utility theory and social choice by first learning a general random utility model for each agent, and subsequently using a voting rule to aggregate individual preferences to make a group decision. To circumvent the computational intractability associated with the latter step, we show that fractional voting outcomes, which are often easy to compute, are identical to those under the randomized modeling with high probability. Preliminary experiments on the efficacy of the framework were conducted on a newly-collected dataset.

1 Introduction

Suppose a family is deciding which college their child should attend after receiving ten offers. The family members may have different preferences over the colleges: for example, the child likes college A’s atmosphere; the dad prefers college B because of its low cost; and the mom prefers college C because it is close to home. How should the decision be made?

This example illustrates the problem of group decision making, which is a central topic in the field of social choice. One of the most popular approaches is voting—agents are asked to submit their preferences over the alternative to the center, who then applies a voting rule to make a group decision for the agents.

The classical paradigm of voting works well in group decision making scenarios with few alternatives and low frequency, in particular presidential elections. However, it faces two major challenges in modern, large-scale, and more-frequent group decision making scenarios: the preference bottleneck [8] and the computational bottleneck [9].

The preference bottleneck arises when the agents’ preferences are too complicated to be efficiently communicated. This often happens when the number of alternatives is large but can also be a common phenomenon when the preferences are uncertain or even unknown to the agents themselves. A typical approach to addressing preference bottleneck is by conducting preference elicitation [7, 11, 41, 24], where interactive questions are computed to efficiently elicit agents’ preferential information that are sufficient for making a group decision.

The computational bottleneck arises when it is computationally hard to compute the outcomes of voting rules based on the preferential information provided by the agents. This typically happens when the number of alternatives is larger than a few under certain voting rules such as Kemeny [12], or can happen under many voting rule when agents’ preferences are compactly represented [20]. Various algorithms and computational theories were developed to address the computational bottleneck, as summarized in the Handbook of Computational Social Choice [9].
In this paper, we aim at providing an immersive solution to the preference bottleneck and the computational bottleneck by leveraging the power of AI. Take the college-choice example for instance, we want to build an intelligent system that silently learns the family members’ preferences from their conversations in natural language (without explicitly querying them), and then compute a group decision based on the learned preferences. The key questions we want to answer in this paper is:

How can we learn agent preferences from natural language to make a group decision?

While the idea of aggregating sentiments expressed in natural language is not new [14, 15], successfully addressing the problem appears to be technically challenging to tackle, despite many previous work on sentiment analysis and stance detection [21, 26] and many voice-enabled systems, including the business group decision-making system developed by IBM [10]. In particular, we are not aware of a dataset on agents’ collective preferences (not simply their sentiment), and we are not aware of a previous technical work that explicitly combines preference learning from natural language and group decision making.

Our Contributions

Our primary conceptual contribution is a group decision-making framework illustrated in Figure 1. We focus on natural language-based preferences in this paper and leave its extensions to multi-modal inputs as future work. The framework consists of three stages. In the opinion mining stage, features that indicate preferences, such as sentiment scores, are extracted from natural language. In the preference learning stage, we use the features extracted in the first stage to train a machine learning model to predict agents’ preferences. Finally, in the preference aggregation stage, voting rules are applied to agents’ predicted preferences to make a group decision.

For experimental analysis, we created a new natural language dataset of discussions about preferences over colleges from the College Confidential forum. The discussions involve two or three alternatives (colleges) per thread. We use Amazon Mechanical Turk to crowd-label agents’ preferences in the discussions. The dataset currently has 53 different discussions of varying lengths. We apply our framework on the dataset to predict group decisions to get high training accuracy but moderate testing accuracy (53 ∼ 61% test

---

Figure 1: The proposed framework.

Our primary conceptual contribution is a group decision-making framework illustrated in Figure 1. We focus on natural language-based preferences in this paper and leave its extensions to multi-modal inputs as future work. The framework consists of three stages. In the opinion mining stage, features that indicate preferences, such as sentiment scores, are extracted from natural language. In the preference learning stage, we use the features extracted in the first stage to train a machine learning model to predict agents’ preferences. Finally, in the preference aggregation stage, voting rules are applied to agents’ predicted preferences to make a group decision.

For experimental analysis, we created a new natural language dataset of discussions about preferences over colleges from the College Confidential forum. The discussions involve two or three alternatives (colleges) per thread. We use Amazon Mechanical Turk to crowd-label agents’ preferences in the discussions. The dataset currently has 53 different discussions of varying lengths. We apply our framework on the dataset to predict group decisions to get high training accuracy but moderate testing accuracy (53 ∼ 61% test

---

[1]https://talk.collegeconfidential.com/
accuracy for group-decision prediction for three-alternative cases for multiple voting rules). This indicates that the problem of predicting group decisions from natural language can be challenging given low amount of data and using direct NLP features like sentiment values.

To evaluate our aggregation results more than just three alternatives, we experiment with the preference data from the Moral Machine (MM) experiment [3], which consists of agents’ preferences over moral dilemmas. Our results indicate that with high probability the fractional profile winner will also be the randomized profile winner, showing effectiveness of the aggregation method.

2 Related Work and Discussions

Sentiment analysis [21, 4, 2] is one of the most popular NLP tools used, and it is very relevant to the problem of learning preferences. Since we are interested about preferences between alternatives, target-specific sentiment analysis is of more concern [5, 37]. At the moment, there are online tools that complete the task of entity-specific sentiment analysis as well. However, sentiment alone can not help model preferences. Another interesting task is stance detection [26, 34, 27, 31], which identifies whether an agent is in favor of, or against a target. Work has also been done in collaborative filtering literature, where additional text features like sentiment have been considered [13, 29, 19]. While the goal is different, this aspect of the work has similar motivation as this work. Sentiment and opinion features are typically considered for single pieces of text. While simple voting based ideas have been proposed to get from sentiment analysis to preference aggregation [14, 15], our work considers more general preference models and expands on that concept.

There are several NLP datasets built for closely related tasks, such as, [25, 40] for predicting approval rating from text; [27, 34] for stance detection. [1] has a large amount of debate data but more are about abstract topics rather than specific alternatives. The UGI corpus [6] focuses on group decision-making, but the discussions express explicit rankings instead of discussing the alternatives.

Random utility models (RUM) [36] are a family of models for modeling probabilistic preferences. We show results using two special cases of general RUMs, the Plackett-Luce model (PL) [30, 22] with features, and Thurstone’s Case V model (TV) [35] with features. PL models are in particular popular for easy-to-compute probabilities for rankings and comparisons. General RUMs subsume the model with bilinear features used in [42] and is similar to use in modeling dyad ranking [17, 32, 33]. See [39] for an exposition to different random utility models and algorithms for learning them.

Aggregation over distribution over preferences, which we also consider, was studied in different extents in [31, 28, 23]. Hazon et al. [16] discusses a dynamic programming algorithm to aggregate arbitrary distribution over preferences, which can be combined with our model but ends up being computationally expensive. Zhao et al. [42] introduce randomized voting rules which require similar computation as this paper. However, instead of assigning winning probabilities proportional to score, we predict a single winner under different voting rules and provide theoretical results about this result. We cover a larger umbrella of voting rules in this paper compared to most other papers and present general results for all anonymous voting rules.

3 Preliminaries

Fractional Preference Profiles. Assume we have \( n \) agents and set of alternatives \( A = \{a_1, \ldots, a_m\} \). Let \( \mathcal{L}(A) \) be the set of all rankings over \( A \). For a particular ranking \( \sigma \), if alternative \( a_j \) is ranked \( i \)-th in \( \sigma \), \( \sigma(j) = i \); e.g. for the highest ranked alternative \( a_j \) in \( \sigma \),
σ(j) = 1. Instead of a single preferred ranking, we may say each agent has some partial preference for each ranking. So, we define a fractional preference profile π ∈ [0, 1]^m such that π(σ) is the fractional preference for a particular ranking σ ∈ L(A). If we have fractional preference profiles π₁, . . . , πₙ for each agent, we may interpret πᵢ(σ) in two different ways. In a probabilistic setting, we may assume that πᵢ(σ) is the probability that agent i’s preferred ranking is σ. Or that agent i gives fractional vote of weight πᵢ(σ) to the ranking σ.

Random Utility Models (RUM). A RUM assumes an associated utility distribution for each alternative. Given utilities u₁, . . . , uₘ for the alternatives in A, the probability of a particular ranking Pr[aⱼ₁ ≻ . . . ≻ aⱼₘ] = Pr[uⱼ₁ ≻ . . . ≻ uⱼₘ].

The Plackett-Luce (PL) model [30, 22] and Thurstone’s Case V (TV) model are two of the most popular random utility models, where PL uses Gumbel as the distributions of all models. These definitions and more properties of RUMs are discussed in Appendix A.

Given utilities u₁, . . . , uₘ for the alternatives in A, the probability of a particular ranking Pr[aⱼ₁ ≻ . . . ≻ aⱼₘ] = Pr[uⱼ₁ ≻ . . . ≻ uⱼₘ].

The Plackett-Luce (PL) model [30, 22] and Thurstone’s Case V (TV) model are two of the most popular random utility models, where PL uses Gumbel as the distributions of all utilities [22] while TV uses Gaussian distributions. PL and TV are formally defined below.

**Definition 3.1 (Plackett-Luce model (PL)).** The parameter space is Θ = {θ = {θ₁|1 ≤ j ≤ m}}. The sample space is L(A)ⁿ. Given a parameter θ ∈ Θ, the probability of any full ranking σ = aⱼ₁ ≻ aⱼ₂ ≻ . . . aⱼₘ is PrPL(σ|θ) = 1/∏_{j=1}^{m-1} exp(θⱼⱼ) / ∑_{j=1}^{m} exp(θⱼⱼ).

**Definition 3.2 (Thurstone’s Case V Model (TV)).** The parameter space is M = {μ = {μⱼ|1 ≤ j ≤ m}}. The sample space is L(A)ⁿ. Given a parameter μ ∈ M, the utility for any alternative aⱼ = aⱼ₁ ≻ aⱼ₂ ≻ . . . aⱼₘ is PrTV(σ|μ) = Pr[u₁ > . . . > uₘ].

Probabilities for PL model are more tractable for computation. Next, we define k-mixture of PL models (k-PL in short), a generalization of PL.

**Definition 3.3 (k-mixture of PL models (k-PL)).** Given k ∈ N, the k-mixture Plackett-Luce model is defined as follows. The sample space is L(A)ⁿ. The parameter space has two parts. The first part is the mixing coefficients α = (α₁, . . . , αₖ), where for all 1 ≤ k ≤ k, αₖ ≥ 0 and ∑_{k=1}^{k=1} αₖ = 1. The second part is (θ(₁), . . . , θ(ₖ)), where θ(κ) ∈ Θ is the parameter of the ℜ-th Plackett-Luce component. The probability of any ranking σ is Prk-PL(σ|θ) = ∑_{k=1}^{k} αₖ PrPL(σ|θ(κ)).

In short, the probability of k-PL is a convex combination of k PL models. For agent aᵢ, we use θ(i)(κ) (or αₖ(i)) to denote the ℜ-th PL component (or the ℜ-th mixing coefficient).

For general RUMs, let Xᵢⱼ ∈ Rᵈ denote the feature vector of alternative aⱼ for the i-th agent. Then the perceived utilities for agent i are given by the following process: uⱼᵢ = Pr(·|Xᵢⱼ, B). Here B is the parameter of the general RUM.

We define k-mixture of Plackett-Luce model (k-PL-X) with features as follows.

**Definition 3.4 (k-mixture of Plackett-Luce model with features (k-PL-X)).** Given any agent i, each alternative aⱼ is characterized by a d-dimensional feature vector Xᵢⱼ. The parameter space has two parts. The first part is the mixing coefficients α = (α₁, . . . , αₖ), where for all 1 ≤ k ≤ k, αₖ ≥ 0 and ∑_{k=1}^{k=1} αₖ = 1. The second part is (β(₁), . . . , β(ₖ)), where β(κ) ∈ Θ is the parameter of the ℜ-th Plackett-Luce component. The sample space is L(A)ⁿ. Given a parameter β ∈ Θ, the probability of any ranking σ = aⱼ₁ ≻ aⱼ₂ ≻ . . . aⱼₘ given by agent i is

Prk-PL-X(σᵢ|α, β) = ∑_{k=1}^{k} αₖ · ∏_{j=1}^{m-1} exp(β(κ) X(i,j)/αₖ) / ∑_{κ=1}^{κ} exp(β(κ) X(i,j)/αₖ).

k-PL-X can be treated as a convex combination of k PL models with features. Similarly k-TV mixture models and models with features can also be defined for Thurstone’s Case V models. These definitions and more properties of RUMs are discussed in Appendix A.
The distribution over rankings given by PL and k-PL models enables us to compute various fractional preferences efficiently. Here, we present the distribution of k-PL model and the distribution for PL can be treated as a special case of k = 1. For a given k-PL parameters \( \bar{\alpha} = (\alpha_1, \ldots, \alpha_k) \), \( \bar{\theta} = (\bar{\theta}_1, \ldots, \bar{\theta}_{(k)}) \) and any ranking \( \sigma \), we have \( \pi(\sigma) = \Pr_{k-PL}(\sigma|\bar{\alpha}, \bar{\theta}) \). The probability of many partial rankings (i.e., \( a_1 \succ \text{others} \)) can also be computed efficiently. For example, let \( \pi^{(j)} \) denote the probability of \( a_j \) being ranked highest. We have \( \pi^{(j)} = \sum_{\sigma(j)=1} \pi(\sigma) = \sum_{i=1}^{k} \alpha_i \cdot \frac{\exp(\theta_i)}{\sum_{i=1}^{k} \exp(\theta_i)} \). Similarly, let \( \pi^{(a_j \succ a_i)} \) denote the probability of \( a_j \succ a_i \). Then we have \( \pi^{(a_j \succ a_i)} = \sum_{\sigma(j)<\sigma(i)} \pi(\sigma) = \sum_{i=1}^{k} \alpha_i \cdot \frac{\exp(\theta_i)}{\exp(\theta_i) + \exp(\theta_{(i)})} \).

Similarly, for given k-TV parameters \( \bar{\alpha} = (\alpha_1, \ldots, \alpha_k) \), \( \bar{\mu} = (\bar{\mu}_1, \ldots, \bar{\mu}_{(k)}) \), we will have \( \pi^{(a_j \succ a_i)} = \sum_{\alpha=1}^{k} \alpha \cdot \Phi(\mu_i - \mu_j(\alpha)) \), where \( \Phi \) is the standard normal CDF. Even though the probability for full rankings are computationally expensive to compute for TV model, probabilities of pairwise preferences can still be calculated efficiently.

**Voting Rules.** A voting rule \( r : \mathcal{L}(\mathcal{A})^n \rightarrow \mathcal{A} \) is a function that maps a preference profile \( (n \text{ ranking over alternatives in } \mathcal{A}) \) to a single winner. For anonymous voting rules, where the agent identity does not matter, the definition of voting rule may easily be extended to provide a winner from a fractional preference as \( r : \mathbb{R}^m \rightarrow \mathcal{A} \). For example, a positional scoring rule is characterized by a score vector \( \bar{s} = (s_1, \ldots, s_m) \) such that \( s_1 \geq \ldots \geq s_m \). And for each ranking \( \sigma \), the alternative \( a_j \) gets a score of \( s_{\sigma(j)} \). For fractional preference profile \( \pi \), an alternative \( a_j \) would get additive score \( \pi(\sigma) \times s_{\sigma(j)} \) for each \( \sigma \in \mathcal{L}(\mathcal{A}) \) and the alternative with maximum total score shall be the winner. Some popular scoring rules are: Plurality, whose scoring vector is \( \{1,0,\ldots,0\} \); Borda, whose scoring vector is \( \{m-1,\ldots,1,0\} \); approval voting with top-\( \ell \) approval, whose scoring vector is \( \{1,\ldots,\ell,0,\ldots,0\} \).

**Definition 3.5 (Weighted Majority Graph (WMG)).** Given any fractional profile \( \pi \), WMG is a directed graph where the nodes are all the alternatives, there exists edges in both directions for each pair of alternatives and weight for edge \( (a_j, a_k) \) is \( \pi^{(a_j \succ a_k)} \).

\( \pi^{(a_j \succ a_k)} \) is defined as before as the fractional preference for all rankings with \( a_j \succ a_k \). Voting rules that depend on pairwise preferences only, such as Copeland and maximin, can be considered WMG-based voting rules, because once the WMG is constructed, it is easy to compute the winner, generally in polynomial time for \( m \).

When \( r \) is Copeland, the winner is the alternative beating most other alternatives in pairwise elections. Formally, define pairwise indicator

\[
 w_r(a_j, a_\ell, F) = \begin{cases} 1 & \sum_{i=1}^{n} \pi_{i}^{(a_j \succ a_\ell)} > \sum_{i=1}^{n} \pi_{i}^{(a_\ell \succ a_j)} \\ 0 & \text{otherwise} \end{cases}
\]

Then, the Copeland winner maximizes Copeland score \( \text{score}_r(a_j, F) = \sum_{\ell \neq j} w_r(a_j, a_\ell, F) \). The maximin winner maximizes the maximin score \( \text{score}_r(a_j, F) = \min_{\ell \neq j} \sum_{i=1}^{n} \pi_{i}^{(a_j \succ a_\ell)} - \pi_{i}^{(a_\ell \succ a_j)} \). It is a well-known fact that Borda winner can also be computed using WMG.

**Multi-round Voting Rules.** There are several multi-round voting rules, mostly all of which focus on eliminating alternatives repeatedly on some criteria until an alternative has definite majority. For example, Single Transferable Voting (STV) has \( m-1 \) rounds. In each round, the alternative with least plurality score is eliminated. And in the next round, all rankings where the eliminated alternative was preferred would be counted for the next ranked alternative.
4 The Framework

In this section we formally propose the framework that learns and aggregates agents’ preferences based on natural language interactions, which consists of three parts: Opinion mining (Section 4.1), preference learning (Section 4.2), and preference aggregation (Section 5).

We view the framework mostly as a conceptual contribution, and while we implement an instance of this framework, it was not our goal to find a state-of-the-art solution to the problem, and our solution can definitely be improved upon. The main technical contributions are the theoretical guarantees in the third components, which will be presented in the next section.

4.1 Opinion mining

A discussion is an array of individual comments, where each comment is associated with an agent $i$. Each agent can have one or more comments associated with them. We assume that we know beforehand the set of alternatives, $\mathcal{A}$, that are being discussed. With the comments and the set of alternatives as input, output of the opinion mining section will be individual NLP based features for each alternative for each comment.

For example, a common NLP feature associated with preferences is sentiment. A positive sentiment expressed towards an alternative can mean preference for that alternative. So, in our implementation, using state-of-the-art tools for target-specific sentiment extractions (such as Google’s Natural Language API\footnote{https://cloud.google.com/natural-language/} and Watson Natural Language Understanding API\footnote{https://www.ibm.com/cloud/watson-natural-language-understanding}), we extract sentiment values for specific alternatives. See Table 1 for some examples of this. For each comment by agent $i$, the sentiment scores and magnitude expressed toward alternative $j$ are part of the opinion features.

In addition to extracted sentiment values, direct language features like n-grams and word-embeddings can also be considered as features for the learning problem.

4.2 Preference Learning

For each comment, we have NLP features representing the comment, and labels regarding to the preferences expressed. The goal of the preference learning problem would be to learn a model that can predict preferences over the alternatives. So, given a learned model, we can get a distribution over preferences over alternatives.

For each agent $i$ and alternative $a_j$, we can have a feature vector $X_{ij}$ consisting of NLP features, agent features and alternative features. We consider all features we can extract from the Opinion Mining stage. Additionally, we may use features of the agents and alternatives themselves in the form $\vec{u}_i \otimes \vec{v}_j$, where $\otimes$ denotes the Kronecker product, $\vec{u}_i$ is the feature vector of agent $u_i$, and $\vec{v}_j$ is the feature vector of $a_j$.

<table>
<thead>
<tr>
<th>Comment Text</th>
<th>Sentiment Score</th>
<th>Preference</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. We thought that Harvard (H) was a much more prestigious school than Yale (Y).</td>
<td>+0.7 +0.1</td>
<td>$H \succ Y$</td>
</tr>
<tr>
<td>2. My daughter chose to go to Harvard (H) over Yale (Y) but she was happy with both offers.</td>
<td>0.0 0.0</td>
<td>$H \succ Y$</td>
</tr>
</tbody>
</table>

Table 1: Examples of opinion features in text
In our implementation, we learn two types of models based on this data. First, considering pairwise preferences between alternatives, we can get a pair of alternative features and a preference label. For a pair of alternatives, A, B, possible labels can be $A \succ B, B \succ A, A = B, \text{No preference}$. So, we get a multiclass classification problem. We considered traditional classification algorithms like random forests, SVM, logistic regression, shallow neural nets for our implementation.

Secondly, we can also learn random utility models with features. As we shall see in Section 4.3, this allows us to aggregate distributions over preferences in a meaningful way. For example, Plackett-Luce model with features (PL-X) are used to model the preference profile in our datasets. See [39] for various methods to learn general RUMs. We use maximum likelihood estimation (MLE) to learn the Plackett-Luce parameters $\beta$:

$$\beta^* = \argmax_\beta \sum_{i} \ln \Pr_{PL-X}(\sigma_i | \beta)$$

4.3 Preference aggregation

Based on the learned model, given a new set of alternatives with feature vectors $\{X_{ij}\}$, we can get specific predictions for preferences or distributions over preferences. To get a group decision, we need to aggregate these predictions/distributions, and we can apply various voting rules there.

In case we have specific prediction for pairwise preferences, any weighted-majority-graph based voting rule can be used to aggregate the preference predictions. For distributional preferences it becomes computationally intractable, which leads to our proposed solutions and theoretical results. In particular, we consider the case where distribution over preferences are compactly represented using random utility models and we want to learn the output of a specific voting rule. We present theoretical result for some voting rules under this scenario in Section 5.1. In our implementation, given the learned PL-X parameters, $\beta$, we get different PL parameters, $\theta_j$ for each agent over the alternatives in the test scenarios. We then aggregate the distributions represented by these models to get a group decision.

We should mention that the preference aggregation part of the framework would also work for non-NLP problems as well. For example, if agents made pairwise comparisons for sample scenarios from a large set of alternatives, we can still learn a model to express preferences and then apply the aggregation results. An example of this would be learning preference models from pairwise data in the Moral Machine dataset, and in a new (test) scenario, predicting the group decision based on the learned models.

5 Efficient preference aggregation of distributional preferences

Based on the learned models, given a new discussion, we can predict distributions over preferences for each agent, represented compactly using random utility models. In this section, we consider the problem of aggregating these distribution over preferences to get a group decision using voting rules.

5.1 Aggregating Fractional Preferences

Assume that we have $n$ agents and agent $i$ has $k$-PL (or $k$-TV) mixing coefficients $\tilde{\alpha} = (\alpha_1, \cdots, \alpha_k)$ and parameters $(\tilde{\theta}_{i(1)}, \cdots, \tilde{\theta}_{i(k)})$ or $(\tilde{\mu}_{i(1)}, \cdots, \tilde{\mu}_{i(k)})$.

$\tilde{\alpha}$ and $\tilde{\theta}_{i(\alpha)}$’s can then be used to compute the fractional preference profile $\pi_i$ for agent $i$. As mentioned in Section 3, getting full fractional profiles for TV model can be computation-
ally expensive, but we can make use of them, when we are considering pairwise preferences only.

Let \( \Pi = \{\pi_1, \ldots, \pi_m\} \) be the set of all fractional profiles. The total preference for a particular ranking \( \sigma \) is just the sum of fractional preference for all agents. We define this as the sum-fractional profile \( F(\sigma) = \sum_{i=1}^{n} \pi_i(\sigma) \). So \( F(\Pi) = \{F(\sigma)\}_{\sigma \in \mathcal{A}(\Lambda)} \in \mathbb{R}^{m^2} \) will be a vector representing the total fractional preference among all agents for each ranking. In general, when clear from context we will just use \( F \) to mean \( F(\Pi) \). All voting rules introduced in this paper can be extended to fractional profiles. For a voting rule \( r \), we say \( r(F) \) is the fractional profile winner (FP-winner) for \( \Pi \). Numeric examples of voting rules with fractional profiles are given in the Appendix.

The following proposition can be considered a simple extension of a result of [12].

**Proposition 5.1.** Given \( m \) alternatives and \( n \) agents, where agent \( i \) has \( k\)-PL mixing coefficients \( \alpha_1, \ldots, \alpha_k \) and parameters \( (\vec{\theta}_i(1), \ldots, \vec{\theta}_i(k)) \), FP-winner \( r(F) \) can be computed in \( \mathcal{O}(k m n) \) time when \( r \) is plurality.

Positional scoring rules other than plurality can be generally hard to compute. Fortunately, there are scoring rules such as approval voting where only the top-\( \ell \) ranked alternatives get any score. For such cases, we have the following general theorem (for which Proposition 5.1 is a special case with \( \ell = 1 \)).

**Theorem 5.2.** Given \( m \) alternatives and \( n \) agents, where agent \( i \) has \( k\)-PL mixing coefficients \( \alpha_1, \ldots, \alpha_k \) and parameters \( (\vec{\theta}_i(1), \ldots, \vec{\theta}_i(k)) \), for scoring rules that have score vectors with \( \ell < m \) non-zero scores, FP-winner can be computed in \( \mathcal{O}(k m n^2) \) time.

Now, we turn to WMG-based voting rules. Once the WMG is constructed, both Copeland and maximin winners can be computed in \( \mathcal{O}(m^2) \) time. The following lemma asserts that the WMG can be computed in polynomial time from agents’ RUM parameters, if the models are \( k\)-PL or \( k\)-TV.

**Lemma 5.3.** Given \( m \) alternatives and \( n \) agents, where agent \( i \) has \( k\)-PL (or \( k\)-TV) mixing coefficients \( \alpha_1, \ldots, \alpha_k \) and parameters \( (\vec{\theta}_i(1), \ldots, \vec{\theta}_i(k)) \) (Or \( (\vec{\mu}_i(1), \ldots, \vec{\mu}_i(k)) \)), the WMG of \( F \) can be computed in \( \mathcal{O}(k m n^2) \) time.

In light of Lemma 5.3, the FP-winner of the WMG-based rules discussed in this paper can be computed in polynomial time.

**Theorem 5.4.** Given \( m \) alternatives and \( n \) agents, where agent \( i \) has \( k\)-PL (or \( k\)-TV) mixing coefficients \( \alpha_1, \ldots, \alpha_k \) and parameters \( (\vec{\theta}_i(1), \ldots, \vec{\theta}_i(k)) \) (Or \( (\vec{\mu}_i(1), \ldots, \vec{\mu}_i(k)) \)), FP-winner for Borda, Copeland, and maximin can be computed in polynomial time.

In most multi-round voting rules, a simple voting rule like plurality is applied in repeatedly. The main challenge is recomputing the score in each round as score received by the eliminated alternatives is then distributed among remaining ones. The next Theorem states that for a wide range of multi-round rules this can be done efficiently.

**Theorem 5.5.** For a multi-round voting rule \( r \), if FP-winner for voting rule applied in each round can be computed in polynomial time, then the multi-round FP-winner \( r(F) \) can also be computed in polynomial time.

For example, Theorem 5.5 and Proposition 5.1 together implies that STV FP-winner can be computed in polynomial time, and we get the following corollary.

**Corollary 5.6.** Given \( m \) alternatives and \( n \) agents, where agent \( i \) has \( k\)-PL mixing coefficients \( \alpha_1, \ldots, \alpha_k \) and parameters \( (\vec{\theta}_i(1), \ldots, \vec{\theta}_i(k)) \), FP-winner for STV can be computed in \( \mathcal{O}(k m^2 n^2) \) time.
5.2Aggregating Randomized Preferences

So far, we have been treating $\pi_i(\sigma)$ as a fractional vote by agent $i$ for ranking $\sigma$. But we can also consider the problem in a randomized setting, in which we assume that $v_i(\sigma)$ is the indicator random variable (RV) defining whether agent $i$'s preference is $\sigma$. We define the total number of agents with preference $\sigma$ as $H(\sigma)$ and $H(\Pi) = \{H(\sigma)\}_{\sigma \in \mathcal{L}(A)}$. $H(\Pi) \in \mathbb{R}^{m!}$ is an $m!$-dimensional RV, denoting the total number votes in favor of each rankings.

**Lemma 5.7.** Given set of fractional preferences $\Pi$, for all $\sigma \in \mathcal{L}(A)$, $\mathbb{E}(H(\sigma)) = F(\sigma)$

Just like $F, F(\Pi)$, we use $H$ instead of $H(\Pi)$ when it is clear from context.

Now, since $H$ is an RV, so the winner under the randomized setting would also be a random variable. We call this $r(H)$, the randomized-profile winner (RP-winner). and we would get a probability for each alternative to be the winner. Let us define the winning probability for alternative $a_j$ as $p_j = \Pr[r(H) = a_j]$.

Since we can compute $\pi_i(\sigma)$ for all $\sigma$ and all $i \leq n$, it is possible use a dynamic programming algorithm in [9] to exactly compute $p_j$’s. However, this algorithm would not be computationally tractable. So we approximately compute probability $p_j$ for $j \leq m$ with Algorithm [1]. The sampling complexity for this problem is expressed in Theorem 5.8.

**Theorem 5.8.** Given any $\epsilon, \delta > 0$, if $p = \langle p_1, \ldots, p_m \rangle$ are true winning probabilities and $\bar{p}$ is the estimate from Algorithm [1] we need $O\left(\frac{m^2}{\epsilon^2} \ln\left(\frac{2m}{\delta} \right)\right)$ samples so that $\text{TVD}(p, \bar{p}) \leq \epsilon$ with probability $1 - \delta$ where $\text{TVD}(p, \bar{p})$ is the total variation distance between the two distributions.

How Often is Fractional-Profile Winner also the Randomized-Profile Winner? For a group decision-making setting, computing the probability of winning for each alternative is not always necessary. Finding the alternative with highest probability of winning can be a much higher priority. Since we have an efficient aggregation method to compute $r(F)$, we can focus on finding $\Pr[r(H) = r(F)]$ and in particular whether we can guarantee that $r(F)$ is the alternative with highest probability of winning in $H$.

Before we delve further in this section, we define the concept of margin of victory (MoV) [38]. In a regular scenario, MoV of an election, is the smallest number $k$ such that $k$ agents can change the winner by voting differently. It can be considered a measure for how robust the election result is. We change the definition slightly to consider fractional profiles and FP-winners.

**Definition 5.1** (Margin of Victory). Given a fractional preference profile $\Pi$ and a voting rule $r$, the MoV is defined as $\text{MoV}(F(\Pi), r) = \min_{\Pi'} \frac{1}{2} \| F(\Pi) - F(\Pi') \|_1$, where $r(F(\Pi)) \neq r(F(\Pi'))$.

Now, we present the main theorem for this subsection.

**Theorem 5.9.** Given a set of $m$ alternatives $A$, $n$ agents with fractional preference profiles $\pi_1, \ldots, \pi_n$, let fractional profile be $F$ and randomized profile be $H$. For any anonymous voting rule $r$, for any $d \geq 0$, $\Pr \left[ r(H) = r(F) \mid \text{MoV}(F, r) \geq d \right] \geq 1 - 2m! \cdot \exp \left( - \frac{d^2}{2\epsilon^2} \right)$.
This theorem does not depend on any voting rule based properties and thus will hold for any anonymous voting rule \( r \). While the bound itself may be weak for cases with small \( N \) and relatively large \( m \), this fact actually works to our advantage. Because the true probability of \( r(F) \) being the winner could be much higher. Also note that while the theorem uses the value of MoV, we actually never need to compute it, because none of our voting rule algorithms depend on it. Now, if the probability of an alternative winning under randomized preferences is \( \geq 0.5 \), that guarantees that it is the highest probable winner. For the FP-winner, we get the following sufficient condition for this to happen. For low \( m \) (compared to \( \sqrt{n} \)), such a margin of victory is often attainable for real preference profiles.

**Theorem 5.10.** Given a set of \( m \) alternatives \( A \), \( n \) agents with fractional preference profiles \( \pi_1, \ldots, \pi_n \), let fractional profile be \( F \) and randomized profile be \( H \). For any anonymous voting rule \( r \), we can get \( \Pr[ r(H) = r(F) ] \geq 1/2 \) for a margin of victory \( \text{MoV}(F,r) \) that is \( \Omega(m \log(m) + \sqrt{n}) \).

6 Experiments and Results

**College Confidential Data.**

**Instances:** The dataset consists of multiple “discussions”. Each discussion has fixed alternatives (set of alternatives can be different for each discussion) and a number of users (agents) discussing their preferences over the alternatives. The discussion consists of multiple serialized comments. For each comment made by the agents, for all possible pairs of alternatives, we have human annotation of preference expressed over the alternatives.

**Collection Process:**

![Figure 2: Sample Task faced by agents](image)

We scraped all discussions in the College Confidential forum made between Jan 2017 through Jun 2020 that had two or more alternatives and had at least 40 comments. The threshold of 40 comments were chosen so that there was at least some meaningful discussion of ideas instead of few isolated comments.

For the annotation, we recruited participants on Amazon Mechanical Turk. Every participant was given 20 comments to annotate, and each comment in turn was shown to up to 3 participants. For each comment, the participants choose whether the agent expressed a preference or not, and in case they did what the exact preference was. A sample task faced by participants is shown in Figure 2.
Statistics: In total, we have labeled data from 53 discussions, 40 two-alternative and 13 three-alternative discussions. Each discussion has at least 30 individual comments which compare between the alternatives. Considering the three pairwise comparisons in a comparison of three alternative, the complete dataset consists of 2974 pairwise comparisons.

Analysis and Results:
To learn from the data, we extract entity-level sentiment features and additional text features e.g. Parts of speech (PoS) tagging, n-grams etc. to create feature vector for the scenarios. We found that the text features (PoS, n-grams), while useful to identify whether there is a preference expressed in a comment, is much less helpful than the sentiment features themselves. Thus, we consider two ways to train classifiers.

1. Two stage learning - first stage: preference vs no-preference, second stage three class classifier with labels $A > B$, $A < B$, $A = B$

2. Single stage four class classifier with labels no preference, $A > B$, $A < B$, $A = B$

When learning the PL-X parameters, we also implement a two-stage method. The first stage learns a preference vs no-preference classifier, while the PL-X model is trained only on comments that have expressed preferences. To have a baseline for comparison, we consider a predictor that predicts the class based on a single value: sentiment score, predicting preference for the alternative that has higher sentiment associated with it.

We show the results for random forest (RF), shallow neural nets (NN) and PL-X models in Table 2. While some classifiers have high training accuracy, test accuracy is low and comparable to the simple baseline for all models. This indicates that predicting preference can be a challenging task based on sentiment and other simple text features. While we also tried multinomial logistic regression, SVM and decision trees among other classifiers, they had poorer training and test accuracy, hence we do not present those results.

Aggregation results: We then predict aggregated group decisions based on the individual comment level predictions. For regular classifiers, we use the two-stage variants and we take the predictions as strict preference for one alternative or a tie and then aggregate the predictions. For PL-X however, we may additionally use fractional profiles as indicated in Section 5 to make the group decision. Note that since the labels are for pairwise comparisons, it makes sense to apply WMG-based voting rules. Hence, we test on Copeland, maximin and Borda rules. Here, also we present both training and testing accuracy. Again, we see that the training accuracy can be particularly high for RF and NN, we can get high training accuracy, the test set performance

<table>
<thead>
<tr>
<th></th>
<th>Copeland</th>
<th>Maximin</th>
<th>Borda</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Test</td>
<td>Train</td>
<td>Test</td>
</tr>
<tr>
<td>Baseline</td>
<td>6/13</td>
<td>29/53</td>
<td>6/13</td>
</tr>
<tr>
<td>RF</td>
<td>8/13</td>
<td>41/53</td>
<td>8/13</td>
</tr>
<tr>
<td>NN</td>
<td>8/13</td>
<td>37/53</td>
<td>7/13</td>
</tr>
<tr>
<td>PL-X</td>
<td>8/13</td>
<td>33/53</td>
<td>8/13</td>
</tr>
<tr>
<td>PL-X (classifier)</td>
<td>7/13</td>
<td>34/53</td>
<td>8/13</td>
</tr>
</tbody>
</table>

Table 3: Group prediction accuracy for various voting rules when trained on two alternative discussions and tested on three alternative discussions
does not improve much from the baseline. For the testing scheme, we trained on all two-alternative discussions and tested on three-alternative discussions. Since all voting rules become trivial for two alternatives, we tested on three-alternative scenarios. These results (Table 3) with somewhat high training accuracy but poor test accuracy again indicate the challenge of successfully learning and aggregating group decision from natural language.

**Moral Machine Data.** The Moral Machine dataset [3] consists of pairwise comparisons between alternatives posing a moral dilemma. Using the pairwise comparison data and alternative features, we learn an individual PL-X models for each agent. Given a new set of alternatives, using the PL-X parameters, we can learn a distribution over the alternatives $\pi_i$ for each agent $i$. Using these distributions we can check our theoretical results for preference aggregation (Section 5).

For a set of alternatives, $A$, we can compute sum-fractional preference profile $F(\Pi)$. For this experiment, we consider the scoring rules, plurality and Borda for voting. From $F(\Pi)$, we can compute $r(F)$ and MoV$(F, r)$ when $r$ is plurality and Borda. We modify Algorithm 1 for computing MoV for scoring rules presented in Xia [38] to work with fractional profiles. Additionally, for $A$, using the Algorithm 1 defined in Section 5.2, we can estimate the probability of the fractional-profile winner also being the randomized profile winner, i.e $\Pr[r(H) = r(F)]$. We repeat this process for different randomly sampled alternative sets of size $|A| = 4$ and $N = 1000$ random agents.

![Figure 3: Margin of Victory vs Probability that FP-winner is also RP-winner](image)

Figure 3 is a scatter plot for estimates and lower bounds for $\Pr[r(H) = r(F)]$, plotted against MoV. We notice that even for low margin of victory, when the theoretical guarantee is not that strong, the probability that the fractional winner is the randomized-profile winner as well is close to 1. This gives confidence in our aggregation framework. We also show further experimental results in Appendix D.

### 7 Discussion and Future Work

We propose a framework that learns preference models for agents from natural language and then efficiently aggregate preferences to provide a group decision. We also present an initial implementation of the framework in a newly created a dataset for the specific problem of group decision-making from natural language. From our experiments, we notice that even sentiment extracted using state-of-the-art sentiment analysis methods are alone not enough to express sentiments. For preferences, we noticed that sentiments about alternatives are not always what matters. But rather figuring out the context of other entities about which sentiments were expressed is an important problem of preference learning and may improve the framework performance. Our dataset, in its current state, may not be large enough to train deep learning based models and it is our goal to expand the dataset with more labeled discussions. In that case, making use of modern deep learning methods in NLP to directly solve the preference learning problem might become more tractable. Also, it would also be interesting to expand on this framework towards building a multi-modal group decision-making system by learning preferences from other possible inputs besides natural language.
Acknowledgments

We thank Tania Bedrax-Weiss, Craig Boutilier, participants of the COMSOC video seminar, and anonymous reviewers for helpful discussions and comments. This work is supported by NSF #1453542, ONR #N00014-17-1-2621, and a gift fund from Google.

References


Farhad Mohsin (Corresponding author)
Rensselaer Polytechnic Institute
Troy, NY, USA
Email: mohsif@rpi.edu

Lei Luo
Duke University
Durham, NC, USA

Wufei Ma
Purdue University
West Lafayette, IN, USA

Inwon Kang
Rensselaer Polytechnic Institute
Troy, NY, USA

Zhibing Zhao
Microsoft Research
Bellevue, WA

Ao Liu
Rensselaer Polytechnic Institute
Troy, NY, USA

Rohit Vaish
Tata Institute of Fundamental Research (TIFR)
Mumbai, India

Lirong Xia
Rensselaer Polytechnic Institute
Troy, NY, USA
A Discussion of Random Utility Models

In this, we use a more general Plackett-Luce model with features for preference learning. Let $X_{ij} \in \mathbb{R}^d$ denote the feature vector of $a_j$ given agent $i$. We define the Plackett-Luce model with features as follows.

**Definition A.1 (Plackett-Luce model with features (PL-X)).** Given any agent $i$, each alternative $a_j$ is characterized by a $d$-dimensional feature vector $X_{ij}$. The parameter space is $\Theta = \{ \vec{\beta} = \{ \beta_k | 1 \leq k \leq d \} \}$. The sample space is $\mathcal{L}(A)^n$. Given a parameter $\vec{\beta} \in \Theta$, the probability of any ranking $\sigma_i = a_{j_1} \succ a_{j_2} \succ \ldots \succ a_{j_m}$ given by agent $i$ is

$$\Pr_{\text{PL-X}}(\sigma_i | \vec{\beta}) = \frac{m-1}{\sum_{q=1}^{m} \exp(\vec{\beta} \cdot X_{ij_q})} \prod_{p=1}^{m-1} \frac{\exp(\vec{\beta} \cdot X_{ij_p})}{\sum_{q=p}^{m} \exp(\vec{\beta} \cdot X_{ij_q})}.$$

**Internal Consistency of RUMs:** Because of the internal consistency that fractional preferences have when defined using PL parameters, the updated score each round may also be computed efficiently.

**Proposition A.1 (Internal consistency).** Given PL parameters $\theta_1, \ldots, \theta_m$, consider any partial ranking $\sigma_B = \{a_{j_1} \succ \ldots \succ a_{j_b}\}$ over alternatives in $B = \{a_{j_1}, \ldots, a_{j_b}\}$ where $B \subseteq A$. The fractional preference for all rankings $\sigma \in \mathcal{L}(A)$ which agree with $\sigma_B$ regarding the order of alternatives in $B$ is

$$\sum_{\sigma, \sigma_B \text{ are consistent}} \pi(\sigma) = \pi(\sigma_B) = \prod_{p=1}^{b-1} \frac{\exp(\theta_{j_p})}{\sum_{q=p}^{b} \exp(\theta_{j_q})}.$$

This result, proven in [18], implies that computing the updated results in each round is equivalent to ignoring the already eliminated alternatives and just considering the PL parameters for existing alternatives. This is useful both for computing pairwise preferences and also preferences in multi-round voting rules.

B Numerical Example for Fractional Profiles

**Example 1:** For $A = \{a_1, a_2, a_3\}$, suppose we have fractional profiles as in Table 4 for 3 agents.

<table>
<thead>
<tr>
<th>$\sigma$</th>
<th>$\pi_1(\sigma)$</th>
<th>$\pi_2(\sigma)$</th>
<th>$\pi_3(\sigma)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_1 \succ a_2 \succ a_3$</td>
<td>1/2</td>
<td>0</td>
<td>1/3</td>
</tr>
<tr>
<td>$a_1 \succ a_3 \succ a_2$</td>
<td>1/2</td>
<td>0</td>
<td>1/3</td>
</tr>
<tr>
<td>$a_2 \succ a_1 \succ a_3$</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$a_2 \succ a_3 \succ a_1$</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$a_3 \succ a_1 \succ a_2$</td>
<td>0</td>
<td>0</td>
<td>1/6</td>
</tr>
<tr>
<td>$a_3 \succ a_2 \succ a_1$</td>
<td>0</td>
<td>0</td>
<td>1/6</td>
</tr>
</tbody>
</table>

Table 4: Example of fractional profiles for $m = 3, n = 3$

Based on these fractional preferences we can calculate alternative scores for different voting rules. For example, we show plurality, Borda and maximin scores in Table 5.

<table>
<thead>
<tr>
<th>alternative</th>
<th>Plurality score</th>
<th>Borda score</th>
<th>Maximin score</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_1$</td>
<td>5/3</td>
<td>9/2</td>
<td>11/6</td>
</tr>
<tr>
<td>$a_2$</td>
<td>1</td>
<td>3</td>
<td>7/6</td>
</tr>
<tr>
<td>$a_3$</td>
<td>1/3</td>
<td>3/2</td>
<td>1/3</td>
</tr>
</tbody>
</table>

Table 5: Example of fractional profiles with $m = 3, n = 3$
We see that, under all three voting rules, $a_1$ has the maximum score and is FP-winner for these fractional profiles.

C  Various Proofs

Proof for Proposition 5.1

**Proposition.** Given $m$ alternatives and $n$ agents, where agent $i$ has $k$-PL mixing coefficients $\alpha_1, \ldots, \alpha_k$ and parameters $(\tilde{\theta}_{i(1)}, \ldots, \tilde{\theta}_{i(k)})$, FP-winner $r(F)$ can be computed in $O(kmn)$ time when $r$ is plurality.

**Proof.** The score vector when $r = \text{plurality}$ is $s = \{1, 0, \ldots, 0\}$. So, to compute plurality winner, we only need $\pi_i^{(j)}$ for all $i \leq n$ and $j \leq m$. And by definition, plurality winner is

$$r(F) = \arg\max_{a_j} \sum_{i=1}^{n} \pi_i^{(j)} = \arg\max_{a_j} \sum_{i=1}^{n} \frac{\exp(\theta_i^{(j)})}{\sum_{k=1}^{m} \exp(\theta_k^{(j)})}.$$ 

Clearly, the sum for each $j$ can be computed in $O(n)$ and thus the argmax can be computed in $O(mn)$ time. \qed

Proof for Theorem 5.2

**Theorem.** Given $m$ alternatives and $n$ agents, where agent $i$ has $k$-PL mixing coefficients $\alpha_1, \ldots, \alpha_k$ and parameters $(\tilde{\theta}_{i(1)}, \ldots, \tilde{\theta}_{i(k)})$, for scoring rules that have score vectors with $\ell < m$ non-zero scores, FP-winner can be computed in $O(klmn)$ time.

**Proof.** Only the first $\ell$ ranks contribute to alternative scores. Thus, if we enumerate all possible permutations over the top $\ell$ positions and compute the fractional preference for each permutation, we can compute score for each alternative. There are $\frac{m!}{(m-\ell)!}$ such permutations.

Now, Assume $\sigma_{1\ell} = a_{j_1} > \ldots > a_{j_\ell}$ is such one permutation. For $k$-PL parameters as described,

$$\pi_i(\sigma_{1\ell}) = \sum_{\kappa=1}^{k} \alpha_\kappa \cdot \prod_{p=1}^{\ell} \frac{\exp(\theta_{i(\kappa)}^{(p)j_p})}{\sum_{q=p}^{m} \exp(\theta_{i(\kappa)}^{(q)}j_q)}.$$ 

So computing each $\pi_i(\sigma_{1\ell})$ would take $O(\ell)$ time, there are $O(m^\ell)$ such permutations, and $n$ users, we can compute score for all alternatives in $O(klmn)$ time. \qed

Proof for Lemma 5.3

**Lemma.** Given $m$ alternatives and $n$ agents, where agent $i$ has $k$-PL (or $k$-TV) mixing coefficients $\alpha_1, \ldots, \alpha_k$ and parameters $(\tilde{\theta}_{i(1)}, \ldots, \tilde{\theta}_{i(k)})$ (or $(\tilde{\mu}_{i(1)}, \ldots, \tilde{\mu}_{i(k)})$), the WMG of $F$ can be computed in $O(kmn)$ time.

**Proof.** The WMG for $F$, same as that for a single fractional profile $\pi_i$ would have $m$ nodes and thus $O(m^2)$ edges. Weight for edge $(a_j, a_\ell)$ for $k$-PL model would be $\sum_{i=1}^{n} \pi_i^{(j)\sim(\ell)} = \sum_{i=1}^{n} \sum_{\kappa=1}^{k} \alpha_\kappa \frac{\exp(\theta_{i(\kappa)}^{(j)})}{\exp(\theta_{i(\kappa)}^{(j)}) + \exp(\theta_{i(\kappa)}^{(\ell)})}.$

For $k$-TV model, this becomes $\pi^{(j)\sim(\ell)} = \sum_{\kappa=1}^{k} \alpha_\kappa \cdot \Phi(\mu_{i(\kappa)}^{(j)} - \mu_{i(\kappa)}^{(\ell)})$

For each case, each edge weight is obviously computable in $O(kn)$ time, which leads to the desired result. \qed
Lemma C.1. Given $m$ alternatives and $n$ agents, where agent $i$ has $k$-PL mixing coefficients $\alpha_1, \ldots, \alpha_k$ and parameters $(\tilde{\theta}_i(1), \ldots, \tilde{\theta}_i(k))$, consider a multi-round voting rule given where voting rule $r_q$ is used in round $q$. Let run-time of $r_q$ for $m$ voters be $T(r_q, m)$, then total run-time for the multi-round procedure shall be $\mathcal{O}(\sum_{q=1}^{m} T(r_q, m - q + 1))$.

Proof. For each round until the last one, the eliminated alternative needs to be computed instead of the winner. For all rules we have discussed, finding lowest score alternative requires the same time as finding winning alternative. Now, Lemma A.1 tells us that for a single PL, computing the fractional preference for a partial ranking is as simple as ignoring the other alternatives. Now, the probability of $k$-PL is a convex combination of $k$ PL models. In round $q$, if the set of remaining alternatives is $B \subset A$ with $|B| = m - q + 1$, for some $\sigma_B \in \mathcal{L}(B)$

$$\pi_i(\sigma_B) = \sum_{k=1}^{\infty} \prod_{p=1}^{b-1} \frac{\exp(\theta_{i,p})}{\sum_{j=p}^{\infty} \exp(\theta_{i,j})}$$

Thus, computing $\pi_i$ for $i \leq n$ and consequentially fractional preference for any other partial ranking in round $q$ concerns computing over only the remaining $m - q + 1$ alternatives. So, we can say that the run-time for round $q$ is $T(r_q, m - q + 1)$, thus completing the proof. \qed

Proof for Lemma 5.7

Lemma. Given set of fractional preferences $\Pi$, for all $\sigma \in \mathcal{L}(A)$, $\mathbb{E}(H(\sigma)) = F(\sigma)$

Proof. The probability that agent $i$’s preferred ranking is $\sigma$ is $\pi_i(\sigma)$ and the related indicator RV is $v_i(\sigma)$. So, $\mathbb{E}(v_i(\sigma)) = \pi_i(\sigma)$. Now, $H(\sigma) = \sum_{i=1}^{n} v_i(\sigma)$. So, $\mathbb{E}(H(\sigma)) = \sum_{i=1}^{n} \mathbb{E}(v_i(\sigma)) = \sum_{i=1}^{n} \pi_i(\sigma) = F(\sigma)$. \qed

Proof for theorem 5.8

Theorem. Given any $\epsilon, \delta > 0$, if $p = (p_1, \ldots, p_m)$ are true winning probabilities and $\bar{p}$ is the estimate from Algorithm 1, we need $\mathcal{O}(\frac{m^2 \epsilon \ln(\frac{2m}{\delta})}{\delta^2})$ samples so that $\text{TVD}(p, \bar{p}) \leq \epsilon$ with probability $1 - \delta$ where $\text{TVD}(p, \bar{p})$ is the total variation distance between the two distributions.

Proof. For alternative $a_i$, define indicator random variable $X_j$, which takes the value 1 when $a_i$ wins, and 0 otherwise. Then $X_j$ is Bernoulli distributed with probability $\pi(a_i)$, and each trial is independent. Let $\epsilon > 0$ be arbitrary. Define $\epsilon_i = \frac{\epsilon}{m}$.

Then we use Chernoff bound and have

$$\Pr[|\sum_{j=1}^{T} X_{i,j}^{(i)} - \pi(a_i)| \geq \epsilon_i] \leq 2e^{-\frac{\epsilon_i^2}{\pi(a_i)}}$$

The total variation distance between the winner distributions computed using the Monte Carlo algorithm $\hat{p}$ and the underlying ground truth $p$ can be defined as $\sum_{a_i \in A} |p_i - \hat{p}_i|$. Applying the union bound, with probability at least $1 - 2\sum_{i=1}^{m} e^{-\frac{\epsilon_i^2}{\pi(a_i)}}$, the total variation distance is below $\sum_{i=1}^{m} \epsilon_i = \epsilon$.

So, to have prob. $> \delta$, we would need at least $T = \frac{3m^2 \epsilon \ln(\frac{2m}{\delta})}{\epsilon^2}$ samples to have $\text{TVD}(p, \bar{p}) \leq \epsilon$ \qed

Proof for theorem 5.9

Theorem. Given a set of $m$ alternatives $A$, $n$ agents with fractional preference profiles $\pi_1, \ldots, \pi_n$, let fractional profile be $F$ and randomized profile be $H$. For any anonymous voting rule $r$, for any $d \geq 0$, $\Pr[r(H) = r(F) | \text{MoV}(F, r) \geq d] \geq 1 - 2m! \cdot \exp\left(-\frac{d^2}{m + 2N}\right)$. 

Proof. Assuming that $F(\sigma) > 0$ for all $\sigma \in \mathcal{L}(A)$, we get the following Chernoff bounds for $\delta > 0$

$$\Pr[|H(\sigma) - F(\sigma)| \geq \delta F(\sigma)] \leq 2 \exp \left( -\frac{\delta^2}{2} F(\sigma) \right)$$

$$\implies \Pr[|H(\sigma) - F(\sigma)| \geq d] \leq 2 \exp \left( -\frac{d^2}{2F(\sigma)} \right) \leq 2 \exp \left( -\frac{d^2}{2N} \right)$$

with $d = \delta F(\sigma)$. The last inequality comes from the fact that $F(\sigma) \leq N$.

Using the union bound with this for all $\sigma$, we get

$$\Pr \left[ \bigcap_{\sigma \in \mathcal{L}(A)} (|H(\sigma) - F(\sigma)| \leq d) \right] \geq 1 - 2m! \exp \left( -\frac{d^2}{2N} \right)$$

This basically means that with high probability, total vote for all rankings in $H$ will be within $d$ of what it is in $F$. Suppose, now for some voting rule $r$, we compute the winner and MoV based on $F$ to get $r(F)$ and MoV($F, r$). Now, if MoV($F, r$) $\geq d$, and $H(\sigma)$ is within $d$ to $F(\sigma)$ for all $\sigma \in \mathcal{L}(A)$ that would be sufficient for $r(F) = r(H)$, independent of the voting rule and that concludes the proof.

Proof for Theorem 5.10

**Theorem.** Given a set of $m$ alternatives $A$, $n$ agents with fractional preference profiles $\pi_1, \ldots, \pi_n$, let fractional profile be $F$ and randomized profile be $H$. For any anonymous voting rule $r$, we can get $\Pr[r(H) = r(F)] \geq \frac{1}{2}$ for a margin of victory MoV($F, r$) that is $\Omega(m \log(m) + \sqrt{n})$.

**Proof.** Instead of applying the union bound where we did in the proof of Theorem 5.9, we could use it before to get the following-

$$\Pr \left[ \bigcap_{\sigma \in \mathcal{L}(A)} (|H(\sigma) - F(\sigma)| \leq d) \right] \geq 1 - \sum_{\sigma \in \mathcal{L}(A)} \exp \left( -\frac{d^2}{2F(\sigma)} \right)$$

We know the added fact that $\sum_{\sigma \in \mathcal{L}(A)} F(\sigma) = N$. Because the fractional votes must add up to 1 for each agent. Thus, to get a tighter lower bound, we can solve the following optimization problem

$$\maximize \sum_{p=1}^{m!} \exp \left( -\frac{d^2}{2x_p} \right)$$

where $\sum_{p=1}^{m!} x_p = n$.

Because of symmetry, the KKT points for this $m!$-dimensional optimization problem would be all points where some $x_p$ are 0 and the rest are all equal. Thus, we can represent the optimum value as

$$\max \left\{ (m! - k) \exp(-d) + k \exp(-\frac{d^2}{d+n}) \right\}_{p=1}^{m!}$$

Note that this already leads to a tighter bound than Theorem 5.9; however we proceeded to present the simpler result because there is no unique $k$ that gives a general inequality for all values of $d$. Thus we chose to present the simpler but weaker result there.

Although we get no fixed $k$ for all values of $d$, as $d$ grows higher, the value for $k = 1$ dominates the other values. Particularly, for $d > \sqrt{n}$,

$$\max \left\{ (m! - k) \exp(-d) + k \exp(-\frac{d^2}{d+n}) \right\}_{p=1}^{m!} = (m! - 1) \exp(-d) + \exp(-\frac{d^2}{d+n})$$
Now, we can find \( d = \Omega(m \log m) \) s.t. \((m! - 1) \exp(-d)\) goes to zero. Also, as \( d \) grows slightly from \( \sqrt{n} \), \( \exp(-\frac{d^2}{m+n}) \) also becomes low. And thus, combined if \( d = \Omega(m \log(m) + \sqrt{n}) \), we get the desired result.

\[\square\]

D Additional Simulation Plots for Moral Machine Data

Figure 4: Margin of Victory vs Probability that FP-winner is also RP-winner

\(\Pr[r(F) = r(H)]\) was predicted with less than ideal number of samples for these plots to ensure probabilistic guarantees as in Theorem 5.8. So they are slightly noisy. But the trend indicates that the probability is near 1 in all cases.