Soft Condorcet Optimization for Ranking of General Agents

Marc Lanctot
Google DeepMind
Montreal, Canada

Kate Larson
Google DeepMind,
University of Waterloo
Waterloo, Canada

Michael Kaisers
Google DeepMind
Paris, France

Quentin Berthet
Google DeepMind
Paris, France

Ian Gemp
Google DeepMind
London, United Kingdom

Manfred Diaz
Mila, University of Montreal
Montreal, Canada

Roberto-Rafael Maura-Rivero
Google DeepMind, LSE
London, United Kingdom

Yoram Bachrach
Meta[1]
London, United Kingdom

Anna Koop
Google DeepMind
Montreal, Canada

Doina Precup
Google DeepMind
Montreal, Canada

ABSTRACT

Driving progress of AI models and agents requires comparing their performance on standardized benchmarks; for *general* agents, individual performances must be aggregated across a potentially wide variety of different tasks. In this paper, we describe a novel ranking scheme inspired by social choice frameworks, called Soft Condorcet Optimization (SCO), to compute the optimal ranking of agents: the one that makes the fewest mistakes in predicting the agent comparisons in the evaluation data. This optimal ranking is the maximum likelihood estimate when evaluation data (which we view as votes) are interpreted as noisy samples from a ground truth ranking, a solution to Condorcet's original voting system criteria. SCO ratings are maximal for Condorcet winners when they exist, which we show is not necessarily true for the classical rating system Elo. We propose three optimization algorithms to compute SCO ratings and evaluate their empirical performance. When serving as an approximation to the Kemeny-Young voting method, SCO rankings are on average 0 to 0.043 away from the optimal ranking in normalized Kendall-tau distance across 865 preference profiles from the PrefLib open ranking archive. In a simulated noisy tournament setting, SCO achieves accurate approximations to the ground truth ranking and the best among several baselines when 59% or more of the preference data is missing. Finally, SCO ranking provides the best approximation to the optimal ranking, measured on held-out test sets, in a problem containing 52,958 human players across 31,049 games of the classic seven-player game of Diplomacy.

KEYWORDS

agent evaluation; social choice theory; ranking

Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

ACM Reference Format:
Marc Lanctot, Kate Larson, Michael Kaisers, Quentin Berthet, Ian Gemp, Manfred Diaz, Roberto-Rafael Maura-Rivero, Yoram Bachrach, Anna Koop, and Doina Precup. 2025. Soft Condorcet Optimization for Ranking of General Agents. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 16 pages.

1 INTRODUCTION

Progress in the field of artificial intelligence has been driven by measuring the performance of agents on common benchmarks and challenge problems [15, 33, 66, 69, 75, 76, 79]. In machine learning, common benchmarks like the UCI data set repository allowed direct comparisons of supervised learning algorithms [41]. Competitions, such as ImageNet, led to breakthroughs in deep learning [44].

All of these examples require comparing agents (or models). Original success stories such as DeepBlue, TD-Gammon, and AlphaGo focused on a single domain. In the past ten years, agents have become increasingly more generally capable. AlphaZero extended application of AlphaGo to chess and Shogi [70]. The Arcade Learning Environment [6], which steered much of the agent development in deep reinforcement learning, evaluated agents across 57 different Atari games. Recently, language models have been evaluated across suites of tasks such as in HELM [49], BIG-bench [7] AgentBench [51], and via a public leaderboard such as Chatbot Arena driven by human voting [17]. Answering simple questions for these generally capable agents, such as "Which is the best agent?" or "Is agent X better than agent Y?" or "What is the relative ranking of agents X, Y, and Z?" become increasingly more difficult when aggregating over many different contexts: how agents are scored may vary wildly across tasks, data collected for evaluation may not be balanced evenly across tasks (or agents), and classical rating systems were simply not designed for this use case.

To address these problems, recent ranking methods such as Vote'N'Rank [65] and Voting-as-Evaluation (VasE) [46] use voting methods to aggregate results across tasks. Using computational

[1]Work performed while at Google DeepMind.

social choice as a basis for ranking agents has several benefits: well-studied consistency properties of the voting methods are inherited, they do no require score normalization, and are less sensitive to score values and agent population than game-theoretic evaluation schemes [5]. However, classic voting schemes, and related tournament solutions, typically assume that the data (e.g. agent comparisons) is complete. This assumption is not necessarily valid in the agent evaluation setting. While there has been research on identifying "necessary and possible winners" when there is incomplete voting or comparison data, the results are mixed [3, 61, 82] and most of the findings are focused on identifying top-ranked agents, not ranking all agents as is our focus.

In this paper, we introduce a new ranking scheme for general agents inspired by the interpretation of voting rules as maximum likelihood estimators [20]. Starting with Condorcet's original model of voting [14, Chapter 8], Young showed that the maximum likelihood estimate (MLE) of the true ranking is the one that minimizes the sum of Kendall-tau distances to all the votes. Soft Condorcet Optimization (SCO) solves an optimization problem that assigns a numerical *rating* (score) θ_a to each agent (alternative) a. SCO then treats these ratings as a parameter vector, the votes as a data set, and defines a differentiable loss function as the objective, which can be optimized in several different ways. The final ranking of agents is obtained by sorting the ratings.

In summary, this paper makes the following contributions.

1. SCO ranking scheme with the following properties: (1a) Three optimization methods to find ratings and corresponding rankings: gradient descent applied to a soft Kendall-tau distance ("sigmoid loss"), or a Fenchel-Young loss (perturbed optimization) [8]; or solving a sigmoidal program with a branch-and-bound method [78]. (1b) Online form that can update ratings, and thus rankings, from individual outcomes as evaluation data arrives. (1c) Theorem 1 guarantees that the top-ranked agent by SCO ratings according to the sigmoid loss is the Condorcet winner when one exists.

2. Empirical evaluations that demonstrate the following: (2a) SCO ranking using sigmoid loss solves a failure mode of classical Elo rating system which may top-rank an agent that is not a Condorcet winner even when one exists. (2b) SCO can serve as an approximation to the Kemeny-Young voting method, indeed empirically finding low approximation error to the optimal ranking: on average 0 to 0.043 away in normalized Kendall-tau distance across 865 preference profiles from the PrefLib [55]. (2c) In a noisy tournament setting with sparse data, SCO approximates the true ranking best when a large proportion (59% or more) of the data is missing. (2d) SCO ratings are closer to optimal rankings than Elo and voting-as-evaluation methods on held out test sets over 31,049 human Diplomacy games played by 52,958 players.

2 BACKGROUND

In this section, we describe the building blocks and introduce some basic terminology required to understand our method.

2.1 Evaluation of General Agents

We first paraphrase several key descriptions from [5, 46]. The problem of evaluating agents is that of ranking agents according to their skill. Skill can be determined in several ways. In the **Agent-versus-Task (AvT)** setting, agents compete individually in different tasks and compare outcomes (scores) to each other in each task (*e.g.*, language models and the various metrics for assessing their abilities). In the **Agent-versus-Agent (AvA)** setting, agents directly compete against each other (*e.g.*, online games such as chess or Diplomacy).

2.1.1 *Classical Evaluation.*

Elo is a classic rating system that uses a simple logistic model learned from win/loss/draw outcomes [32]. A rating, r_i, is assigned to each player i such that the probability of player i beating player j is predicted as $\hat{p}_{ij} = \frac{1}{1+10^{(r_j - r_i)/400}}$. While Elo was designed specifically to rate players in the two-player zero-sum, perfect information game of Chess, it has been widely applied to other domains, including in evaluation of large language models [84]. TrueSkill [38] and bayeselo [22] are rating systems based on similar foundations (Bradley-Terry models of skill) that also model uncertainty over ratings using Bayesian methods.

Elo has a number of positive qualities. First, it is a simple rule. Second, it can be used to estimate win rates between any two agents. Third, it can be easily employed *online*, *i.e.*, to modify players' ratings from the result of a single game. It is also a special case of logistic regression (for details, see Appendix D.2). Hence, Elo has been widely applied and is a common default choice for evaluation of agents. However, Elo has a number of well-known drawbacks [5, 9, 46]. Of particular interest is the incompatibility of Elo with the concept of a Condorcet winner from social choice theory; examples are summarized in Section 4.1.

2.1.2 *Voting as Evaluation of General Agents.*

Another way to evaluate agents is to use social choice theory, called Voting-as-Evaluation (VasE) recently proposed in [46]. In VasE, the alternatives correspond to general agents and votes to assessments of their performance. In the AvT setting, agents are ordered based on their performance on different tasks (such as each game in the Atari Learning Environment [6] or on different benchmarks for large language models [51]). In the AvA setting, agents compete directly in a multi-agent environment, such as multiplayer games (like chess or poker), and each game outcome corresponds to a ranking over a subset of agents. Chatbot Arena [84], where language models compete head-to-head to provide the best answer to the same question, is another instance of the AvA setting. Casting agent evaluation as an application of computational social choice provides the benefit of robustness in the form of Condorcet consistency, clone/composition consistency, agenda consistency, and/or population consistency depending on the choice of voting rule used to evaluate agents. However, it is unclear how well these methods perform when the data is missing or unevenly distributed, which often occurs in the agent evaluation setting. SCO is motivated similarly to VasE and, as such, will be empirically assessed mainly for agent evaluation. To make the connection to ideas from the social choice literature, we will often use voting language when discussing SCO and its use in evaluation. Relevant terminology is introduced later in the paper.

2.2 Permutation and Ranking Distances

We now define distance metrics over rankings that will play a key role when describing our method and loss function. Informally, the

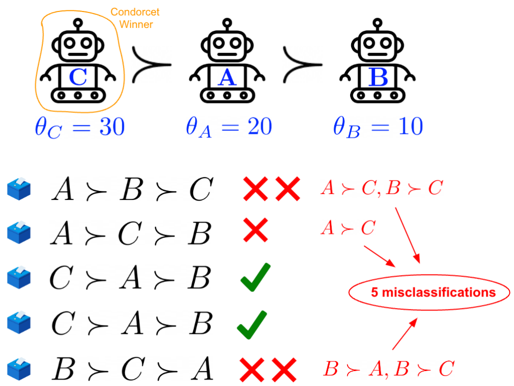

Figure 1: An example calculation of the Kendall-tau distances of each vote in the preference profile from Table 1 to the optimal ranking $R = C \succ A \succ B$. The sum of the distances is 5. This sum for any other ranking $R' \neq R$ is greater than 5.

1:	$A \succ B \succ C$
1:	$A \succ C \succ B$
2:	$C \succ A \succ B$
1:	$B \succ C \succ A$

	A	B	C
A	0	4	2
B	1	0	2
C	3	3	0

	A	B	C
A	0	3	-1
B	-3	0	-1
C	1	1	0

Table 1: Left: an example preference profile with five votes [46, Figure 1]. The number on the left of the colon represents the number of votes of that type. Middle: the voter preference matrix, N. Right: the voter margin matrix, M.

Kendall-tau distance counts the number of pairwise disagreements between permutations.

Definition 1. *Let S_1, S_2 be finite sets of elements such that $S_1 \subseteq S_2$. Let π_1, π_2 be permutations over elements in S_1 and S_2, respectively. The* ***Kendall-tau distance*** *between two permutations is defined as*

$$K_d(\pi_1, \pi_2) = \sum_{\{i,j\} \in C_2(S_1)} \bar{K}_{i,j}(\pi_1, \pi_2), \quad (1)$$

where $C_2(S)$ is the set of unordered pairs of S (combinations of size 2), $\bar{K}_{i,j}(\pi_1, \pi_2) = 0$ if i and j are in the same order in π_1 and π_2, and $\bar{K}_{i,j}(\pi_1, \pi_2) = 1$ otherwise.

Note that this definition allows one set to be a subset of the other, which is more general than the standard definition; this distinction is necessary for the evaluation metric used in Section 4.4, and corresponds to the standard definition when $S_1 = S_2$.

Since the maximum distance is $\binom{|S_1|}{2} = \frac{|S_1|(|S_1|-1)}{2}$, this can be easily normalized to be in $[0, 1]$:

Definition 2. *Let S_1, S_2 be finite sets of elements such that $S_1 \subseteq S_2$. Let π_1, π_2 be permutations over elements in S_1 and S_2, respectively. The* ***normalized Kendall-tau distance*** *π_1 and π_2 is defined as*

$$K_n(\pi_1, \pi_2) = \frac{2K_d(\pi_1, \pi_2)}{|S_1|(|S_1|-1)}. \quad (2)$$

2.3 Social Choice Theory

A **voting scheme** is defined as $\langle A, V, f \rangle$ where $A = \{a_1, \ldots, a_m\}$ is the set of **alternatives** (agents), $V = \{v_1, \ldots, v_n\}$ is the set of **voters**, and f is the **voting rule** that determines how votes are aggregated. Voters have **preferences** over alternatives: $a_1 \succ_{v_i} a_2$ indicates voter v_i strictly prefers alternative a_1 over alternative a_2. In this paper, we assume strict preferences only, however the ideas can easily be extended to the case that allow weak preference (including ties/indifference between alternatives, *e.g.*, $a_1 \succeq s_2$). These preferences induce total orders over alternatives, which we denote by $\mathcal{L}$. A **preference profile**, $[\succ] \in \mathcal{L}^n$, is a vector specifying the preferences of each voter in V. It can be useful to summarize the preference profile in a **voter preference matrix** N or **vote margin matrix** M. The preference count $N(x, y)$, $x, y \in A$ is the number of voters in $[\succ]$ that strictly prefer x to y. The vote margin is the difference in preference count: $\delta(x, y) = N(x, y) - N(y, x)$. Table 1 shows a preference profile and its resulting preference matrix, $\mathbf{N} = (N(i, j))_{i,j}$, and margin matrix $\mathbf{M} = (\delta(i, j))_{i,j} = \mathbf{N} - \mathbf{N}^T$.

The central problem of social choice theory is how to aggregate preferences of a population so as to reach some collective decision. A voting rule that determines the "winner" (a non-empty subset, possibly with ties), is a **social choice function** (SCF). A voting rule that returns an aggregate ranking (total order) over all the alternatives is a **social welfare function** (SWF). Much of the social choice literature focuses on understanding what properties different voting rules support.

The **Condorcet winner** defines a fairly intuitive concept: A is the (strong) Condorcet winner if the number of votes where A is ranked higher than B is greater than vice versa for all other alternatives B. A weak Condorcet winner wins or ties in every head-to-head pairing. Formally, given a preference profile, $[\succ]$, a weak Condorcet winner is an alternative $a^* \in A$ such that $\forall a' \in A, \delta(a^*, a') \geq 0$. If the inequality is strict for all pairs except (a^*, a^*) then we call it a strong Condorcet winner. In the example shown in Table 1, alternative C is the strong Condorcet winner. It dominates A since three out of the five voters prefer C to A. A similar situation holds when C is compared to alternative B. While many have argued that this definition captures the essence of the correct collective choice [27], in practice preference profiles may have no Condorcet winner. Condorcet-consistent voting schemes (*e.g.*, Kemeny-Young introduced next) return a Condorcet winner when it exists, but differ on how they handle settings with no Condorcet winners.

2.3.1 Kemeny-Young Voting Method. The voting method was initially proposed by Kemeny [42]. Later, its properties were characterized by Young & Levenglick [83]. Let each ranking (total order) be represented as a permutation π over $|A|$ alternatives. Define the Kemeny score of permutation π as $\textsc{KemenyScore}(\pi) = \sum_{(i,j),i<j} N(\pi[i], \pi[j])$. The Kemeny rule returns the ranking that maximizes this Kemeny score: $\text{argmax}_{\pi \in \Pi(|A|)} \textsc{KemenyScore}(\pi)$. Kemeny-Young is *Condorcet-consistent*: if a Condorcet winner exists, it will be top-ranked by Kemeny-Young. It also satisfies the majority criterion, the Smith criterion [71], and monotonicity. The Kemeny rule always returns an **optimal ranking**, *i.e.*, one whose sum of Kendall-tau distances to the votes is minimal, but its computational complexity is prohibitively expensive when m is large.

2.4 Learning-to-Rank

Another related field is that of learning-to-rank [50]. The canonical example is that a user enters a keyword (query) and the problem is to retrieve the most relevant documents in a database, ranked by relevance to the keyword. There are several algorithms that learn to rank; Google's PageRank, which powers their search engine, is one example. Our setting can be characterized by a learning-to-rank problem with a constant keyword (or no keyword) as there is no query. An important class of statistical models in this setting is random utility models [81]. In random utility models, assessments are perceived as some ground truth assessment plus some noise.

Perturbed optimizers [8, 11] transform non-differentiable functions (such as sorting and ranking) into smoothed versions by adding noise; these perturbed functions can then be optimized by gradient descent. This is part of a growing effort to allow end-to-end training through discrete operators, using classical stochastic smoothing and perturbation approaches [35, 37]. Included are optimal transport, clustering, dynamic time-warping and other dynamic programs [25, 26, 57, 60, 67, 73, 80] applied in fields such as computer vision, audio processing, biology, and physical simulators [4, 16, 21, 45, 48, 52] and other optimization algorithms [30].

3 SOFT CONDORCET OPTIMIZATION

Soft Condorcet Optimization (SCO) is a ranking scheme for evaluation of general agents inspired by social choice theory. An SCO ranking is derived from and represented by **SCO ratings**, θ_a, for each alternative $a \in A$. The rating $\theta_a \in [\theta_{\min}, \theta_{\max}]$ serves only to determine a's relative order compared to other alternatives in the ranking, such that $a \succ b$ if and only if $\theta_a > \theta_b$. The ranking is induced by numerically sorting these ratings (*e.g.*, see Figure 1). We then formulate an optimization problem by carefully constructing a loss function that that penalizes discrepancies or misclassifications in the ordinal relationships between alternatives.

An SCO rating is a numerical value representing an agent's level of skill. SCO is closely related to several prior works: Elo [32], probabilistic ranking [2, 28, 54], and perturbed optimizers [8, 11]. We elaborate on these relationships in Appendix D.

3.1 SCO Ratings and the Sigmoid Loss

In this section, we first explain the **sigmoid loss** function. For some set of ratings $\boldsymbol{\theta}$, this loss function quantifies the amount of error: the level of disagreement between the specific values of each rating and the data (preference profiles obtained through agent evaluation). The goal is then to find an assignment of ratings that minimizes this loss, *i.e.*, the set of ratings that best explain the preference data.

Let $\Pi(A)$ denote the set of permutations over alternatives A. We will call the data we work with "votes" to emphasize that we are working with (partial) rankings over alternatives, and borrow terminology from preferences and social choice. In particular, we view the entire dataset to be a collection of votes and so will refer to it as *preference profile*, $[\succ]$. Let $v \in [\succ]$ refer to each vote in the profile, where v is a permutation over subsets of A, with length $|v|$. For a vote v, denote $v[i]$ as the alternative in position i such that vote v is represented as: $v[0] \succ v[1] \succ \cdots \succ v[|v|-1]$.

The ultimate goal is to find an **optimal ranking** R:

Definition 3. *Given some a profile $[\succ]$ (i.e., set of votes), an* **optimal ranking** *minimizes the sum of the Kendall-tau distances*

$$R = argmin_{R \in \Pi(A)} \sum_{v \in [\succ]} K_d(v, R).$$

Let index pairs $I_2(v) = \{(i,j) \mid i,j \in \{0,1,\cdots,|v|-1\}$ and $i < j\}$. Given a preference profile and ratings $\boldsymbol{\theta}$, we define a **discrete loss**:

$$L([\succ], A, V, \boldsymbol{\theta}) = \sum_{v \in [\succ]} \sum_{(i,j) \in I_2(v)} D_v(\theta_{v[i]}, \theta_{v[j]}), \tag{3}$$

where $D_v(\theta_a, \theta_b)$ is a function that measures the *discrepancy* between the positions of alternatives a and b:

$$D_v(\theta_a, \theta_b) = \begin{cases} 1 & \text{if } \theta_b - \theta_a > 0 \text{ in } v; \\ 0 & \text{otherwise,} \end{cases}, \tag{4}$$

then the minimum of the discrete loss function corresponds to a ratings assignment $\boldsymbol{\theta}$ such that ranking induced by $\boldsymbol{\theta}$ minimizes the sum of the Kendall-tau distances to all the votes.

Example 1. *Recall the example from Figure 1 showed the computation of the sum of Kendall-tau distances from the ranking $R = C \succ A \succ B$ to votes depicted in preference profile in Table 1. In this example, we show how the value is computed under rating vector:*

$$\boldsymbol{\theta} = (\theta_A, \theta_B, \theta_C) = (20, 10, 30).$$

Let $[\succ]$ be the preference profile depicted in Table 1. We now show that the main loss function (Equation (3)) leads to same value as in Figure 1. The outer sum of Equation 3 enumerates the votes, which we will assume is in the same order as listed in Figure 1. The inner sum computes the Kendall-tau distance from the vote v to the ranking R induced by $\boldsymbol{\theta}$ (red exes in Figure 1). For the first vote $v = A \succ B \succ C$, the discrepancy function D outputs 1 two times: once with pair $(i,j) = (0,2)$ and once with pair $(i,j) = (1,2)$ because the preferences between agents (A, C) and agents (B, C) disagree between $\boldsymbol{\theta}$ and v, so the inner sum for the first vote is 2. Similarly for the other votes: they correspond precisely to the same values as in Figure 1. Hence, the loss function (Equation (3)) is simply computing the sum of Kendall-tau distances between the ranking induced by $\boldsymbol{\theta}$ and all the votes.

Since D is a step function discontinuous at $\theta_a = \theta_b$, it is not differentiable in $\boldsymbol{\theta}$. To solve this, we replace D with a smooth approximation, *i.e.*, the logistic function

$$\tilde{D}_v(\theta_a, \theta_b) = \sigma(\theta_b - \theta_a) = \frac{1}{1 + e^{(\theta_a - \theta_b)/\tau}}, \tag{5}$$

leading to a soft Kendall-tau (differentiable) **sigmoid loss**:

$$\tilde{L}([\succ], A, V, \boldsymbol{\theta}) = \sum_{v \in [\succ]} \sum_{(i,j) \in I_2(v)} \tilde{D}_v(\theta_{v[i]}, \theta_{v[j]}). \tag{6}$$

The sigmoid loss is a differentiable version of the Kendall-tau distance sum and acts as a smooth approximation to the discrete loss.

Note that while we focus on the sigmoid loss as a soft approximation to Kendall-tau distance in this paper, the same approach can be used for other ranking distances that can be approximated by differentiable functions, such as Spearman's footrule distance [29].

3.2 Sigmoid Loss Minimization

SCO ratings can be computed using the sigmoid loss in two ways.

Algorithm 1: Learning SCO ratings by gradient descent

Input: A preference profile $[\succ]$
Input: An initial parameter vector $\boldsymbol{\theta}^0 = \left(\frac{\theta_{max}-\theta_{min}}{2}\right)\mathbf{1}$, where $\mathbf{1}$ is a vector of ones of length $|A|$
Input: Learning rates for each step α^t
Input: Batch size K
1 **for** $t \in \{1, 2, \cdots, T\}$ **do**
2     $B \leftarrow K$ votes sampled uniformly from $[\succ]$
3   Define $\nabla_{\boldsymbol{\theta}} \tilde{L}(B, \boldsymbol{\theta})$ based on equation (7).
4     $\boldsymbol{\theta}^t = \text{Proj}(\boldsymbol{\theta}^{t-1} - \alpha^t \nabla_{\boldsymbol{\theta}} \tilde{L}(B))$
5 **return** $\boldsymbol{\theta}^T$

3.2.1 Gradient Descent. The most straight-forward way is to apply gradient descent [36]: update ratings by following the gradient of the sigmoid loss, as shown in Algorithm 1. After applying the gradient to the ratings on line 4, the ratings may escape the bounded constraint space so we project them back. This is a straight-forward application of standard gradient descent [36, Chapter 2]. A common variant is stochastic gradient descent (SGD) which estimates the gradient by sampling subsets, *i.e.*, "batches", of the data set, computing the gradient using the sampled batch only. The standard ℓ_2 projection step, Proj, projects the ratings back to the hypercube by clipping any ratings that fall outside the valid range $[\theta_{min}, \theta_{max}]$.

We compute the gradient for a subset of the votes (*i.e.*, batch $B \subseteq [\succ]$), $\nabla_{\boldsymbol{\theta}} \tilde{L}(B, \boldsymbol{\theta})$, from the batch loss $\tilde{L}(B, \boldsymbol{\theta})$, which resembles equation (6) but summed only over the votes in B using the continuous $\tilde{D}_v$ from equation (5):

$$\tilde{L}(B, \boldsymbol{\theta}) = \sum_{v \in B} \sum_{(i,j) \in I_2(v)} \tilde{D}_v(\theta_{v[i]}, \theta_{v[j]}). \tag{7}$$

This allows an online form of the algorithm, similar to Elo, where ratings for players can be updated in a decentralized fashion after receiving the outcome of a single game ($|B| = 1$).

3.2.2 Sigmoidal Programming. A different way to compute SCO ratings is via sigmoidal programming [78]: solve for the (soft) optimum directly, *i.e.*, find $\boldsymbol{\theta}^*$ that minimizes $\tilde{L}$ defined in equation 6. Note the $\tilde{L}$ can be rewritten in terms of the number of pairwise interactions between agents, quantified in the $\mathbf{N}$ matrix:

$$\tilde{L}([\succ], A, V, \boldsymbol{\theta}) \quad = \sum_{a,b \in A \times A} N(a,b)\sigma(\theta_b - \theta_a) \tag{8}$$

which is a sum of *sigmoidal functions* σ defined as functions which are strictly convex on domain $\theta_b - \theta_a \leq z$ and then strictly concave on $\theta_b - \theta_a \geq z$ (*i.e.*, $z = 0$). With a variable per difference in pair of ratings and appropriate constraints on variables and their feasible regions, the resulting optimization problem is known as a *sigmoidal program* which can be solved using a branch-and-bound algorithm [78]. A detailed construction is presented in Appendix C[1]

3.3 Fenchel-Young Loss Optimization

Here, we give an overview of the implementation of Fenchel-Young loss optimization; for more detail on the precise formulation and relationship perturbed optimizers, please see Appendix B.

[1]All appendices are available in the technical report version of the paper [47].

In practice, Fenchel-Young loss optimization follows Algorithm 1, with a different definition of the loss and hence gradient on line 3. For a single vote v, a stochastic version of the gradient $\hat{g}_v$ can be computed: let $\boldsymbol{\theta}_v$ be the ratings for agents compared in v. Let $\mathbf{X}$ be a vector of Gumbel-distributed random variables of size $|v|$. Then $\tilde{\boldsymbol{\theta}}_v = \boldsymbol{\theta}_v + \sigma X$ is the *perturbed ratings* and let $\text{ArgSort}(-\tilde{\boldsymbol{\theta}}_v)$ be the the indices of the elements that would sort the values. The Fenchel-Young gradient is then:

$$\hat{g}_v = \text{ArgSort}(-\tilde{\boldsymbol{\theta}}_v) - (0, 1, \cdots, |v| - 1), \tag{9}$$

for all agents in v, and 0 otherwise. Inuitively, if the perturbed ranking obtained by sorting $\tilde{\boldsymbol{\theta}}_v$ would yield the same order of agents as in v, then the gradient for this vote would be zero. Otherwise, it is nonzero and each element of the gradient corresponds to the difference in rank position between the vote and perturbed ranking. Similarly to standard gradient descent, these gradients can be accumulated over batches $|B| \geq 1$ as $\nabla_{\boldsymbol{\theta}} \tilde{L}^{FY}(B, \boldsymbol{\theta}) = \sum_{v \in B} \hat{g}_v$.

3.4 Theoretical Properties

Given the SCO framework, defined through the loss function introduced in equation 6, the first question to ask is whether its solution does lead to rankings with desired properties. We answer this question in the affirmative.

Theorem 1. *Given the sum of soft Kendall-tau distances:*

$$\tilde{L}([\succ], A, V, \boldsymbol{\theta}) \quad = \sum_{a,b \in A \times A} N(a,b)\sigma(\theta_b - \theta_a), \tag{10}$$

if for preference profile $[\succ]$, voters V, there exists a candidate $c \in A$ that is a Condorcet winner, the loss is monotonically decreasing with θ_c. As a consequence, if $\boldsymbol{\theta}^*$ is a global minimum of $\tilde{L}$ on the ℓ_∞ ball of radius $\theta_{\max}$, then $\theta^*_c = \theta_{\max}$.

Proof (sketch – for a full proof, please see Appendix A). Let $c \in A$ be the Condorcet winner. The loss $\tilde{L}$ as expressed in equation 8 is expressable in terms of a constant K (that does not depend on $\boldsymbol{\theta}$) and a sum of sigmoids multiplied by coefficients from $\mathbf{M}$, by symmetry of the logistic function: $\sigma(x) = 1 - \sigma(-x)$. The second term is decomposable into contributions from comparisons to agent c, which is monotonically decreasing in θ_c, and a sum which does not depend on θ_c. As a result, increasing θ_c always decreases the loss, hence the minimum must correspond to $\theta^*_c = \theta_{max}$. □

Theorem 1 assumes that it is possible to find $\boldsymbol{\theta}^*$. The challenge is $\tilde{L}$ is nonconvex in its parameters, $\boldsymbol{\theta}$, and thus standard gradient descent is not guaranteed to find a global minimum. However, the sigmoidal programming approach described in Section 3.2.2 is guaranteed to find a point that approximately minimizes $\tilde{L}$ within a specified tolerance region, though the problem may take exponential time in the number ($\Omega(m^2)$) of variables. Furthermore, as we will show in Section 4, stochastic gradient descent, while without any guarantees, tends to perform very well in practice.

The ranking loss used by Fenchel-Young optimization method described in Section 3.3 is convex and Lipschitz with respect to its parameters. Since the parameters correspond to the ratings themselves, the loss is also convex with respect to the parameters. Hence, assuming we restrict the ratings to a compact, convex set, there exists a global minimum that gradient descent via Fenchel-Young gradients is guaranteed to approach assuming standard learning rate conditions (e.g., square-summable, not summable).

4 EMPIRICAL EVALUATION

We run experiments to demonstrate a number of properties of interest. In particular, we are interested in understanding how closely the rankings obtained from SCO approximate those obtained via Kemeny-Young, compare to outcomes returned by perturbed optimizers and Elo using several sources of data:

Example data. This is a preference profile used in Section 4.1, example similar to the one from Table 1. **PrefLib data.** These are examples from the voting literature on Wikipedia and on real data from elections, sports analytics, and others from PrefLib [55]. Note that we restrict ourselves to the strictly-ordered incomplete (SOI) and strictly-ordered complete (SOC) data types in PrefLib, yielding a total of 12,680 preference profiles.

Synthetic evaluation data. The synthetic data is generated to resemble those coming from online matching data, such as from a gaming site or tournament, based on TrueSkill [38]. Agents' true skill values are normally distributed and contests (matches) between them are generated such that each agent's individual performance is stochastic with mean centered at their skill level. The outcome of each match-up is then a sorted list of each player's performance in the match-up, equal to their true skill plus normally-distributed noise. Generated data allows us to mimic the structured sparsity present in real online game-play data but also to compare results to actual ground truth rankings.

Diplomacy game data. This is our largest challenge problem: an anonymized agent-vs-agent data set of 7-player Diplomacy games played on the webDiplomacy web site (webdiplomacy.net) between 2008 and 2019 [46] with $m = 52,958$ agents (players) and $n = 31,049$ votes (games). Each game outcome is a strict order between seven players; the goal is to find a ranking over agents that minimizes the average Kendall-tau distance to all the votes. Consequently, only 0.0011% of the margin matrix entries are nonzero.

We chose these data sets to show specific properties of SCO ratings: top-ranking Condorcet winners, PrefLib ranking data capturing real human preferences, the online game regime evaluation systems are commonly deployed but with ground truth ratings, and finally a very large challenge human evaluation problem.

For Elo ratings, the majorization-minorization algorithm of Hunter (used by bayeselo [22]) is used to efficiently compute the best fit to the evaluation data [39], and the SigmoidalProgramming package [77] to solve sigmoidal programs. By default we use gradient descent to minimize the sigmoid loss (Algorithm 1) to compute the SCO ratings, but also compare Fenchel-Young gradients and sigmoidal programming. Unless otherwise noted, $\theta_{min} = 0$ and $\theta_{max} = 100$. Full details such as specific hyperparameter values, please see Appendix E. We also show two additional experiments in Appendix E: one which shows SCO used to approximate a Bayesian posterior over rating and one that shows SCO's online performance.

4.1 Warmup: Top-Ranking Condorcet Winners

Lanctot et al. showed that Elo assigns the same rating to agent A and C in the example in Table 1, despite A not being a Condorcet winner [46]. In contrast, SCO is designed to find the optimal ranking according to Definition 3, which top-ranks Condorcet winners when they exist. Consider the following 5-vote preference profile:

$$2 : A \succ B \succ C, \qquad 3 : C \succ A \succ B. \tag{11}$$

Note that $\delta(C, A) = \delta(C, B) = 3 - 2 > 0$, hence agent C is a strong Condorcet winner. However, the win rate of A ($\frac{7}{15}$) is higher than C ($\frac{6}{15}$), hence Elo assigns strictly higher rating to agent A. Since there are only six possible rankings, it is easy to verify that $\tilde{L}$ is minimized for the optimal ranking $C \succ A \succ B$. Since $n = 5$, we use full gradient descent (no batching) and compare to stochastic gradient descent with a batch size of 2. In all cases, Algorithm 1 using the sigmoid loss converges to the optimal ranking, and so does sigmoidal programming.

We also find that Fenchel-Young gradient descent top-ranks agent A. This is because the gradient of a rating using the Fenchel-Young loss is weighted by the difference in ranks rather than just order misclassifications like the soft Kendall-tau distance. We elaborate on this in Section 5. Full results are shown in Appendix E.1.

$m_\perp$	$m_\top$	size	$\bar{m}$	$\bar{n}$	C^{gd}	C^{sp}	$\bar{K}_n^{gd}$	$\bar{K}_n^{sp}$
2	2	11	2.00	29	1.00	1.00	0	0
3	3	115	3.00	1878	1.00	1.00	0	0
4	4	162	4.00	7189	1.00	0.99	0.005	0.009
5	5	135	5.00	34666	1.00	0.66	0.024	0.039
6	6	109	6.00	33266	0.99		0.043	
7	7	92	7.00	28755	0.97		0.029	
8	8	73	8.00	18336	0.96		0.032	
9	9	88	9.00	4190	0.94		0.027	
10	10	80	10.00	3289	0.97		0.023	
11	20	1721	16.48	127	0.99			
21	50	1465	30.75	34	0.98			
51	100	567	71.50	40	0.92			
101	200	2989	124.00	25	0.99			
201	500	4540	302.81	65	0.98			
501	–	533	2190.04	50	0.56			

Table 2: Kemeny-Young approximation quality on 12,680 PrefLib instances, grouped by number of alternatives where $m_\perp \le |A| \le m_\top$. Each row corresponds to a group, size to the number of instances in each group, $\bar{m}$ and $\bar{n}$ are the average number of alternatives and votes in each group. C^{gd} and C^{sp} refer to the Condorcet match proportions for gradient descent and sigmoidal programming, respectively. Similarly, $\bar{K}_n$ refers to normalized Kendall-tau distance to the Kemeny ranking, averaged over all instances in the group.

4.2 Kemeny-Young Approximation Quality

We evaluate approximation quality (compared to the Kemeny-Young ranking) of Algorithm 1 on PrefLib instances [55]. We run Algorithm 1 with a batch size $|B| = 32$, learning rates $\alpha \in \{0.01, 0.1\}$, iterations $T \in \{10^4, 10^5\}$, and temperature $\tau \in \{1, \frac{1}{2}\}$ averaged across 3 seeds per instance on all 12,680 PrefLib data instances. On the instances where $|A| \le 10$ we also run the Kemeny-Young method. Denote an instance by $i \in \{1, 2, \cdots, 12680\}$. Each produces a ranking which we denote $R_{i,\text{SCO}}$ and $R_{i,\text{Kem}}$.

We compute two metrics: (i) *Condorcet Match Proportion*: this is the proportion of instances that top-ranks a Condorcet winner when one exists. (ii) *Normalized Kendall-tau Distance*: For all instances i with $|A| \le 10$, the average value of $K_n(R_{i,\text{SCO}}, R_{i,\text{Kem}})$.

We show the aggregated metrics for 15 groupings of alternatives ($m = |A|$) that partition the 12,680 preference profiles in Table 2. Generally, when using gradient descent (Algorithm 1, Section 3.2.1) the Condorcet winner is top-rated when it exists 92% - 100% of the time when $|A| \le 500$ and the average normalized Kendall-tau distance to the Kemeny solution is low ($\bar{K}_n(R_{i,\text{SCO}}, R_{i,\text{Kem}}) \le 0.043$). We found that sigmoidal programming worked as well as Algorithm 1 for instances where $m \le 4$. On instances with five or more alternatives, there were numerical instabilities with the sigmoidal programming approach leading to a high number of failures. Hence, we run only gradient descent when $m \ge 6$.

4.3 Sparse Data Regime

In this subsection, we generate synthetic data by simulating evaluations from match-ups played in an online game setting. We do this in two ways: one that is uniform (reflecting a round-robin style tournament), and one that leads to a structured form of sparsity often encountered in competitive gaming (*i.e.*, skill-matching platforms). We use $|A| = 20$ agents where for each agent i: $\theta_i \sim N(100, 30)$, and contests between 4 agents (*i.e.*, four-player games). To generate contests, two separate distributions are used: the **uniform distribution** samples agents uniformly at random, and the **skill-matched distribution** which incrementally builds each contest, drawing 3 new candidates at random and choosing the one whose true rating is closest to the average of the set of agents so far. Then, for each contest c, we simulate the performance of agent i in that contest, $P_i(c) = \theta_i + \epsilon_{c,i}$, where each $\epsilon_{c,i} \sim N(0, 5.0)$. The outcome of the contest (vote among contestants) is then obtained by sorting the performances of all the agents in that contest.

We run Algorithm 1, Elo, and several VasE methods across many simulated n-contest tournaments, where each value of n corresponds to a proportion of missing data $p_\emptyset$ (alternative pairs that have not been evaluated in a contest together), with $p^u_\emptyset$ and $p^s_\emptyset$ referring the the uniform and skill-match distributions respectively:

n	5	10	20	30	50	75	100	200
$p^u_\emptyset$	0.85	0.72	0.52	0.38	0.20	0.09	0.04	0.001
$p^s_\emptyset$	0.88	0.75	0.59	0.49	0.36	0.28	0.23	0.15

For each value of n we run 200 instances using different seeds and report average values. For each run, we used 10000 iterations with batch size 16. We show two different metrics. First, the Kendall-tau distance between the final ranking found by Algorithm 1 and the true ranking given the true ratings (maximum value of $\frac{20 \cdot 19}{2} = 190$). This first metric identifies the pairs of agents whose relative order disagree between SCO and the true ranking; we denote these discordant pairs $D_{SCO,true}$. The second metric, which we call "mean true ratings distance" (MTRD), is then defined to be the average absolute difference in true ratings between all pairs of agents in these disagreements $\sum_{(i,j) \in D_{\text{SCO,true}}} |\theta_i - \theta_j| / |D_{\text{SCO,true}}|$, which allows us to take a nuanced look at the optimized parameters in addition to the associated ranking. The results are shown in Fig. 2.

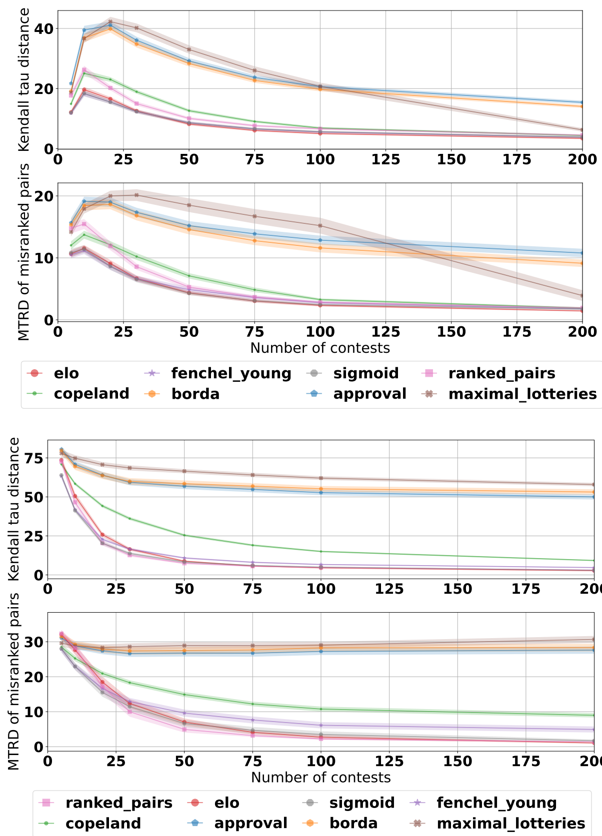

Figure 2: Kendall-tau distance (KTD) of ranking to true ranking, and Mean True Rating Distance (MTRD) of misranked pairs for ranking methods in tournament settings. Top: uniform distribution, Bottom: skill-matched distribution. Error bars represent 95% confidence intervals.

Of the VasE voting methods: approval, Borda, and maximal lotteries have the highest KTD and MRTD, and struggle especially in the skill-matched distribution. This is unsurprising; for example, approval and Borda will naturally weight alternatives according to their representation in the data. Under both distributions, when 59% or more of the match-ups are missing, SCO ratings (computed using both sigmoid and Fenchel-Young losses) achieve the lowest KTD and MRTD. Under the uniform distribution, SCO achieves the lowest when 38% or more of the data is missing. Under the skill-matched distribution, ranked pairs achieves comparable KTD to SCO ratings, and lower MRTD when the amount of missing data is less than 50%. Elo values are comparable to SCO in the uniform distribution and higher in the skill-matched distribution.

4.4 Diplomacy Game Outcome Prediction

In this subsection we investigate the question of how well SCO rankings predict human game outcomes. Recall that this data set consists of all human-played seven-player games taken from the `webDiplomacy.net` spanning 11 years, resulting in a data set of size $m = 52,958$ and $n = 31,049$. As a result, less than 0.01% of

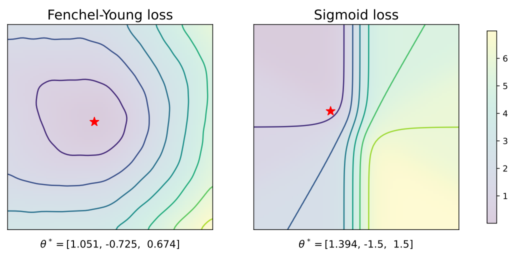

Figure 4: Loss landscapes for the Fenchel-Young (left) and sigmoid (right), with minimizer (red star) over $[-1.5, 1.5]^3$.

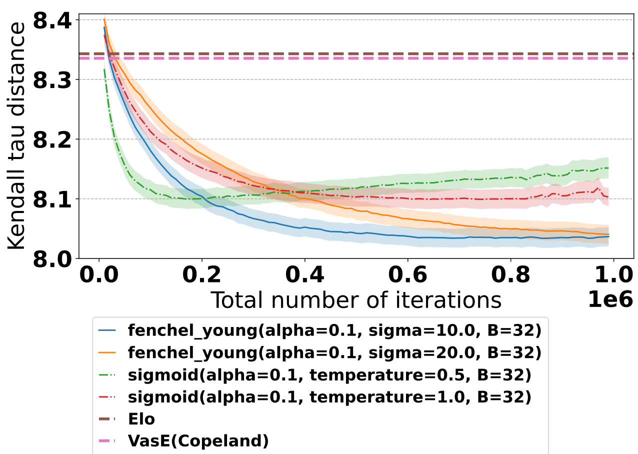

Figure 3: Average $\mathrm{KTD}_{test}(t)$ between rankings and Diplomacy game outcomes across 50 seeds and train/test splits. All runs use batch size 32. Error bars depict 95% conf. intervals.

the unique player combinations are observed in the data. Finding a single ranking that sufficiently predicts the unseen test data is a difficult due to the extreme sparsity of the preference data.

We reflect the evaluation used in [46] with a small modification to adopt common practice in the supervised learning setting . First, we create 50 random splits of the data into training sets $\mathcal{D}_R$ and testing sets $\mathcal{D}_T$, with $|\mathcal{D}_R| = 28049$ game outcomes (votes) and $|\mathcal{D}_T| = 3000$ game outcomes. The random splits are such that each alternative in the test set is seen at least once on the training set, but no game outcomes (data points) are shared across train / test split. At each iteration t the method has a ranking denoted R_t learned from data in $\mathcal{D}_R$; we then compute and report the average Kendall-tau distance over the test set: $\mathrm{KTD}_{test}(t) = \frac{1}{3000} \sum_{g \in \mathcal{D}_T} K_d(g, R_t)$.

The resulting $\mathrm{KTD}_{test}(t)$ is shown in Figure 3. We found that batch size had a minor effect on the results, so we show results for batch size 32 only. SCO with sigmoid loss reaches an average Kendall-tau distance of 8.10 after roughly 190000 iterations, and Fenchel-Young loss reaches 8.05 at 600000 iterations. There is an effect of increasing error after reaching this low point, likely due to overfitting; this could be reduced with early stopping, annealing learning rates, or other forms of regularization. As a point of comparison, Elo and the best VasE method on this data set, VasE(Copeland), achieve a value of 8.34. The next best VasE method was plurality, achieving a value of 8.57. The more complex Condorcet methods, such as ranked pairs and maximal lotteries, cannot be run on this dataset due to their complexity, since $m = 52,958$.

5 DISCUSSION

Which algorithm should be used to compute SCO ratings? In our experience, sigmoidal programming produced similar results to standard gradient descent using the sigmoid loss; however, it sometimes suffered from numerical instability and was hard to scale to large number of agents due to programs requiring m^2 variables. Hence, we recommend using the sigmoidal programming approach only when there are a relatively low number of alternatives. The Fenchel-Young loss is convex and hence gradient descent is guaranteed to converge to a global minimum; however, Condorcet winners (when they exist) are not necessarily top-ranked at that global minimum. The practical performance of Fenchel-Young loss minimization is comparable to sigmoid loss minimization and slightly better in the large Diplomacy problem. Both the sigmoid and Fenchel-Young losses are minimized by gradient descent so can be optimized online (batch size $|B| = 1$) and work particularly well in approximating the optimal rankings when a large portion of the evaluation data is missing.

The differences between sigmoid and Fenchel-Young loss minimization is further illustrated in Figure 4, where we plot two landscapes of the Fenchel-Young and sigmoid losses for the same data over three agents, using the example of vote profiles in Equation 11. Since both losses are invariant by adding a constant, we plot over a 2D slice, for all values of θ with the same sum. As discussed above, the Fenchel-Young loss is strictly convex, its global minimizer exists is found by gradient descent. Like Elo, it favours the win rate and assigns the highest rating not to the Condorcet winner C, but to A. In contrast, the sigmoid loss is nonconvex, and has no global minimizer (it keeps decreasing at infinity). Optimizing over constrained rating, it assigns the highest rating to the Condorcet winner C.

If Condorcet-consistency or distance to the optimal ranking is important, we recommend using the sigmoid loss as it optimizes to minimize the distance to it directly; despite being non-convex, in practice it finds the Condorcet winner when it exists $\geq 96\%$ of the time when the number of alternatives $m \leq 500$ and returns almost-optimal rankings on instances where it can be compared (Section 4.2). Whereas, if assured convergence or weighting the loss functions by win rates (like Elo) is more important, we recommend using the Fenchel-Young loss optimization.

For future work, we would like to compare the performance of SCO to faster methods for finding Kemeny rankings (or approximations thereof) [1, 40, 43, 63, 64] or other ranking methods inspired by tournament solutions [62], or alternatives [18]. Considering similar differential approximations to other ranking distance functions such as Spearman's footrule distance [29] as well as other ways to aggregate rankings [31] could be worthwhile. Finally, we would like to investigate using SCO and social choice theory for driving post-training for alignment of language models [19, 56, 74].

REFERENCES

[1] Alnur Ali and Marina Meilă. 2012. Experiments with Kemeny Ranking: What Works When? *Mathematical Social Sciences* 64, 1 (2012), 28–40.

[2] Mayer Alvo and Philip L H Yu. 2014. *Statistical Methods for Ranking Data*. Springer New York.

[3] Haris Aziz, Markus Brill, Felix Fischer, Paul Harrenstein, Jérôme Lang, and Hans Georg Seedig. 2015. Possible and necessary winners of partial tournaments. *Journal of Artificial Intelligence Research* 54 (2015), 493–534.

[4] Gunjan Baid, Daniel E Cook, Kishwar Shafin, Taedong Yun, Felipe Llinares-López, Quentin Berthet, Anastasiya Belyaeva, Armin Töpfer, Aaron M Wenger, William J Rowell, et al. 2023. DeepConsensus improves the accuracy of sequences with a gap-aware sequence transformer. *Nature Biotechnology* 41, 2 (2023), 232–238.

[5] David Balduzzi, Karl Tuyls, Julien Perolat, and Thore Graepel. 2018. Re-evaluating evaluation. In *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), Vol. 31. Curran Associates, Inc.

[6] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. 2013. The Arcade Learning Environment: An Evaluation Platform for General Agents. *Journal of Artificial Intelligence Research* 47 (jun 2013), 253–279.

[7] BIG bench authors. 2023. Beyond the Imitation Game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research* (2023). https://openreview.net/forum?id=uyTL5Bvosj

[8] Quentin Berthet, Mathieu Blondel, Olivier Teboul, Marco Cuturi, Jean-Philippe Vert, and Francis Bach. 2020. Learning with Differentiable Pertubed Optimizers. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 9508–9519. https://proceedings.neurips.cc/paper_files/paper/2020/file/6bb56208f672af0dd65451f869fedfd9-Paper.pdf

[9] Quentin Bertrand, Wojciech Marian Czarnecki, and Gauthier Gidel. 2023. On the Limitations of the Elo, Real-World Games are Transitive, not Additive. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 206)*, Francisco Ruiz, Jennifer Dy, and Jan-Willem van de Meent (Eds.). PMLR, 2905–2921.

[10] Mathieu Blondel, André FT Martins, and Vlad Niculae. 2020. Learning with Fenchel-Young losses. *The Journal of Machine Learning Research* 21, 1 (2020), 1314–1382.

[11] Mathieu Blondel, Olivier Teboul, Quentin Berthet, and Josip Djolonga. 2020. Fast Differentiable Sorting and Ranking. In *Proceedings of the 37th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 119)*, Hal Daumé III and Aarti Singh (Eds.). PMLR, 950–959. https://proceedings.mlr.press/v119/blondel20a.html

[12] Meriem Boubdir, Edward Kim, Beyza Ermis, Sara Hooker, and Marzieh Fadaee. 2023. Elo Uncovered: Robustness and Best Practices in Language Model Evaluation. (Nov. 2023). arXiv:2311.17295 [cs.CL]

[13] R. A. Bradley and M. E. Terry. 1952. Rank analysis of incomplete block designs I: The method of paired comparisons. *Biometrika* 39 (1952), 324–345.

[14] Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D. Procaccia (Eds.). 2016. *Handbook of Computational Social Choice*. Cambridge University Press.

[15] Murray Campbell, Jr. Hoane, A. Joseph, and Feng-hsiung Hsu. 2002. Deep Blue. *Artificial Intelligence* 134, 1–2 (2002), 57–83.

[16] Andrew N Carr, Quentin Berthet, Mathieu Blondel, Olivier Teboul, and Neil Zeghidour. 2021. Self-supervised learning of audio representations from permutations with differentiable ranking. *IEEE Signal Processing Letters* 28 (2021), 708–712.

[17] Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Banghua Zhu, Hao Zhang, Joseph E. Gonzalez Michael Jordan, and Ion Stoica. 2024. Chatbot Arena: An Open Platform for Evaluating LLMs by Human Preference. In *Proceedings of the 41st International Conference on Machine Learning (PLMR, Vol. 235)*.

[18] Ari Conati, Andreas Niskanen, and Matti Järvisalo. 2022. SAT-based Judgment Aggregation. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.

[19] Vincent Conitzer, Rachel Freedman, Jobst Heitzig, Wesley H. Holliday, Bob M. Jacobs, Nathan Lambert, Milan Mosse, Eric Pacuit, Stuart Russell, Hailey Schoelkopf, Emanuel Tewolde, and William S. Zwicker. 2024. Position: Social Choice Should Guide AI Alignment in Dealing with Diverse Human Feedback. In *Proceedings of the 41st International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 235)*, Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (Eds.). PMLR, 9346–9360. https://proceedings.mlr.press/v235/conitzer24a.html

[20] V. Conitzer and T. Sandholm. 2005. Common Voting Rules as Maximum Likelihood Estimators. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*. 145–152.

[21] Jean-Baptiste Cordonnier, Aravindh Mahendran, Alexey Dosovitskiy, Dirk Weissenborn, Jakob Uszkoreit, and Thomas Unterthiner. 2021. Differentiable patch selection for image recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*. 2351–2360.

[22] Rémi Coulom. 2005. Bayesian Elo Rating. https://www.remi-coulom.fr/Bayesian-Elo.

[23] J.S. Cramer. 2002. *The origins of logistic regression*. Technical Report. 167–178 pages.

[24] Douglas E Critchlow, Michael A Fligner, and Joseph S Verducci. 1991. Probability models on rankings. *Journal of mathematical psychology* 35, 3 (Sept. 1991), 294–318.

[25] Marco Cuturi. 2013. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in Neural Information Processing Systems* 26 (2013).

[26] Marco Cuturi and Mathieu Blondel. 2017. Soft-DTW: a differentiable loss function for time-series. In *International Conference on Machine Learning*. 894–903.

[27] Marquis de Condorcet. 1785. Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix.

[28] Persi Diaconis. 1988. *Group representations in probability and statistics*. Lecture Notes-Monograph Series, Vol. 11. Institute of Mathematical Statistics. i–192 pages.

[29] P. Diaconis and R. Graham. 1977. Spearman's footrule as a measure of disarray. *Royal Statistical Society. Series B (Methodological)* 39, 2 (1977), 262–268.

[30] Benjamin Dubois-Taine, Francis Bach, Quentin Berthet, and Adrien Taylor. 2022. Fast Stochastic Composite Minimization and an Accelerated Frank-Wolfe Algorithm under Parallelization. In *Advances in Neural Information Processing Systems*.

[31] Cynthia Dwork, Ravi Kumar, Moni Naor, and D. Sivakumar. 2001. Rank aggregation methods for the Web. In *Proceedings of the 10th international conference on World Wide Web*. 613–622.

[32] Arpad E. Elo. 1978. *The Ratings of Chess Players, Past and Present* (2nd ed.). Arco Publishing, Inc.

[33] Meta Fundamental AI Research Diplomacy Team (FAIR)†, Anton Bakhtin, Noam Brown, Emily Dinan, Gabriele Farina, Colin Flaherty, Daniel Fried, Andrew Goff, Jonathan Gray, Hengyuan Hu, Athul Paul Jacob, Mojtaba Komeili, Karthik Konath, Minae Kwon, Adam Lerer, Mike Lewis, Alexander H. Miller, Sasha Mitts, Adithya Renduchintala, Stephen Roller, Dirk Rowe, Weiyan Shi, Joe Spisak, Alexander Wei, David Wu, Hugh Zhang, and Markus Zijlstra. 2022. Human-level play in the game of Diplomacy by combining language models with strategic reasoning. *Science* 378, 6624 (2022), 1067–1074. https://doi.org/10.1126/science.ade9097 arXiv:https://www.science.org/doi/pdf/10.1126/science.ade9097

[34] M A Fligner and J S Verducci. 1986. Distance Based Ranking Models. *Journal of the Royal Statistical Society. Series B, Statistical methodology* 48, 3 (1986), 359–369.

[35] Emil Julius Gumbel. 1954. *Statistical Theory of Extreme Values and some Practical Applications: A Series of Lectures*. Number 33. US Govt. Print. Office.

[36] Elad Hazan. 2015. *Introduction to online convex optimization*. Now Publishers Inc. Foundations and Trends in Optimization, Volume 2, Issue 3-4.

[37] Tamir Hazan, George Papandreou, and Daniel Tarlow. 2016. *Perturbations, Optimization, and Statistics*. MIT Press.

[38] Ralf Herbrich, Tom Minka, and Thore Graepel. 2006. TrueSkill™: A Bayesian Skill Rating System. In *Advances in Neural Information Processing Systems*, B. Schölkopf, J. Platt, and T. Hoffman (Eds.), Vol. 19. MIT Press.

[39] David R. Hunter. 2004. MM Algorithms for Generalized Bradley-Terry Models. *The Annals of Statistics* 32, 1 (2004), 384–406.

[40] M. Karpinski and W. Schudy. 2010. Faster Algorithms for Feedback Arc Set Tournament, Kemeny Rank Aggregation and Betweenness Tournament. *ISAAC* 6506 (2010), 3–14.

[41] Markelle Kelly, Rachel Longjohn, and Kolby Nottingham. [n.d.]. The UCI Machine Learning Repository. https://archive.ics.uci.edu.

[42] J. Kemeny. 1959. Mathematics Without Numbers. *Daedalus* 88, 4 (1959), 577–591.

[43] Claire Kenyon-Mathieu and Warren Schudy. 2007. How to rank with few errors. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing* (San Diego, California, USA) *(STOC '07)*. Association for Computing Machinery, New York, NY, USA, 95–103. https://doi.org/10.1145/1250790.1250806

[44] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems*. 1097–1105.

[45] Abhinav Kumar, Garrick Brazil, and Xiaoming Liu. 2021. Groomed-nms: Grouped mathematically differentiable nms for monocular 3d object detection. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*. 8973–8983.

[46] Marc Lanctot, Kate Larson, Yoram Bachrach, Luke Marris, Zun Li, Avishkar Bhoopchand, Thomas Anthony, Brian Tanner, and Anna Koop. 2023. Evaluating Agents using Social Choice Theory. arXiv:2312.03121 [cs.AI]

[47] Marc Lanctot, Kate Larson, Michael Kaisers, Quentin Berthet, Ian Gemp, Manfred Diaz, Roberto-Rafael Maura-Rivero, Yoram Bachrach, Anna Koop, and Doina Precup. 2024. Soft Condorcet Optimization for Ranking of General Agents. arXiv:2411.00119 [cs.MA] https://arxiv.org/abs/2411.00119

[48] Quentin Le Lidec, Ivan Laptev, Cordelia Schmid, and Justin Carpentier. 2021. Differentiable rendering with perturbed optimizers. *Advances in Neural Information Processing Systems* 34 (2021), 20398–20409.

[49] Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher Ré, Diana Acosta-Navas, Drew A. Hudson,

Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue Wang, Keshav Santhanam, Laurel Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. 2022. Holistic Evaluation of Language Models. *CoRR* abs/2211.09110 (2022). arXiv:2211.09110 [cs.CL] See latest results at: https://crfm.stanford.edu/helm/latest/.

[50] Tie-Yan Liu. 2011. *Learning to Rank for Information Retrieval.* Springer.

[51] Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. 2023. AgentBench: Evaluating LLMs as Agents. arXiv:2308.03688 [cs.AI]

[52] Felipe Llinares-López, Quentin Berthet, Mathieu Blondel, Olivier Teboul, and Jean-Philippe Vert. 2023. Deep embedding and alignment of protein sequences. *Nature Methods* 20, 1 (2023), 104–111.

[53] Stephan Mandt, Matthew Hoffman, and David Blei. 2016. A variational analysis of stochastic gradient algorithms. In *International conference on machine learning.* PMLR, 354–363.

[54] John I Marden. 1995. *Analyzing and Modeling Rank Data.* Chapman and Hall.

[55] Nicholas Mattei and Toby Walsh. 2013. PrefLib: A Library of Preference Data HTTP://PREFLIB.ORG. In *Proceedings of the 3rd International Conference on Algorithmic Decision Theory (ADT 2013) (Lecture Notes in Artificial Intelligence).* Springer.

[56] Roberto-Rafael Maura-Rivero, Marc Lanctot, Francesco Visin, and Kate Larson. 2025. Jackpot! Alignment as a Maximal Lottery. arXiv:2501.19266 [cs.AI] https://arxiv.org/abs/2501.19266

[57] Arthur Mensch and Mathieu Blondel. 2018. Differentiable dynamic programming for structured prediction and attention. In *Proc. of ICML.*

[58] Steven Morse. 2019. Elo as a Statistical Learning Model. https://stmorse.github.io/journal/Elo.html. Retrieved January 2024.

[59] Kevin P. Murphy. 2022. *Probabilistic Machine Learning: An introduction.* MIT Press. probml.ai

[60] Max Paulus, Dami Choi, Daniel Tarlow, Andreas Krause, and Chris J Maddison. 2020. Gradient estimation with stochastic softmax tricks. *Advances in Neural Information Processing Systems* 33 (2020), 5691–5704.

[61] Maria Silvia Pini, Francesca Rossi, Kristen Brent Venable, and Toby Walsh. 2011. Possible and necessary winners in voting trees: majority graphs vs. profiles. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 1.* 311–318.

[62] Arun Rajkumar, Suprovat Ghoshal, Lek-Heng Lim, and Shivani Agarwal. 2015. Ranking from Stochastic Pairwise Preferences: Recovering Condorcet Winners and Tournament Solution Sets at the Top. In *Proceedings of the 32nd International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 37).* 665–673.

[63] Noelia Rico, Camino R. Vela, and Irene Diaz. 2023. Kemeny ranking aggregation meets the GPU. *Journal of Supercomputing* 79 (2023), 10335–10352.

[64] Noelia Rico, Camino R. Vela, and Irene Diaz. 2023. Reducing the time required to find the Kemeny ranking by exploiting a necessary condition for being a winner. *European Journal of Operational Research* 305, 3 (2023).

[65] Mark Rofin, Vladislav Mikhailov, Mikhail Florinsky, Andrey Kravchenko, Tatiana Shavrina, Elena Tutubalina, Daniel Karabekyan, and Ekaterina Artemova. 2023. Vote'n'Rank: Revision of Benchmarking with Social Choice Theory. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, Andreas Vlachos and Isabelle Augenstein (Eds.). Association for Computational Linguistics, Dubrovnik, Croatia, 670–686. https://doi.org/10.18653/v1/2023.eacl-main.48

[66] Arthur L. Samuel. 1959. Some Studies in Machine Learning Using the Game of Checkers. *IBM Journal of Research and Development* 44 (1959), 206–226.

[67] Michael E Sander, Joan Puigcerver, Josip Djolonga, Gabriel Peyré, and Mathieu Blondel. 2023. Fast, Differentiable and Sparse Top-k: a Convex Analysis Perspective. *arXiv preprint arXiv:2302.01425* (2023).

[68] Nihar B Shah and Martin J Wainwright. 2018. Simple, Robust and Optimal Ranking from Pairwise Comparisons. *Journal of machine learning research: JMLR* 18, 199 (2018), 1–38.

[69] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature* 529 (2016), 484–489.

[70] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. 2018. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science* 632, 6419 (2018), 1140–1144.

[71] J. H. Smith. 1973. Aggregation of preferences with variable electorate. *Econometrica* 41 (1973), 1027–1041.

[72] Mandt Stephan, Matthew D Hoffman, David M Blei, et al. 2017. Stochastic gradient descent as approximate bayesian inference. *Journal of Machine Learning Research* 18, 134 (2017), 1–35.

[73] Lawrence Stewart, Francis S Bach, Felipe Llinares López, and Quentin Berthet. 2022. Differentiable Clustering with Perturbed Spanning Forests, In Advances in Neural Information Processing Systems. *arXiv preprint arXiv:2305.16358.*

[74] Gokul Swamy, Christoph Dann, Rahul Kidambi, Zhiwei Steven Wu, and Alekh Agarwal. 2024. A Minimaximalist Approach to Reinforcement Learning from Human Feedback.

[75] Aamir Syed, Omar; Syed. [n.d.]. Arimaa – a New Game Designed to be Difficult for Computers. *International Computer Games Association Journal* 26 ([n. d.]), 138–139.

[76] Gerald Tesauro. 1995. Temporal Difference Learning and TD-Gammon. *Commun. ACM* 38, 3 (1995), 58–68.

[77] Madaleine Udell. 2020. SigmoidalProgramming (Julia package). https://github.com/madeleineudell/SigmoidalProgramming.jl. Retrieved August 2024.

[78] Madeleine Udell and Stephen P. Boyd. 2014. Maximizing a Sum of Sigmoids. https://api.semanticscholar.org/CorpusID:18061910

[79] Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander S. Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom L. Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature* 575, 7782 (2019), 350–354. https://doi.org/10.1038/s41586-019-1724-z

[80] Marin Vlastelica, Anselm Paulus, Vít Musil, Georg Martius, and Michal Rolínek. 2019. Differentiation of blackbox combinatorial solvers. *arXiv preprint arXiv:1912.02175* (2019).

[81] Lirong Xia. 2019. *Learning and Decision-Making from Rank Data.* Springer.

[82] Lirong Xia and Vincent Conitzer. 2011. Determining possible and necessary winners given partial orders. *Journal of Artificial Intelligence Research* 41 (2011), 25–67.

[83] H. P. Young and A. Levenglick. 1978. A consistent extension of Condorcet's election principle. *SIAM Journal of Applied Mathematics* 35, 2 (1978), 285–300.

[84] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. arXiv:2306.05685 [cs.CL] See also https://lmsys.org/blog/2023-05-03-arena/ and https://huggingface.co/spaces/lmsys/chatbot-arena-leaderboard. Accessed August 13th, 2023.