Learning How to Vote with Principles

Axiomatic Insights Into the Collective Decisions of Neural Networks

Levin Hornischer and Zoi Terzopoulou

Abstract

Can neural networks be applied in voting theory, while satisfying the need for transparency in collective decisions? We propose *axiomatic deep voting*: a framework to build and evaluate neural networks that aggregate preferences, using the well-established axiomatic method of voting theory. Our findings are: (1) Neural networks, despite being highly accurate, often fail to align with the core axioms of voting rules, revealing a disconnect between mimicking outcomes and reasoning. (2) Training with axiom-specific data does not enhance alignment with those axioms. (3) By solely optimizing axiom satisfaction, neural networks can synthesize new voting rules that often surpass and substantially differ from existing ones. This offers insights for both fields: For AI, important concepts like bias and value-alignment are studied in a mathematically rigorous way; for voting theory, new areas of the space of voting rules are explored.[1]

1 Introduction

Artificial intelligence (AI) is increasingly applied in many domains, including not just scientific and technological but also societal problems. This poses a dilemma when it comes to *social choice*, i.e., voting, preference aggregation, and other processes of collective decisions [11]. On the one hand, voting systems should be transparent, but the neural networks on which modern AI is built are notoriously opaque. On the other hand, neural networks could unearth novel and tailor-made collective decision procedures. Already, state-of-the-art techniques for alignment of Large Language Models (LLMs) with human values—like RLHF [5] or DPO [53]—rely on the aggregation of human preferences about the generated outputs. This triggered recent research in guiding such AI alignment using social choice [16].

In this paper, we study how neural networks aggregate preferences. When they form collective decisions, do they adhere to the normative principles that social choice theory formulates as axioms? This offers new insights for both AI and voting theory. For AI, this provides a rich testing ground to study pressing machine learning concepts like bias, value-alignment and interpretability in a mathematically rigorous way. For example, a network is not biased towards specific individuals if it aggregates their preferences in accordance with the axiom of anonymity; the so-called Pareto principle requires the neural network to align with any preference shared among all individuals; and the well-known axiom of independence entails a certain compositional interpretability of the network. For voting theory, axiomatic deep voting provides a new method for the central quest of exploring the space of voting rules.

Formally, a *voting rule* is a function that takes as input a *profile*—i.e., a list of each individual's preferences among a set of alternatives—and produces as output a *collective decision*, i.e., the alternative(s) that the rule takes to be most preferred for the group as a whole. A straightforward rule is *Plurality* (picking the alternative that is considered best by the most individuals); other classic rules include *Borda* and *Copeland*, while a more recent suggestion is *Stable Voting*. To study the collective decisions of neural networks, we develop the *axiomatic deep voting* framework.[2] Deep neural networks are (parametrized) functions that map vectors (typically of a high dimension) to vectors (typically of a low dimension). So, after suitably *encoding* profiles and collective decisions as vectors, neural networks realize voting

[1] After submission, the paper got accepted and published as [34] (COMSOC 2025 does not preclude such publications).
[2] The source code is available here: https://github.com/LevinHornischer/AxiomaticDeepVoting.

rules, i.e., functions from profiles to collective decisions. Discovering a voting rule can be seen as an *optimization problem*: updating the neural network parameters until a given desired property is fulfilled. We *evaluate* neural networks in terms of accuracy and axiom satisfaction. While the former is standard in machine learning, the latter is specific to voting theory and its *axiomatic method* [55, 38]: as already mentioned, different axioms describe different desirable properties of voting rules.

Research questions. We investigate how neural networks aggregate preferences via three questions.

(1) *Correct for the right reasons?* Neural networks can accurately learn standard voting rules, but do they adhere to the normative principles expressed by voting axioms?

We observe eminent violations of the axioms, so we focus on teaching neural networks the expert knowledge expressed by them, in two common ways. First, using *dataset augmentation* [56]:

(2) *Learning principles by example?* Can neural networks be trained to adhere to voting axioms by training with data exemplifying the axioms?

The second way is using *semantic loss functions* [57]. For this, we develop a translation of the axioms into loss functions; so, by optimizing this loss during training, the network increases the corresponding axiom satisfaction. Importantly, though, perfect axiom satisfaction is impossible according to the infamous theorem by Arrow [4]. So we search for the best possible axiom satisfaction:

(3) *Rule synthesis guided by principles?* When neural networks optimize axiom satisfaction, can they develop new voting rules that surpass existing ones in axiom satisfaction?

We compare the discovered rules to a wide range of known voting rules, to test if neural networks can advance the current state of the art in voting theory.

Key findings. We answer these questions in three experiments, testing three paradigmatic neural network architectures (multi-layer perceptrons, convolutional neural networks, and word embedding based classifiers) and a variety of standard distributions of voter preferences. We find, respectively:

(1) The employed architectures demonstrate similar behavior both regarding accuracy and axiom satisfaction. Importantly, despite high accuracy, they markedly violate critical axioms like anonymity—yet, the news is not as bad for other axioms.

(2) Data augmentation does not seem to boost the principled learning of neural networks. However, it drastically decreases the amount of required training data.

(3) Neural networks that perform the unsupervised learning task of optimizing axiom satisfaction discover voting rules that are substantially different from existing ones and are comparable—and often better—in axiom satisfaction.

Thus, we combine two approaches: Drawing on machine learning, we use neural networks qua universal function approximators to *explore* the space of voting rules; and drawing on voting theory, we *evaluate* points in that space—i.e., voting rules—by their axiom satisfaction, guiding the exploration.

Related work. We identify three main streams of relevant literature. First, social choice theory has extensively studied the axiomatic satisfaction of voting rules, particularly focusing on manipulability [23, 22, 48] and Condorcet consistency [24, 43, 50]. In line with our findings, the Borda rule is found to

elect the Condorcet winner more often than Plurality [50], and to satisfy independence more frequently than both Copeland and Plurality [19]. Even in simple settings (e.g., 3 voters and 3 alternatives), independence is rarely satisfied by anonymous voting rules [51]. Our work aligns with this tradition of evaluating voting rules based on axioms, but also extends it to learning.

Second, the synergy between voting theory and machine learning has recently garnered increasing attention. Neural networks like MLPs are used to predict outcomes of voting rules across various settings [36, 12], showing high accuracy for simpler rules like Borda, and moderate success for more complex ones. In these studies, MLPs tend to mimic Borda even when trained on other rules, and sample size impacts their behavior. More advanced architectures like Set Transformers and DeepSets improve prediction performance slightly [2]. Our approach differs by evaluating not only the outcome accuracy but also the axiomatic behavior of voting rules learned by neural networks.

Some other works also incorporate axioms directly. Armstrong and Larson [3] use Condorcet consistency to guide learning with real data from Canadian elections.[3] Mohsin et al. [46] explore trade-offs of fairness and efficiency using synthetic data and discover new rules that compete with Plurality and Borda. Similarly, Matone et al. [42] apply MLPs to probabilistic voting and learn voting rules with improved properties. In settings of participatory budgeting, Set Transformers learn existing rules and discover new ones that satisfy axioms of fair representation [21]. Our work builds on these insights by proposing a systematic framework for the axiomatic evaluation of ML-driven voting rules.

Other research directions include learning a rule given examples about its choices [52] and optimizing for social welfare [2]. Holliday et al. [32] find that several voting rules can be manipulated by large MLPs that only have information about pairwise majority victories between alternatives, though some other rules (e.g., Split Cycle) are more resistant. Broader connections to social choice and RLHF are explored by Dai and Fleisig [18], who include fundamental axioms like Condorcet consistency. Considering AI agents as voters, Yang et al. [58] find that LLMs are influenced by the presentation order of the alternatives and demographic stereotypes about the voters they are supposed to mimic, sometimes reducing the diversity of the collective decisions (notably under Borda). However, Gudiño Rosero et al. [27] show that LLMs outperform naive predictors in modelling individual votes in real elections.

Third, when social choice theory is applied to AI alignment, it can model moral aggregation [49], capture fairness limits in AI via Arrow-like results [45], or design RL-based redistribution mechanisms that agree with human preferences [35]. While we do not directly engage in discussions about ethics, we contribute foundational insights into principled decision-making via learned voting rules.

2 Preliminaries

We work in the standard setting of voting theory, where a finite set N of *voters* have preferences that are linear orders (also called *rankings*) over a finite set A of *alternatives* [59]. Set $m := |A|$ and $n := |N|$. We denote by $\boldsymbol{P} = (P_1, ..., P_n)$ a preference *profile*, i.e., a vector with the preference P_i for every voter $i \in N$. This is illustrated in Figure 1.

For a permutation of the alternatives $\sigma : A \to A$, the ranking $\sigma(P)$ is obtained by applying σ elementwise to the ranking P, and $\sigma(\boldsymbol{P}) = (\sigma(P_1), \ldots, \sigma(P_n))$. For a permutation of the voters $\pi : N \to N$, we define $\pi(\boldsymbol{P}) = (P_{\pi(1)}, ..., P_{\pi(n)})$. A *voting rule* is a function F that determines the winning alternatives for each such profile. Formally, $F : \boldsymbol{P} \mapsto S$, where $\emptyset \neq S \subseteq A$.[4]

[3] Recently (after submission), Caiata et al. [13] study the similar setting of committee-selection. They compute, by exhaustive search, data points by minimizing axiom violations in this setting, and train a multi-layer perceptron on this dataset to find a new committee selecting rule. As an anonymous reviewer very helpfully remarks: In their setting, the underlying profile distribution matters, while this is not significantly the case in our rule-learning experiment (Section 4.3), where the models 'directly' optimize axiom satisfaction (not 'indirectly' via a dataset). It would be very interesting to further explore this.

[4] We use the Python package pref-voting in all our experiments [31].

1	2	3	4	5	6
c	d	d	c	a	d
a	b	b	b	b	a
b	c	a	d	c	b
d	a	c	a	d	c

Figure 1: A voting profile, with voters $N = \{1, \dots, 6\}$ and alternatives $A = \{a, b, c, d\}$. Each column depicts the preference of the individual voter; e.g., voter 1 prefers alternative c most, followed by alternative a, etc.

2.1 Voting Rules

Voting rules usually fit into one of two categories: scoring rules and tournament solutions. *Scoring rules* assign a score to each alternative depending on its position in the linear preference of each voter and declare as winners those alternatives with the highest score across all voters. The two primary scoring rules are *Plurality* (assigning score 1 to an alternative each time it is ranked first by a voter, and score 0 otherwise) and *Borda* (assigning score $m-1$ to an alternative ranked first by a voter, score $m-2$ to an alternative ranked second, and so on, until score 0 is assigned to an alternative ranked last by a voter).[5]

Tournament solutions on the other hand are based on tournaments that capture pairwise comparisons between the alternatives, induced by the voters' preferences. For $x, y \in A$, let $N^{\boldsymbol{P}}_{x \succ y}$ be the set of voters i in the profile $\boldsymbol{P}$ that consider x better than y in P_i, and $n^{\boldsymbol{P}}_{x \succ y} := |N^{\boldsymbol{P}}_{x \succ y}|$. A classical tournament solution is the *Copeland* rule, which selects as winners the alternatives that beat the most other alternatives in a pairwise majority contest: $\operatorname{argmax}_{x \in A} |\{y \in A : n^{\boldsymbol{P}}_{x \succ y} \geq n^{\boldsymbol{P}}_{y \succ x}\}|$. To define the recently proposed *Stable Voting* rule [30], we first need to describe—the more computationally expensive, and thus left aside in our analysis—*Split Cycle* [29]. The weighted tournament of a profile is a weighted directed graph the nodes of which are alternatives with an edge from x to y of weight $n^{\boldsymbol{P}}_{x \succ y}$. Suppose that in each cycle of the graph, we simultaneously delete the edges with minimal weight. Then the alternatives with no incoming edges are the winners of Split Cycle. If there is only one Split Cycle winner in a profile $\boldsymbol{P}$, then this also is the winner of Stable Voting; otherwise x is a winner of Stable Voting if for some alternative y it holds that x is a Split Cycle winner with the maximal margin $n^{\boldsymbol{P}}_{x \succ y}$ such that x is a Stable Voting winner in the profile $\boldsymbol{P}_{-y}$ obtained from $\boldsymbol{P}$ after deleting alternative y.

The aforementioned rules and several others are included in our last experiment that compares a wide pool of voting rules that vary in nature. Our remaining two experiments are focused on the three most standard voting rules, Plurality, Borda, and Copeland, which are the most frequently studied in the literature to date at the intersection of voting and machine learning. For detailed definitions of all the rules, we refer to the introductory chapter of Zwicker [59] and to the `pref-voting` documentation.

2.2 Data Generation

A line of work called 'map of elections' within computational social choice attempts to systematize simulation experiments relying on synthetic voting data [7, 8, 9]. In this vein, we aim to ensure that our results are independent of the specific choice of the distribution of preference profiles. We employ four representative distributions that cover elections of different nature, and, since our experimental results are robust across those distributions, additional ones are not expected to provide new insights.

Impartial Culture (IC) assumes that all preference profiles have the same probability of appearing. Each preference of a voter in a profile is sampled uniformly at random. The *Mallows* distribution [41] fixes a reference ranking P and assumes that each voter's preference is close to that ranking. Closeness to the reference ranking is defined using the Kendall Tau distance, parameterized by a dispersion

[5] Plurality and Borda are often contrasted in voting [28, 54].

parameter $\phi \in (0, 1]$.[6] We use a parameter rel-ϕ (randomly generated) that, together with the number of alternatives, determines the value of the dispersion parameter ϕ, following Boehmer et al. [7, 8]: this methodology generates data that more closely resemble those of real elections. In Appendix A, we also define the *2D-Euclidean* and the *Urn* distribution.

We work with $n_{\max} = 77$ and $m_{\max} = 7$ in the first experiment and $n_{\max} = 55$ and $m_{\max} = 5$ in the other experiments. The first experiment does not show a qualitative difference between these settings, but the latter is computationally more efficient. To generate our synthetic data, we sample preference profiles according to a given distribution and produce their corresponding winning sets according to a known voting rule.

2.3 Axioms

We define axioms as functions that map a voting rule and a preference profile to a value in $\{-1, 1, 0\}$, where 0 means that the axiom is not applicable, -1 means that the axiom is violated, and 1 that it is satisfied. The *satisfaction degree* of a rule with respect to a given axiom is the ratio of the number of sampled profiles in which the axiom is satisfied to the number of sampled profiles in which it is applicable. We focus on axioms that capture basic and diverse normative properties of a voting rule F.

- *Anonymity* is always applicable; it is satisfied in $\boldsymbol{P}$ if for all permutations of voters $\pi : N \to N$, $F(\pi(\boldsymbol{P})) = F(\boldsymbol{P})$. In words, the winners should be invariant under permutations of the voters.
- *Neutrality* is always applicable; it is satisfied in $\boldsymbol{P}$ if for all permutations of alternatives $\sigma : A \to A$, $F(\sigma(\boldsymbol{P})) = \sigma(F(\boldsymbol{P}))$. In words, under permutations of the alternatives, the winners should be permuted respectively.
- *Condorcet principle* is applicable in $\boldsymbol{P}$ if some $x \in A$ is such that $n^{\boldsymbol{P}}_{x \succ y} > n/2$ for all $y \in A \setminus \{x\}$; it is satisfied if $F(\boldsymbol{P}) = \{x\}$. In words, if a Condorcet winner exists, then it should be the unique winner of the voting rule.
- *Pareto principle* is applicable in $\boldsymbol{P}$ if there exist two alternatives $x, y \in A$ such that $n^{\boldsymbol{P}}_{x \succ y} = n$; it is satisfied if $y \notin F(\boldsymbol{P})$. In words, if an alternative is considered inferior to a certain other alternative by all voters, then it should not win.
- *Independence* is applicable in $\boldsymbol{P}$ if $F(\boldsymbol{P}) \neq A$; it is satisfied if for all $x \in F(\boldsymbol{P})$, $y \notin F(\boldsymbol{P})$, and $\boldsymbol{P'}$ such that $N^{\boldsymbol{P}}_{x \succ y} = N^{\boldsymbol{P'}}_{x \succ y}$, it holds that $y \notin F(\boldsymbol{P'})$. In words, if the relative ranking between a winning alternative and a losing alternative remains the same for all voters, then the losing alternative should not win.

All voting rules we defined satisfy anonymity and neutrality, as well as the Pareto principle, for all preference profiles. They all violate independence for some preference profile. Several of them such as Copeland and Stable voting always satisfy the Condorcet principle (but no scoring rule does) .

To evaluate the axiom satisfaction of a voting rule, we sample 400 profiles on which the axioms are applicable. We use the same profile distribution μ as was used for training the neural network, and we again randomly choose integers $n \in [1, n_{\max}]$ and $m \in [1, m_{\max}]$ before μ-sampling a profile with n voters and m alternatives. To compute whether an axiom is satisfied for a profile, the axioms of anonymity, neutrality, and independence require sampling of permutations. We sample, per profile, 50, 50, and $w(m-w)256$ permutations, respectively (where w is the number of winners according to the rule on the profile, and hence $m - w$ is the number of losers).[7]

[6] The *Kendall Tau distance* between two rankings P and Q over the same set of alternatives is the number of pairs of alternatives (a, b) such that a is preferred over b in P but not so in Q.

[7] Independence requires permutations of voters and of alternatives, hence we sample more permutations of the profile.

3 Method

The *axiomatic deep voting* framework is built around a neural network, which is a function $f_w : \mathbb{R}^i \to \mathbb{R}^j$ parametrized by wights $w \in \mathbb{R}^k$. We instantiate this with three neural network architectures. Every profile $\boldsymbol{P}$ is mapped, via an *encoding* function e, to a vector $x = e(\boldsymbol{P}) \in \mathbb{R}^i$, for which the neural network produces an output $\hat{y} \in \mathbb{R}^j$, [8] and the *decoding* function d turns this output into a winning set $S = d(\hat{y})$). This realizes the voting rule $F_w\big(\boldsymbol{P}\big) := d\big(f_w(e(\boldsymbol{P}))\big)$. The network is trained using backpropagation with respect to a *loss function* relying on training data. Finally, we *evaluate* the trained network not only with respect to its accuracy (how well it fits the test dataset), but, crucially, also by how much it satisfies the various voting axioms. We calculate the accuracy on a given test set in two ways: *Identity accuracy* is the percentage of pairs $(\boldsymbol{P}, S)$ in the test set for which $F_w(\boldsymbol{P}) = S$. *Subset accuracy* is defined in the same way but considering $F_w(\boldsymbol{P}) \subseteq S$. We calculate the satisfaction degrees for the various axioms of the voting rule F_w realized by the trained neural network as in Section 2.3.

Architectures. We use three paradigmatic neural network architectures from modern machine learning. Section C.1 in the Appendix explains the selected hyperparameters for each architecture. First, *multi-layer perceptrons* (MLPs)—also known as feed-forward neural network—are the classic deep neural net [see, e.g., 26, ch. 6]. They consist of an input layer of neurons, one or more hidden layers, and an output layer. Second, *convolutional neural networks* (CNNs) are a standard architecture to process grid-like input data such as images [see, e.g., 26, ch. 9], and in our case profiles. Third, we devise an architecture that satisfies the anonymity axiom by design: We view profiles as sentences whose words are the rankings. We use the *word embedding* algorithm *Word2vec* [44] to map each ranking to a high-dimensional embedding vector. These vectors are averaged—hence we get anonymity—and an MLP then classifies this average into a winning set. This combined architecture we call here *word embedding classifiers* (WECs).

Encoding and decoding. To ensure our neural networks learn general patterns, we do not work with a fixed number of voters and alternatives, but only with a maximal number of voters $n_{\max}$ and a maximal number of alternatives $m_{\max}$. So the model should allow as input any profile $\boldsymbol{P}$ over the set of voters $N = \{0, \dots, n-1\}$ with $n \leq n_{\max}$ and set of alternatives $M = \{0, \dots, m-1\}$ with $m \leq m_{\max}$. We write a_r^s for the r-th most preferred alternative of voter s, so the profile $\boldsymbol{P}$ is represented as the matrix $(a_r^s)_{r,s}$, whose columns are the rankings as in Figure 1. We write $\tilde{\boldsymbol{P}} = (\tilde{a}_r^s)_{r,s}$ for the result of padding the $m \times n$ matrix $\boldsymbol{P}$ with the symbol $\sim$ to the maximal input dimensions $m_{\max} \times n_{\max}$. So $\tilde{a}_r^s$ is a_r^s if $r \leq m$ and $s \leq n$, and otherwise it is $\sim$.

How should we encode $\tilde{\boldsymbol{P}}$ so it can be inputted to a neural network? The most straightforward way is to read each alternative $a_r^s \in M$ as the number that it is and the padding symbol $\sim$ as, say, -1. Then the matrix $\tilde{\boldsymbol{P}}$ is regarded as a vector of dimension $m_{\max} n_{\max}$. However, this does not perform well, so, following Anil and Bao [2], we represent an alternative not as a number but as a one-hot vector. For $a \in \{0, \dots, m_{\max} - 1\}$, let $\overline{a}$ be the vector of length $m_{\max}$ that is 1 at position a and 0 everywhere else. For the padding symbol, let $\overline{\sim}$ be the vector of length $m_{\max}$ that is 0 everywhere. We write $\overline{\boldsymbol{P}} = (\overline{\tilde{a}_r^s})_{r,s}$.

The *encoding function for MLPs*, e_{MLP}, maps profile $\boldsymbol{P}$ to the vector x obtained by casting the matrix $\overline{\boldsymbol{P}}$ column by column into a flattened vector (of dimension $m_{\max}^2 n_{\max}$). This can then be inputted into the MLP. The *encoding function for CNNs* regards the matrix $\overline{\boldsymbol{P}}$ as a pixel image: the 'pixel' at position (r, s) has the 'color value' $\overline{\tilde{a}_r^s}$. Thus, e_{CNN} maps profile $\boldsymbol{P}$ to the matrix $\overline{\boldsymbol{P}}$ recast as a tensor with dimensions $(channel, height, width) = (m_{\max}, m_{\max}, n_{\max})$. This tensor can then be inputted into the CNN. The *encoding function for WECs* regards the profile $\boldsymbol{P} = (P_1, \dots, P_n)$ as a sentence with words P_i. We train it to embed these words into vectors of a fixed high dimension. Thus, unlike the previous encoding

[8] For our third architecture, the encoding function is part of the neural network, while for the first two it is independent; hence we treat e as a separate entity here.

functions, this one is not separate from the neural network but rather forms the first layer of the WEC, with the remaining layers processing the embedding vectors. More precisely, we first pre-train the embeddings as follows. For a given corpus size c, we sample c-many profiles from a given distribution of profiles (e.g., IC) to form our corpus (i.e., a set of sentences). The rankings occurring in the profiles form the vocabulary of this corpus, to which we add the `unk` token (to later represent *unknown* rankings, i.e., rankings that are not in the vocabulary) and the `pad` token (to *pad* a profile to length $n_{\max}$). Due to the `unk` token, this encoding applies to *all* profiles, even if it contains rankings that are not part of the model's vocabulary. Using Word2vec, we train embeddings which represent words in the vocabulary as vectors. When instantiating the WEC architecture, these embeddings form the first layer: it maps the profile $(P_1, \dots, P_n)$ to the corresponding embedding vectors $(v_1, \dots, v_n)$. The next layer averages these vectors into a single vector v, followed by several linear layers ending with the output layer.

Given a profile $\boldsymbol{P}$ as input, all architectures produce as output the logits $\hat{y} = (\hat{y}_0, \dots, \hat{y}_{m_{\max}})$ in $\mathbb{R}^{m_{\max}}$. We apply the sigmoid function sig elementwise to obtain the probability that alternative r is in the winning set and define the decoding function $d_m(\hat{y}) := \{r \in \{0, \dots, m\} : \mathsf{sig}(\hat{y}_r) > 0.5\}$.[9]

Loss functions. Since multiple alternatives can win, we cast the task of finding a voting rule as a *multi-label classification* problem. Each input profile $\boldsymbol{P}$ is associated with m binary labels (where m is the number of alternatives in $\boldsymbol{P}$), and the r-th label is 1 if and only if the r-th alternative is in the winning set associated with $\boldsymbol{P}$. Hence we use *binary cross entropy* as loss function. A main contribution of this paper is that, for each axiom, we also design a loss function that enforces satisfaction of that axiom. So, for each axiom ax, we define a function $L_{\mathsf{ax}}(f_w, \boldsymbol{P})$ that takes as input the function f_w computed by the neural network with weights w and a profile $\boldsymbol{P}$. It outputs a non-negative real number describing numerically how much the axiom is satisfied: 0 means perfect axiom satisfaction, while higher numbers mean worse axiom satisfaction. For example, for *anonymity*, uniformly sample N-many permutations $\pi_1, \dots, \pi_N$ of the set of voters of $\boldsymbol{P}$ and define

$$L_A(f_w, \boldsymbol{P}) := \frac{1}{N} \sum_{r=1}^{N} \mathrm{KL}\Big(f_w\big(e(\boldsymbol{P})\big), f_w\big(e(\pi_r(\boldsymbol{P}))\big)\Big),$$

where KL is Kullback–Leibler divergence. The loss functions for the other axioms are in Appendix B.

4 Results and Analysis

We discuss the three experiments that provide answers to our three main research questions.

4.1 Experiment 1: Correct for the Right Reasons?

Recent papers address neural networks learning voting rules [2, 12], without asking if "the system performs well for the right reasons" [6, p. 5192]. Rather, we use axioms to focus on principled learning.

Design. We train each neural network architecture (MLP, CNN, and WEC) on data from each basic voting rule (Plurality, Borda, and Copeland), using four different sampling distributions (IC, Urn, Mallows, and Euclidean). We report the results as *relative* accuracy and axiom satisfaction, i.e., ⟨relative evaluation⟩ = ⟨rule evaluation⟩ − ⟨model evaluation⟩. For example, if the model has 95% accuracy, then, since the rule trivially has 100% accuracy, there is a relative accuracy *loss* of 100% − 95% = 5%. If the model has 35% satisfaction of an axiom and the rule only 30%, then the relative axiom satisfaction is 30% − 35% = −5%, so there is a relative *gain* of 5%.

[9] A priori, it can happen that the neural network does not assign any winner, in contrast to our definition of a voting rule. We check (and train) that this happens, if at all, only with a negligible probability.

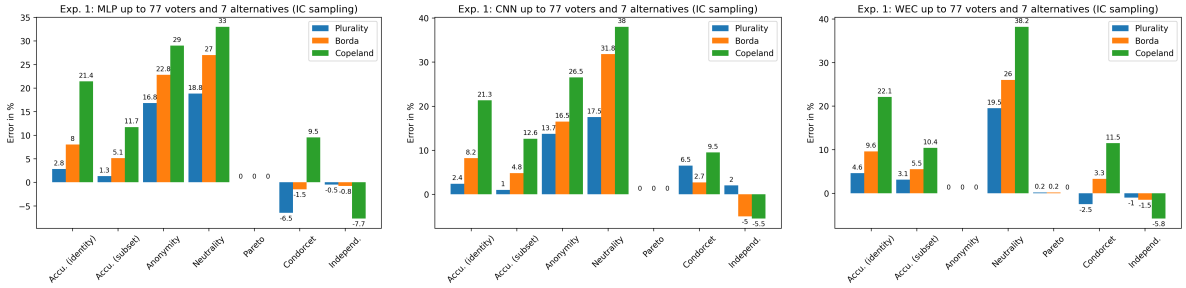

Figure 2: Training the three architectures (MLP, CNN, and WEC) on data from Plurality, Borda, and Copeland (the three bars in each plot) with IC samples and comparing the errors in both accuracy and axiom satisfaction. The error describes the rule's evaluation minus the model's evaluation. Intuitively, 2.8% accuracy error means 2.8% loss in accuracy: the rule by definition is correct so it has 100% accuracy, but the model obtains only 97.2% accuracy; similarly, -6.5% Condorcet error means 6.5% gain in accuracy: the rule has 73.25% Condorcet satisfaction, but the model obtains 79.75% Condorcet satisfaction.

Results. Figure 2 presents the relative evaluations with IC sampling. (The otehr distributions are in Appendix D, as they do not yield significantly different results.) The three architectures do not differ much in accuracy. The best (resp. worst) accuracy is achieved for the simple Plurality rule (resp. the complex Copeland rule). Across all voting rules, architectures, and distributions, we see large losses in neutrality despite only low losses in accuracy (e.g., 4.6% relative identity-accuracy loss but 19.5% relative neutrality loss for the WEC architecture when trained on Plurality). Large anonymity losses are also observed under the MLP and CNN architectures. This is particularly noteworthy since anonymity and neutrality are 100% satisfied by the given voting rules. The MLP and CNN models show larger neutrality than anonymity losses (with the models trained on Plurality demonstrating the smallest such difference). Regarding the other axioms, all models adhere perfectly to Pareto, in accordance with the voting rules on which they are trained. The MLP and WEC models trained on Plurality seem to exhibit relative Condorcet gains, but Condorcet losses are found for the CNN model. The MLP model trained on Borda obtains relative Condorcet gains, but this is not the case for CNN and WEC. Since Copeland always satisfies the Condorcet principle, there is a relative small Condorcet loss for all models. The MLP and WEC models trained on Plurality and Borda, as well as the CNN model trained on Plurality, satisfy independence to a similar degree as the rules on which they are trained. All models trained on Copeland exhibit relative independence gains, and the same holds for the CNN model trained on Borda.

Discussion. The simplicity of Plurality is probably the reason behind its advanced learnability. Notably, the models take a stance on the well-documented tension between anonymity and neutrality:[10] they tend to favor outcomes that align more closely with the former than with the latter. As the architectures are not invariant to permutations of the input data, the severe violations of neutrality (and of anonymity for MLPs and CNNs) are not *a priori* surprising. What is surprising is that the violations persist even for high accuracy with respect to rules that are perfectly neutral and anonymous.As the architectures are not invariant to permutations of the input data, the severe violations of neutrality (and of anonymity for MLPs and CNNs) are not *a priori* surprising. However, these violations persist even for high accuracy with respect to rules that are perfectly neutral and anonymous.[11] Overall, this experiment highlights the importance of the *reasons* behind automated decision-making. Outcomes that mimic well-defined voting rules are arguably still unreliable, since they do not come with a guarantee

[10]No voting rule that always elects a single winner can simultaneously be anonymous and neutral.

[11]To elaborate: On the one hand, these results are surprisingly bad. Given the high accuracy, we may expect that the neural network should have 'gotten the idea' of the voting rule, and hence of its anonymity and neutrality. On the other hand, for anonymity and, respectively, neutrality to be satisfied on a given profile, we require the neural network to output the correct answer on 50 permutations of the profile, while accuracy requires being correct only on that very profile. So struggles with anonymity and neutrality are not surprising. However, surprising or not, the results stay the same: we do *not* have high certainty (at least one failure in 50 checks) that the neural network outputs the desired answer under permutations.

of respecting the principles on which those rules are built.

4.2 Experiment 2: Learning Principles by Example?

A natural approach to integrating expert knowledge in neural networks is data augmentation. In the voting context, this was proposed by Xia [56] but has not been tested in practice, to the best of our knowledge. We focus on the anonymity and neutrality axioms since they were violated most in our first experiment. We also test the effects of data augmentation on the model's accuracy. Asking if data augmentation helps can be understood in two ways: First, does training with augmented data increase axiom satisfaction without diminishing accuracy? Second, if so, does it do it better than just training with sampled data points? A 'yes' to the first question improves data efficiency: we get at least as good a model even when only part of the data is 'real' and the rest is augmented. A 'yes' to the second question means that we can actually improve our model's performance.

Design. We test data augmentation in two versions. In the *first version*, we form an initial dataset (i.e., pairs of a sampled profile with corresponding winning set) and train an architecture on it. Using a copy, we continue training either with sampled data points or with the same number of augmented data points obtained from the initial data points by renaming alternatives ('neutrality variations') or voters ('anonymity variations'). If the model trained on augmented data improves axiom satisfaction without worsening accuracy, we get a 'yes' to the first question. If it also is better than the copied model, we get a 'yes' to the second question; otherwise, any improvement only comes from a mere increase in the quantity and not from the quality of data. A potential issue is that, during the continued training with augmented data, the model does not see any further sampled data and hence might lose in accuracy.

The *second version* maks sure that each training batch consists of p percent sampled data points with the remaining data points being neutrality variations (resp., anonymity variations) of those sampled data points. For different choices of p, we then test the models' achieved axiom satisfaction and accuracy. We get a 'yes' to the first (resp., second) question if axiom satisfaction and accuracy are not worse (resp., better) for lower values of p when compared to $p = 100\%$ (i.e., only sampled data).

Results. Results for IC-sampling and neutrality augmentation are exhibited in Figure 3 for the first version of the experiment. (Appendix E presents results for the second version, other distributions, and anonymity augmentation.) It is again apparent that even when the models excel in accuracy, their satisfaction of neutrality and anonymity remains consistently below perfect—this does not change when considering augmented data. Now, we find a 'yes' to the first question but a 'no' to the second: training with augmented data does not substantially hurt accuracy (though in some cases, sometimes it does: e.g., the identity accuracy of the CNN after 1000 gradient steps on top of 500 steps of pretraining is up to 10% lower with augmented data than with sampled data), but it does not reliably improve axiom satisfaction. In Figure 3, augmented data does not seem to be advantageous for neutrality relatively to sampled data (on the contrary, it often seems harmful, e.g., when applying CNN or WEC). In the second version of the experiment that can be found in the Appendix, the ratio p between sampled and augmented data does not seem to correlate with neutrality satisfaction either.

Discussion. Learning voting-theoretic principles by examples—augmented to the training data—does not seem to work for neural networks. However, an advantage of data augmentation is a drastic increase in data efficiency when we only aim for accuracy. This is crucial if we use real election data, where having access to a vast amount of data points is practically impossible. Even when more data is needed to increase the accuracy of network, we could build an appropriate data set based on a limited amount of real data points and then augment it via the neutrality (or anonymity) axiom.

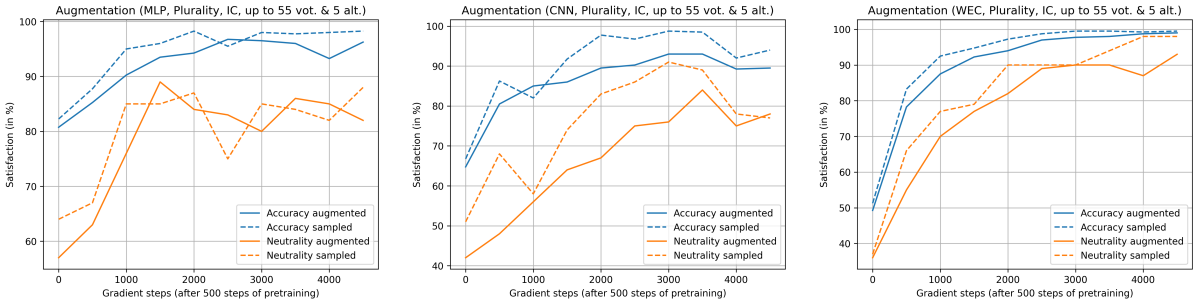

Figure 3: The first version of the experiment 3. Pretraining on IC-sampled data from the Plurality rule, then continue training on neutrality variations of this data (solid line) and comparing it to continued training with sampled data (dashed line), with identity accuracy.

4.3 Experiment 3: Rule Synthesis Guided by Principles?

We saw that neural networks, when trained on data from established voting rules, struggle to vote with principles. But can we directly train neural networks to form principled collective decisions, without relying on any pre-existing voting rules? We are limited by Arrow's Impossibility Theorem [4]: a voting rule (ML-based or not) cannot simultaneously satisfy anonymity, Pareto, and independence. However, how close can we get to full axiom satisfaction? We design an optimization task, using custom loss functions, to guide neural networks in learning novel and principled voting rules.

Design. We train each architecture (MLP, CNN, and WEC) on the loss functions (defined in Section B of the Appendix) that represent the axioms of anonymity, neutrality, Condorcet, Pareto, and independence. Since neural networks could attempt to vacuously satisfy the axioms by proposing no winner, we also consider the "No-winner" loss function, which demands the winning sets to be nonempty. Moreover, by Arrow's Theorem, the axioms cannot be jointly satisfied and will, hence, negatively influence each other. So optimizing for all axioms is not necessarily the best. We pick, for each architecture, a set $\mathcal{O}$ of objectives that we optimize for. For WEC, we choose: no winner, Condorcet, and Pareto. (Appendix F.5 establishes in an ablation study the optimality of this choice.) For MLP and CNN, we add: anonymity and independence. Then the optimization problem is $\operatorname{argmin}_w \sum_{O \in \mathcal{O}} \mathbb{E}_{\mathbf{P} \sim D} \left[L_O(f_w, \mathbf{P}) \right]$, where the loss functions L_O are described in Section 3 and D is the chosen distribution of profiles $\mathbf{P}$. Note that, unlike the previous experiments, this is an unsupervised learning task.

To have a model that is also neutral by design (not just anonymous like the WEC), we add the *neutrality-averaged* decoding [cf. 12]: Given an input profile, we first generate all alternative-permuted versions of the profile, then compute the logits-predictions of the model on each of those permuted profiles (in one batch), next de-permute the predictions again and average all of them, and finally turn those average logits into a winning set with the decoding function used so far. The WEC with neutrality-averaged decoding is anonymous and neutral by design. (We also tested a *neutrality-and-anonymity-averaged* decoding for the other architectures, but the WEC results were consistently superior.)

Results. Table 1 shows the axiom satisfaction of different neural networks (bottom) and of several known voting rules (top), using IC sampling. (Appendix F includes other distributions.) The best ML-based rule in terms of axiom satisfaction is always the neutrality-averaged WEC: it outperforms the classic Plurality, Borda, and Copeland rules in every single axiom, except for a slight loss on Condorcet. Even when we consider more modern rules in voting theory, the neutrality-averaged WEC is competitive: the existing rule with highest axiom satisfaction is Stable Voting and its edge is marginal, with its average axiom satisfaction being less than 1% higher than that of the neutrality-averaged

	Anon.	Neut.	Condorcet	Pareto	Indep.	Average
Plurality	100	100	80.2	100	28.5	81.8
Borda	100	100	95.5	100	37.2	86.5
Anti-Plurality	100	100	74.2	100	24.8	79.8
Copeland	100	100	100	100	28.0	85.6
Llull	100	100	100	100	26.8	85.4
Uncovered Set	100	100	100	100	27.8	85.5
Top Cycle	100	100	100	100	29.0	85.8
Banks	100	100	100	100	27.8	85.5
Stable Voting	100	100	100	100	43.0	88.6
Blacks	100	100	100	100	35.2	87.1
Instant Runoff TB	100	100	94.8	100	28.2	84.6
PluralityWRunoff PUT	100	100	95.0	100	25.5	84.1
Coombs	100	100	96.2	100	34.5	86.2
Baldwin	100	100	100	100	39.2	87.9
Weak Nanson	100	100	100	100	40.0	88.0
Kemeny-Young	100	100	100	100	39.2	87.9
MLP (NW, A, C, P, I)	77.8	75.8	92.5	100	39.5	77.1
CNN (NW, A, C, P, I)	85.2	67.2	92.0	100	39.5	76.8
WEC (NW, C, P)	100	72.5	94.2	100	41.8	81.7
WEC n (NW, C, P)	100	100	96.8	100	41.2	87.6

Table 1: Axiom satisfaction of known rules and ML models. IC sampling. WEC n is the neutrality-averaged WEC. The following letters indicate the axioms on which we optimize: NW–No winner, A–Anonymity, C–Condorcet, P–Pareto, I–Independence. All models have been trained for 15k gradient steps with batch size 200.

WEC.[12] In fact, when averaging five runs of checking axiom satisfaction (which always involves some stochasticity), the neutrality-averaged WEC even comes out better than the rules: see Table 6 in the Appendix. (This is also true for the Euclidean distribution but not for Mallows and Urn.)

We also consider how often the examined rules produce the same outcomes, because similar axiom satisfaction does not imply similar outcomes.[13] Table 2 considers the five closest rules, using IC sampling (again, see Appendix F for the other distributions). The rule discovered by the neutrality-averaged WEC is substantially different from existing ones: it proposes different outcomes than each of them, according to identity accuracy, at least 9.3% of the time (resp., 10.6% for Mallows, 11.1% for Urn, and 7.8% for Euclidean). In comparison, Stable Voting, which was found best in Table 1, disagrees with Borda and Copeland 8.9% of the time and with Weak Nanson and Blacks only 6.6% of the time. Thus, the discovered rule not only is competitive in axiom satisfaction, it also is novel. Figure 4 shows an example of a profile where the winning set provided by the neutrality-averaged WEC differs to all existing voting rules. Note however that even if the results of the model differ from those of existing voting rules, very frequently they do so by only excluding or by only adding certain winning alternatives (see Table 4 in the Appendix). This is not unique to our ML model—it is also the case between known voting rules that exhibit high axiomatic satisfaction: for example, the outcome of Stable Voting is a subset (resp., superset) of the outcome of Weak Nanson 97.9% (resp., 94.35%) of the time.

Discussion. WEC outperforms the other two architectures because it is anonymous by design, so it is enough to use the neutrality-averaged decoding to get a model that is anonymous and neutral, without dealing with the tension between anonymity and neutrality. Moreover, just three optimization objectives were enough to obtain competitive results, while the MLP and CNN models needed to optimize for anonymity and independence as well. The WEC model interestingly had enough *implicit* inductive

[12]Table 5 in the Appendix suggests that more gradient steps do not further improve the results.

[13]For example, the Blacks and Weak Nanson rules are close in average axiom satisfaction (less than 1% difference), but Table 2 shows that more than 8% of the time they propose a different set of winners.

Identity accuracy	WEC n	Blacks	Stable Voting	Borda	Weak Nanson	Copeland
WEC n	100	91	90.5	89.5	88.4	87.7
Blacks		100	95.71	95.13	91.26	90.57
Stable Voting			100	90.84	93.5	91.67
Borda				100	86.39	85.7
Weak Nanson					100	92.43
Copeland						100

Table 2: Similarities between the rules, on 10,000 IC-sampled profiles.

1	2	3	4	5	6	7	8
a	e	d	a	e	b	e	a
b	b	b	c	b	a	a	b
e	d	c	b	c	e	c	d
d	a	e	e	a	c	d	e
c	c	a	d	d	d	b	c

Figure 4: Profile where the WEC outcome disagrees with known voting rules. The winners are: $\{a\}$ for the 'WEC n' (with sigmoids a:.51, b:.49, c:.31, d:.32, e:.43); $\{b\}$ for Blacks, Stable Voting, Borda, Weak Nanson, Copeland; $\{a, e\}$ for Plurality, PluralityWRunoff PUT; $\{e\}$ for Instant Runoff TB, Anti-Plurality; $\{a, b\}$ for Llull, Uncovered Set, Banks, Coombs, Baldwin, and Kemeny-Young; $\{a, b, e\}$ for Top Cycle. The WEC choice is intuitive: alternative a is three times the most preferred and two times the second-most preferred option among eight voters. The sigmoids indicate that alternative b was a close competitor, and would indeed win under many known rules.

bias toward satisfying independence—again highlighting non-trivial interference of the axioms and the network architecture. In summary, ML-driven voting rules derived from axiom optimization beat many classic ones in terms of axiom satisfaction while being comparable to the best rules known today, even if they inherit the opacity of neural networks. This may suggest that existing rules are already close to optimal axiom satisfaction, but since the newly discovered rules are substantially different, they extend the boundaries of what is so far explored in voting theory.

5 Conclusion

Within our axiomatic deep voting framework, we answer the three questions: (1) Are preference-aggregating neural networks correct for the right reasons? No. (2) Can they learn voting-theoretic principles by example? No. (3) Can they synthesize new rules guided by the principles? Yes.

Axiomatic deep voting offers a new tool for the exploration of the space of all voting rules. This provides a promising starting point for studying new, axiom-optimal voting rules, and the influence the axioms exert on each other. The universal approximation theorems [33, 17] ensure that the neural networks are dense in the space of all voting rules, so all areas of that space can be explored with axiomatic deep voting. Moreover, the axiomatic evaluation offers another cautionary tale that accuracy is not everything: Neural networks can have high accuracy without following the right reasons. This changes when we move from the supervised setting of learning rules from examples to the unsupervised setting of directly optimizing axiom satisfaction (translating voting axioms into corresponding loss functions).

Future work could investigate further architectures and axioms, and more options in generating the dataset. For example, we could consider the *extrapolation* task (in which the model sees data where different rules agree and has to find a general rule) or the *interpolation* task (in which the model sees data of different rules and has to find a compromise). Finally, it seems intriguing to bridge notions of explainability in voting theory [14, 47, 10] and in AI [1], in particular to try extracting a symbolic representation (e.g., in logic programming) of the rule that a model learns.

Acknowledgments For very helpful comments and discussions, we would like to thank Ben Armstrong, Balder ten Cate, Timo Freiesleben, Ronald de Haan, Thomas Icard, Alina Leidinger, Christian List, Ignacio Ojea, as well as the audiences at the 'Workshop on Learning and Logic 2025' at the University of Amsterdam and the 'Social Choice for AI Ethics and Safety 2025' workshop at AAMAS in Detroit. We are especially grateful to the COMSOC 2025 anonymous reviewers as well as the JAIR editor Alessandro Farinelli and two anonymous reviewers.

References

[1] A. Adadi and M. Berrada. Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). *IEEE Access*, 6:52138–52160, 2018. doi: 10.1109/ACCESS.2018.2870052.

[2] Cem Anil and Xuchan Bao. Learning to elect. *Advances in Neural Information Processing Systems*, 34:8006–8017, 2021.

[3] Ben Armstrong and Kate Larson. Machine learning to strengthen democracy. In *NeurIPS Joint Workshop on AI for Social Good*, 2019.

[4] Kenneth J Arrow. *Social Choice and Individual Values*. John Wiley & Sons, 1951.

[5] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv*, 2022. arXiv:2204.05862.

[6] Emily M. Bender and Alexander Koller. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198, 2020.

[7] Niclas Boehmer, Robert Bredereck, Piotr Faliszewski, Rolf Niedermeier, and Stanisław Szufa. Putting a compass on the map of elections. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence (IJCAI)*, 2021.

[8] Niclas Boehmer, Piotr Faliszewski, and Sonja Kraiczy. Properties of the mallows model depending on the number of alternatives: A warning for an experimentalist. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 2023.

[9] Niclas Boehmer, Piotr Faliszewski, Łukasz Janeczko, Andrzej Kaczmarczyk, Grzegorz Lisowski, Grzegorz Pierczyński, Simon Rey, Dariusz Stolicki, Stanisław Szufa, and Tomasz Wąs. Guide to numerical experiments on elections in computational social choice. *arXiv*, 2024. arXiv:2402.11765.

[10] Arthur Boixel, Ulle Endriss, and Ronald de Haan. A calculus for computing structured justifications for election outcomes. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI)*, 2022.

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

[12] Dávid Burka, Clemens Puppe, László Szepesváry, and Attila Tasnádi. Voting: A machine learning approach. *European Journal of Operational Research*, 299(3):1003–1017, 2022.

[13] Joshua Caiata, Ben Armstrong, and Kate Larson. Learning to elect a committee. In *The 7th Games, Agents, and Incentives Workshop (GAIW-25)*, Detroit, Michigan, USA, May 2025. Held as part of the Workshops at the 23rd International Conference on Autonomous Agents and Multiagent Systems.

[14] Olivier Cailloux and Ulle Endriss. Arguing about voting rules. In *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2016.

[15] Ioannis Caragiannis and Evi Micha. Learning a ground truth ranking using noisy approval votes. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, 2017.

[16] Vincent Conitzer, Rachel Freedman, Jobst Heitzig, Wesley Holliday, Bob Jacobs, Nathan Lambert, Milan Mossé, Eric Pacuit, Stuart Russell, Hailey Schoelkopf, Emanuel Tewolde, and William Zwicker. Position: Social choice should guide ai alignment in dealing with diverse human feedback. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024.

[17] G. Cybenko. Approximations by superpositions of sigmoidal functions. *Mathematics of Control, Signals, and Systems*, 2(4):303–314, 1989.

[18] Jessica Dai and Eve Fleisig. Mapping social choice theory to RLHF. *arXiv*, 2024. arXiv:2404.13038.
[19] Keith L Dougherty and Jac C Heckelman. The probability of violating arrow's conditions. *European Journal of Political Economy*, 65:101936, 2020.
[20] Florian Eggenberger and George Pólya. Über die statistik verketteter vorgänge. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, 3(4): 279–289, 1923.
[21] Roy Fairstein, Dan Vilenchik, and Kobi Gal. Learning aggregation rules in participatory budgeting: A data-driven approach. *arXiv*, 2024. arXiv:2412.01864.
[22] Pierre Favardin and Dominique Lepelley. Some further results on the manipulability of social choice rules. *Social Choice and Welfare*, pages 485–509, 2006.
[23] Pierre Favardin, Dominique Lepelley, and Jérôme Serais. Borda rule, Copeland method and strategic manipulation. *Review of Economic Design*, 7:213–228, 2002.
[24] Peter C Fishburn and William V Gehrlein. Majority efficiencies for simple voting procedures: Summary and interpretation. *Theory and Decision*, 14(2):141–153, 1982.
[25] Eleonora Giunchiglia, Mihaela Catalina Stoian, and Thomas Lukasiewicz. Deep learning with logical constraints. In Luc De Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 5478–5485. International Joint Conferences on Artificial Intelligence Organization, 7 2022. doi: 10.24963/ijcai.2022/767. URL https://doi.org/10.24963/ijcai.2022/767. Survey Track.
[26] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. The MIT Press, Cambridge, Massachusetts, 2016.
[27] Jairo Gudiño Rosero, Umberto Grandi, and César A. Hidalgo. Large language models (LLMs) as agents for augmented democracy. *arXiv*, 2024. arXiv:2405.03452.
[28] Aleksandar Hatzivelkos. Borda and Plurality comparison with regard to compromise as a sorites paradox. *Interdisciplinary Description of Complex Systems: INDECS*, 16(3B):465–484, 2018.
[29] Wesley Holliday and Eric Pacuit. Split cycle: A new Condorcet-consistent voting method independent of clones and immune to spoilers. *Public Choice*, 197(1):1–62, 2023.
[30] Wesley Holliday and Eric Pacuit. Stable voting. *Constitutional Political Economy*, 34(3):421–433, 2023.
[31] Wesley H Holliday and Eric Pacuit. pref_voting: The preferential voting tools package for python. *Journal of Open Source Software*, 10(105):7020, 2025.
[32] Wesley H. Holliday, Alexander Kristoffersen, and Eric Pacuit. Learning to manipulate under limited information. *arXiv*, 2024. arXiv:2401.16412.
[33] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. doi: https://doi.org/10.1016/0893-6080(89)90020-8.
[34] Levin Hornischer and Zoi Terzopoulou. Learning how to vote with principles: Axiomatic insights into the collective decisions of neural networks. *Journal of Artificial Intelligence Research*, 83(25), 2025. doi: https://doi.org/10.1613/jair.1.18890.
[35] Raphael Koster, Jan Balaguer, Andrea Tacchetti, Ari Weinstein, Tina Zhu, Oliver Hauser, Duncan Williams, Lucy Campbell-Gillingham, Phoebe Thacker, and Matthew Botvinick. Human-centred mechanism design with Democratic AI. *Nature Human Behaviour*, 6(10):1398–1407, 2022.
[36] Hanna Kujawska, Marija Slavkovik, and Jan-Joachim Rückmann. Predicting the winners of borda, kemeny, and dodgson elections with supervised machine learning. In *Multi-Agent Systems and Agreement Technologies Workshop. At the 17th European Conference on Multi-Agent Systems (EUMAS)*, pages 440–458, 2020.
[37] David Lee, Ashish Goel, Tanja Aitamurto, and Helene Landemore. Crowdsourcing for participatory democracies: Efficient elicitation of social choice functions. In *Proceedings of the 2nd AAAI Conference on Human Computation and Crowdsourcing*, 2014.
[38] Christian List. The logical space of democracy. *Philosophy & public affairs*, 39(3):262–297, 2011.
[39] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In

International Conference on Learning Representations (ICLR), 2017.

[40] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

[41] Colin L Mallows. Non-null ranking models. *Biometrika*, 44(1/2):114–130, 1957.

[42] Leonardo Matone, Ben Abramowitz, Nicholas Mattei, and Avinash Balakrishnan. Deepvoting: Learning voting rules with tailored embeddings. *arXiv*, 2024. arXiv:2408.13630.

[43] Samuel Merrill. A comparison of efficiency of multicandidate electoral systems. *American Journal of Political Science*, pages 23–48, 1984.

[44] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[45] Abhilash Mishra. AI alignment and social choice: Fundamental limitations and policy implications. *arXiv*, 2023. arXiv:2310.16048.

[46] Farhad Mohsin, Ao Liu, Pin-Yu Chen, Francesca Rossi, and Lirong Xia. Learning to design fair and private voting rules. *Journal of Artificial Intelligence Research*, 75:1139–1176, 2022.

[47] Oliviero Nardi, Arthur Boixel, and Ulle Endriss. A graph-based algorithm for the automated justification of collective decisions. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2022.

[48] Shmuel Nitzan. The vulnerability of point-voting schemes to preference variation and strategic manipulation. *Public choice*, 47:349–370, 1985.

[49] Ritesh Noothigattu, Snehalkumar Gaikwad, Edmond Awad, Sohan Dsouza, Iyad Rahwan, Pradeep Ravikumar, and Ariel Procaccia. A voting-based system for ethical decision making. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

[50] Hannu Nurmi. Discrepancies in the outcomes resulting from different voting schemes. *Theory and Decision*, 25:193–208, 1988.

[51] Robert C Powers. The number of times an anonymous rule violates independence in the 3×3 case. *Social Choice and Welfare*, 28(3):363–373, 2007.

[52] Ariel D Procaccia, Aviv Zohar, Yoni Peleg, and Jeffrey S Rosenschein. The learnability of voting rules. *Artificial Intelligence*, 173(12-13):1133–1149, 2009.

[53] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024.

[54] Zoi Terzopoulou. Voting with limited energy: A study of Plurality and Borda. *Proceedings of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2023.

[55] William Thomson. On the axiomatic method and its recent applications to game theory and resource allocation. *Social Choice and Welfare*, 18(2):327–386, 2001.

[56] Lirong Xia. Designing social choice mechanisms using machine learning. In *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2013.

[57] Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang, and Guy Broeck. A semantic loss function for deep learning with symbolic knowledge. In *International conference on machine learning (ICML)*, pages 5502–5511, 2018.

[58] Joshua Yang, Damian Dailisan, Marcin Korecki, Carina Hausladen, and Dirk Helbing. Llm voting: Human choices and AI collective decision-making. *arXiv*, 2024. arXiv:2402.01766.

[59] William S Zwicker. Introduction to the theory of voting. In Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D. Procaccia, editors, *Handbook of Computational Social Choice*. Cambridge University Press, 2016.

Levin Hornischer
Munich Center for Mathematical Philosophy, LMU Munich
Munich, Germany
Email: levin.hornischer@lmu.de

Zoi Terzopoulou
GATE, Jean Monnet University, Saint-Etienne School of Economics
Saint-Etienne, France
Email: zoi.terzopoulou@cnrs.fr

A More on distributions

We continuing our discussion of distributions of profiles from Section 2.2. The IC and Mallows distributions are complementary: IC is simplistic and widely employed in theoretical works on voting rules discussed earlier in the literature review; it captures an extreme case with no correlation between preferences of voters. Mallows is often employed in numerical studies of voting rules that use artificial data but wish to capture more realistic voting scenarios [15, 37].

The next two distributions also capture more intricate relationships between the preferences in a profile. According to the *2D-Euclidean* distribution, voters and alternatives are distributed randomly in 2-dimensional Euclidean space, and the closer an alternative is to a voter the more the voter prefers that alternative. Finally, in *Urn* distribution [20] voters randomly draw their ranking from an urn. Initially, the urn includes all possible rankings over the alternatives. After a voter randomly draws from the urn, we add to the urn $\alpha n!$ copies of that ranking for some parameter $\alpha \in [0, \infty)$. When $\alpha = 0$, this reduces to IC. We use the Urn-R distribution [7], where, for each generated profile, α is chosen according to a Gamma distribution with shape parameter $k = 0.8$ and scale parameter $\theta = 1$.

B Specific loss functions

We define the specific loss functions for several basic axioms, which we use in the third experiment. (For more background on semantic loss functions, see, e.g., Xu et al. [57] or Giunchiglia et al. [25].)

Anonymity. Given the network f_w and profile $\boldsymbol{P}$, uniformly sample N-many permutations $\pi_1, \ldots, \pi_N$ of the set of voters of $\boldsymbol{P}$ and define

$$L_A(f_w, \boldsymbol{P}) := \frac{1}{N} \sum_{r=1}^{N} \mathrm{KL}\Big(f_w\big(e(\boldsymbol{P})\big), f_w\big(e(\pi_r(\boldsymbol{P}))\big)\Big),$$

where KL is Kullback–Leibler divergence.[14]

Condorcet. If $\boldsymbol{P}$ has no Condorcet winner, $L_C(f_w, \boldsymbol{P}) := 0$, and otherwise, if that Condorcet winner is alternative a, define (recall $\overline{a}$ is the one-hot vector for alternative a)

$$L_C(f_w, \boldsymbol{P}) := \mathrm{KL}\big(f_w(e(\boldsymbol{P})), \overline{a}\big).$$

Pareto. We define (recall that $n^{\boldsymbol{P}}_{a \succ b} = n$ means that all voters in $\boldsymbol{P}$ rank a above b)

$$L_P(f_w, \boldsymbol{P}) := \sum_{a,b \text{ with } n^{\boldsymbol{P}}_{a \succ b} = n} \mathsf{sig}\Big(f_w(e(\boldsymbol{P}))_b\Big).$$

Independence. Define $L_I(f_w, \boldsymbol{P}) := 0$ if $\boldsymbol{P}$ does not have at least two alternatives. Otherwise, randomly sample N-many pairs (a_r, b_r) of distinct alternatives in $\boldsymbol{P}$ and randomly sample, for each ranking P_k of $\boldsymbol{P} = (P_1, \ldots, P_n)$, a shuffling P'_k of P_k in which, however, the order of a_r and b_r is the same as in P_k, and set $\boldsymbol{P}_r := (P'_1, \ldots, P'_n)$. Write $\hat{y} := f_w(e(\boldsymbol{P}))$ and $\hat{y}^r := f_w(e(\boldsymbol{P}_r))$, and define

$$L_I(f_w, \boldsymbol{P}) := \sum_{r=1}^{N} \mathrm{KL}\Big(\big(\hat{y}_{a_r}\hat{y}_{b_r}\big), \big(\hat{y}^r_{a_r}\hat{y}^r_{b_r}\big)\Big).$$

No winner. Recall that voting rules are required to output at least one winner. This is usually not called an axiom, and we did not hard-code this into our architectures. So we also want to optimize our neural

[14] Though, in principle, other distance/similarity functions can be considered.

networks to align with this requirement. Hence we define the 'no winner' loss as follows. Writing $\hat{y} = f_w(e(\boldsymbol{P}))$, we want that at least one of the numbers in $p := \big(\mathsf{sig}(\hat{y}_1), \dots, \sigma(\hat{y}_m)\big)$ is above 0.5, i.e., the maximum norm $\|p\|_\infty$ should be above 0.5. Hence the more it is below that, the worse the loss:

$$L_{NW}(f_w, \boldsymbol{P}) := \max\big(0.5 - \|p\|_\infty, 0\big).^{15}$$

C Hyperparameter tuning

In this section, we first explain the hyperparameter choices for our models and then investigate different choices to justify the ones we made.

C.1 Hyperparameters

All models use ReLU as the activation function. Our MLP has four hidden layers with 128 neurons each, like those of [2]. The CNN has two convolution layers with kernel size $(5, 1)$ and $(1, 5)$, respectively (and 32 or 64 channels), followed by three linear layers with 128 neurons. Thus, the first kernel can pick up local patterns in the rankings of the voters, while the second kernel can pick up local patterns among the i-th preferred alternatives of the voters. (Appendix C.3 establishes the optimality of this choice when compared to other kernel sizes and additional pre-processing.) The WEC has the word embedding layer, then the averaging layer, and then three linear layers with 128 neurons. For pre-training the word embedding layer with word2vec, we use a corpus size of 10^5, an embedding dimension of 200, and a window size of 7.[16] The corpus size is chosen large enough so that no occurrences of the `unk` token are observed in $1,000$ sampled profiles.

This results in the following numbers of parameters in the setting $n_{\max} = 77$ and $m_{\max} = 7$: $500,487$ (MLP), $1,834,439$ (CNN), and $1,226,143$ (WEC). In the setting $n_{\max} = 55$ and $m_{\max} = 5$ this reduces to: $193,285$ (MLP), $232,165$ (CNN), and $45,585$ (WEC). Thus, the models have roughly comparable capacities. Section C in the Appendix motivates these choices via hyperparameter tuning.

For training, we use the *AdamW* algorithm [40]. We use a batch size of 200. Since we have synthetic data, we do not use epochs and hence only specify the number of gradient steps. In experiments 1, 2, and 3, these are 15,000, 5,000, and 15,000, respectively. Similar to Anil and Bao [2], we use as a learning rate scheduler cosine annealing with warm restarts [39]. All results are reported for one fixed seed. (In the Appendix, Table 3 performs cross validation and Tables 6 and 9 report averaged results across different seeds.) All experiments were run on a laptop without GPU.

C.2 Model sizes

First, regarding the MLPs, Figure 5 tests the performance of different sizes. As mentioned in Section C.1, we use the same sizes for our MLPs as Anil and Bao [2]: four hidden layers with 128 neurons each. To test this choice, we compare it to a smaller MLP with only two hidden layers with 128 neurons each, and to a larger MLP with six hidden layers with 128 neurons each plus layer norm. Figure 5 shows that all three MLPs achieve very similar accuracy. The larger MLP learns a bit more quickly than the other two, but it also has a higher variance in achieved accuracy. Hence the larger MLP does not yield a performance improvement. The results suggest that a smaller MLP might work, too, but, for continuity with the literature on the topic, when then choose their model sizes.

[15] To see almost-everywhere differentiability of the loss functions, use the distributivity of the differential operator over sums, the chain rule, and the almost-everywhere differentiability of the involved functions (KL, sig, max, $\|\cdot\|_\infty$).

[16] That is in the setting $n_{\max} = 77$ and $m_{\max} = 7$. When $n_{\max} = 55$ and $m_{\max} = 5$, we reduce this to a corpus size of 2×10^4, an embedding dimension of 100, and a window size of 5.

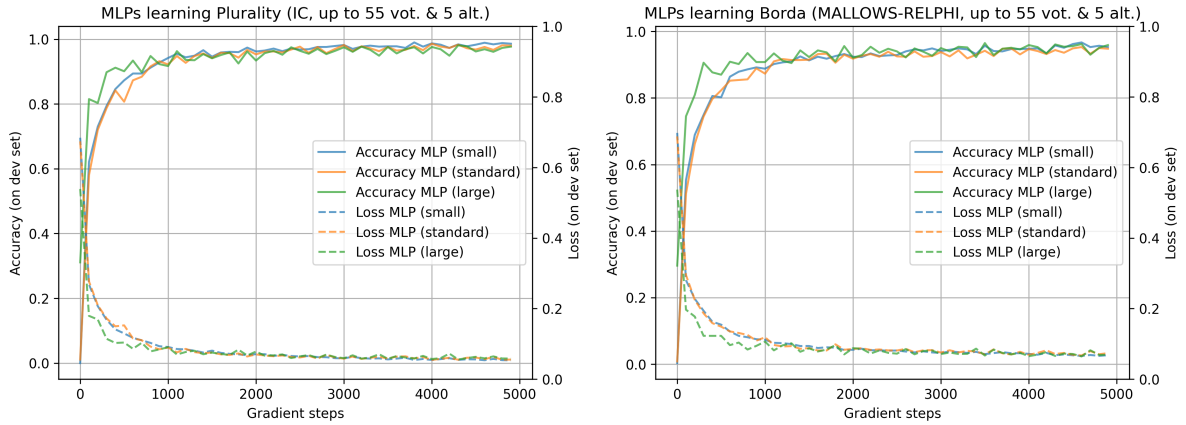

Figure 5: Different sized MLP and their learning performance. The 'small' MLP has two hidden layers with 128 neurons each, the 'standard' MLP has four hidden layers with 128 neurons each (our and the literature's choice), and the 'large' MLP has six hidden layers with 128 neurons each plus layer norm.

Second, regarding the CNNs and the WECs, the choice for the MLP size also dictates their sizes: in order to have a comparable capacity, they should have a roughly similar number of parameters. Indeed, with our choices of numbers and kinds of layers for the CNN and the WEC, we get, as described in Section C.1, models that are roughly comparable in size.

C.3 CNN kernels and pre-processing

Figure 6 investigates different choices for the hyperparameters of the CNN architecture. In Section C.1, we described our choice: the two convolution layers have kernel sizes $(5, 1)$ and $(1, 5)$, respectively. Thus, the first kernel can pick up local patterns in the rankings of the voters, while the second kernel can pick up local patterns among the i-th preferred alternatives of the voters.

In image processing, 'quadratic' kernel sizes—e.g., $(3, 3)$—are more common, to pick up correlations of pixels with their surrounding pixels. In the voting setting, at least conceptually speaking, a quadratic surrounding does not make too much sense: Why should there be important 'diagonal' correlations, say between the i-th preferred alternative of voter k and the $i + 2$-th preferred alternative of voter $k + 3$, especially if the voters should be permutable? On the contrary, vertical and horizontal correlations are important: Vertically, the first kernel captures patterns of correlation between a given alternative in a voter's ranking and more or less preferred alternatives in that ranking; horizontally, the second kernel captures patterns of correlation between the i-th preferred alternative of a voter and the i-th preferred alternative of other voters.

Figure 6 shows that our choice of kernel size (top left) indeed achieves overall better results than a quadratic choice of kernel size (top right). Only for Condorcet and Independence, the quadratic choice is slightly better for some rules.

One might wonder, if one could still leverage diagonal correlations by first reordering the rankings of the voters in a profile so that 'similar' rankings are next to each other, before feeding the profile into the CNN. With such a pre-processing of the input, a quadratic kernel could be used since diagonal comparisons now make sense in such a similarity reordered profile. A standard way to formalize this notion of similarity is via Kendall Tau distance (as defined in footnote 6). We consider two versions of reordering a given profile $\boldsymbol{P} = (P_1, \ldots, P_n)$:

- *Global*: Compute, for $k = 2, \ldots, n$, the Kendall Tau distance d_k between P_1 and P_k. Then reorder the profile starting with P_1 followed by the other rankings with ascending d_k. (In case of a tie, pick the ranking with minimal index first.)

- *Local*: The reordered profile $\boldsymbol{P'} = (P'_1, \ldots, P'_n)$ is computed recursively. Start with $P'_1 := P_1$. Given P'_k, we determine P'_{k+1} as follows. Go through the rankings that have not been picked yet (i.e., $\{P_1, \ldots, P_n\} \setminus \{P'_1, \ldots, P'_k\}$) and compute their Kendall Tau distance to P'_k. Then P'_{k+1} is the ranking among these with the smallest Kendall Tau distance. (Again, tie-break via the indices.)

Figure 6 shows that adding either the global or the local version of Kendall Tau pre-processing overall does not reliably help the performance compared to our chosen setting (neither for our choice of kernel size nor for the quadratic choice). In some cases we do observe an improvement, as for example in the independence axiom satisfaction when learning the Copeland rule—however, this always comes with an additional loss, either in the accuracy or in the satisfaction of other axioms such as anonymity and neutrality.

C.4 Cross validation

Finally, we corroborate our choice of hyperparameters by establishing their robust learning capabilities via cross validation in Table 3. For this, we IC-sample a fixed dataset of $100,000$ data points. We split the dataset into 10 folds (each of size $10,000$). Looping over $k = 0, \ldots, 9$, we take fold k as the test set and train the model on the data in the other 9 folds for 8 epochs. We record the achieved accuracy and loss (both on the training and the test set). Table 3 shows that, for all architectures with their chosen hyperparameters, we always get a high accuracy with little variance. This corroborates the robust learning capabilities of our architectures.

D Experiment 1

We run the 'correct for the right reasons' experiment from Section 4.1 in more settings, reported in Figure 7 (part 1) and Figure 8 (part 2).

E Experiment 2

We add further results to the 'learning principles by example' experiment from Section 4.2. Figure 9 and 10 show different choices of architecture, rules, and distribution for the first version of the experiment. Figure 12, 11, and 13 show different choices of architecture, rules, and distribution for the second version of the experiment.

Regarding the second version and IC sampling (see Figure 11), when $p < 10\%$, i.e., with almost only augmented data, both accuracy and neutrality satisfaction are unsatisfactory, so data augmentation only becomes relevant for $p \geq 10\%$. Here accuracy is stable: it does not vary by more than 5%. In some cases, neutrality is equally stable: for the CNN on all rules and the MLP on Borda (certainly for $p \geq 25\%$, with slightly worse neutrality satisfaction for smaller p). In the remaining non-stable cases, the best neutrality satisfaction is achieved for $p = 100\%$, i.e., without augmented data—with only negligible exceptions.[17] Thus, neither in the stable nor the unstable cases can we see reliable comparative improvements in neutrality satisfaction with more neutrality augmented data.

[17]The only two exceptions are the CNN on Plurality (where neutrality is most satisfied at $p = 75\%$ but to a very similar degree as for $p = 100\%$) and the CNN on Copeland (where neutrality is minimized at $p = 25\%$). Moreover, CNN on Borda and MLP on Copeland have a local—albeit not global—minimum at $p = 25\%$. Thus, while there might be some special cases where neutrality is improved in the highly augmented scenario, this is not enough to consider data augmentation as a successful strategy to improve neutrality satisfaction (which is what we are concerned with here).

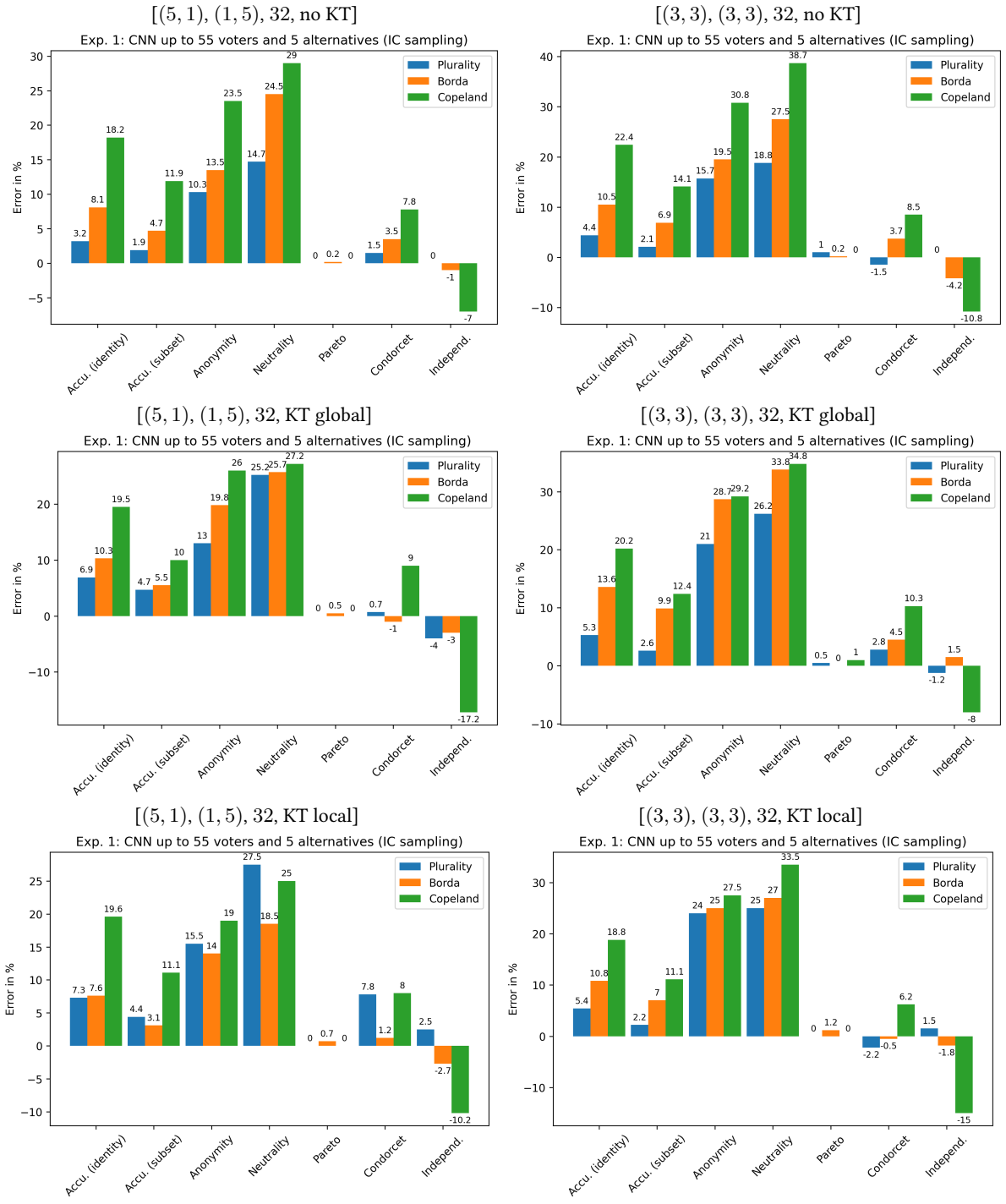

Figure 6: Different hyperparameters of CNNs and their performance. The description $[(5, 1), (1, 5), 32, \text{no KT}]$ means that, for this CNN, the first kernel has size $(5, 1)$, the second kernel has size $(1, 5)$, the number of channels is 32, and the input is not Kendall Tau preprocessed. Similarly for the other descriptions. (The top left plot is our standard CNN setting and repeated from Figure 7 for convenience.)

MLP

Testing fold number	Train loss	Train accuracy (in %)	Test loss	Test accuracy (in %)
0	0.013	97.7	0.031	95.1
1	0.013	97.7	0.033	94.8
2	0.012	97.9	0.032	95.4
3	0.009	98.5	0.031	95.4
4	0.018	96.9	0.038	94.4
5	0.018	96.9	0.04	94.1
6	0.025	95.8	0.042	93.7
7	0.011	97.9	0.03	95.1
8	0.013	97.7	0.029	95.9
9	0.008	98.7	0.025	96
Avg.	0.014	97.6	0.033	95
Std. dev.	0.005	0.8	0.005	0.7

CNN

Testing fold number	Train loss	Train accuracy (in %)	Test loss	Test accuracy (in %)
0	0.021	96.5	0.023	96.1
1	0.015	97.2	0.019	96.8
2	0.009	98.7	0.011	98.3
3	0.012	98	0.015	97.3
4	0.016	97.3	0.019	96.9
5	0.01	98.5	0.011	98
6	0.009	98.4	0.012	98
7	0.02	96.3	0.02	96.4
8	0.014	97.3	0.019	96.6
9	0.024	95.8	0.029	95.4
Avg.	0.015	97.4	0.018	97
Std. dev.	0.005	0.9	0.005	0.9

WEC

Testing fold number	Train loss	Train accuracy (in %)	Test loss	Test accuracy (in %)
0	0.005	99.4	0.006	99.5
1	0.005	99.7	0.005	99.8
2	0.006	99.6	0.006	99.5
3	0.007	99.3	0.008	99.1
4	0.035	96.3	0.037	95.8
5	0.004	99.8	0.004	99.9
6	0.012	98.6	0.014	98.5
7	0.01	99	0.011	98.7
8	0.01	98.5	0.009	98.6
9	0.011	98.4	0.011	98.3
Avg.	0.01	98.8	0.011	98.8
Std. dev.	0.009	1	0.009	1.1

Table 3: Cross validation of the three architectures on a dataset with $100,000$ data points IC-sampled from the Plurality rule with up to 55 voters and 5 alternatives. Training is for 8 epochs with a batch size of 200 (hence $8 \times \frac{90,000}{200} = 3,600$ gradient steps).

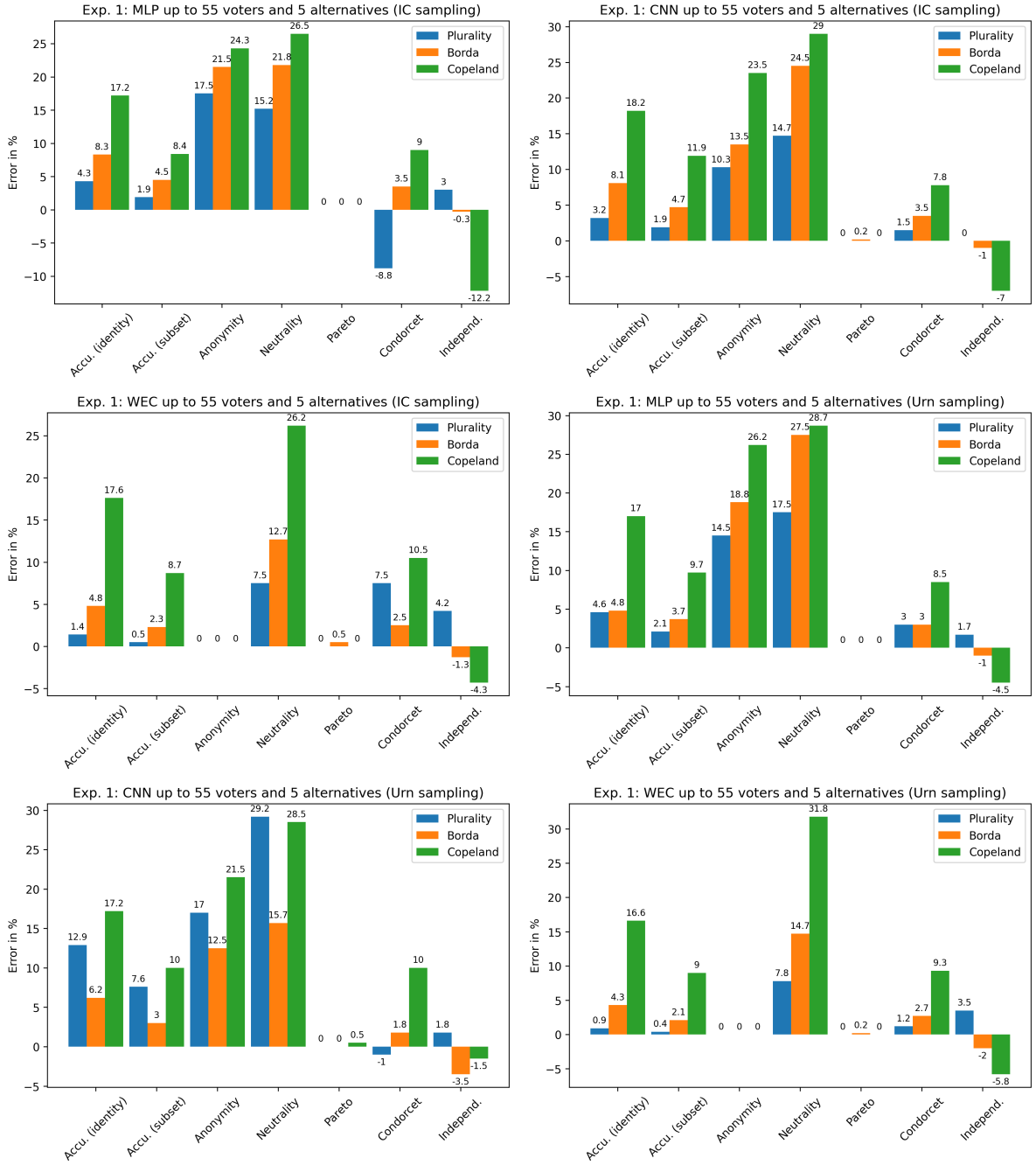

Figure 7: Part 1 of more settings of experiment 1 (Section 4.1). Varying architectures, rules, and sampling, while comparing the errors in both accuracy and axiom satisfaction.

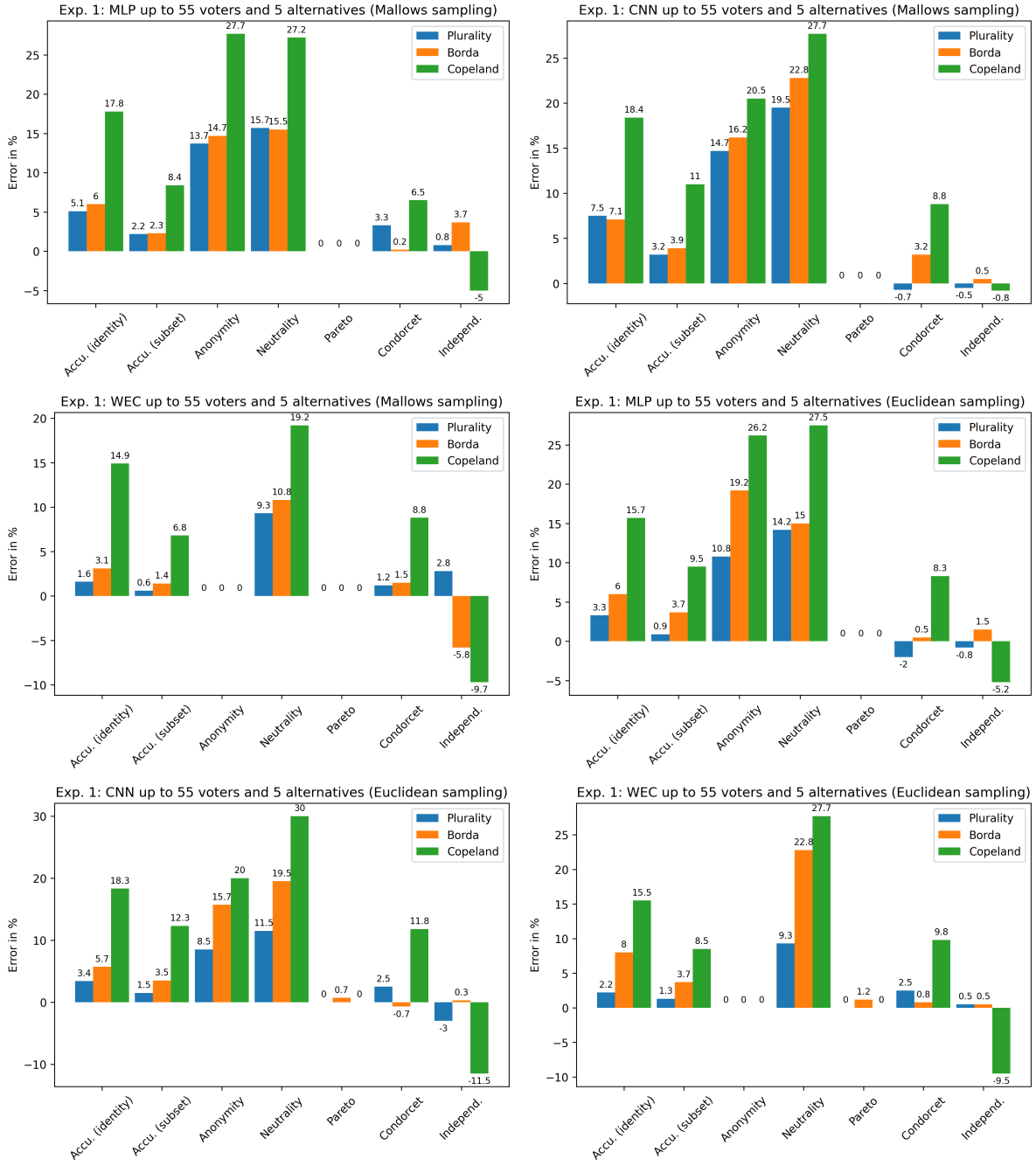

Figure 8: Part 2 of more settings of experiment 1 (Section 4.1). Varying architectures, rules, and sampling, while comparing the errors in both accuracy and axiom satisfaction.

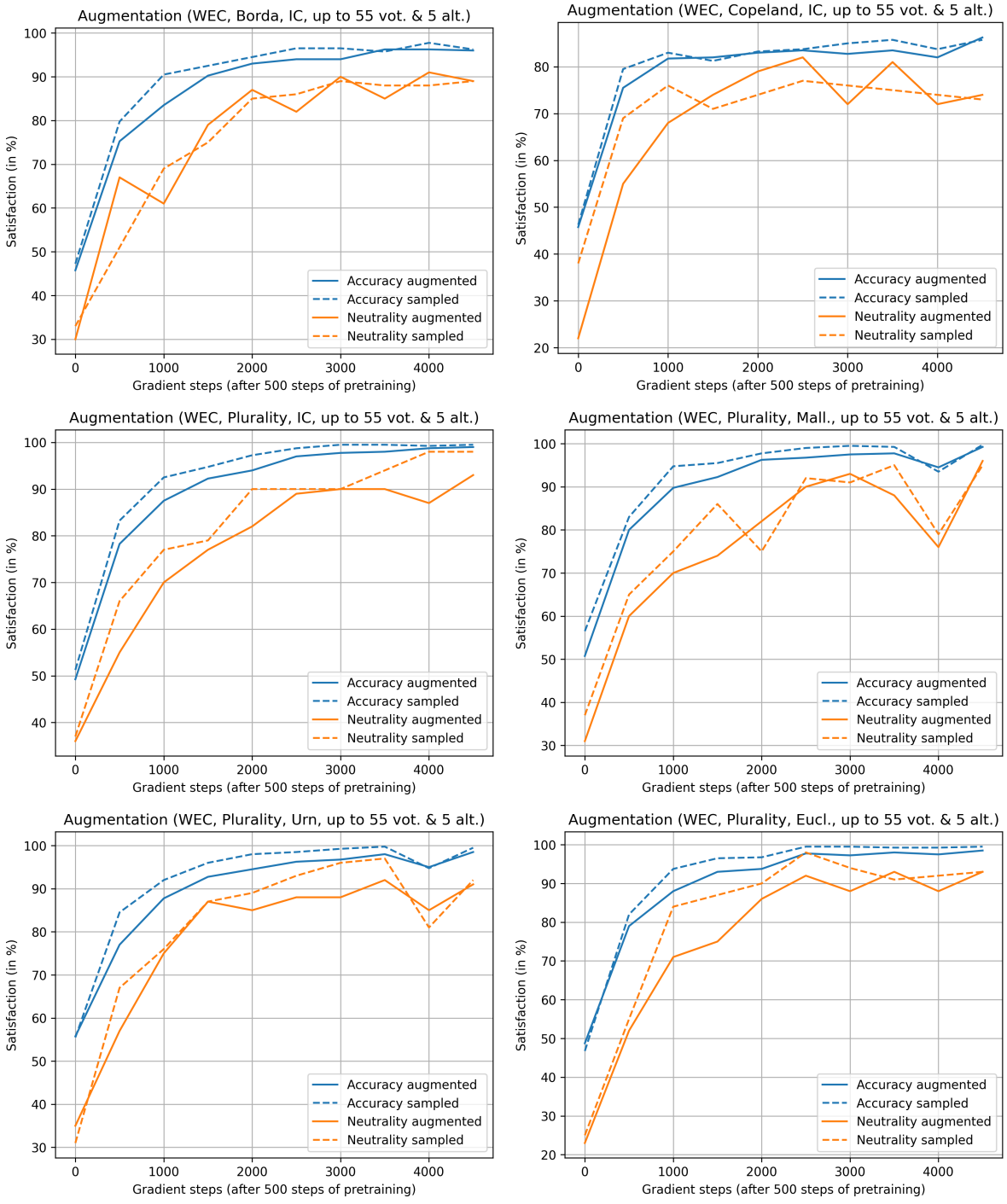

Figure 9: The first version of experiment 2 (Section 4.2) with further architectures, rules, and distributions.

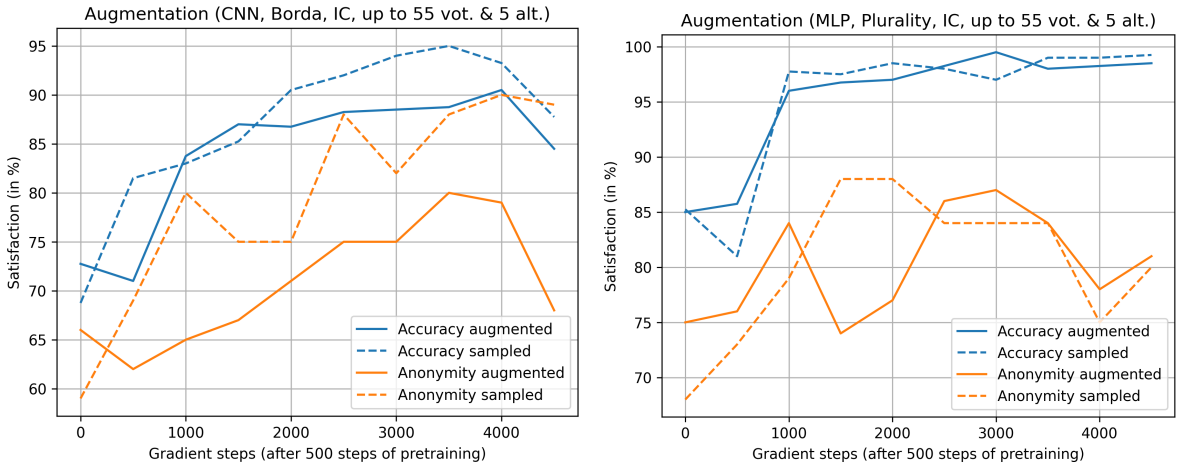

Figure 10: The first version of experiment 2 (Section 4.2), but for the anonymity axiom (instead of neutrality). Since the WEC is anonymous by design, this can only be tested for MLP and CNN.

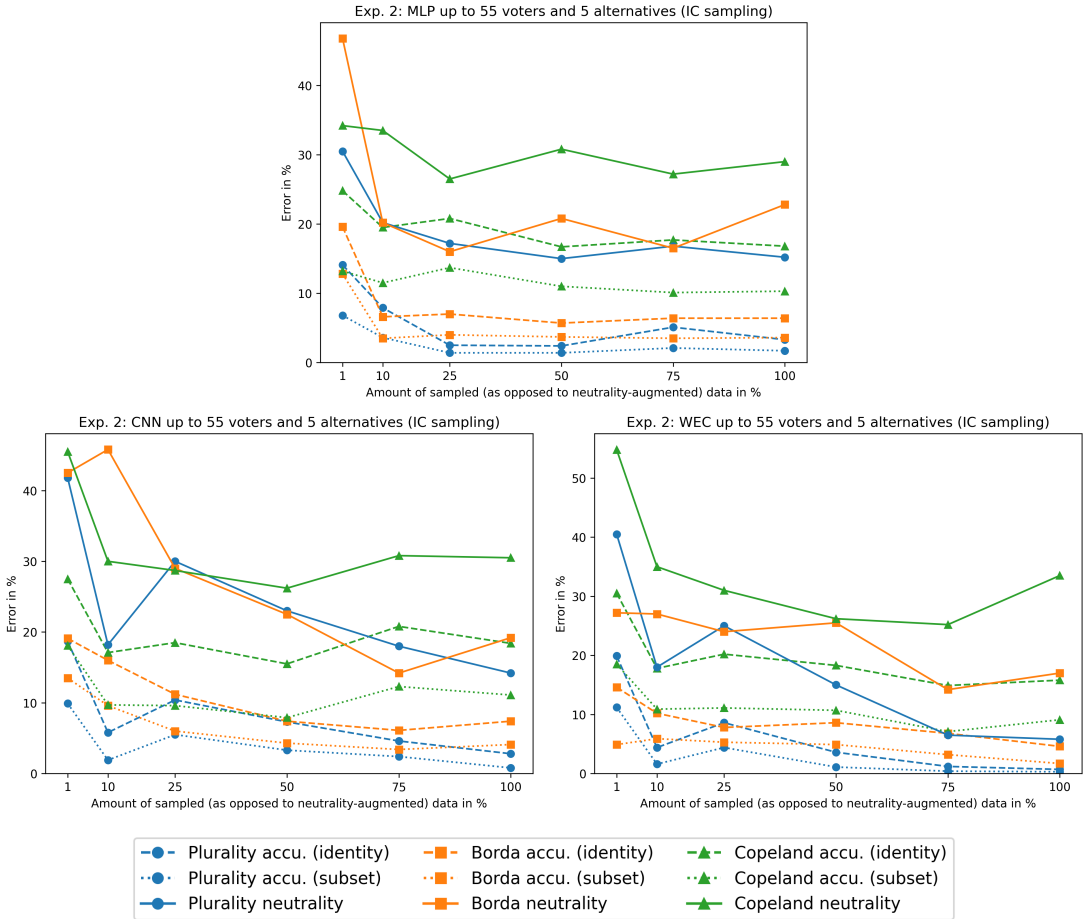

Figure 11: The second version of experiment 2 (Section 4.2). The identity accuracy, subset accuracy, and neutrality error across different ratios of augmented data (for the rules shown in the legend). E.g., '10' on the x-axis means the data is 10% sampled and 90% augmented.

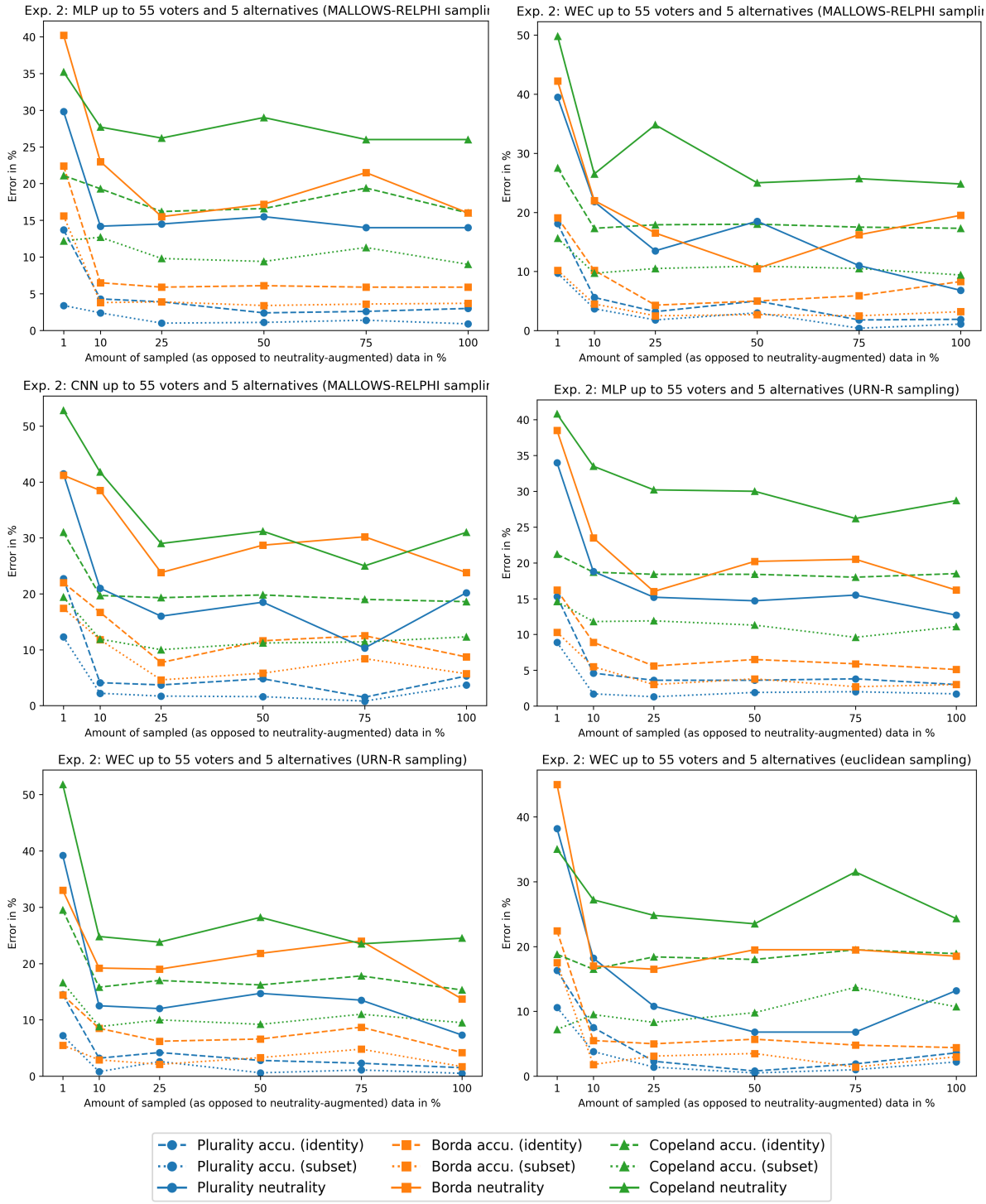

Figure 12: The second version of experiment 2 (Section 4.2) with different distributions.

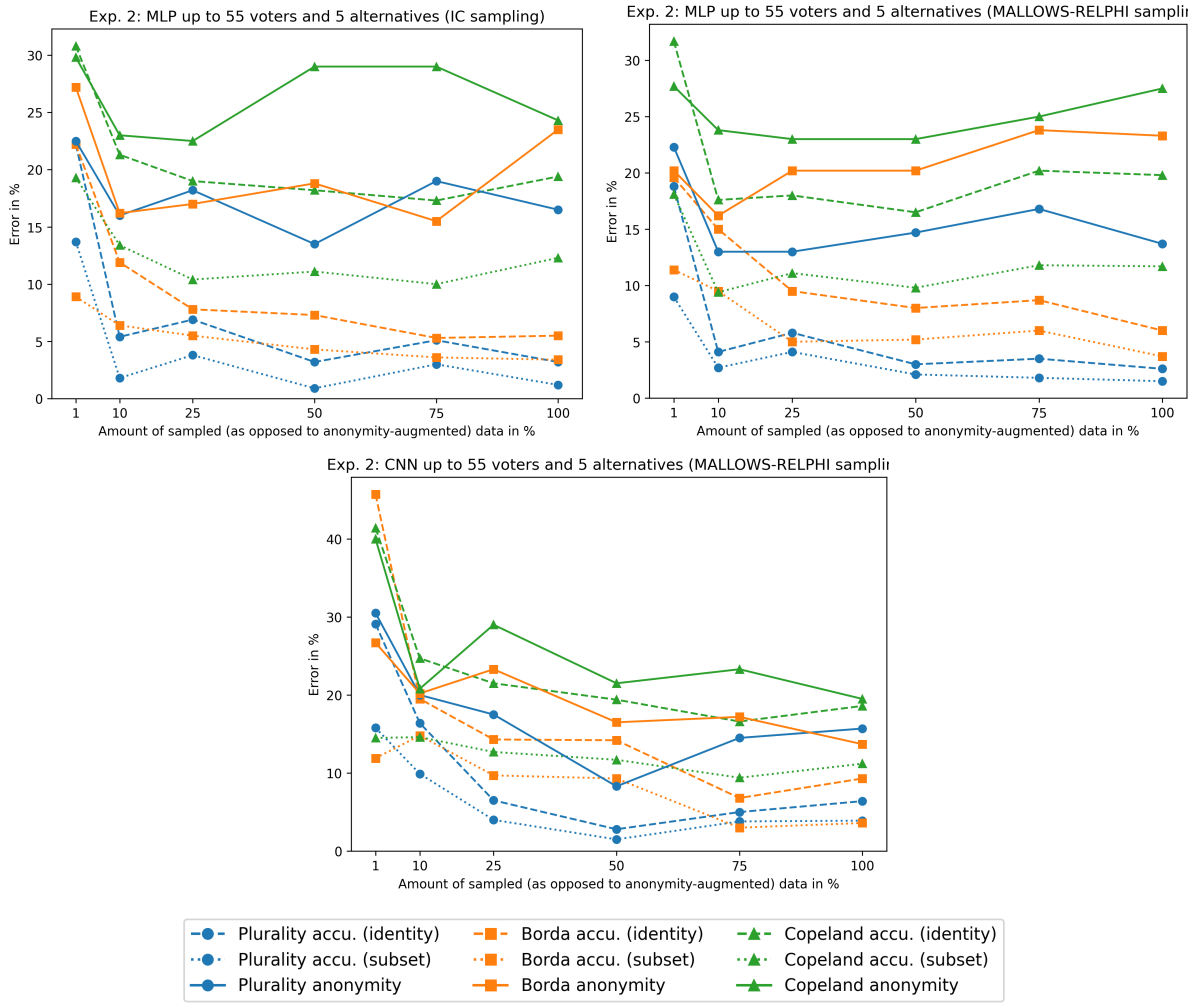

Figure 13: The second version of experiment 2 (Section 4.2), but for the anonymity axiom (instead of neutrality). Since the WEC is anonymous by design, this can only be tested for MLP and CNN.

Subset accuracy	WEC n	Blacks	Stable Voting	Borda	Weak Nanson	Copeland
WEC n	100	93.6	92.7	93.9	93	95.3
Blacks	95.6	100	96.53	97.3	95.57	98.2
Stable Voting	95.4	97.21	100	94.51	97.9	99.59
Borda	93.8	95.13	91.66	100	90.7	93.33
Weak Nanson	92	92.69	94.35	89.99	100	97.71
Copeland	90.4	91.29	91.73	88.59	94.15	100

Table 4: Similarities between the rules. Computed on 10,000 IC-sampled profiles. The entries show when the rule in the row outputs a winning set that is a subset of the rule in the column. So the entry 92.7 in row 'WEC n' and column 'Stable Voting' means that, in 92.7% of the sampled profiles, the winning set outputted by the neutrality-averaged WEC is a subset of the winning set outputted by Stable Voting. This table is not symmetric, because the entry 95.4 in row 'Stable Voting' and column 'WEC n' means that, in 95.4% of the sampled profiles, the winning set outputted by Stable Voting is a subset of the model's winning set (equivalently, the model's winning set is a superset of the rule's winning set).

1	2	3	4	5	6	7
a	b	c	e	b	b	d
e	d	e	c	c	c	e
d	c	a	d	a	a	b
b	a	d	a	d	d	c
c	e	b	b	e	e	a

$\{b, c\}$	neutrality-averaged WEC, with sigmoids (rounded) a:.38, b:.54, c:.50, d:.39, e:.39
$\{b\}$	Plurality, Weak Nanson, Kemeny-Young
$\{c\}$	Borda, Copeland, Llull, Blacks, Coombs
$\{d\}$	Anti-Plurality, Baldwin
$\{e\}$	Instant Runoff TB
$\{b, d\}$	Stable Voting
$\{b, c, d\}$	Uncovered Set, Banks
$\{b, c, e\}$	PluralityWRunoff PUT
$\{a, b, c, d, e\}$	Top Cycle

Figure 14: IC sampling: Profile where the 'WEC n' model weakly disagrees (i.e. non-identical winning sets) with existing voting rules.

F Experiment 3

We add further results on the 'rule synthesis guided by principles' experiment (Section 4.3).

F.1 More on the experiment with IC

Regarding difference-making profiles, Figure 4 in the main text shows a profile where the model *strongly disagrees* with its 5 closest voting rules, i.e., the model's winning set does not intersect the winning set of these rules. Figure 14 here shows a profile where the model *weakly disagrees* with all considered voting rules, i.e., the model's winning set is not identical with the winning set of these rules. (For the main text profile, the model also happens to weakly disagree with all considered rules.) We found these profiles by going through 10,000 IC-sampled profiles and picking the weakly or strongly disagreeing profile with the smallest number of voters.

Table 4 shows the similarities between the rules based on subset (i.e., soft) accuracy.

Table 5 suggests that further optimization does not further improve axiom satisfaction.

	Anon.	Neut.	Condorcet	Pareto	Indep.	Average
WEC n (NW, C, P, round 0)	100	100	97.5	100	46	88.7
WEC n (NW, C, P, round 1)	100	100	100	100	38.5	87.7
WEC n (NW, C, P, round 2)	100	100	100	100	34.8	87
WEC n (NW, C, P, I, round 3)	100	100	100	100	31.8	86.3

Table 5: IC sampling: The result of keeping on training an WEC model. Each round adds 20k gradient steps to the previous one. Round 1 is with a learning rate of 10^{-3}, round 2 with 10^{-4}, round 3 with $5 * 10^{-5}$, and round 4 the same but with added optimization of independence.

	Anon.	Neut.	Condorcet	Pareto	Indep.	Avg.
Blacks	100	100	100	100	36.04	87.2
Stable Voting	100	100	100	100	40.48	88.1
Borda	100	100	93.82	100	37.72	86.32
Weak Nanson	100	100	100	100	38.28	87.68
Copeland	100	100	100	100	28.54	85.72
WEC n (NW, C, P)	100	100	96.78	100	45.9	88.54

Table 6: IC sampling: Take the average over 5 runs of checking the axiom satisfaction of the 'WEC n' model and its closest rules.

Table 6 shows the statistical robustness of the axiom satisfaction achieved by the model.

F.2 The experiment with Mallows

Table 7 shows the result of the experiment from Section 4.3 but with Mallows sampling (instead of IC sampling), using the parameters discussed in Section 2.2. Table 8 shows how similar the best model is to its closest rules. Figure 15 presents a profile where the model differs from all considered rules. Table 9 shows the statistical robustness of the axiom satisfaction achieved by the model.

F.3 The experiment with Urn

Table 10 shows the result of the experiment from Section 4.3 but with Urn sampling (instead of IC sampling). Table 11 shows how similar the best model is to its closest rules. Figure 16 presents a profile where the model strongly differs (i.e., had non-intersecting winning sets) from its five closest rules. Table 12 shows the statistical robustness of the axiom satisfaction achieved by the model.

1	2	3	4
c	b	b	c
d	d	d	a
a	a	c	b
b	c	a	d

$\{b\}$	neutrality-averaged WEC, with sigmoids (rounded) a:.24, b:.51, c:.49, d:.32
$\{c\}$	Instant Runoff
$\{b, c\}$	Plurality, Borda, Copeland, Llull, Uncovered Set, Stable Voting, Blacks, Plurality-WRunoff PUT, Baldwin, Weak Nanson, Kemeny-Young
$\{b, c, d\}$	Banks
$\{a, b, c, d\}$	Anti-Plurality, Top Cycle, Coombs

Figure 15: Mallows sampling: Profile where the 'WEC n' model weakly disagrees (i.e. non-identical winning sets) with existing voting rules.

	Anon.	Neut.	Condorcet	Pareto	Indep.	Average
Plurality	100	100	83.0	100	30.2	82.7
Borda	100	100	92.8	100	32.8	85.1
Anti-Plurality	100	100	76.5	100	26.2	80.5
Copeland	100	100	100	100	27.8	85.5
Llull	100	100	100	100	26.2	85.2
Uncovered Set	100	100	100	100	29.5	85.9
Top Cycle	100	100	100	100	25.2	85.0
Banks	100	100	100	100	25.2	85.0
Stable Voting	100	100	100	100	39.0	87.8
Blacks	100	100	100	100	33.8	86.8
Instant Runoff TB	100	100	96.8	100	29.0	85.2
PluralityWRunoff PUT	100	100	94.0	100	27.0	84.2
Coombs	100	100	95.5	100	30.2	85.2
Baldwin	100	100	100	100	39.2	87.9
Weak Nanson	100	100	100	100	33.8	86.8
Kemeny-Young	100	100	100	100	38.2	87.7
MLP (NW, A, C, P, I)	78.8	76.0	94.0	100	38.8	77.5
CNN (NW, A, C, P, I)	80.5	68.8	94.5	100	38.8	76.5
WEC (NW, C, P)	100	65.5	91.8	100	37.8	79
WEC n (NW, C, P)	100	100	97.0	100	44.0	88.2

Table 7: Mallows sampling: Axiom satisfaction of different rules (top part of the table) and models (bottom part of the table). Otherwise like Table 1 from the main text.

Identity accuracy	WEC n	Stable Voting	Blacks	Borda	Weak Nanson	Copeland
WEC n	100	89.1	89.4	88.3	87.3	87.2
Stable Voting		100	95.61	91.04	93.47	92.16
Blacks			100	95.43	91.71	90.82
Borda				100	87.14	86.25
Weak Nanson					100	92.08
Copeland						100

Subset accuracy	WEC n	Stable Voting	Blacks	Borda	Weak Nanson	Copeland
WEC n	100	91.7	92.2	92.5	92.1	94.4
Stable Voting	95.8	100	97.09	94.4	97.78	99.49
Blacks	95.8	96.63	100	97.31	95.96	97.97
Borda	94.2	92.06	95.43	100	91.39	93.4
Weak Nanson	92.7	94.5	93.02	90.33	100	97.4
Copeland	91.3	92.23	91.6	88.91	94.17	100

Table 8: Mallows sampling: Similarities between the rules. Computed on 10,000 sampled profiles. Otherwise like Table 2 from the main text.

	Anon.	Neut.	Condorcet	Pareto	Indep.	Avg.
Stable Voting	100	100	100	100	38.14	87.62
Blacks	100	100	100	100	34.04	86.84
Borda	100	100	94.16	100	35.02	85.84
Weak Nanson	100	100	100	100	37.18	87.46
Copeland	100	100	100	100	27.5	85.48
WEC n (NW, C, P)	100	100	94.92	100	41.72	87.34

Table 9: Mallows sampling: Take the average over 5 runs of checking the axiom satisfaction of the 'WEC n' model and its closest rules.

	Anon.	Neut.	Condorcet	Pareto	Indep.	Avg.
Plurality	100	100	84.2	100	24.5	81.8
Borda	100	100	94.8	100	35	85.9
Anti-Plurality	100	100	76.8	100	25	80.3
Copeland	100	100	100	100	28.2	85.6
Llull	100	100	100	100	27.3	85.5
Uncovered Set	100	100	100	100	25.2	85
Top Cycle	100	100	100	100	27.8	85.5
Banks	100	100	100	100	27.5	85.5
Stable Voting	100	100	100	100	38	87.6
Blacks	100	100	100	100	34.2	86.9
Instant Runoff TB	100	100	97	100	28.2	85
PluralityWRunoff PUT	100	100	94.2	100	26.2	84.1
Coombs	100	100	96.5	100	28.5	85
Baldwin	100	100	100	100	39	87.8
Weak Nanson	100	100	100	100	38.5	87.7
Kemeny-Young	100	100	100	100	39.2	87.9
MLP (NW, A, C, P, I)	79.8	74.2	92.8	100	34.8	76.3
CNN (NW, A, C, P, I)	82.8	73.8	94	100	37.8	77.6
WEC (NW, C, P)	100	75.2	93	100	37.8	81.2
WEC n (NW, C, P)	100	100	93.5	100	39	86.5

Table 10: Urn sampling: Axiom satisfaction of different rules (top part of the table) and models (bottom part of the table). Otherwise like Table 1 from the main text.

Identity accuracy	WEC n	Blacks	Stable Voting	Borda	Copeland	Weak Nanson
WEC n	100	88.9	88.5	88.2	86.5	86.3
Blacks		100	95.52	95.32	90.92	91.6
Stable Voting			100	90.84	92.17	93.61
Borda				100	86.24	86.92
Copeland					100	92.09
Weak Nanson						100

Subset accuracy	WEC n	Blacks	Stable Voting	Borda	Copeland	Weak Nanson
WEC n	100	92.3	91.4	93.2	94.3	91.3
Blacks	95.1	100	96.44	97.48	98.07	95.4
Stable Voting	94.9	97.03	100	94.51	99.47	97.48
Borda	93.9	95.32	91.76	100	93.39	90.72
Copeland	90.4	91.83	92.19	89.31	100	93.9
Weak Nanson	91.7	93.04	94.65	90.52	97.43	100

Table 11: Urn sampling: Similarities between the rules. Computed on 10,000 sampled profiles. Otherwise like Table 2 from the main text.

	Anon.	Neut.	Condorcet	Pareto	Indep.	Avg.
Blacks	100	100	100	100	34.38	86.9
Stable Voting	100	100	100	100	39.2	87.84
Borda	100	100	93.46	100	35.66	85.84
Copeland	100	100	100	100	26.9	85.38
Weak Nanson	100	100	100	100	37.82	87.54
WEC n (NW, C, P)	100	100	95.68	100	38.16	86.76

Table 12: Urn sampling: Take the average over 5 runs of checking the axiom satisfaction of the 'WEC n' model and its closest rules.

1	2	3	4	5
b	a	b	d	e
c	c	e	a	c
d	e	d	c	a
e	d	a	b	b
a	b	c	e	d

$\{b\}$	neutrality-averaged WEC, with sigmoids (rounded) a:.44, b:.50, c:.48, d:.40, e:.44
$\{c\}$	Blacks, Stable Voting, Borda, Copeland, Weak Nanson, Llull
$\{b\}$	Plurality, Instant Runoff TB
$\{a\}$	Baldwin
$\{a, b\}$	PluralityWRunoff PUT
$\{a, c\}$	Kemeny-Young
$\{a, c, e\}$	Uncovered Set, Banks
$\{a, b, c, d, e\}$	Anti-Plurality, Top Cycle, Coombs

Figure 16: Urn sampling: Profile where the 'WEC n' model strongly disagrees (i.e. non-intersecting winning sets) with its 5 closest rules (among the remaining rules it only agrees with Plurality and Instant Runoff TB).

1	2
d	b
a	e
c	a
b	c
e	d

$\{b\}$	neutrality-averaged WEC, with sigmoids (rounded) a:.34, b:.50, c:.13, d:.30, e:.13
$\{a\}$	Coombs
$\{d\}$	Instant Runoff TB
$\{a, b\}$	Weak Nanson
$\{b, d\}$	Plurality, PluralityWRunoff PUT
$\{a, b\}$	Borda, Copeland, Stable Voting, Blacks
$\{a, b, d\}$	Llull, Uncovered Set, Banks, Baldwin, Kemeny-Young
$\{a, b, c\}$	Anti-Plurality
$\{a, b, c, d, e\}$	Top Cycle

Figure 17: Euclidean sampling: Profile where the 'WEC n' model weakly disagrees (i.e. non-identical winning sets) with existing voting rules.

F.4 The experiment with Euclidean

Table 13 shows the result of the experiment from Section 4.3 but with Euclidean sampling (instead of IC sampling). Table 14 shows how similar the best model is to its closest rules. Figure 17 presents a profile where the model differs from all considered rules. Table 15 shows the statistical robustness of the axiom satisfaction achieved by the model.

F.5 Ablation study

Figure 18 displays an ablation study in the choice of axioms to optimize for. For the best performing model, i.e., the neutrality-averaged WEC, there remain three axioms that can be optimized for: Condorcet (C), Pareto (P), and independence (I). The 'No winner' loss (NW) is always needed to prevent the model from never outputting any winner. Which subset of the three axioms is the optimal choice for optimization? In the main text, we chose C and P. The figure shows that this choice indeed is the best one. Figure 19 shows, for each choice of axiom optimization, the evolution of the losses during training.

	Anon.	Neut.	Condorcet	Pareto	Indep.	Avg.
Plurality	100	100	79.8	100	25.5	81
Borda	100	100	93.8	100	37.2	86.2
Anti-Plurality	100	100	77.2	100	25	80.5
Copeland	100	100	100	100	29.5	85.9
Llull	100	100	100	100	29.8	86
Uncovered Set	100	100	100	100	26.2	85.2
Top Cycle	100	100	100	100	28.7	85.8
Banks	100	100	100	100	28	85.6
Stable Voting	100	100	100	100	39.5	87.9
Blacks	100	100	100	100	37	87.4
Instant Runoff TB	100	100	96.8	100	30	85.4
PluralityWRunoff PUT	100	100	94.8	100	26.5	84.2
Coombs	100	100	96	100	26.2	84.5
Baldwin	100	100	100	100	40.2	88
Weak Nanson	100	100	100	100	40	88
Kemeny-Young	100	100	100	100	36.5	87.3
MLP (NW, A, C, P, I)	79.2	78	92.2	100	36	77.1
CNN (NW, A, C, P, I)	83.2	74.5	93.2	100	35	77.2
WEC (NW, C, P)	100	75	96.8	100	38.5	82
WEC n (NW, C, P)	100	100	97.8	100	42.2	88

Table 13: Euclidean sampling: Axiom satisfaction of different rules (top part of the table) and models (bottom part of the table). Otherwise like Table 1 from the main text.

Identity accuracy	WEC n	Stable Voting	Blacks	Weak Nanson	Copeland	Borda
WEC n	100	92.2	91.5	89.5	88.7	89.1
Stable Voting		100	95.65	93.22	92.24	91.05
Blacks			100	91.37	90.85	95.4
Weak Nanson				100	92.46	86.77
Copeland					100	86.25
Borda						100

Subset accuracy	WEC n	Stable Voting	Blacks	Weak Nanson	Copeland	Borda
WEC n	100	94.9	94.5	94.6	96.7	93.9
Stable Voting	95.6	100	97.19	97.5	99.58	94.57
Blacks	94.7	96.69	100	95.61	97.92	97.38
Weak Nanson	92	94.24	92.82	100	97.53	90.2
Copeland	90.8	92.3	91.67	94.39	100	89.05
Borda	92.1	92.09	95.4	91.01	93.32	100

Table 14: Euclidean sampling: Similarities between the rules. Computed on 10,000 sampled profiles. Otherwise like Table 2 from the main text.

	Anon.	Neut.	Condorcet	Pareto	Indep.	Avg.
Stable Voting	100	100	100	100	39.12	87.8
Blacks	100	100	100	100	34.98	86.98
Weak Nanson	100	100	100	100	39.32	87.86
Copeland	100	100	100	100	27.68	85.54
Borda	100	100	95.08	100	35.46	86.08
WEC n (NW, C, P)	100	100	97.74	100	45.16	88.58

Table 15: Euclidean sampling: Take the average over 5 runs of checking the axiom satisfaction of the 'WEC n' model and its closest rules.

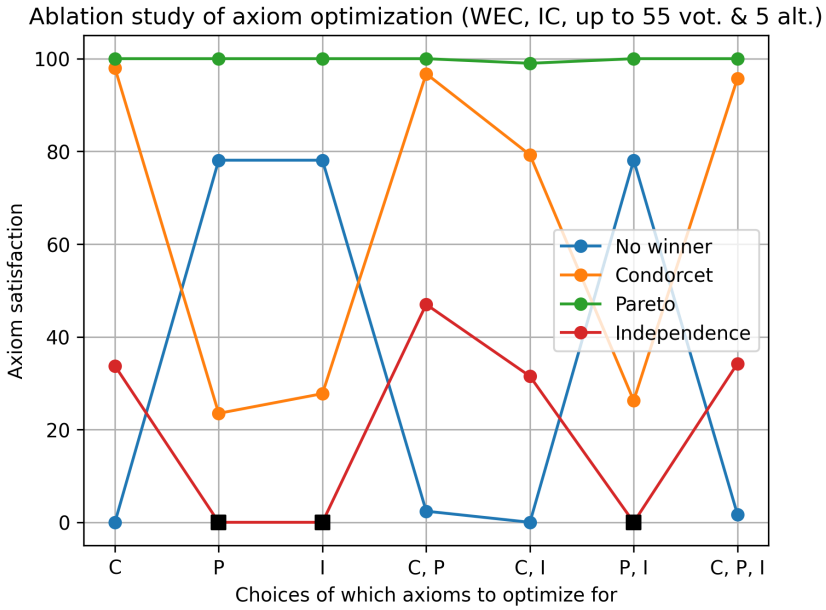

Figure 18: Ablation study in axiom optimization with the neutrality-averaged WEC. For each possible (nonempty) choice of axioms to optimize for among Condorcet (C), Pareto (P), and independence (I), the achieved axiom satisfaction is shown. The 'No winner' loss (NW) is always optimized for. Its reported satisfaction is 0 if the model always outputs at least one winner. The axioms of anonymity and neutrality are not shown since they are satisfied by design. The black squares in the independence satisfaction indicate that the axiom was applicable on too few of the sampled test profiles to warrant an estimate. The best choice is C, P since it has the highest axiom satisfaction combined with a low 'No winner' satisfaction.

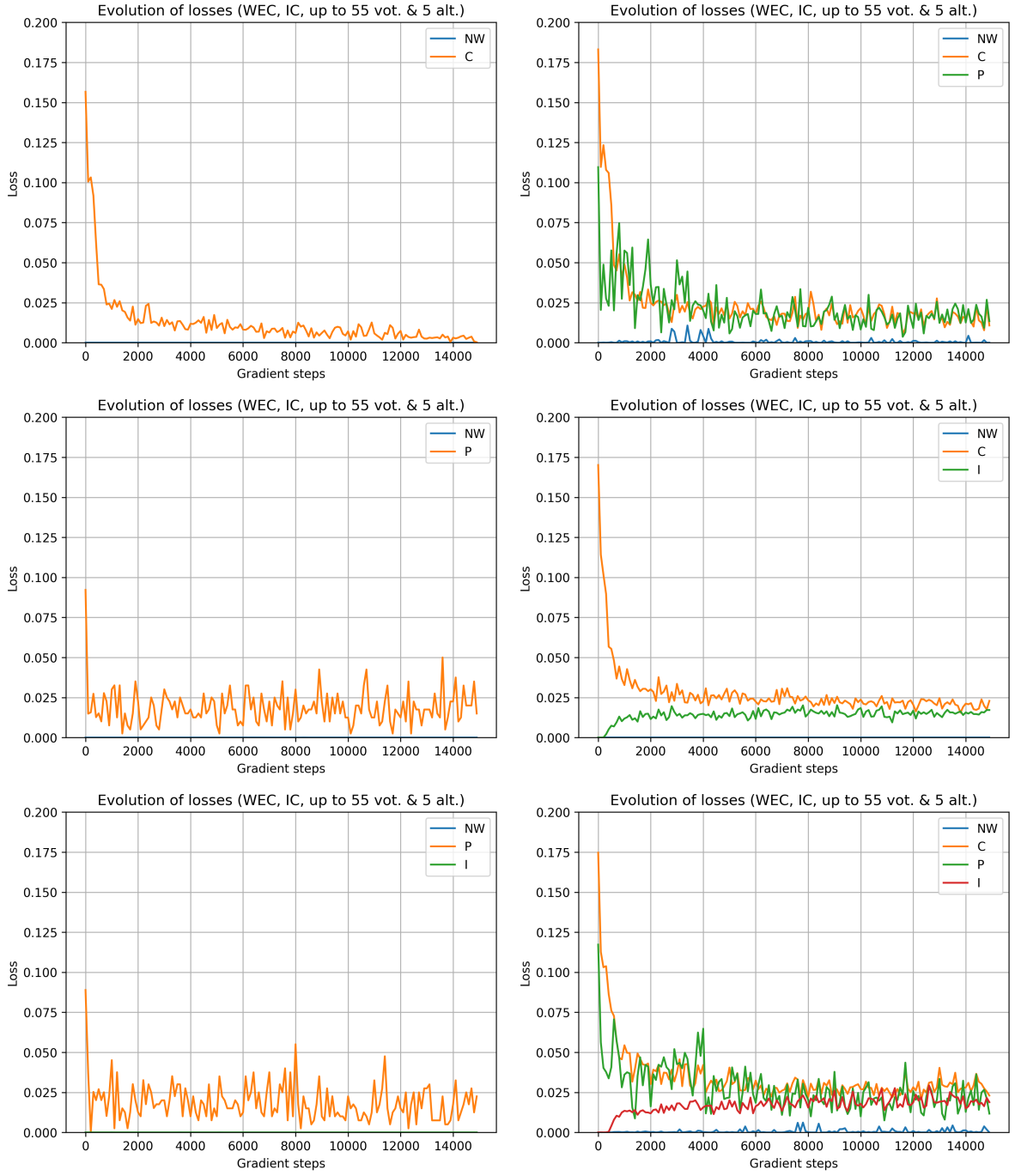

Figure 19: The evolution of the losses during axiom optimization with the neutrality-averaged WEC, for each choice of which axioms to optimize among Condorcet (C), Pareto (P), and independence (I), with 'no winner' (NW) always being optimized for. The loss curves for the NW+I-optimization are not shown, since they are so close to 0 that they are indistinguishable from the x-axis. In the other two cases that did not yield good axiom satisfaction—i.e., P and P, I in figure 18—, the loss evolution also shows no convergence.