Generative Social Choice

Sara Fish[1], **Paul Gölz**[2], **David C. Parkes**[1], **Ariel D. Procaccia**[1], **Gili Rusak**[1], **Itai Shapira**[1], and **Manuel Wüthrich**[1]

[1]Harvard University [2]Simons Laufer Mathematical Sciences Institute

Traditionally, social choice theory has only been applicable to choices among a few predetermined alternatives but not to more complex decisions such as collectively selecting a textual statement. We introduce *generative social choice*, a framework that combines the mathematical rigor of social choice theory with large language models' capability to generate text and extrapolate preferences. This framework divides the design of AI-augmented democratic processes into two components: first, proving that the process satisfies rigorous representation guarantees when given access to oracle queries; second, empirically validating that these queries can be approximately implemented using a large language model. We illustrate this framework by applying it to the problem of generating a slate of statements that is representative of opinions expressed as free-form text, for instance in an online deliberative process.

1. Introduction

Social choice theory is the field of mathematics, economics, and political science that studies the aggregation of individual preferences towards collective decisions. The typical social choice setting involves a *small, predetermined set of alternatives* (such as candidates in an election) and a set of participants who specify their preferences regarding these alternatives, often in the form of rankings.

When democratic input is sought for more nuanced decisions, however, the process may not fit into this neat template. Citizens' assemblies, for example, provide policy recommendations to governments on complex issues, such as climate change and constitutional reform [Flanigan et al., 2021]. In response to the rapid progress of generative AI in recent years, a number of companies, for instance Meta [Clegg, 2023] and OpenAI [Zaremba et al., 2023], are experimenting with democratic processes for AI value alignment, with open-ended questions such as "how far [...] personalization of AI assistants like ChatGPT to align with a user's tastes and preferences should go?"

These developments motivate us to introduce a new paradigm for the design of democratic processes: *generative social choice*. It fuses the rigor of social choice theory with the flexibility and power of generative AI, in particular large language models (LLMs), to facilitate collective decisions on complex issues in a principled way.

How LLMs address the limitations of classical social choice. In our view, there are two fundamental obstacles to using classical social choice to answer open-ended questions; both obstacles can be circumvented by LLMs.

- *Unforeseen Alternatives.* In classical social choice, the set of alternatives is explicitly specified and static. Take the 2016 Brexit referendum, for example, in which the alternatives were either to maintain the status quo or make a clean break with the European Union. Since intermediate options were not specified, they could not be selected by voters, even if they might have enjoyed a much larger degree of support. Even in participatory budgeting [Cabannes, 2004], the set of alternatives is limited to the budget-feasible subsets of *previously proposed* projects.

  By contrast, LLMs have the capability of *generating* alternatives that were not initially anticipated but find common ground between participants. In principle, the possible outcomes of an LLM-augmented democratic process may span the universe of all relevant outcomes for the problem at hand, e.g., all possible bills, images, or statements.

- *Extrapolating Preferences.* In classical social choice theory, agents specify their preferences in a rigid format. Typically, agents evaluate each alternative independently, or, if the alternatives form a combinatorial domain[1], a voting rule might assume that preferences have a restricted parametric shape, and only elicit its parameters. This approach clearly does not suffice if a democratic process may produce alternatives that were not previously anticipated, and therefore not elicited: to even know which alternatives would be promising to generate, the process must be able to extrapolate participants' preferences.

  LLMs can address this problem by allowing participants to *implicitly* specify their preferences by expressing their opinions, values, or criteria in natural language. The LLM can act as a proxy for the participant, predicting their preferences over any alternative, whether foreseen or newly generated.

A framework for generative social choice. It is clear, at this point, what LLMs can contribute to social choice. LLMs and social choice theory make an odd couple, however, because social choice focuses on rigorous guarantees whereas LLMs are notoriously impervious to theoretical analysis. We propose a framework for generative social choice that addresses this difficulty by breaking the design of democratic processes into two interacting components.

- *First component: Guarantees with perfect queries.* Assume that the LLM is an oracle that can precisely answer certain types of queries, which may involve generating new alternatives in an optimal way or predicting agents' preferences. Once appropriate queries have been identified, the task is to design algorithms that, when given access to an oracle for these queries, provide social choice guarantees.

- *Second component: Empirical validation of queries.* Assuming the LLM to be a perfect oracle is helpful for guiding the design of a democratic process, but of course not an accurate reflection of reality. In the second component, the task is to empirically validate the required queries. While the LLM will not provide perfect answers in practice, it can be shown to provide accurate or reliable answers.

[1] This is the case, for example, in multi-winner elections or participatory budgeting.

Naturally, the first component and the second component interact: The theory identifies queries that are useful for social choice and should hence be validated empirically. Conversely, experiments show which queries can be answered accurately in practice, raising the question of which guarantees algorithms relying on these queries might provide.

A key benefit of this framework is that theoretical results derived from it are future-proof: as LLMs continue to rapidly improve, they will only grow more reliable in answering queries, making the LLM-based aggregation methods ever more powerful.

Our results: A case study in generative social choice. In addition to introducing the general framework presented above, we pilot it in one particular setting: the selection of representative opinion statements. In this setting, agents share their opinions about a given policy issue on an online platform such as *Polis* [Small et al., 2021], and a voting rule selects a slate of k statements that is representative of the user population's opinions. This setting was formalized by Halpern et al. [2023] in the language of multi-winner approval elections: If we think of statements as candidates, and of an agreement between participants and statements as approval votes, the slate should satisfy axioms for proportional representation from this literature such as *justified representation* (JR) and *extended justified representation* (EJR).

While previous systems can only select a slate among users' statements, our process can *generate new statements*, which might find new common ground between participants and allow for more representative slates. Our process takes as input each user's interactions on the platform (e.g., in the form of votes on statements, statements submitted by the user, or free-text interaction) as a description of their preferences. The process then employs an LLM to (1) translate these descriptions to participants' approvals regarding any new statements (*discriminative* queries, in the language of machine learning), and (2) generate statements that maximize the number of approvals of a given set of participants, based on their descriptions (*generative* queries).

Following our framework's first component, we show that, with access to polynomially many of these queries, a democratic process based on *Greedy Approval Voting* [Aziz et al., 2017] guarantees JR, and a democratic processes based on the *Greedy Justified Candidate Rule* [Brill and Peters, 2023] guarantees EJR. Crucially, these guarantees hold not just relative to a set of predetermined statements but to the space of all possible statements (Section 3.1).

A potential issue with this process is that, through the generative query, it calls the LLM with a prompt whose length scales linearly in the number of participants. This is problematic since LLMs can only handle input of bounded length. We show that, unless one makes assumptions on the structure of preferences, this problem of linear-size queries is unavoidable for any process guaranteeing EJR and for any process guaranteeing JR with subexponentially many queries (Section 3.2). If, however, the space of statements and approval preferences is structured, specifically, if it has finite VC dimension [Vapnik, 1998], democratic processes based on sampling can guarantee EJR (with high probability) using a polynomial number of queries whose length is independent of the number of participants (Section 3.3).

In Section 4, we execute the second component of the framework by empirically validating our discriminative and generative queries on three real-world datasets. We show that the LLM can extrapolate approval preferences well in its discriminative queries, and that generative queries consistently identify statements with wider support than the statements made by the participants themselves. We also assess the performance of two democratic processes on our

datasets. The process based on Greedy Approval Voting produces slates of statements without any apparent violations of JR (as predicted by the theory), which are moreover qualitatively meaningful. We view these findings as an encouraging indication of the potential of generative social choice. By contrast, the process based on the Greedy Justified Candidate Rule tends to produce panels with many near-identical statements, even though the generative query used by this process explicitly instructs the LLM to avoid such repetitions. We conclude with take-aways for the design of practical democratic processes and future work.

Related work. In a very recent position paper that is independent of our work, Small et al. [2023] discuss the opportunities and risks of LLMs in the context of Polis. The opportunities they identify include topic modeling, summarization, moderation, comment routing, identifying consensus, and vote prediction. Most relevant to us are their experiments for the vote prediction task, which is related to our discriminative queries. Through an internal API of the *Claude* LLM, they can predict *probabilities* of agreement, and demonstrate that these vote predictions are well calibrated. This differs from our experiments because we require a binary output of the discriminative query, rather than a probability. While this gap could likely be bridged, a more fundamental difference is that we evaluate generative queries, along with discriminative queries, and the interaction between them.

Our discriminative queries using LLMs are also related to work by Konya et al. [2022], who integrate an LLM with a latent factor model to predict preferences. More broadly, the paradigm of *virtual democracy* facilitates automated decisions on ethical dilemmas by learning the preferences of stakeholders and, at runtime, predicting their preferences over the current alternatives and aggregating the predicted preferences; example papers, which employ classical machine-learning algorithms, apply the paradigm to domains such as autonomous vehicles [Noothigattu et al., 2018], food rescue [Lee et al., 2019], and kidney exchange [Freedman et al., 2020]. These papers all aim to predict preferences on a fixed set of alternatives — they do not generate new alternatives.

As source of inspiration for our work is the paper of Bakker et al. [2022]. They fine-tune an LLM to generate a single consensus statement for a specific group of people, based on written opinions and ratings of candidate statements. Reward models are trained to capture individual preferences, and the acceptability of a statement for the group is measured through a social welfare function. One difference from our work is that we do not attempt to find a single statement that builds consensus across the entire group — we instead allow for multiple statements representing distinct opinions. A more fundamental difference is that we view our experiments as an instance of a broader framework that allows for a systematic investigation of the types of queries an LLM can perform and the theoretical guarantees they provide.

We also build, in our application, on the rich literature on justified representation (and its extensions) in approval-based committee elections [Aziz et al., 2017], and in particular make use of a voting rule by Brill and Peters [2023] in this setting. As we have already mentioned, Halpern et al. [2023] also study representation axioms from this literature in a statement-selection context. The key technical challenge in their work is that they only have access to partial approval votes. The learning-theoretic approach they adopt, as well as a later refinement by Brill and Peters [2023], bears technical similarity to the algorithm we propose for obtaining representation with size-constrained generative queries. All previous papers in this literature assume a non-generative setting with a fixed set of alternatives.

2. Model

Let N be a set of n *agents*, and let $\mathcal{U}$ denote the *universe of* (well-formed, on-topic) *statements*, which might be finite or infinite. Each agent $i \in N$ *approves* a subset of statements $A_i \subseteq \mathcal{U}$. For a given statement $\alpha \in \mathcal{U}$, we denote by $A^{-1}(\alpha) \coloneqq \{i \in N \mid \alpha \in A_i\}$ the *supporters* of α. (In Section 3.3, we study a setting that imposes additional structure on approval preferences, which we introduce there.) An *instance* of the statement-selection problem consists of N, $\mathcal{U}$, $\{A_i\}_{i \in N}$, and a *slate size* $k \in \mathbb{N}_{\geq 1}$.

A *democratic process* is an algorithm that, when run on an instance, returns a *slate*, i.e., a set of at most k statements from the universe. Crucially, however, this algorithm receives only N and k, but not $\mathcal{U}$ or the A_i, in its input.

2.1. Queries

Instead, the democratic process accesses statements and approval preferences indirectly through *queries*. The democratic processes we develop make use of two query types:

Approval Queries. Our discriminative query predicts an agent's approval preferences: For an agent i and statement α, $\textsc{Approval}(i, \alpha)$ returns 1 if $\alpha \in A_i$, else 0.

Generative Queries. For a set of agents S of size at most t, t-$\textsc{Generate}(S, T)$ returns the statement most widely approved among S, excluding some statements T. That is, the query returns an element of $\operatorname{argmax}_{\alpha \in \mathcal{U} \setminus T} |A^{-1}(\alpha) \cap S|$, breaking ties arbitrarily.

For convenience, we will simply write $\textsc{Generate}(\cdot, \cdot)$ to refer to generative queries without a size limit (i.e., n-$\textsc{Generate}(\cdot, \cdot)$, which we sometimes spell out for emphasis) or to talk generally about generative queries with different size limits t.

2.2. Representation Axioms

The aim of our democratic processes is to produce slates of statements W that are representative of the agents. We capture this notion of representation through the following axioms introduced by Aziz et al. [2017]:

Justified Representation (JR): We say that W satisfies JR if there is no *coalition* $S \subseteq N$ and no statement $\alpha \in \mathcal{U}$ such that (i) $|S| \geq n/k$, (ii) $\alpha \in A_i$ for all $i \in S$, and (iii) $A_i \cap W = \emptyset$ for all $i \in S$.

Extended Justified Representation (EJR): The slate W satisfies EJR if there is no nonempty coalition $S \subseteq N$ and set $T \subseteq \mathcal{U}$ such that (i) $|S| \geq |T|\, n/k$, (ii) $T \subseteq \bigcap_{i \in S} A_i$, and (iii) $|A_i \cap W| < |T|$ for all $i \in S$.

A slate that satisfies EJR must also satisfy JR, since the definition of JR is a special case of EJR for a singleton set T.

3. First Component: Guarantees with Perfect Queries

In this section, we instantiate the first component of the generative social choice framework.

3.1. Unconstrained Queries

To illustrate the appeal of generative social choice in a simple setting, we begin by constructing a democratic process that guarantees JR in polynomial time. This algorithm will use queries of type $\textsc{Approval}(\cdot,\cdot)$ and n-$\textsc{Generate}(\cdot,\cdot)$; i.e., generative queries without constraints on the size of S.

The democratic process we propose, shown in Process 1, is the familiar *Greedy Approval Voting Rule* [Aziz et al., 2017], which iteratively adds statements to the slate that have the widest support among agents who do not approve any already selected statement. The proof of Aziz et al. [2017] that this algorithm satisfies justified representation directly applies to our setting. We simply highlight that, if the LLM can indeed greedily choose a statement from $\mathcal{U}$ in constant time, we can efficiently simulate Greedy Approval Voting over arbitrarily large, possibly infinite, universes of statements $\mathcal{U}$.

Proposition 1 (Aziz et al., 2017, Thm. 1)**.** *Process 1 satisfies justified representation in polynomial time in n and k, using queries of types n-$\textsc{Generate}(\cdot,\emptyset)$ and $\textsc{Approval}(\cdot,\cdot)$.*

Process 1: Democratic Process for Justified Representation

$S \leftarrow N$
$W \leftarrow \emptyset$
for $t = 1, 2, \ldots, k$ **do**
  $\alpha \leftarrow \textsc{Generate}(S, \emptyset)$
  $W \leftarrow W \cup \{\alpha\}$
  $S \leftarrow \{i \in S \mid \textsc{Approval}(i, \alpha) = 0\}$
return W

Whereas Process 1 never used the generative query's second argument to exclude statements, a democratic process for EJR must do so to prevent some statement α^* from "overshadowing" other statements needed for representation. Indeed, suppose that $\mathcal{U}$ contains only $k \geq 2$ statements, each approved by all agents. Depending on tie breaking, *any* n-$\textsc{Generate}(\cdot,\emptyset)$ query might return the same statement α^*. A process based only on such queries and approval queries can never access statements other than α^*, and can thus only include α^* in its returned slate. Such a slate, however, would violate EJR for the coalition N with $T = \mathcal{U}$. General queries of type n-$\textsc{Generate}(\cdot,\cdot)$ allow a process to circumvent this issue by excluding previously generated statements from future queries. We will encounter more subtle forms of overshadowing in Section 3.2.

Using the full power of n-$\textsc{Generate}(\cdot,\cdot)$ queries, a polynomial-time democratic process can ensure EJR. We had originally conjectured this to be impossible since, among the EJR voting rules known as of last year, both the variants of Proportional Approval Voting [Aziz et al., 2017, 2018] and the Method of Equal Shares [Peters et al., 2021] centrally rely on weighting agents, which our generative query does not allow.[2] However, Brill and Peters [2023] recently proposed the *Greedy Justified Candidate Rule*, which guarantees EJR while being compatible with our generative queries (Process 2).

[2]The Greedy Cohesive Rule [Peters et al., 2021], does not require weighting agents, but requires simultaneously selecting multiple statements in a non-greedy way.

Proposition 2 (Brill and Peters, 2023, Prop. 7)**.** *Process 2 satisfies extended justified representation*[3] *in polynomial time in n and k, using queries of types n-*Generate$(\cdot,\cdot)$ *and* Approval$(\cdot,\cdot)$*.*

Process 2: Democratic Process for Extended Justified Representation [Brill and Peters, 2023]

$W \leftarrow \emptyset$
$\ell \leftarrow k$
while $\ell \geq 1$ **do**
  $S \leftarrow \{i \in N \mid \sum_{\alpha \in W} \text{Approval}(i, \alpha) < \ell\}$
  $\alpha \leftarrow \text{Generate}(S, W)$
  if $\sum_{i \in S} \text{Approval}(i, \alpha) \geq \ell n/k$ **then**
    $W \leftarrow W \cup \{\alpha\}$
  else
    $\ell \leftarrow \ell - 1$
return W

3.2. Size-Constrained Generative Queries

So far, our generative queries could generate maximally approved statements even if the queried set S of agents was as large as n. When implementing a generative query using an LLM, however, the prompt to the LLM must include, for each agent in S, enough information to extrapolate the agent's preference across the universe of statements. Since this information can easily take hundreds of tokens per agent in S, the context windows of current LLMs (GPT-4 base model, PaLM: 8K tokens, Llama: 4K tokens) severely limits the size of S. Even recent models with extended context windows (GPT-4: 32K tokens, Claude 2: 100K tokens) struggle to effectively use the entirety of their context window [Liu et al., 2023]. As a result, democratic processes might for now be restricted to generative queries for moderate $|S|$. In response to this limitation of current LLMs, we investigate in this section whether democratic processes can still ensure JR and EJR when generative queries are limited to sets of agents of size of at most some value t, substantially smaller than n.

Unfortunately, neither JR nor EJR can be guaranteed with such t-Generate$(\cdot,\cdot)$ queries in a satisfactory manner: Even if queries can refer to a constant fraction of agents, EJR cannot be guaranteed with any number of t-Generate$(\cdot,\cdot)$ and approval queries (Theorem 3). If we lower our demands on representation to JR, the situation improves only mildly: though there exists a simple democratic process that guarantees JR with n/k-Generate$(\cdot,\cdot)$ and approval queries (Theorem 4), it requires exponential time. This turns out to be necessary, since no democratic process can guarantee JR with subexponentially many $n/8$-Generate$(\cdot,\cdot)$ queries and any number of approval queries (Theorem 5). In Section 3.3, we will overcome these impossibilities by making assumptions on the structure of approval preferences.

Conceptually, our impossibility theorems are based on the idea of overshadowing, a simple example of which we previously discussed. In our proofs, we construct instances that have few "popular" statements and many "unpopular" statements with lower support. For a given set S

[3] It even satisfies the stronger *EJR+* [Brill and Peters, 2023].

of at most t agents, our instances will ensure that some unpopular statement has a support *within* S that is at least at large as that of any popular statement. Thus, the democratic process will be unable to access any popular statements through its generative queries, from which we derive a violation of representation. Using this proof idea, it is straight-forward to show that JR cannot be attained with $\frac{n}{k}(1-\frac{1}{k})$-GENERATE$(\cdot,\cdot)$ and approval queries (see Proposition 7 in Appendix A). Our larger lower bound for EJR, below, additionally requires a careful argument to show that a slate of unpopular statements necessarily violates EJR. We defer its proof to Appendix B.

Theorem 3. *No democratic process can guarantee extended justified representation with arbitrarily many* $\frac{n}{2}(1-\frac{1}{k})$*-*GENERATE$(\cdot,\cdot)$ *and* APPROVAL$(\cdot,\cdot)$ *queries.*

On the face of it, size-constrained generative queries are more promising for achieving JR, since there is a democratic process that achieves JR with queries of size $t = \lceil n/k \rceil$. This process is quite naïve, however: it iterates over all sets S of t agents, performs a t-GENERATE$(S, \emptyset)$ query for each, and adds the resulting statement to the slate iff this statement is approved by at least n/k agents who do not yet approve any statement in the slate (as determined by APPROVAL$(\cdot,\cdot)$ queries). While it is easy to show that this democratic process satisfies JR (proof deferred to Appendix B), its time complexity is obviously prohibitive.

Theorem 4. *There exists a democratic process that satisfies justified representation using (exponentially many) queries of type* $\lceil n/k \rceil$*-*GENERATE$(\cdot,\emptyset)$ *and* APPROVAL$(\cdot,\cdot)$*.*

In our most involved lower bound, we show that the exponential running time of this naïve democratic process is unavoidable, even if the generative queries can have linear size in n. Our proof must necessarily be more complicated than our previous lower bounds, in which we constructed explicit instances on which *any* democratic process with t-bounded generative queries had to violate representation. By contrast, for any approval instance, there exists a democratic process that satisfies JR in polynomial time on this instance; namely, a variant of the algorithm from Theorem 4 that skips all queries whose result the original algorithm did not add to the slate. Instead, we prove our impossibility (in Appendix B) by showing that, for any fixed polynomial-time algorithm, there exists an instance on which this algorithm violates JR, through an application of the probabilistic method.

Theorem 5. *No democratic process can guarantee justified representation with any number of* APPROVAL$(\cdot,\cdot)$ *queries and fewer than* $\frac{2}{k}e^{n/(12k)}$ *queries of type* $\frac{n}{8}$*-*GENERATE$(\cdot,\cdot)$*. As a corollary, if* $k \in O(n^{0.99})$*, then any democratic process guaranteeing JR with* $\frac{n}{8}$*-*GENERATE$(\cdot,\cdot)$ *and* APPROVAL$(\cdot,\cdot)$ *queries has exponential running time.*

3.3. Structured Approval Settings

Amid the last section's impossibilities, a silver lining is that the instances we used to prove them were contrived. Our impossibility proofs were constructed by drowning popular statements in an overwhelming number of relatively unpopular statements: For any set of agents (of a given size), there was a statement that was approved by only these agents and no one else. Since statements and preferences in the real world presumably have some structure, it seems highly implausible that such an abundance of orthogonal statements would exist for real-world populations. Note that, in contrast to work on spatial models of voting, we are not referring

to any fixed geometry of alternatives. Instead, we only require that preferences do not have infinite "complexity".

To be able to formally define complexity, we introduce the notion of a *statement space* $(\mathcal{U}, \mathcal{A})$, which consists of a universe of statements $\mathcal{U}$ and a set $\mathcal{A} \subseteq 2^{\mathcal{U}}$ of possible approval preferences. A statement-selection instance belongs to $(\mathcal{U}, \mathcal{A})$ if its universe of statements is $\mathcal{U}$ and if each agent i's set of approved statements A_i appears in $\mathcal{A}$.

To measure the complexity of a statement space, we borrow a fundamental complexity notion from learning theory, the *VC-dimension* (see, e.g., Vapnik 1998). We extend the definition of VC-dimension to statement spaces in the natural way: The VC-dimension of $(\mathcal{U}, \mathcal{A})$ is the largest $d \in \mathbb{N}$, for which there exist $A_1, A_2, \ldots, A_d \in \mathcal{A}$ such that, for any $\mathcal{I} \subseteq \{1, \ldots, d\}$, there is a statement $\alpha \in \mathcal{U}$ that lies in all $\{A_i\}_{i \in \mathcal{I}}$ and none of the $\{A_i\}_{i \notin \mathcal{I}}$. If no largest integer d exists, the VC-dimension is infinite. In other words, d is the size of the largest set of participants, such that for any subset of participants there is a statement that is approved by this subset an no one else. It seems unlikely that d would be huge in real-world settings, as it would imply, for instance, that we could find a statement such that people that lie at opposite sides of the space of opinions all approve of that statement, while people that lie in the middle all disapprove.

Theorem 6. *Let d be the VC-dimension of the statement space and $\delta > 0$ the maximum admissible error probability. There is a democratic process (taking d, δ as input) that runs in polynomial time in n, k (independent of d) and satisfies EJR with probability at least $1 - \delta$ using the following queries: Approval$(\cdot, \cdot)$ and t-Generate$(\cdot, \emptyset)$ with $t = O(k^4(d + \log \frac{k}{\delta}))$.*

For example, suppose that opinions on a discussion topic vary along three dimensions, say (1) socially conservative vs. liberal, (2) fiscally conservative vs. liberal, (3) religious vs. secular. Suppose that participants can be represented as points, and statements as axis-aligned boxes, such that a participant approves a statement iff the participant's point lies inside the statement's box. Since this statement space has VC-dimension 6 (see, e.g., Despres 2017), the democratic process from Theorem 6 produces EJR slates (up to a failure probability below 10^{-6}) using t-Generate$(\cdot, \cdot)$ queries with $t \leq \text{const} \cdot k^4(1 + \log(k))$ — a much smaller t than in our impossibility theorems if n is large.

Importantly, Theorem 6 extends to far more complicated structures, and does not require the structure to be known, but only (an upper bound on) the VC-dimension. If, for example, the set of statements $\mathcal{U}$ consists of all sequences of w many words in English (which has below 10^6 words), a naive upper bound on the VC-dimension is $d \leq \log_2(|\mathcal{U}|) \leq w \log_2(10^6)$. Thus, $t \leq \text{const} \cdot k^4(w + \log(k))$ suffices to virtually guarantee EJR.

In summary, despite the negative worst-case results from Section 3.2, it is highly likely that in reality the statement space has enough structure to allow for an EJR guarantee with high probability and a relatively small number of queries, which is *independent of the number of agents n*. This means that we can scale the democratic process to any number of participants, say to a national audience, for an LLM with bounded context window size.

4. Second Component: Empirical Validation of Queries

We established in the previous section that with access to *perfect* generative queries (for large enough t) and approval queries we can guarantee JR and EJR. We will now assess to what

extent we can implement these queries in practice using LLMs. In all our experiments, we implement the queries using GPT-4, a state-of-the-art LLM. In Appendix C, we describe our prompt-design process and provide example prompts and responses by GPT-4.

Datasets. We perform our evaluation on three real-world datasets. The first two, `minimum-wage` and `bowlinggreen`, describe deliberations hosted on Polis [2023], respectively on whether increasing the minimum wage would positively impact Seattle, WA and on possible ways to improve the city of Bowling Green, KY. In both cases, the platform allowed users to submit statements and to up- and downvote other users' statements. We employ a user's statements and some of their votes to inform the LLM about their preferences, and held-out votes serve as a ground truth for evaluating our approval query. Our third dataset, `changemyview`, is based on a post[4] from Reddit's /r/changemyview forum, discussing global warming, along with 23 top-level comments responding to the post. We selected this post because the comments were relatively high in quality and expressed distinct points of view. Since we do not have data on upvotes (as Reddit only shows cumulative votes), we hand-labeled approval between users and other users' statements. We summarize the datasets and provide details on data processing in Appendix D.

Evaluating approval queries. We begin by evaluating how accurately GPT-4 predicts approval votes. For each dataset, we draw a number of agents[5] and evaluate GPT-4's accuracy by predicting each agent's approval for 20 statements.[6] For the Polis datasets, GPT-4 receives information about the agent's preferences in the form of statements and votes (on statements not being evaluated) made by the agent. For `changemyview`, we provide only the comment made by the agent.[7] The following table shows, for each dataset, the mean and median accuracy across agents (for distributional information, see Appendix E):

dataset	mean acc.	median acc.
`minimumwage`	69.2%	70%
`bowlinggreen`	57.0%	55%
`bowlinggreen` (more preference data)	72.3%	70%
`changemyview`	84.0%	90%

The table shows that our implementation of the Approval$(\cdot,\cdot)$ query successfully detects patterns in agents' preferences, which allows it to predict their preferences on unseen statements. The accuracy appears to depend on the dataset's complexity and ambiguity: Preferences in `minimumwage` are largely clustered into proponents and opponents of the minimum wage, a pattern easily picked up by GPT-4. By contrast, the `bowlinggreen` dataset contains more diverse preferences, which explains why GPT-4, when given the same amount of information about an agent's preferences (1 comment, 2 votes), gives less accurate predictions. When we

[4] https://www.reddit.com/r/changemyview/comments/15bqufp

[5] For `minimumwage` and `changemyview` we use all agents (13 and 15 respectively). For `bowlinggreen` we sample 20.

[6] Statements might be drawn multiple times for the same agent, especially if the agent left few up- or downvotes. Balancing votes ensures that a constant classifier would achieve 50% accuracy.

[7] We do, however, include a handful of comments and approval votes (made by agents excluded from the evaluation) in the prompts as few-shot demonstrations of what approval means.

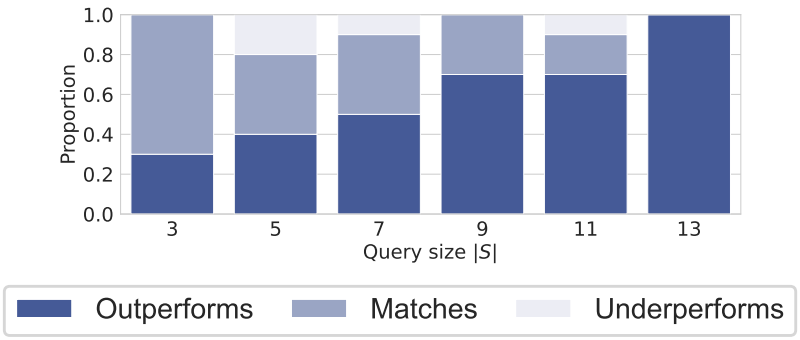


Figure 1: Out of 10 randomly chosen sets of agents S per query size $|S|$ (`changemyview` dataset), we give the rate at which the statement produced by $\textsc{Generate}(S, \emptyset)$ strictly outperforms, matches, or strictly underperforms the most-approved statement in the query input.

increase the amount of data about the agent from this default value to 3 comments and 20 given votes, the accuracy on `bowlinggreen` increases noticeably ("more preference data" in the table). This indicates that the prediction errors may be partially due to a lack of information available to GPT-4. Manual inspection confirms that most errors seem to be the result of ambiguity: even for a human it is unclear, based on the data provided to GPT-4, whether that person would approve of the given statement. If data were collected through a democratic process, we would mitigate this problem by requesting more detailed and diverse information from participants, perhaps even in an iterative manner. The high accuracy on `changemyview`, which is less ambiguous, is therefore encouraging. We expect that in an appropriately-designed democratic process, GPT-4 could reach even higher accuracy.

Evaluating generative queries for JR. We now investigate whether our generative queries succeed in producing new statements with high support. We start with generative queries where the set T of excluded comments is empty, i.e. $\textsc{Generate}(\cdot, \emptyset)$ queries, which by Proposition 1 are sufficient to guarantee JR. We implement $\textsc{Generate}(S, \emptyset)$ by providing GPT-4 with one comment made by each agent in S, and we ask it to generate a statement that would be approved by as many of the agents in S as possible. Since there are no ground-truth approval labels for these generated statements, we henceforth evaluate the support of statements relative to approval predicted by $\textsc{Approval}(\cdot, \cdot)$, which above proved to capture people's preferences well. Ideally, we would like to compare the statements generated by $\textsc{Generate}(\cdot, \emptyset)$ to the most-approved statement out of the universe of all possible statements, which is naturally not feasible. Instead, we compare to a weaker baseline: the comment with the most support out of all comments given as inputs to GPT-4. As Fig. 1 shows, $\textsc{Generate}(\cdot, \emptyset)$ often outperforms, and rarely underperforms this baseline. This also holds true on the other datasets, `minimumwage` and `bowlinggreen`.[8] This result shows that our generative queries indeed generate meaningful new statements that represent participants better than any of their own statements.

In a second experiment in Appendix F.2, we evaluate generative queries on a synthetic dataset, in which we embed a propositional logic problem into a toy application domain. In

[8] Analyses of other datasets and effect sizes are in Appendix F.1.

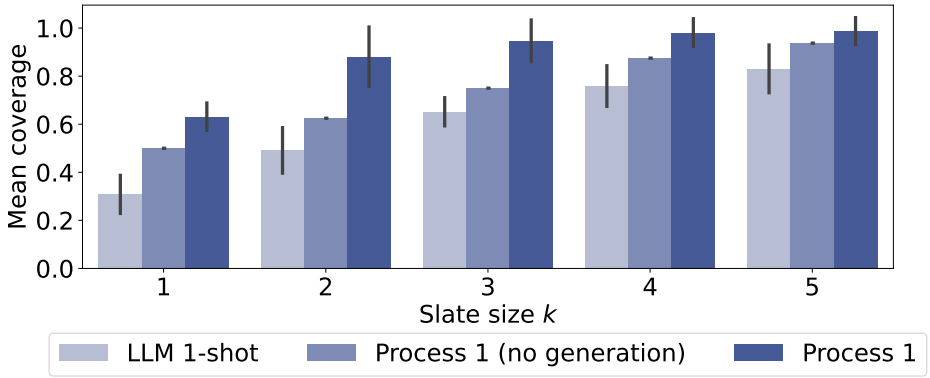


Figure 2: Fraction of agents covered as a function of k (`changemyview`, 20 runs per algorithm). Light blue: result of directly asking GPT-4 for a JR slate. Blue: classical greedy approval voting (i.e., identical to Process 1, but instead of generating statements, only picks among statements made by participants). Dark blue: our generative Process 1. Error bars show standard deviations.

this dataset, statements correspond to truth-value assignments setting a fixed number ℓ of variables to true, each agent corresponds to a monotone propositional formula, and agents approve a statement iff the truth-value assignment satisfies their formula. Thus, the generative query specifies a kind of maximum satisfiability problem, for which we can compute optimal solutions on small instances. Across our experiments, the generative query obtains around 90% of the maximum approval, clearly outperforming random assignments, which obtain around 70% approval. The generative query performs on par with a heuristic that selects the ℓ variables occurring most frequently in the formulas, which simultaneously shows an encouraging sensitivity to the query's input, but also limitations to its ability for elaborate logical reasoning.

Finally, we evaluate whether the promising performance of $\textsc{Approval}(\cdot,\cdot)$ and $\textsc{Generate}(\cdot,\emptyset)$ translates to Process 1, which relies on these queries. On the two Polis datasets, satisfying JR is easy since certain high-approval comments allow to cover almost all agents with one or two statements. By contrast, approvals in `changemyview` are sparser since the agents' opinions, based on longer and more precise statements, are more clearly delineated.

Across our evaluation on this dataset, the slates produced by Process 1 *never* admitted a JR violation for any of the users' statements, which is nontrivial since, e.g., a slate randomly selected among users' statements often violates JR in this way (Appendix G). To obtain a more fine-grained understanding of the performance of Process 1, we plot in Fig. 2 how many participants approve at least one statement in the generated slate. Interestingly, Process 1 performs better than a "classical" version of the same greedy algorithm, which can only select among the statements written by users but in exchange can precisely choose the comment with maximum marginal approval. This underscores that generation of new statements can lead to better representation than selecting among existing ones.

Evaluating generative queries with exclusion for EJR. As discussed in Section 3, EJR requires $\textsc{Generate}(S,T)$ queries with nonempty sets T, i.e., the LLM must be able to generate statements that are distinct from a set of statements T given as input. The performance of this

type of query is hard to measure, as it requires determining whether the generated statement is substantively distinct from those in T. We therefore resort to (1) a qualitative comparison of the generated statements to T and (2) plotting the approval rate of generated statements among the agents in S, as we keep excluding more and more statements. Both evaluations reveal a challenge with this type of query: Qualitative inspection reveals that the generated statements are often semantically close to the statements in T (see Appendix H.2). Further, as we include more and more statements in T, we do not observe a decrease in the approval rate of generated statements (see Appendix H.1). This also suggests that $\textsc{Generate}(S, T)$ fails to exclude statements in T, as otherwise one would expect the approval rate to drop eventually, as an increasing number of exclusion constraints are added.

This potential repetitiveness is particularly problematic for Process 2, which often makes multiple generative queries for the same group of agents, as it attempts to provide agents more than one approved statement. By contrast, the simpler Process 1 is less prone to repeated comments since its generative queries aim to satisfy only those agents who do not yet approve any selected statement, which encourages a diversity of statements. See Appendix I for details.

Conclusion. Our experiments paint an encouraging, albeit not yet conclusive, picture. Our LLM queries perform nontrivial tasks, which we could not have solved with classical algorithms. We saw that $\textsc{Approval}(\cdot, \cdot)$ can accurately extrapolate a person's preferences given limited amounts of data, that $\textsc{Generate}(\cdot, \emptyset)$ can generate novel statements whose approval exceeds that of any statement made by participants, and the resulting Process 1 shows promise.

The challenges we encountered with $\textsc{Generate}(\cdot, T)$ given $T \neq \emptyset$, and therefore Process 2, point towards interesting future research directions: Maybe few-shot learning, e.g. providing examples in the prompt, might work better for communicating abstract notions, such as equivalence of two statements, to LLMs. In particular, an adaptive method, in which we keep adding examples where the LLM made mistakes, seems to be a fruitful direction. Another interesting question is whether it is possible to guarantee EJR with generative queries that may be more well-suited for LLMs, instead of $\textsc{Generate}(\cdot, \cdot)$.

5. Discussion

The design of a democratic process for selecting representative statements is not just a hypothetical exercise, but one of practical relevance. In fact, we are close to launching a democratic process, with a nationally representative sample of participants, to elicit the population's stances on AI regulation. We thus conclude with three ways in which this study informs the design of real-world democratic processes:

First, our theoretical analysis highlighted two greedy voting rules augmented with generative queries (Processes 1 and 2) as promising structures for a practical democratic process. Indeed, these democratic processes are simple and efficient, guarantee representation axioms under perfect queries, and can be adapted through sampling if the electorate's size prevents generative queries from including all voters at once.

Second, since the queries' accuracy is sensitive to the clarity and quantity of the information describing the agents' preferences, eliciting this information well is crucial for a practical democratic process. For example, one should ask voters to write detailed and precise statements. If a process involves voting, voters should not only approve or disapprove, but also explain their

reasoning, to help the LLM generalize this vote to other statements. Finally, the information about an agent's preferences should cover all major aspects of the topic, which could be encouraged by exposing agents to other agents' statements in a targeted way.

Third, we derive from our experiments both optimism and some caution about LLMs' role in democratic processes. For the time being, LLMs navigate preferences and possible statements only imprecisely, which is all the more pertinent in light of well-documented biases of LLMs against groups of people [Basta et al., 2019, Kurita et al., 2019] and viewpoints [e.g. Hartmann et al., 2023], as well as their susceptibility to adversarial attacks such as prompt injection [Wallace et al., 2019]. Naturally, each of these vulnerabilities must be mitigated to the largest possible extent before deployment. In addition, we propose to "trust, but verify" in high-stakes settings: Once the democratic process has produced a slate of statements, one should return these statements to the voters and only adopt the slate if, based on ground-truth votes and any alternative statements proposed by the users, justified representation cannot be falsified.[9] In this way, the democratic process can tap into the power of LLMs while ensuring that the voters, not machines, have the final word.

Acknowledgments

We thank Nika Haghtalab and Abhishek Shetty for helpful pointers on how to apply sampling bounds to sampling without replacement. This work was partially supported by OpenAI through the "Democratic Inputs to AI" program and by the Office of Naval Research under grant N00014-20-1-2488.

References

H. Aziz, M. Brill, V. Conitzer, E. Elkind, R. Freeman, and T. Walsh. Justified representation in approval-based committee voting. *Social Choice and Welfare*, 42(2):461–485, 2017.

H. Aziz, E. Elkind, S. Huang, M. Lackner, L. Sánchez-Fernández, and P. Skowron. On the complexity of extended and proportional justified representation. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI)*, pages 902–909, 2018.

M. Bakker, M. Chadwick, H. Sheahan, M. Tessler, L. Campbell-Gillingham, J. Balaguer, N. McAleese, A. Glaese, J. Aslanides, M. Botvinick, and C. Summerfield. Fine-tuning language models to find agreement among humans with diverse preferences. In *Proceedings of the 36th Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

P. L. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

C. Basta, M. R. Costa-jussà, and N. Casas. Evaluating the underlying gender bias in contextualized word embeddings. In *Proceedings of the 1st Workshop on Gender Bias in Natural Language Processing*, 2019.

[9] If it is falsified, one might repeat the process, or select a slate that satisfies EJR with respect to the comments explicitly voted on. In this way, the LLM-augmented process does no worse than a classical process, even for arbitrary corruptions of LLM outputs.

M. Brill and J. Peters. Robust and verifiable proportionality axioms for multiwinner voting. In *Proceedings of the 14th ACM Conference on Economics and Computation (EC)*, 2023.

Y. Cabannes. Participatory budgeting: A significant contribution to participatory democracy. *Environment and Urbanization*, 16(1):27–46, 2004.

N. Clegg. Bringing people together to inform decision-making on generative AI. Blog post, 2023. Available at https://about.fb.com/news/2023/06/generative-ai-community-forum/.

C. J. J. Despres. The Vapnik-Chervonenkis dimension of cubes in $\mathbb{R}^d$. arXiv:1412.6612v3, 2017.

R. El-Yaniv and D. Pechyony. Transductive Rademacher Complexity and its Applications. *Journal of Artificial Intelligence Research*, 35:193–234, June 2009. ISSN 1076-9757. doi: 10.1613/jair.2587.

B. Flanigan, P. Gölz, A. Gupta, B. Hennig, and A. D. Procaccia. Fair algorithms for selecting citizens' assemblies. *Nature*, 596:548–552, 2021.

R. Freedman, J. Schaich Borg, W. Sinnott-Armstrong, J. P. Dickerson, and V. Conitzer. Adapting a kidney exchange algorithm to align with human values. *Artificial Intelligence*, 283, 2020.

D. Halpern, G. Kehne, A. D. Procaccia, J. Tucker-Foltz, and M. Wüthrich. Representation with incomplete votes. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence (AAAI)*, 2023.

J. Hartmann, J. Schwenzow, and M. Witte. The political ideology of conversational AI: Converging evidence on ChatGPT's pro-environmental, left-libertarian orientation. *arXiv:2301.01768*, 2023.

A. Konya, Y. L. Qiu, M. P. Varga, and A. Ovadya. Elicitation inference optimization for multi-principal-agent alignment. Manuscript, 2022.

K. Kurita, N. Vyas, A. Pareek, A. W. Black, and Y. Tsvetkov. Measuring bias in contextualized word representations. In *Proceedings of the 1st Workshop on Gender Bias in Natural Language Processing*, pages 166–172, 2019.

M. K. Lee, D. Kusbit, A. Kahng, J. T. Kim, X. Yuan, A. Chan, R. Noothigattu, D. See, S. Lee, C.-A. Psomas, and A. D. Procaccia. WeBuildAI: Participatory framework for fair and efficient algorithmic governance. In *Proceedings of the 22nd ACM Conference on Computer-Supported Cooperative Work and Social Computing (CSCW)*, article 181, 2019.

N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang. Lost in the middle: How language models use long contexts. arXiv:2307.03172, 2023.

R. Noothigattu, S. S. Gaikwad, E. Awad, S. Dsouza, I. Rahwan, P. Ravikumar, and A. D. Procaccia. A voting-based system for ethical decision making. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI)*, pages 1587–1594, 2018.

D. Peters, G. Pierczynski, and P. Skowron. Proportional participatory budgeting with additive utilities. In *Proceedings of the 35th Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 12726–12737, 2021.

Polis. Open Polis data, 2023. https://github.com/compdemocracy/openData, last accessed on 08/15/23.

S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

C. Small, M. Bjorkegren, T. Erkkilä, L. Shaw, and C. Megill. Polis: Scaling deliberation by mapping high dimensional opinion spaces. *Revista De Pensament I Anàlisi*, 26(2), 2021.

C. T. Small, I. Vendrov, E. Durmus, H. Homaei, E. Barry, J. Cornebise, T. Suzman, D. Ganguli, and C. Megill. Opportunities and risks of LLMs for scalable deliberation with Polis. arXiv:2306.11932, 2023.

V. N. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

E. Wallace, S. Feng, N. Kandpal, M. Gardner, and S. Singh. Universal adversarial triggers for attacking and analyzing NLP. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019.

W. Zaremba, A. Dhar, L. Ahmad, T. Eloundou, S. Santurkar, S. Agarwal, and J. Leung. Democratic inputs to AI. Blog post, 2023. Available at https://openai.com/blog/democratic-inputs-to-ai.

APPENDIX

A. Additional Results

Proposition 7. *No democratic process can guarantee justified representation with arbitrarily many $n/k\,(1-1/k)$-*GENERATE$(\cdot,\cdot)$ *and* APPROVAL$(\cdot,\cdot)$ *queries.*

Proof. Set $t \coloneqq n/k\,(1-1/k)$. Let n be some multiple of k^2, so that t is an integer. Suppose that there is one "popular" statement α, which is approved by all agents. Furthermore, for each set S of at most t agents, let there be infinitely many "unpopular" statements approved by S and no one else. With the right tie breaking, one can implement all t-GENERATE$(\cdot,\cdot)$ queries to return unpopular statements, from which it follows that the process will have to return a slate W entirely of unpopular comments.

Since each unpopular statement is approved by at most t agents, at most $k \cdot t = n\,(1-1/k) = n - n/k$ agents approve a statement in W. In other words, n/k agents do not approve of any statement in W, but approve α. This demonstrates a violation of JR. □

B. Deferred Proofs

Theorem 3. *No democratic process can guarantee extended justified representation with arbitrarily many $\frac{n}{2}(1-\frac{1}{k})$-*GENERATE$(\cdot,\cdot)$ *and* APPROVAL$(\cdot,\cdot)$ *queries.*

Proof. Let $t \coloneqq \frac{n}{2}(1-\frac{1}{k})$. As in the proof of Proposition 7, let every coalition S of size at most t have an infinite set of "unpopular" statements only they approve, and in addition let there be an infinite number of "popular" statements approved by all agents. We may answer all t-GENERATE$(\cdot,\cdot)$ queries with an unpopular statement, in which case any process as described in the theorem statement must return a slate W of only unpopular statements.

Let the utility of an agent for a slate of statements be defined as their number of approved statements. For a slate of unpopular statements, its (utilitarian) social welfare is at most $k\,t$, as each statement is approved by at most t agents.

Now consider the social welfare of any slate W' satisfying EJR in this instance. Observe that for any $\ell \in \{1,\dots,k-1\}$, there must be more than $n - \ell(n/k)$ agents with utility at least ℓ in the winning set. Indeed, otherwise there would be a coalition of size $\ell(n/k)$ who all approve fewer than ℓ statements in the winning set, but there exist ℓ popular statements they all approve, in contradiction to EJR.

What is the minimal social welfare of a slate W' satisfying EJR?

$$\begin{aligned}
\sum_{i\in N} |A_i \cap W'| &= \sum_{\ell=1}^{k} \left|\{i \in N \,|\, |A_i \cap W'| \geq \ell\}\right| \\
&> \sum_{\ell=1}^{k} \left(n - \ell(n/k)\right) \\
&= n\,k - \frac{n}{k} \cdot \sum_{\ell=1}^{k} \ell \\
&= n\,k - \frac{n}{k} \cdot \frac{k\,(k+1)}{2}
\end{aligned}$$

$$
\begin{aligned}
&= n\,k - n\,(k+1)/2 \\
&= \frac{n\,k}{2} - n/2 \\
&= \frac{n\,k}{2}\,(1 - 1/k) \\
&= k\,t \\
&\geq \sum_{i \in N} |A_i \cap W|.
\end{aligned}
$$

Thus, we have shown that any slate that satisfies EJR must have social welfare strictly larger than that of the slate returned by the process. We conclude that W must violate EJR. □

Theorem 4. *There exists a democratic process that satisfies justified representation using (exponentially many) queries of type* $\lceil n/k \rceil$*-*GENERATE$(\cdot, \emptyset)$ *and* APPROVAL$(\cdot, \cdot)$*.*

Proof. The process was already described in the body of the paper. Call the result of the process W. On the one hand, if $|W| \geq k$, then, by the definition of the process, the number of agents who approve at least one statement in W is at least $|W|\,\frac{n}{k}$. Thus, W is a valid slate (i.e., $|W| \leq k$), and it obviously satisfies JR since all n agents approve some comment in W.

On the other hand, suppose that $|W| < k$. If there was a JR violation, call the witnessing coalition S. By definition, S has size at least n/k (and thus at least $\lceil n/k \rceil$), and cannot approve any statement in W. Let $S' \subseteq S$ denote an arbitrary subset with exactly $\lceil n/k \rceil$ agents, and consider what happened when the process considered the set S'. The generative query t-GENERATE$(S', \emptyset)$ must have shown that no comment is satisfied by at least n/k agents, which contradicts the assumption that the coalition S witnesses a JR violation. □

Theorem 5. *No democratic process can guarantee justified representation with any number of* APPROVAL$(\cdot, \cdot)$ *queries and fewer than* $\frac{2}{k}e^{n/(12k)}$ *queries of type* $\frac{n}{8}$*-*GENERATE$(\cdot, \cdot)$*. As a corollary, if* $k \in O(n^{0.99})$*, then any democratic process guaranteeing JR with* $\frac{n}{8}$*-*GENERATE$(\cdot, \cdot)$ *and* APPROVAL$(\cdot, \cdot)$ *queries has exponential running time.*

Proof. Choose k to be an even integer and n as a multiple of 8, such that $t \coloneqq n/8$ is integer as well. Fix a process that makes fewer than $\frac{2}{k}\,e^{n/(12k)}$ many t-GENERATE$(\cdot, \cdot)$ and any number of approval queries. We will prove the claim using the probabilistic method: we will define a random instance and show that the process will fail JR with positive probability, which means that there exists a deterministic instance where the process fails JR.

For given n, k, construct our instance as follows: Each set S of $\frac{n}{2\,k}$ many agents has infinitely many "unpopular" statements that exactly they approve. Furthermore, each agent is uniformly and independently assigned a color in $\{1, 2, \ldots, k/2\}$, and all agents with the same color c approve a "popular" statement β_c. We will break ties in the queries according to some canonical ordering of statements.

Consider the trajectory of the process on an instance with just the unpopular statements, i.e., where each t-GENERATE$(\cdot, \cdot)$ query of the process for a set of agents S is answered by an unpopular statement approved by some subset $S' \subseteq S$, following the canonical ordering of unpopular comments.

Now, consider the random instance with unpopular and popular comments. We will show that, with positive probability, all t-GENERATE$(\cdot, \cdot)$ queries made by the process are still

answered with the canonical unpopular comments, which means that the process will follow the same trajectory as above. This will be the case if, for each t-GENERATE$(\cdot,\cdot)$ query corresponding to a set of agents S and for each color c, fewer than $\frac{n}{2\,k}$ agents in S have color c. For a specific S and c, the probability of this event can be upper-bounded using Chernoff as

$$\begin{aligned}
&\mathbb{P}\left[\text{at least } \frac{n}{2\,k} \text{ agents in } S \text{ have color } c\right] \\
&= \mathbb{P}\left[\mathsf{Binomial}(n/8, 2/k) \geq 2 \cdot \frac{n}{4\,k}\right] \\
&\leq \exp\left[-\frac{n}{12\,k}\right].
\end{aligned}$$

By a union bound, it follows that, with positive probability, this event does not occur in any of the fewer than $\frac{2}{k}\,e^{n/(12k)}$ queries, for any of the $\frac{k}{2}$ colors. This implies that there is an instance in the support of our random instance on which the trajectory of the process remains the same as if there were no popular statements and where, in particular, the process must return a slate of unpopular statements.

Finally, we show that, when the process only returns unpopular statements, it must violate JR. (This always hold for our random instance, ex post.) Since the unpopular statements only have a support of $\frac{n}{2\,k}$, no more than $\frac{n}{2}$ agents can be covered by a slate of k statements selected by the process. Therefore, there are at least $\frac{n}{2}$ uncovered agents, which are partitioned in some arbitrary manner across the $\frac{k}{2}$ many colors. By an averaging argument, there must be some color c with at least $\frac{n}{k}$ uncovered agents, which means that the process' output violates JR for β_c. □

Lemma 8 (Agnostic PAC learning for sampling without replacement)**.** *Let $\mathcal{H}$ be a hypothesis class, consisting of binary classifiers $h : \mathcal{X} \to \{-1, 1\}$ over some domain $\mathcal{X}$. Let $d < \infty$ denote the VC-dimension of $\mathcal{H}$. For a given hypothesis $h \in \mathcal{H}$, denote its 0–1 loss on a nonempty finite set $S \subseteq \mathcal{X} \times \{-1, 1\}$ of labeled datapoints by $L_S(h) \coloneqq \sum_{(x,y)\in S} \mathbb{1}\{h(x) \neq y\}/|S|$.*

*Let $D \subseteq \mathcal{X} \times \{-1, 1\}$ be a finite set of labeled datapoints. Consider a random process that chooses some number $m \leq |D|/2$ of labeled datapoints $S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$ from D uniformly and **without replacement**, and denote by $\hat{h}$ the empirical risk minimizer* $\operatorname{argmin}_{h\in\mathcal{H}} L_S(h)$*. For any $0 < \epsilon < 1, 0 < \delta < 1$, this process will satisfy*

$$L_D(\hat{h}) \leq \min_{h\in\mathcal{H}} L_D(h) + \epsilon \tag{1}$$

with probability at least $1 - \delta$, as long as

$$m \geq C \cdot \frac{d + \log 1/\delta}{\epsilon^2} \tag{2}$$

for some absolute constant C.

Proof. If $|D| \geq m^2/\delta$, the result will follow from the sampling bounds for i.i.d. samples. Note that we can implement the without-replacement drawing of S through rejection sampling, i.e., by drawing a sample of m datapoints uniformly *with* replacement, and re-drawing if this sample should contain any datapoint multiple times. We will consider only the first round of this rejection sampling. The probability that any two datapoints are identical is

at most $\sum_{i=0}^{m-1} i/|D| = \frac{m\,(m-1)}{2\,|D|} \leq \frac{m^2}{2\,|D|} \leq \delta/2$, so we reject with probability at most $\delta/2$. Moreover, since drawing with replacement is the same as drawing i.i.d. from the uniform distribution over D, we can apply a standard agnostic PAC learning bound [Shalev-Shwartz and Ben-David, 2014, Thm. 6.8] to show that the empirical risk minimizer $\hat{h}$ on the sample with replacement satisfies Eq. (1) with probability at least $1 - \delta/2$ as long as the constant in Eq. (2) is sufficiently large. By a union bound over both events, with probability at least $1 - \delta$, the with-replacement sample is not rejected and additionally satisfies Eq. (1), which proves the claim for our sampling process without replacement in the case of $|D| \geq m^2/\delta$.

From here on, suppose that $|D| < m^2/\delta$. Essentially, our claim will follow from Theorem 2 by El-Yaniv and Pechyony [2009], a bound on transductive learning, but we have to do some work to get their bound into our desired shape. We apply their Theorem 2 twice, with a value of δ that is half of the δ in our theorem, the full sample D, the hypothesis class $\mathcal{H}$, $\gamma = 1$, and setting m once to m and once to $|D| - m$ (swapping the role of sampled and not sampled datapoints). By union-bounding over both invocations and unfolding some definitions in the theorem, we obtain that, with probability at least $1 - \delta$, it holds for all $h \in \mathcal{H}$ that

$$L_{D\setminus S}(h) \leq L_S(h) + R_{trans}(\mathcal{H}) + slack \quad \text{and} \quad L_S(h) \leq L_{D\setminus S}(h) + R_{trans}(\mathcal{H}) + slack \tag{3}$$

where $R_{trans}(\mathcal{H})$ denotes the *transductive Rademacher complexity* of $\mathcal{H}$ on D, and *slack* is defined and bounded in the following.

The slack term is defined as

$$slack \coloneqq c_0\, q\, \sqrt{m} + \sqrt{\frac{s\,q}{2}\, \ln 1/\delta},$$

where $c_0 < 5.05$ is an absolute constant, $q \coloneqq \frac{1}{m} + \frac{1}{|D|-m} \leq \frac{2}{m}$, and $s \coloneqq \frac{|D|}{(|D|-1/2)\cdot(1-\frac{1}{2(|D|-m)})}$. Since m is a positive integer, $m \geq 1$, hence $|D|-m \geq m \geq 1$, and thus $s = \frac{|D|}{|D|-1/2} \cdot \frac{1}{1-\frac{1}{2\,(|D|-m)}} \leq 4/3 \cdot 2 = 8/3$. Thus,

$$slack \leq \frac{5.05 \cdot 2}{\sqrt{m}} + \sqrt{\frac{8/3}{m}\, \ln 1/\delta} = \frac{1}{\sqrt{m}}(10.10 + \sqrt{8/3\, \ln 1/\delta}). \tag{4}$$

Next, we bound the transductive Rademacher complexity, for which we require several definitions: Let $\vec{x} \in \mathcal{X}^{|D|}$ be a vector listing the first components (i.e., the unlabeled datapoints) for all members of D, in arbitrary order. For an index set $\mathcal{I} \subseteq \{1, \ldots, |D|\}$, let $\vec{x}_\mathcal{I} \in \mathcal{X}^{|\mathcal{I}|}$ be the restriction of $\vec{x}$ to the indices $\mathcal{I}$. For a hypothesis h and a vector $\vec{v}$, let $h(\vec{v})$ be the vector that results from applying h element-wise to the entries of $\vec{v}$. For any $t \in \mathbb{N}$, let Σ^t_{trans} denote the probability distribution over vectors of length t, whose entries are drawn i.i.d. and are equal to -1 with probability $\frac{m\,(|D|-m)}{|D|^2}$, equal to 1 with probability $\frac{m\,(|D|-m)}{|D|^2}$, and are 0 otherwise. Furthermore, let Σ^t_{ind} denote the probability distribution over vectors of length t whose entries are independently drawn and -1 or 1 with equal probability. Finally, denote by $\mathcal{B}$ the probability distribution over subsets of $\{1, \ldots, |D|\}$ in which each element is contained in the subset independently with probability $2\,\frac{m\,(|D|-m)}{|D|^2}$.

In this notation, El-Yaniv and Pechyony [2009, Def. 1 and p. 6] define the transductive Rademacher complexity $R_{trans}(\mathcal{H})$ as

$$(\tfrac{1}{m} + \tfrac{1}{|D|-m}) \cdot \mathbb{E}_{\vec{\sigma}\sim\Sigma^{|D|}_{trans}} \sup_{h\in\mathcal{H}} \vec{\sigma}^T h(\vec{x}).$$

Note that we can draw $\vec{\sigma}$ from $\Sigma_{trans}^{|D|}$ in two steps: we first draw the set of indices $\mathcal{I}$ from $\mathcal{B}$ whose entries in $\vec{\sigma}$ are nonzero, and then set $\vec{\sigma}$'s coordinates in $\mathcal{I}$ to -1 or 1 with equal probability. Therefore, we can equivalently write

$$R_{trans}(\mathcal{H}) = (\tfrac{1}{m} + \tfrac{1}{|D|-m}) \cdot \mathbb{E}_{\mathcal{I}\sim\mathcal{B}}\, \mathbb{E}_{\vec{\sigma}\sim\Sigma_{ind}^{|\mathcal{I}|}} \sup_{h\in\mathcal{H}} \vec{\sigma}^T h(\vec{x}_{\mathcal{I}}).$$

By Bartlett and Mendelson [2002, Lemma 4 & Thm. 6], $\mathbb{E}_{\vec{\sigma}\sim\Sigma_{ind}^{|\mathcal{I}|}} \sup_{h\in\mathcal{H}} \vec{\sigma}^T h(\vec{x}_{\mathcal{I}}) \leq c_1\sqrt{d|\mathcal{I}|}$ for some absolute constant c_1. Thus, we can bound

$$\begin{aligned}
R_{trans}(\mathcal{H}) &\leq c_1\,(\tfrac{1}{m} + \tfrac{1}{|D|-m}) \cdot \mathbb{E}_{\mathcal{I}\sim\mathcal{B}}\sqrt{d|\mathcal{I}|} \\
&\leq c_1\,\tfrac{2}{m} \cdot \mathbb{E}_{\mathcal{I}\sim\mathcal{B}}\sqrt{d|\mathcal{I}|} && m \leq |D| - m \\
&\leq \frac{2\,c_1\sqrt{d}}{m} \cdot \mathbb{E}_{t\sim Binomial\left(|D|, \frac{2\,m\,(|D|-m)}{|D|^2}\right)} \sqrt{t} \\
&\leq \frac{2\,c_1\sqrt{d}}{m} \left(\sqrt{6\,\tfrac{m\,(|D|-m)}{|D|}} + \mathbb{P}\left[Binomial\left(|D|, \tfrac{2\,m\,(|D|-m)}{|D|^2}\right) > 6\,\tfrac{m\,(|D|-m)}{|D|}\right] \cdot \sqrt{|D|}\right) \\
&\leq \frac{2\,c_1\sqrt{d}}{m} \left(\sqrt{6\,\tfrac{m\,(|D|-m)}{|D|}} + \exp(-2\,\tfrac{m\,(|D|-m)}{|D|}) \cdot \sqrt{|D|}\right) && \text{(Chernoff bound)} \\
&\leq \frac{2\,c_1\sqrt{d}}{m} \left(\sqrt{6\,m} + \exp\left(\tfrac{\ln|D|}{2} - m\right)\right) && (1/2 \leq \tfrac{|D|-m}{|D|} \leq 1) \\
&\leq \frac{2\,c_1\sqrt{d}}{m} \left(\sqrt{6\,m} + \exp\left(\tfrac{\ln m^2/\delta}{2} - m\right)\right) && (|D| < m^2/\delta) \\
&= \frac{2\,c_1\sqrt{d}}{m} \left(\sqrt{6\,m} + \exp\left(\tfrac{\ln 1/\delta}{2} + \ln m - m\right)\right) \\
&\leq \frac{2\,c_1\sqrt{d}}{m} \left(\sqrt{6\,m} + \exp\left(\tfrac{\ln 1/\delta}{2} - (1-1/e)m\right)\right) && (x - \ln x \geq (1-1/e)\,x))
\end{aligned}$$

By choosing a large enough constant in Eq. (2), we can ensure that $(1-1/e)m \geq \frac{\ln 1/\delta}{2}$. Then, we can continue:

$$\begin{aligned}
&\leq \frac{2\,c_1\sqrt{d}}{m} \left(\sqrt{6\,m} + e^0\right) \leq \tfrac{2\,c_1\sqrt{d}}{m}\,(\sqrt{6}+1)\sqrt{m} \\
&\leq \frac{c_2\sqrt{d}}{\sqrt{m}}, && (5)
\end{aligned}$$

where we set $c_2 \coloneqq 2\,(\sqrt{6}+1)\,c_1$. Putting together Eqs. (3) to (5), we obtain that, for all $h \in \mathcal{H}$,

$$L_{D\setminus S}(h) \leq L_S(h) + \tfrac{10.10+\sqrt{8/3\,\ln 1/\delta}+c_2\sqrt{d}}{\sqrt{m}} \quad \text{and} \quad L_S(h) \leq L_{D\setminus S}(h) + \tfrac{10.10+\sqrt{8/3\,\ln 1/\delta}+c_2\sqrt{d}}{\sqrt{m}}.$$

Finally, set $h^* \coloneqq \operatorname{argmin}_{h\in\mathcal{H}} L_D(h)$. Then, we bound

$$\begin{aligned}
&L_D(\hat{h}) - L_D(h^*) \\
&= \frac{m}{|D|}\left(L_S(\hat{h}) - L_S(h^*)\right) + \frac{|D|-m}{|D|}\left(L_{D\setminus S}(\hat{h}) - L_{D\setminus S}(h^*)\right)
\end{aligned}$$

$$
\begin{aligned}
&\leq \tfrac{m}{|D|}\left(L_S(\hat{h}) - L_S(h^*)\right) + \tfrac{|D|-m}{|D|}\left(L_S(\hat{h}) - L_S(h^*) + 2\,\tfrac{10.10+\sqrt{8/3\,\ln 1/\delta}+c_2\sqrt{d}}{\sqrt{m}}\right)\\
&= \underbrace{L_S(\hat{h}) - L_S(h^*)}_{\leq 0,\text{ by definition of }\hat{h}} + 2\,\tfrac{|D|-m}{|D|}\,\tfrac{10.10+\sqrt{8/3\,\ln 1/\delta}+c_2\sqrt{d}}{\sqrt{m}}\\
&\leq 2\,\frac{10.10+\sqrt{8/3\,\ln 1/\delta}+c_2\sqrt{d}}{\sqrt{m}}
\end{aligned}
$$

By choosing the constant in Eq. (2) large enough, we can ensure[10] that

$$m \geq \frac{4}{\epsilon^2}\cdot 3\,(10.10^2 + 8/3\,\ln 1/\delta + c_2^2 d).$$

By Cauchy's inequality, this implies that

$$m \geq \frac{4}{\epsilon^2}\cdot (10.10 + \sqrt{8/3\,\ln 1/\delta} + c_2\sqrt{d})^2,$$

and, by rearranging, that

$$\epsilon \geq 2\,\frac{10.10+\sqrt{8/3\ln 1/\delta}+c_2\sqrt{d}}{\sqrt{m}}.$$

Thus, with probability at least $1-\delta$, $\epsilon \geq L_D(\hat{h}) - L_D(h^*)$, as claimed. □

Theorem 6. *Let d be the VC-dimension of the statement space and $\delta > 0$ the maximum admissible error probability. There is a democratic process (taking d, δ as input) that runs in polynomial time in n, k (independent of d) and satisfies EJR with probability at least $1-\delta$ using the following queries: Approval$(\cdot,\cdot)$ and t-Generate$(\cdot,\emptyset)$ with $t = O(k^4(d+\log\frac{k}{\delta}))$.*

Proof. We will prove this theorem for $t \coloneqq 2\,C\,(k+1)^4\,(d+\log(2k/\delta))$, where the absolute constant C is the same as in Lemma 8. If $n \leq t$, we can simply run Process 2 to achieve EJR (and EJR+); thus, we from here on focus on the case where $n > t$.

The democratic process we study for this case is an adaptation of Process 2. In this variant, each call to Generate(S,T) is replaced with a call to the subprocedure approx-Generate(S,T), which draws a sample X of $t/2$ agents uniformly without replacement from N, and returns the result of $t/2$-Generate$(X \cap S, T)$. The other modification in this variant is that we replace the threshold condition in Line 6 of Process 2 by $\sum_{i\in S}$ Approval$(i,\alpha) > \frac{\ell n}{k+1}$.

We will not only show that this process satisfies EJR with high probability, but even the stronger axiom EJR+ [Brill and Peters, 2023, Def. 10]. A slate W satisfies EJR+ if there is no nonempty coalition $S \subseteq N$, no $\ell \in \mathbb{N}$, and no statement $\alpha \in \mathcal{U} \setminus W$ such that (i) $|S| \geq \ell\, n/k$, (ii) $\alpha \in A_i$ for all $i \in S$, and (iii) $|A_i \cap W| < \ell$ for all $i \in S$. Note that EJR+ implies EJR since any violation S, T of EJR[11] implies a violation of EJR+ for S, $\ell \coloneqq |T|$, and any $\alpha \in T \setminus W \neq \emptyset$.

[10] We may assume without loss of generality that $d \geq 1$, since, if $d = 0$, $\mathcal{H}$ only contains a single classifier and the claim holds trivially. If $d \geq 1$, we can upper bound the term $\frac{12\cdot 10.10^2}{\epsilon^2}$ by a multiple of $\frac{d}{\epsilon^2}$.

[11] These sets S, T come from the definition of EJR, and need not have any relation with the sets S, T in the generative queries of the previous paragraph.

To show that our democratic process satisfies EJR+ with high probability, we proceed in three steps: First, we show that our adaptation of Process 2 still never selects more than k statements. Second, we show that, if APPROX-GENERATE(S, T) is a sufficiently accurate proxy for GENERATE(S, T), the process will satisfy EJR+. Finally, we show that these queries indeed attain the necessary accuracy with high probability.

To show that the process never selects more than k statements, we repeat the basic charging argument by Brill and Peters [2023]: Imagine keeping track of an amount of money owed by each agent, starting at zero. Whenever the process adds a statement α to W, we spread 1 dollar of charge equally among $A^{-1}(\alpha) \cap S$. Note that this results in a charge strictly below $\frac{k+1}{\ell n}$ for each agent and that, for all previously added statements, the per-agent charge was also strictly below $\frac{k+1}{\ell n}$ since ℓ was at least as large then. Since any agent in S was previously charged at most $\ell - 1$ times, their cumulative charge so far is strictly below $\ell \frac{k+1}{\ell n} = \frac{k+1}{n}$. By summing up the cumulative charges across all agents, we obtain a sum strictly below $k + 1$, which shows that no more than k statements have been added.

For convenience, define $n_{S,T}(R)$ to be the number of agents in S who approve of the statement returned by GENERATE(R, T). We will show that EJR+ will be satisfied if, for any call to APPROX-GENERATE(S, T), it holds that

$$n_{S,T}(S) - n_{S,T}(X \cap S) < n \left(\frac{1}{k} - \frac{1}{k+1} \right) = \frac{n}{k\,(k+1)}. \tag{6}$$

Indeed, suppose that the final slate W did not satisfy EJR+, and that this was witnessed by a coalition S', $\ell' \in \mathbb{N}$, and $\alpha' \in \mathcal{U} \setminus W$. Then, consider the call to APPROX-GENERATE(S, T) that caused the algorithm to decrement ℓ from ℓ' to $\ell' - 1$ because the resulting statement α had at most $\ell n/(k + 1)$ supporters among S, i.e., $n_{S,W}(X \cap S) \leq \ell n/(k + 1)$. Note that $S' \subseteq S$ and thus $n_{S,W}(S) \geq n_{S',W}(S') \geq |A^{-1}(\alpha) \cap S'| = |S'| \geq \ell n/k$. But then, $n_{S,W}(S) - n_{S,W}(X \cap S) \geq \frac{\ell n}{k} - \frac{\ell n}{k+1} \geq \frac{n}{k\,(k+1)}$, which contradicts the assumption in Eq. (6), which therefore implies EJR+.

It remains to show that Eq. (6) indeed holds, with probability at least $1 - \delta$, for all APPROX-GENERATE(S, T) calls made in the process. For each such call, we will apply the PAC sampling bound in Lemma 8.[12] For this, we treat the statements α in $\mathcal{U} \setminus T$ as binary hypotheses over N, specifically as

$$h_\alpha(i) := \begin{cases} +1 & \text{if } i \in S \text{ and } \alpha \in A_i \\ -1 & \text{otherwise.} \end{cases}$$

The VC-dimension of this hypothesis class is at most d. To see this, note that d corresponds to the VC-dimension of this hypothesis class with $T = \emptyset$ and $S = N$. Clearly, excluding hypotheses (i.e., $T \neq \emptyset$) cannot increase the VC-dimension. Further, forcing the datapoints that are not in S to be labeled negatively by all hypotheses also cannot increase the VC-dimension, since they cannot be shattered.

Imagine that our underlying dataset D consists of all points N, all with a label of $+1$. Then, observe that for any α,

$$|A^{-1}(\alpha) \cap S|/n = 1 - L_D(h_\alpha),$$

[12] Note that we cannot simply apply sampling bounds from the literature, due to the complication that we sample *without replacement*, since our generative query does not allow to include the same agent multiple times in the set. For this reason, we prove in Lemma 8 that the same sample bound as for i.i.d. sampling also holds for sampling without replacement.

where $L_D(h) \coloneqq \sum_{(x,y)\in D} \mathbb{1}\{h(x) \neq y\}/|D|$ denotes the 0–1 loss of h on D. Due to this connection, we can think of APPROX-GENERATE(S,T) as returning the empirical risk minimizer for a sample of size $t/2$ and of GENERATE(S,T) as returning the actual hypothesis with minimal risk. By applying Lemma 8 with sample size $t/2$ (recall the definition of t from the beginning of the proof) and $\epsilon = 1/(k+1)^2$, we obtain that, with probability at least $1 - \frac{\delta}{2k}$, it holds that

$$L_D(h_{\text{APPROX-GENERATE}(S,T)}) \leq L_D(h_{\text{GENERATE}(S,T)}) + \frac{1}{(k+1)^2},$$

which means equivalently that

$$\underbrace{|A^{-1}(\text{APPROX-GENERATE}(S,T)) \cap S|}_{n_{S,T}(X \cap S)} /n \geq \underbrace{|A^{-1}(\text{GENERATE}(S,T)) \cap S|}_{n_{S,T}(S)} /n - \frac{1}{(k+1)^2},$$

which clearly implies Eq. (6). Finally, observe that (our variant of) Process 2 makes at most $2\,k$ generative queries, since each such query either adds an element to the slate or decreases ℓ, each of which happens at most k times. The theorem follows from a union bound over these $2\,k$ queries. □

C. Prompt-design Process

Approval queries. For APPROVAL$(\cdot,\cdot)$ queries, relatively simple prompts suffice. For example, for `minimumwage` and `bowlinggreen`, we use the following prompt:

```
Suppose a person holds the following opinion: "{self.comment}". Would they agree with this:
"{other.comment}"? Answer AGREE if they would agree and DISAGREE if they would disagree.
Only put AGREE if you are reasonably confident that they would agree, as a safe default
you should put DISAGREE. Your output should always be AGREE or DISAGREE and nothing else.
```

We started with the core prompt (first three sentences, in black). Upon noticing that the model would sometimes produce outputs other than AGREE or DISAGREE, we added the last sentence (in red). Finally, upon noticing that the model would err on the side of returning AGREE, especially if `{other.comment}` contains widely agreed-upon platitudes, we added the second to last sentence (in blue).

We also tried simple Chain-of-Thought prompts, but abandoned this direction due to increased costs and lack of discernable gains in performance.

Generative queries. For GENERATE$(\cdot,\cdot)$, we observed higher quality model outputs when using more complex Chain-of-Thought prompts. For example, for `changemyview`, we use the following prompt for GENERATE$(\cdot,\emptyset)$:

```
Below is a list of people with various opinions on a topic. Follow the following steps:
1. Determine what topic they are discussing. What are the key axes of disagreement here?
2. For each person, give them a number and write down what their main beliefs are, and how
they compare to the crowd.
3. Reflect on which opinions are commonly held and which opinions are less commonly held.
Where might people find common ground?
4. Write a single specific statement that the most possible people would feel FULLY
REPRESENTED by. You should try to find common ground among the different opinions. But
```

```
also, the statement you generate should not be wishy-washy: it should represent a single
specific viewpoint on the topic. It should sound like an opinionated statement one of the
people in the list would make. Of course, not everyone can be fully represented by a single
statement, I just want you to generate one statement which fully represents the most people
possible. (Don't worry that some people will not be represented yet, I will ask you later to
generate statements which represent the remaining people.) Now here is the list of people
and their opinions:
{agent_opinion_list}
This concludes the list of people, now start following the steps. Your output should be
a Python dictionary with key-val pairs: "STEP 1": """<writing for step 1>""", "STEP 2":
"""<writing for step 2>""", "STEP 3":, """<writing for step 3>""", "STEP 4": """<writing
for step 4>""". Format your Python dict with curly braces at the beginning and end and
NEVER write newlines. Only write the Python dict and nothing else.
```

We started with the core prompt ("Below is a list [...] FULLY REPRESENTED by.", in black). Upon noticing that the model would produce insufficiently specific statements (e.g. "Many people believe X is important, but we also need to consider not X"), we added the next two sentences (in blue). To further encourage specificity we added the text "It should sound like [...] remaining people.)" (in red).

Separately, we also tweaked the parsing instructions (at the end of the prompt, in orange) to minimize the probability of unparseable model outputs. Here it proved beneficial to encourage the use of triple quotes and discourage the use of newlines.

To avoid overfitting, we tweaked prompts based on the results of small pilot experiments, and refrained from further altering prompts after running our full-scale experiments. For a complete list of prompts used in our experiments see Appendix J.

D. Data Processing

Polis datasets. Data from 8 Polis converstations has been made publicly available[13]. Using Polis moderation data, we filter for high-quality comments[14] and participants[15]. We focus on two such conversations, 15-per-hour-seattle (our `minimumwage`) and bowling-green.american-assembly (our `bowlinggreen`), because they are in English and contain high-quality on-topic comments. It is useful to consider both conversations because they occupy two ends of a spectrum: `bowlinggreen` has 280 high-quality participants, but comments often are uncontroversial (average comment upvote rate 72%) and touch on disjoint issues. Meanwhile, `minimumwage` only has 17 high-quality participants, but comments are more controversial (average comment upvote rate 52%) and focus on a single issue.

`Changemyview` dataset. We retrieved all top-level comments from a Reddit post from the subreddit /r/changemyview[16]. The original poster explains their view on climate change (specifically, that the only possible solution to climate change would be a technological breakthrough), and the comments on the post respond to and challenge this view. We chose

[13] https://github.com/compdemocracy/openData

[14] A high-quality comment is a comment with moderation flag +1 ("approved").

[15] A high-quality participant is a participant who wrote at least one comment with a +1 ("approved") moderation flag.

[16] https://www.reddit.com/r/changemyview/comments/15bqufp/cmv_global_warming_will_not_be_solved_by_small/

this post because the responses are high-quality and express well-defined stances regarding the original post.

We only keep comments that contain between 200 and 500 characters and do not contain any urls or quotations from the original post. We hence end up with 23 high-quality posts of similar length. We then hand-label for any pair of comments whether the person that made the first comment would feel represented by the comment made by the second person.

E. Evaluating Approval Queries

Comparison across datasets. The main body of the paper reports summary statistics for this comparison. See Fig. 3 for full distributional information.

Robustness to choice of prompt. Our prompt for $\textsc{Approval}(\cdot, \cdot)$ on the `changemyview` dataset uses few-shot demonstrations to explain to the model what is meant by approval in this setting. Specifically, 5 participant-vote pairs are randomly sampled to include as few-shot demonstrations in the prompt, and we test the accuracy of $\textsc{Approval}(\cdot, \cdot)$ on the remaining participants[17]. In Fig. 4 we demonstrate robustness of our approval query to this choice of few-shot examples (specifically, the seed). Aggregated across seeds, the mean accuracy of our approval query is 86.7%, with a standard deviation of 11.0%.

F. Evaluating Generative Queries

F.1. Comparison with Most-Approved Input Statement

Results on other datasets. See Fig. 6 for the experiment in Fig. 1 run on `minimumwage` and `bowlinggreen`. For a more fine-grained analysis of the gains from using $\textsc{Generate}(\cdot, \emptyset)$ see Fig. 5.

Analysis of `bowlinggreen` outputs. The high performance of $\textsc{Generate}(\cdot, \emptyset)$ on `bowlinggreen` is likely due to the specific nature of the dataset. In `bowlinggreen`, participants suggest ways to improve their city. Representative comments include:

- "Complete the ky. 185 restructuring project."
- "Bring a discount grocer to the downtown area."
- "Expand Nashville Rd all the way past Chaney's & Buchanan park. [...]"
- "Post-secondary training programs for special education students."

Typical $\textsc{Generate}(\cdot, \emptyset)$ outputs are able to please nearly all such participants simultaneously with general statements such as:

```
To create a more inclusive and thriving community, we should invest in education, infras-
tructure improvements, and promote equal opportunities for all residents, regardless of
background or abilities.
```

By contrast, $\textsc{Generate}(\cdot, \emptyset)$ faces a more meaningful challenge on the `minimumwage` and `changemyview` datasets, in which participants' opinions can be in direct opposition.

[17]The number of remaining participants will be between 13 and 18, depending on the choice of seed.

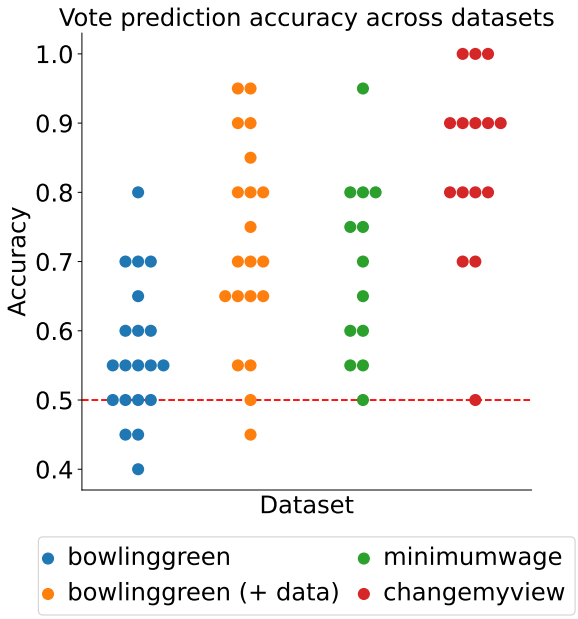


Figure 3: Evaluating approval queries on `minimumwage`, `bowlinggreen`, and `changemyview`. Each dot denotes vote prediction accuracy for a fixed participant over 20 samples (or 10 for `changemyview`).

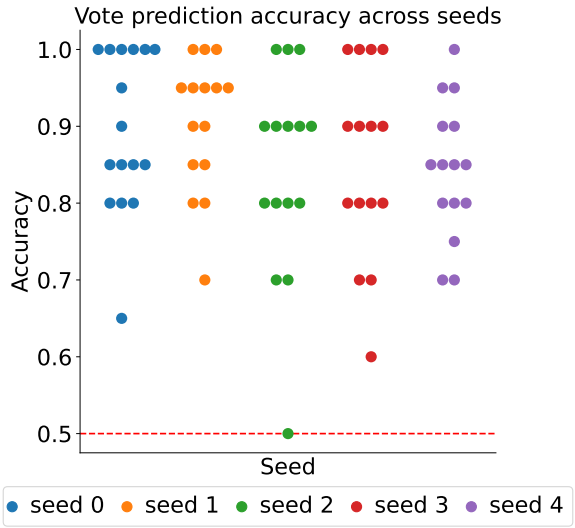


Figure 4: Evaluating approval queries on `changemyview`, for different choices of few-shot examples (determined by the random seed). Each dot denotes vote prediction accuracy for a fixed participant over 20 samples.

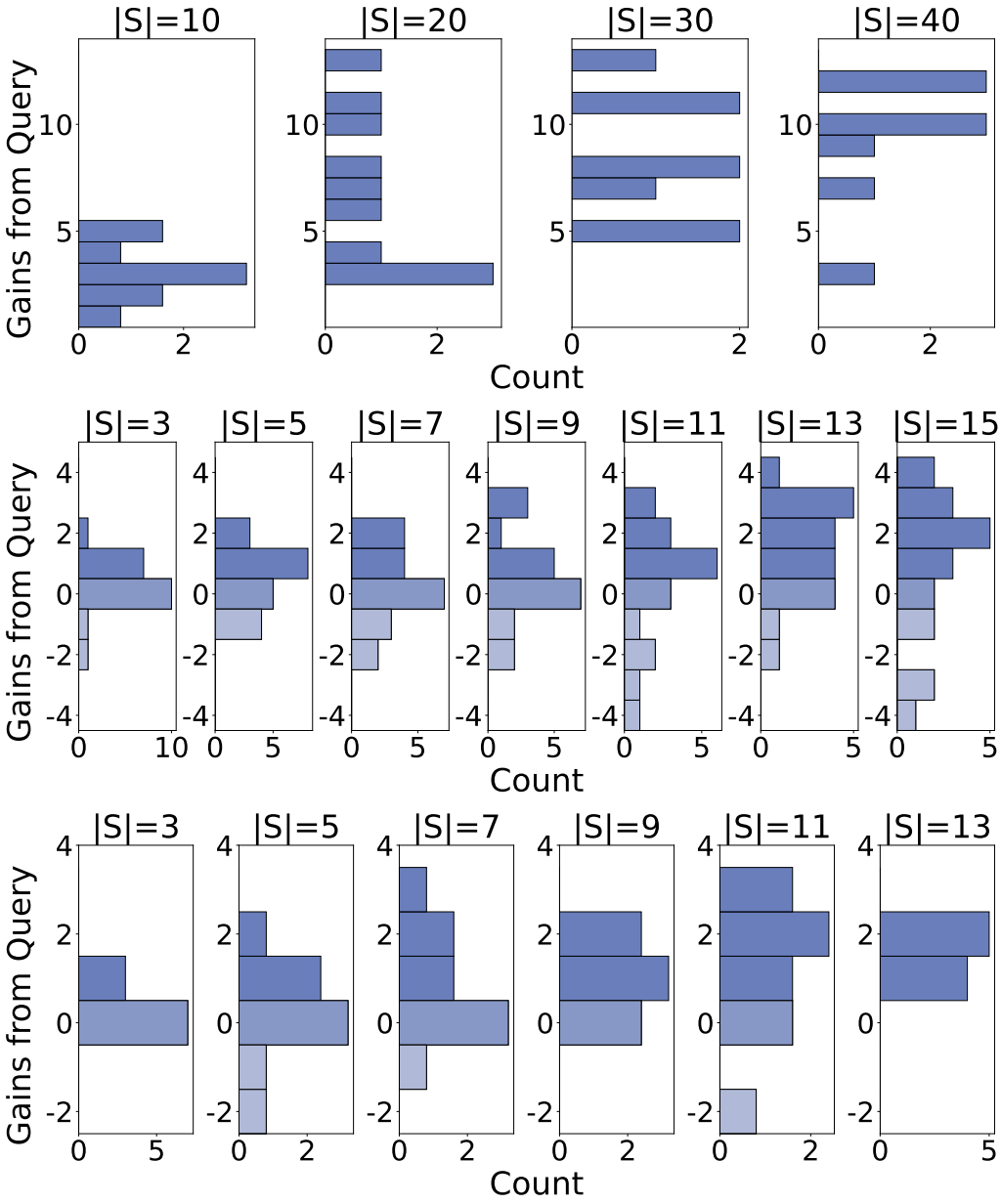


Figure 5: Fine-grained evaluation data for GENERATE$(\cdot, \emptyset)$ (above: `bowlinggreen`, middle: `minimumwage`, below: `changemyview`). For the experiment description, see Fig. 1 and Fig. 6. Here, we show the degree to which GENERATE$(\cdot, \emptyset)$ out- or underperforms the most-approved statement in the query input by measuring the difference between the number of participants who approve the query output and the number of participants who approve the most-approved input statement.

F.2. Measuring Performance on Synthetic Data

Formula generation. Recall that agents' preferences correspond to simple propositional-logic formulas. For this experiment, we restrict ourselves to two shapes of formulas: "$(A \wedge B) \vee C$" and "$A \wedge (B \vee C)$".

In each experiment, we have a number m of distinct subjects and tell the LLM to select ℓ of them. We then i.i.d. generate n many random agents. Each agent has either type of formula with probability 1/2, and the blanks "A", "B", and "C" are filled in with three uniform draws from the m subjects, without replacement. Note that the there are no negations in these formulas.

Prompt experimentation. We had originally tried a variant of the prompt that tells the LLM to interpret the students preferences "as propositional logic" and gave two examples for how to evaluate a preference on an assignment. Since this did not improve performance, we dropped this additional explanation. See Appendix J.4 for the prompt.

Parameters evaluated. We predominantly focused on the case $m = 10$ and $\ell = 5$. These numbers were not optimized, but they struck us as in a good balance between not having too large a number of subjects, and having a number of possible assignments $\binom{10}{5} = 252$, which is small enough to allow us to enumerate it but large enough that the random assignment is not too frequently optimal. We then vary the number of agents n for values $5, 10, 20, 40$.

Results. See Fig. 7.

G. Evaluating Process 1

Frequency of JR violations. In the table below we report frequency of JR violations (see Fig. 2 for experimental details).

algorithm	$k = 2$	$k = 3$	$k = 4$	$k = 5$
Process 1	0.0%	0.0%	0.0%	0.0%
Process 1 (w/o g.)	0.0%	0.0%	0.0%	0.0%
LLM 1-shot	0.0%	0.0%	**7.1%**	0.0%
Random slate	0.0%	**5.0%**	**35%**	**10%**

A more fine-grained way to measure how close a slate is to violating JR is to measure the size of the largest cohesive uncovered coalition, that is, the maximum number of agents unsatisfied with the slate who all approve a common statement outside the slate. Here too Process 1 outperforms competing baselines (see Fig. 8).

H. Evaluating Generative Queries ($T \neq \emptyset$)

To investigate the performance of GENERATE(S, T) for $T \neq 0$, we run the following experiment (Experiment 3).

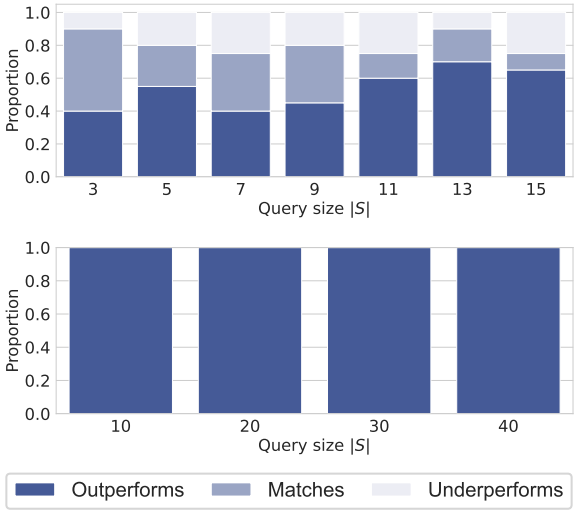
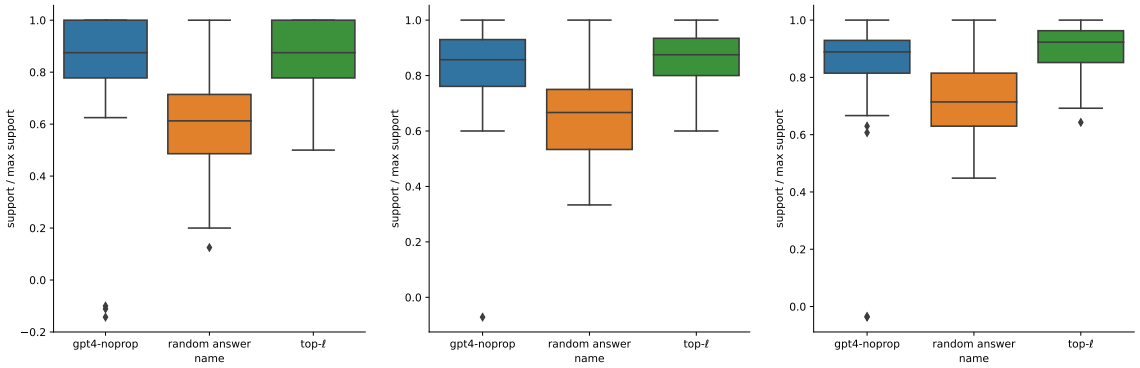


Figure 6: Out of 10 randomly chosen sets of agents S per query size $|S|$ (above: `minimum-wage`, below: `bowlinggreen`), we give the rate at which the statement produced by GENERATE$(\cdot, \emptyset)$ strictly outperforms, matches, or strictly underperforms the most-approved statement in the query input. See also Fig. 1.



(a) $n = 10$ (b) $n = 20$ (c) $n = 40$

Figure 7: Ratio between the level of support of the alternative generated by GPT-4 divided by ground-truth maximal support of any alternative.

Experiment 3: Evaluating GENERATE(S, T)

```
S ← set of statements
T ← ∅
for t = 1, ..., k do
    α_t ← GENERATE(S, T)
    T ← T ∪ {α_t}
return T
```

In Appendix H.1, we quantitatively evaluate whether GENERATE(S, T) is in fact excluding statements in T by plotting APPROVAL($\cdot, \alpha_t$) as a function of t, and in Appendix H.2, we further investigate this question by manually inspecting outputs of Experiment 3.

H.1. Measuring Approval of Generated Statements

To quantify the extent to which GENERATE(S, T) fails to exclude statements in T, we study the dependence of the average approval rate of statement α_t, given by $\frac{1}{N}\sum_{i \in N}$ APPROVAL(i, α_t), on t. Fig. 9 demonstrates that the average approval rate of α_t does not fall with t: instead, this function is roughly constant.

H.2. Manual Inspection of Generated Statements

Despite significant prompting efforts, GENERATE(S, T) often fails to meaningfully exclude statements in T. Below is a representative run of Experiment 3 with $k = 5$ and S a subset of 15 `changemyview` comments[18]. Sentence parts with similar meanings are written in similar colors (and text formatting).

(1) Climate change is an urgent crisis that **requires not just technological breakthroughs**, *but also transformative and systemic changes in society, politics, and the economy*, along with cooperation and strong leadership to implement effective solutions.

(2) Resolving the global warming crisis **demands technological advancement** *in conjunction with transformative social and economic reforms*, as well as collaborative efforts to overcome entrenched political barriers and establish sustainable solutions.

(3) Addressing global warming and climate change calls for a collective effort to **embrace existing technological advancements**, *systemic alterations in our economies,* and cooperation among countries for a greener, more sustainable future.

(4) To truly address global warming, we must act on multiple levels, **integrating existing technological advancements**, *reevaluating societal values and power structures,* while nurturing global cooperation to implement far-reaching solutions.

(5) Tackling global warming effectively demands a combination of **utilizing current technological advancements**, *recognizing the need for large-scale societal shifts,* and fostering cooperation across nations to implement comprehensive and sustainable strategies.

The generated statements are near-identical, despite explicit prompting[19] encouraging the model to produce statements very different from those in the excluded set T. Downstream this

[18] With seed 423.

[19] Specifically, the prompt includes this passage: *"Finally there is one more aspect to the challenge: the statement*

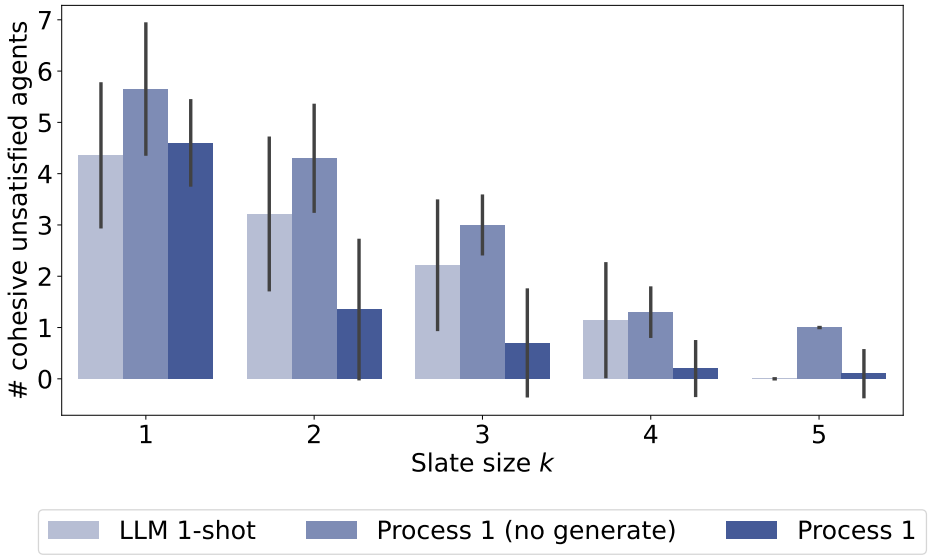


Figure 8: Largest cohesive coalition of unsatisfied agents, as a function of k. (Lower is better.) Slates are generated as in Fig. 2.

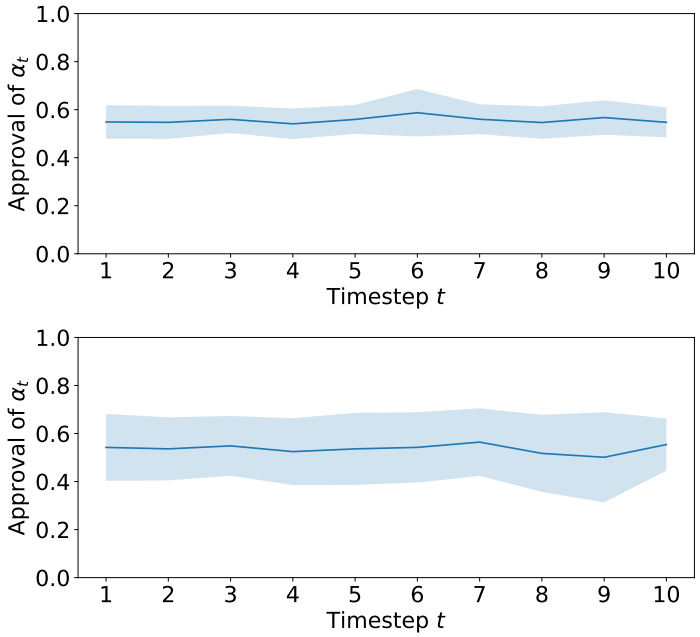


Figure 9: Averaged across 10 random seeds, and 10 runs of Experiment 3 per random seed, we give the proportion of agents i who approve of α_t (according to APPROVAL$(\cdot, \alpha_t)$) for each $t = 1, \ldots, 10$. Both plots use `changemyview` with different GENERATE$(\cdot, \cdot)$ prompts: few-shot above, Chain-of-Thought below. Shaded region indicates 1 standard deviation. In both cases we observe a flat curve, indicating that GENERATE$(\cdot, T)$ is not effectively excluding statements in T.

Figure 10: An example run of Process 1 on `changemyview`. Each column corresponds to an agent (i.e., a comment), and each row correponds to a selected statement (selected in order from top to bottom). The blue squares indicate that the agent approves of the statement. The corresponding comments are in Table 1.

results in difficulties in implementing Process 2 (see Appendix I for additional discussion).

I. Qualitative Comparison of Process 2 with Process 1

Recall that Process 2 satisfies stronger guarantees than Process 1, as it attempts to represent each agent with multiple comments, if possible. This has an unintended side effect here, since GENERATE$(\cdot, T)$ often fails to exclude statements in T: It keeps catering to the same population (see also Appendix H). We observe this for instance in Fig. 11, where we see that the same agents are provided with statements multiple times, while others do not obtain any statement at all. Looking at the corresponding comments in Table 2, we observe that indeed all generated comments are quite similar and very broad.

In contrast, Process 1, which is theoretically weaker, does not suffer from this problem to the same extent in practice. In Fig. 10, we see that it caters to different agents at each step, as agents that are already satisfied are simply ignored in future queries. Therefore, we also see that the corresponding comments in Table 1 are more distinct and specific.

J. Complete List of Prompts

J.1. Polis prompts

Below is a complete list of the prompts we used for our experiments. The blue text highlights specific instructions of the structure of our anticipated response for the LLM. The red text explains how the ideal repsonse should sound. The orange text includes the text for the expected format of the final output.

APPROVAL$(\cdot, \cdot)$ for Polis data

you generate should not be anything close to the statements written below in the forbidden list. If your statement at all resembles a statement in the forbidden list, that is very bad. So this is the challenge: you must represent the participants in the above list, with a statement which is nothing like anything in the below forbidden list."

Figure 11: An example run of Process 2 on `changemyview`. Each column corresponds to an agent (i.e., a comment), and each row correponds to a selected statement (selected in order from top to bottom). The blue squares indicate that the agent approves of the statement. The corresponding comments are in Table 2

```
Suppose a person holds the following opinion: "{self.comment}". Would they agree with this:
"{other.comment}"? Answer AGREE if they would agree and DISAGREE if they would disagree.
Only put AGREE if you are reasonably confident that they would agree, as a safe default
you should put DISAGREE. Your output should always be AGREE or DISAGREE and nothing else.
```

GENERATE$(\cdot, \emptyset)$

```
Below is a list of people with various opinions on a topic. Follow the following steps:
  1. Determine what topic they are discussing. What are the key axes of disagreement here?
  2. For each person, give them a number and write down what their main beliefs are, and
how they compare to the crowd.
  3. Reflect on which opinions are commonly held and which opinions are less commonly held.
Where might people find common ground?
  4. Write a single specific statement that the most possible people would agree with. You
should try to find common ground among the different opinions. But also, the statement you
generate should not be wishy-washy: it should represent a single specific viewpoint on the
topic. It should sound like an opinionated statement one of the people in the list would
make.
  {agent_opinion_list}
  This concludes the list of people, now start following the steps. Your output should
be a Python dictionary with key-val pairs:""STEP 1"": """"""<writing for step 1>""""""",
""STEP 2"": """"""<writing for step 2>""""""", ""STEP 3"":, """"""<writing for step 3>""""""",
""STEP 4"": """"""<writing for step 4>""""""". Output only the Python dict and nothing else.
```

J.2. Changemyview prompts

APPROVAL$(\cdot, \cdot)$

```
Consider a group of people deliberating on a topic. In particular, they are responding the
the following comment:
  "{post_title}"
  Your tasks is to determine whether a particular person would feel represented by a given
statement. Below I will provide some example tasks, in which I will give you the following
information: 1) an opinion some particular person has, and 2) a statement somebody else made.
Given this information, the task is to determine whether the person would feel represented
by the statement, based on the opinion they expressed. This is a yes-no question, the only
two valid answers to the task are "Yes" (the person would feel represented by the statement)
or "No" (the person would not feel represented by the statement).
```

	Comment
Member 1	To effectively address global warming, we must rely on a combination of technological innovations, government regulation, and community-based solutions, all while acknowledging our collective responsibility to make progressive changes in our lifestyles.
Member 2	Global warming is an ongoing issue, to address it effectively we need to overcome societal inertia and work together to implement existing solutions, though skepticism about the role of government remains.
Member 3	The only way to address global warming and climate change at this point might be through a major technological breakthrough, as current efforts seem insufficient and potentially influenced by political and economic motives.
Member 4	The planet is facing the severe consequences of climate change, and unless drastic actions are taken, it may be impossible to reverse the damage we've caused, leading to a devastating impact on the environment and future generations.
Member 5	It's unlikely that human societies will effectively address climate change, as history shows humans often mismanage resources, and governments consistently prioritize maintaining power over finding sustainable solutions.

Table 1: Comments selected by Process 1, corresponding to Fig. 10

```
  {examples}
  This concludes the example tasks. Now it is your turn to solve a task. Your response
should be just "Yes" or "No" and nothing else.
  {incomplete_example}
```

GENERATE$(\cdot, \emptyset)$: Few-shot Prompt

```
Consider a group of people deliberating on a topic. In particular, they are responding the
the following comment:
  "{post_title}"
  Each participant has expressed an opinion, and your task is to generate a statement such
that as many participants as possible feel represented by that statement. To illustrate
what I mean by "feel represented", I give you a few examples in the following. Each example
consists of a participant's opinion, a statement, and the answer to the question whether
the participant who expressed the opinion would feel represented by the statement:
  {examples}
  These examples hopefully made it clear what it means for participants to "feel repre-
sented" by a statement. Now I give you a list of participants' opinions, afterwards I will
ask you to generate a statement such that as many participants as possible feel represented:
  {agent_opinion_list}
  Given these opinions expressed by the participants, write now a statement such that
as many participants as possible feel represented by that statement. Make the statement
precise and short, no more than three sentences.
```

GENERATE$(\cdot, \emptyset)$: Chain-of-Thought Prompt

	Comment
Member 1	Addressing global warming requires systemic changes in the way we design our society and economics, supported by technological advancements, to mitigate its effects and work towards a more sustainable future.
Member 2	Combating climate change requires a multi-faceted approach that includes embracing renewable technologies, fostering cooperation, and implementing systemic changes in our way of life and economy to reduce greenhouse gas emissions and work towards a more sustainable future.
Member 3	Climate change solutions must prioritize both widespread systemic transformations and technological innovation, to enable a more sustainable and resilient civilization in the face of this global challenge.

Table 2: Comments selected by Process 2, corresponding to Fig. 11

```
Below is a list of people with various opinions on a topic. Follow the following steps:
  1. Determine what topic they are discussing. What are the key axes of disagreement here?
  2. For each person, give them a number and write down what their main beliefs are, and
how they compare to the crowd.
  3. Reflect on which opinions are commonly held and which opinions are less commonly held.
Where might people find common ground?
  4. Write a single specific statement that the most possible people would feel FULLY
REPRESENTED by. You should try to find common ground among the different opinions. But
also, the statement you generate should not be wishy-washy: it should represent a single
specific viewpoint on the topic. It should sound like an opinionated statement one of the
people in the list would make. Of course, not everyone can be fully represented by a single
statement, I just one you to generate one statement which fully represents the most people
possible.(Don't worry that some people will not be represented yet, I will ask you later to
generate statements which represent the remaining people.) Now here is the list of people
and their opinions:
  {agent_opinion_list}
  This concludes the list of people, now start following the steps. Your output should be
a Python dictionary with key-val pairs: "STEP 1": """<writing for step 1>""", "STEP 2":
"""<writing for step 2>""", "STEP 3":, """<writing for step 3>""", "STEP 4": """<writing
for step 4>""". Format your Python dict with curly braces at the beginning and end and
NEVER write newlines. Only write the Python dict and nothing else.
```

GENERATE$(\cdot, \cdot)$: Few-shot Prompt

```
Consider a group of people deliberating on a topic. In particular, they are responding the
the following comment:
  {post_title}
  Each participant has expressed an opinion, and your task is to generate a statement such
that as many participants as possible feel represented by that statement. To illustrate
what I mean by "feel represented", I give you a few examples in the following. Each example
consists of a participant's opinion, a statement, and the answer to the question whether
the participant who expressed the opinion would feel represented by the statement:
```

{examples}
These examples hopefully made it clear what it means for participants to "feel represented" by a statement. Now I give you a list of participants' opinions, afterwards I will ask you to generate a statement such that as many participants as possible feel represented:
{agent_opinion_list}
Remember, your task is to generate a statement such that as many participants as possible in this above list feel represented. However, and this is VERY important, the statement you generate should not be anything close to the statements written below:
{excluded_comments_list}
Given the opinions expressed by the participants, and the list of statements you are NOT allowed to write, write now a statement such that as many participants as possible feel represented by that statement. Make the statement precise and short, no more than three sentences.

GENERATE$(\cdot, \cdot)$ for Chain-of-Thought Prompt

Below is a list of people with various opinions on a topic. Follow the following steps:
1. Determine what topic they are discussing. What are the key axes of disagreement here?
2. For each person, give them a number and write down what their main beliefs are, and how they compare to the crowd.
3. Reflect on which opinions are commonly held and which opinions are less commonly held. Where might people find common ground?
4. Write a single specific statement that the most possible people would feel FULLY REPRESENTED by. You should try to find common ground among the different opinions. But also, the statement you generate should not be wishy-washy: it should represent a single specific viewpoint on the topic. It should sound like an opinionated statement one of the people in the list would make. Of course, not everyone can be fully represented by a single statement, I just one you to generate one statement which fully represents the most people possible. (Don't worry that some people will not be represented yet, I will ask you later to generate statements which represent the remaining people.) Finally there is one more aspect to the challenge: the statement you generate should not be anything close to the statements written below in the forbidden list.
If your statement at all resembles a statement in the forbidden list, that is very bad. So this is the challenge: you must represent the participants in the above list, with a statement which is nothing like anything in the below forbidden list. Now here is the list of people and their opinions:
{agent_opinion_list}
This concludes the list of people. Now here is the list of forbidden statements:
{excluded_comments_list}
Now start following the steps. Your output should be a Python dictionary with key-val pairs: "STEP 1": """<writing for step 1>""", "STEP 2": """<writing for step 2>""", "STEP 3":, """<writing for step 3>""", "STEP 4": """<writing for step 4 (VERY IMPORTANT just put the statement, nothing else, and remember to not write anything similar to any statement in the forbidden list)>""". Format your Python dict with curly braces at the beginning and end and NEVER write newlines. Only write the Python dict and nothing else.

J.3. JR One Shot Prompt

Below is a list of n=n statements people made on a particular topic. Your goal is to write a list of k=k statements which generally represent the group as a whole. Specifically, the list of k=k statements you generate should satisfy a property called JR (justified representation). JR requires that there do not exist n/k people, all of whom do not agree with any of the k=k statements you generate, but they would all agree with some different statement not included in your list. In other words, what is important is that your k=k

statements generally ""cover"" the n=n people in terms of agreement. Now here is the list of n=n people and their opinions:
{agent_opinion_list}
This concludes the list of people, now write down your list of k=k statements which satisfy JR with respect to the above people. Format your response as a Python dict in the following way. ""MY_EXPLANATION"" : """"""<your step by step reasoning for solving the task>""""""", ""MY_JR_STATEMENT_LIST"" : [""""""<first statement>""""""", """""" <second statement >""""""", ..., """""""<kth statement >"""""""]. Only write the Python dict and nothing else.

J.4. Synthetic Data prompts

GENERATE$(\cdot, \cdot)$

A school is deciding which classes to offer for a Summer program, and asked students for their opinions. The available subjects are ['Computer Science', 'Business', 'Mathematics', 'Mechanical Engineering', 'Literature', 'Architecture', 'Global Languages', 'Supplemental Resources', 'Planetary Sciences', 'Environmental Engineering'], and the school can offer exactly 5 of them. The students made the following statements:
{examples}
Please explain your answer step by step and break it down into smaller steps. Explain which students will attend based on your choice, and why it is not possible to make more students attend. Do not write code that implements an algorithm - you're instructed to provide an answer. The last line of your answer must begin with "OFFERED COURSES:" and then contain a Python list with 5 distinct elements of ['Computer Science', 'Business', 'Mathematics', 'Mechanical Engineering', 'Literature', 'Architecture', 'Global Languages', 'Supplemental Resources', 'Planetary Sciences', 'Environmental Engineering']. For example, the last line might look like this: OFFERED COURSES: ['Computer Science', 'Business', 'Mathematics', 'Mechanical Engineering', 'Literature']