Appendix

A Algorithm Details **1**

B Subgroup Estimates **3**

B.1 Issue Importance Study 3

B.2 Misinformation Study 3

C Concerns and Caveats **6**

C.1 Late arrivals 6

C.2 Prompting Participants 6

C.3 Costs of LLM Inference 6

C.4 Toxicity and Quality of Responses 7

C.5 Additional Guidance and Modifications of CSAS 7

C.6 Implementation 8

D Large Language Model Prompts **9**

D.1 Misinformation Filter 9

D.2 Claim Summary 9

D.3 Important Issue Summarization 9

E Open-Source and Proprietary Language Model Comparisons **11**

E.1 Toxicity Detection across Model Types 14

E.2 Testing Proprietary and Open-Source Models in Real-Time 15

F Toxicity Filters **16**

G Implementing the CSAS Method **17**

G.1 "Easy CSAS" Replit Example 17

G.2 More Customizable Replit Example 18

G.3 More Customizable Local Hosting Example 18

H Reducing Redundant Items **20**

I Demographic Bias in CSAS Estimates **22**

J Evaluation of Question Selection Methods and Algorithms: A Toy Example **24**

J.1 Expert Selection 24

J.2 Uniform Random Sampling 25

J.3 Gaussian Thompson Sampling (GTS) 26

K An Application to Local Political Concerns **28**

A Algorithm Details

In Thompson Sampling, a Beta distribution is used as the conjugate prior and a Bernoulli distribution is used as the likelihood. This combination yields a Beta posterior distribution. The Beta distribution is ideal for binary outcomes, and could be used in CSAS settings where binary items are the most appropriate way of measuring the outcome (see (Offer-Westort et al., 2021) for an implementation). However, given the prevalence of ordinal rating scales in political science, I draw on Agrawal and Goyal (2017) to implement Thompson sampling with Gaussian priors (also referred to as Gaussian Thompson sampling). In contrast to traditional Thompson sampling, GTS assumes a normal conjugate prior and allows for the modeling of non-binary outcomes. The implementation of the algorithm is presented below:

Algorithm 1 Gaussian Thompson Sampling Algorithm

Initialization: Initialize "seed questions."
for $i = 1, 2, \ldots, I$ **do**
  if $i = 1$ **then**
    Assign initial seed questions with uniform probability.
  else
    if new questions are added by respondents and pass the filtering step **then**
      Add new questions to the question bank.
    end if
    a) Draw Samples:
    for each question q in the question bank **do**
      Draw $\theta_q \sim N\left(\hat{\mu}_{q,i-1}, \frac{1}{n_{q,i-1}+1}\right)$.
    end for
    b) Calculate Probabilities:
    $q_w = \frac{1}{M}\sum_{m=1}^{M} 1\left\{w = \arg\max_w \{\theta^{(1)}, \ldots, \theta^{(|Q|)}\}\right\}$.
    c) Apply Probability Floor and Rescale:
    $e_q = \max(q_w, .01)$ for each q
    Rescale $e_1, \ldots, e_Q$ to sum to one.
    d) Assign Items:
    $q_i \sim$ Multinomial $(e_1, \ldots, e_Q)$.
    e) Observe Response: Observe response and update parameters.
  end if
end for

In plain language, the first participant is randomly assigned to seed questions with a uniform probability and their ratings are used to update the initial parameters. $\hat{\mu}$ is equal to the mean rating for a given question, whereas $\hat{\sigma}$ is equal to $\frac{1}{n+1}$ where n is equal to the number of ratings for a given question. Since each participant rates their own submitted item and a set of adaptive items, parameters for existing and newly added items, assuming those items have passed the filtering step, are set for the subsequent participant. As participants enter the survey, dynamic items are presented with a probability determined by the ratings of the previous participants. This probability is calculated by simulating draws from the normal distribution for each question and then calculating the proportion of the time with which

each question emerges as the most preferred across simulations. This method of updating probabilities ensures that the questions most likely to elicit higher ratings are more frequently presented to future participants.

B Subgroup Estimates

B.1 Issue Importance Study

Figure B1 presents mean estimates of issue importance among Republicans and Democrats. Estimates are ordered by the absolute size of the partisan difference in estimates. As seen on the bottom of the figure, there are some areas of consensus, namely those related to healthcare costs, political polarization, and economic issues (i.e., cost of living, economic stability). However, by and large, issue importance ratings exhibit considerable gaps. For example, Democrats generally rate inequality, progressive policies (e.g., universal healthcare), gun control, and social issues (e.g., women's rights, race relations) as more important than Republicans, whereas Republicans rate issues involving morals, government spending, and immigration higher than Democrats.

B.2 Misinformation Study

Previous research has discussed WhatsApp as a potential vector for misinformation (Velez et al., 2023). In the survey, participants were asked about the credibility of information on WhatsApp using a five-point ordinal scale ranging from "Trust" to "Distrust." Comparing participants below and above the median of WhatsApp trust, several patterns emerge in Figure B2. For the claims rated toward the bottom of perceived factual accuracy, there is a general consensus. However, for claims involving party stereotypes, newsworthy scandals, and salient allegations, differences between the two groups can be seen. For example, those who trust WhatsApp were less likely to have rated true claims about the Supreme Court blocking student loan relief, Ted Cruz visiting Cancun with his family during widespread blackouts in Texas, and Donald Trump being under investigation as accurate relative to other users. These gaps, of course, could reflect other social or demographic causes, but they point to the possibility that observed differences between WhatsApp users and non-users in factual accuracy can be derived from gaps in knowledge of factual political information, rather than a wholesale adoption of misinformation. The relatively flexible nature of the approach, which allows for the inclusion of blatantly false misinformation as well as misperceptions or overgeneralizations allows us to arrive at a more careful description of information environments than if we solely measured accuracy ratings for verifiably inaccurate content.

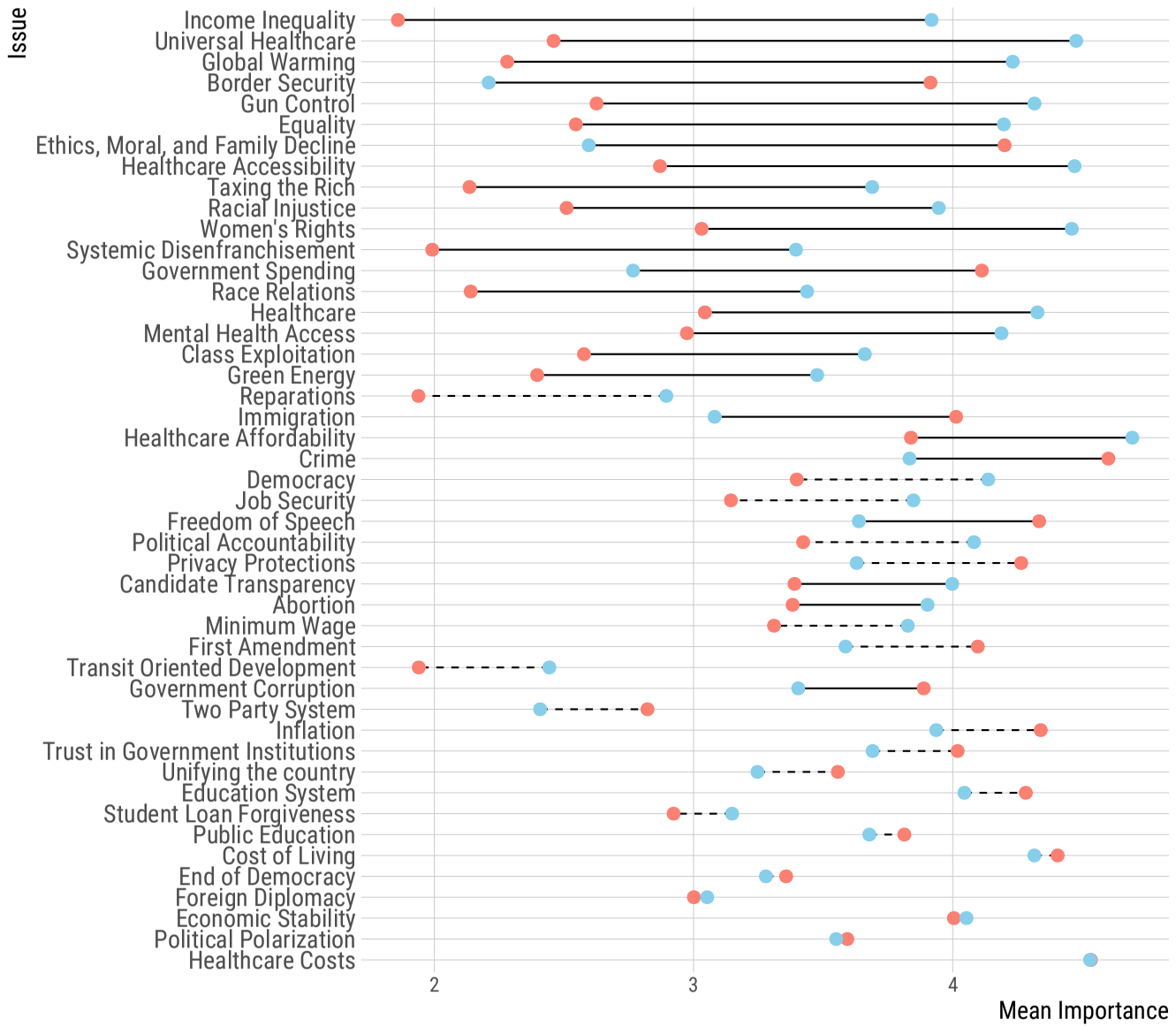

Figure B1: IPW-weighted estimates of survey questions measuring issue importance across Republicans and Democrats with corresponding 95% confidence intervals. Statistically significant (insignificant) differences at conventional levels of significance (α = .05; two-tailed) are depicted using solid (dashed) lines.

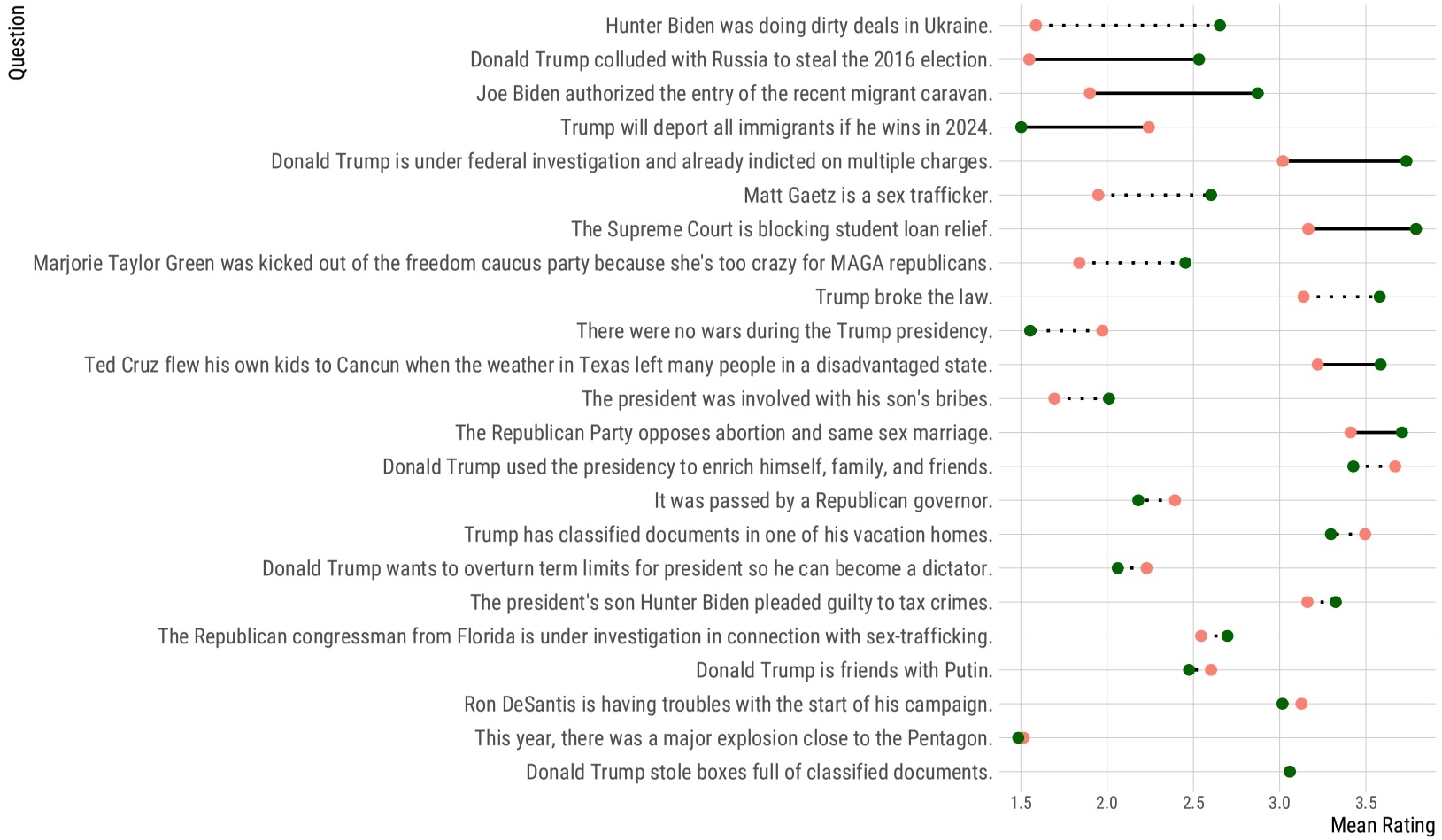

Figure B2: IPW-weighted estimates of survey questions measuring negative political claims across levels of WhatsApp trust with corresponding 95% confidence intervals. Statistically significant (insignificant) differences at conventional levels of significance (α = .05; two-tailed) are depicted using solid (dashed) lines. Green (red) points indicate those scoring above (below) the median on WhatsApp trust.

C Concerns and Caveats

C.1 Late arrivals

Since later submissions to the question bank are reviewed by fewer respondents, there is a risk of not identifying the "best-performing questions." In experimental settings where the determination of optimal treatment allocation may be vital, such as the testing of pharmaceutical interventions, this risk would represent a significant limitation for the approach. However, it is worth highlighting the exploratory nature of the proposed method, and how it performs against the alternative of pre-determined fixed question batteries. In the absence of an adaptive survey design, the selected questions are assumed, by design, to be the best-performing set. If one compares a fixed battery to a dynamic battery comprised of the same items, the dynamic battery, with its ability to adapt and explore new questions, can potentially uncover additional questions that a static design may ignore. Therefore, the risk of failing to identify useful survey items may be even higher with a static design. This being said, the method seems well-suited to cases where items cannot be easily constructed due to a lack of relevant information sources, as in the case of Latino-targeted misinformation, or where opinion is in flux, but may not be as helpful when validated items exist or use of consistent items over time is the objective (e.g., partisan identification).[1]

C.2 Prompting Participants

Leaving aside potential challenges with prompting large language models, prompting participants is also an important step in the process. For example, in the case of misinformation, asking about false claims directly would likely fail to elicit meaningful items, given that participants may not know the veracity of statements or may be inclined to disregard validity when thinking about congenial pieces of information. Instead, asking about claims that possess common features of misinformation, such as negativity, could be more fruitful (Carrasco-Farré, 2022). In extending this approach to construct other kinds of question banks, researchers should carefully consider whether the questions they create may diverge significantly from how respondents conceptualize them. Though best practices involving closed-ended questions are well-developed, the use of LLMs and other text analysis methods to extract meaningful information from text suggests that more thought and care may need to go into identifying optimal open-ended formats. Here, lessons from scholars using semi-structured and unstructured interview modes could prove useful (Leech, 2002).

C.3 Costs of LLM Inference

Though serving large language models to several users at once on personal computers is still largely cost prohibitive, various APIs have emerged that allow scholars to interact with

[1] It is possible that respondent characteristics are associated with time of completion, thus giving a systematic advantage to the priorities of certain groups, as some items gain a "lead" over others by virtue of being submitted earlier. Though this is a valid concern, modifications of the algorithm such as the use of "contextual bandits" that adjust the selection of survey questions in real-time based on user characteristics (e.g., demographics, political affiliation, time) could address this imbalance.

proprietary models such as OpenAI's GPT-3.5 Turbo and GPT-4, as well as open-source alternatives such as Meta's Llama series and MistralAI's Mixtral. At present, these models are cost effective for use in survey settings. Two studies with 100 CloudResearch Connect participants each were carried out to precisely estimate the inference costs of using CSAS with OpenAI's GPT-4 and MistralAI's Mixtral, a high-performing open-source model, at the time of the study. The total cost of inference on AnyScale for processing 100 open-ended responses was 5.4 cents, with a cost per participant of 0.054 cents. In comparison, the total cost of inference on OpenAI amounted to 14 cents, with a cost of 0.14 cents per participant.

C.4 Toxicity and Quality of Responses

The CSAS method depends on user-generated questions, but one might contend that this also poses its own set of problems. For example, toxic user-generated content, such as hate speech, and low-quality content, such as gibberish, could pollute the question bank and compromise the validity of the approach. There are several mitigation strategies that could be implemented.

By clearly defining what counts as an issue or a claim, one can simultaneously filter out expletives, racial slurs, or gibberish. These guidelines can be implemented by fine-tuning models or through the use of in-context learning, where examples are directly used to guide completions by including them in the prompt (see prompt examples in Appendix D).

Relying on moderation endpoints is another approach. OpenAI offers moderation models that are designed to detect and filter out undesirable content when using their API.[2] In many settings, proceeding with automated content filters may be sufficient.

In the three settings explored, zero items including slurs, offensive, or violent content were included in either of the two question banks despite no manual review of items. Moreover, in the misinformation study, some items were not added to the question bank because they mentioned abuse or violence (e.g., "Ted Cruz is the Zodiac killer"). Though these "false negatives" were rare (12% of submissions), they highlight the necessity of a balanced approach in content moderation (see Appendix F for an extended discussion).

C.5 Additional Guidance and Modifications of CSAS

Simpler classification tasks, like those demonstrated in the work on important issue identification (Mellon et al., 2022), may not require the additional investment of fine-tuning pre-trained LLMs, as their existing capabilities may suffice. However, capturing more abstract concepts such as misinformation may require additional steps such as fine-tuning and few-shot learning (i.e., including positive and negative examples in an LLM prompt). Moreover, there may be cases where measuring how members of the public respond to expert-defined constructs may be the primary aim. In these settings, the latent constructs defined by researchers may take precedence due to their correspondence with theoretically important variables. For

[2] Proprietary models such as Google's Bard and Anthropic's Claude, as well as OpenAI's GPT-4, commonly refuse to produce toxic content even without the use of moderation endpoints. There are also fine-tuned chat versions of open-source models such as Llama 2 that are trained to produce less harmful content.

instance, research on the nature of democracy has long debated necessary conditions such as competitiveness, press freedom, and protection of minority rights (Treier and Jackman, 2008). The specific operationalization of these concepts by experts might be crucial, as they pertain to system-level variables that may escape the notice of ordinary citizens. While understanding public perceptions of democracy remains valuable, it may not always align with the theoretical constructs that scholars aim to measure.

The size of the potential question bank also plays a significant role in CSAS performance. Generally, multi-arm bandits perform best when the universe of potential items is small, as it entails an easier process for estimating the reward distributions of each item and requires fewer observations to identify the best-performing items accurately. Consider a scenario where voters take into account at most 30 features of candidates when making voting decisions. In this case, obtaining the best-performing candidate characteristic would be easier than in settings such as campaign events or misinformation, where the universe of items is large and constantly expanding. Even in this setting, algorithms like GTS may be preferable to uniform sampling, as they reduce the likelihood of exposing participants to poorly-performing items.

One may also be concerned with "demographic bias," where over-time shifts in sample composition affect which items are considered as data are being collected. In Appendix I, I examine this concern and fail to find significant differences between "early completers" and "late completers." Recent innovations in multi-arm bandit algoriths such as deconfounded Thompson sampling (Qin and Russo, 2022) could be used to address these concerns in online samples where demographic bias is likely.

Finally, inverse-probability weighting is used to capture uncertainty in estimates, as in other work (Offer-Westort et al., 2021); however, this is an active area of research and future studies could the assess statistical properties of IPW in dynamic adaptive settings, and the requisite sample size for CSAS designs (Hadad et al., 2021).

C.6 Implementation

The method requires interacting with APIs and managing a personal database, which can be daunting for those without a background in programming or database management. Though the implementation used in the first two studies relies on PHP and MySQL, more flexible frameworks exist that can be deployed across a variety of providers. To facilitate adoption of the method, a Django application with a graphical user interface to save user settings and deploy the model on popular survey software has been developed (see Appendix G). Users can use the GUI to submit an initial set of seed items, select an API provider, modify prompts depending on objectives (e.g., summarizing claims versus issues), and various other parameters. The quality of completions can also be assessed before deploying the model in a survey context, such that one can try different user inputs and evaluate the quality of text completions. In Appendix G, instructions for hosting a CSAS survey are described. Given that the application is programmed using Python's Django framework, a wide range of hosting providers are available for deploying the CSAS survey.

D Large Language Model Prompts

D.1 Misinformation Filter

- model: ada
- temperature: 0
- prompt:

> Is the following text a specific, verifiable claim or is it a value judgment/gibberish/evaluative statement? Use 1 to indicate a verifiable claim; 0 to indicate otherwise.
>
> Examples:
> Biden is corrupt.->0
> Bidenś son had profitable business deals in Ukraine.->1
> Trump was a horrible president.->0
> Trump kept classified documents in his home.->1
> George Soros funded the Black Lives Matter protests.->1
> Republicans are hiding a lot of information.->0
> The 2020 election was stolen by the Democrats.->1
> Republicans are racist.->0
> Republicans are in the pocket of big business.->0
> Democrats lie.->0
> Republicans are the best party.->0
> Democrats are the best party.->0
> Democrats hate the country.->0
> Republicans should ban Donald Trump from social media.->0
> Bill Clinton and Hillary Clinton were friends with Jeffrey Epstein.->1
> {text}->{completion}

D.2 Claim Summary

- model: davinci-002
- temperature: 0
- prompt:

> Produce a one-sentence summary of the following point of view. Only summarize the most important claim. This claim is about {party}.
> Claim: {text}
> Summary: {completion}

D.3 Important Issue Summarization

- model: gpt-4
- temperature: 0
- prompt:

> Please extract the political issue or concern mentioned by the respondent using

one to three words. Be descriptive and stay true to what the user has written.
Select only one issue, concern, or topic. Never ask about two issues.
If the issue is not politically relevant, return the phrase Room Temperature Superconductors.
If a related issue or theme has already been mentioned, return the same issue or theme as the output. Do not duplicate broad issue areas or themes.
Examples:
Previously Mentioned Issues () I care about the environment.->Environment
Previously Mentioned Issues (Taxation) My taxes are too high.->Taxation
Previously Mentioned Issues () Abortion should be legal under all circumstances.->Abortion
Previously Mentioned Issues (Immigration) Close the borders.->Immigration
Previously Mentioned Issues (Inflation) I am concerned about rising prices.->Inflation
Previously Mentioned Issue ({matches}) {text}->

Note: A comma-separated list of the five closest matches, labeled as matches, is generated using embeddings. These matches are identified from the question bank based on their cosine similarity.

E Open-Source and Proprietary Language Model Comparisons

OpenAI's large language models are used throughout the manuscript to filter and summarize open-ended responses before submitting them to the question bank. Though models such as OpenAI's GPT-3.5 and GPT-4 score at the upper end of various coding and natural language benchmarks (Achiam et al., 2023), open-source models such as Meta's Llama 2/3 (Touvron et al., 2023) and MistralAI's Mixtral (Jiang et al., 2024) have also demonstrated significant capabilities in these areas.[3] Moreover, unlike proprietary models such as GPT-4 that tend to update their weights with increasing training data or as a result of modifications to inference routines, base model weights are fixed and accessible, facilitating transparency and reproducibility (Palmer et al., 2023).[4] Finally, ethical considerations regarding the use of copyrighted data in the training process for both proprietary and open-source models are also worth highlighting. Recently, OpenAI has faced legal challenges over the use of copyrighted materials in training their models without explicit permission from authors and publishers. However, this issue affects open-source models as well.[5] Despite these issues, it is worth highlighting that the CSAS method does not rely on any specific LLM implementation. As the field evolves and more models become available, researchers can adapt CSAS to use these alternatives without fundamentally altering the method's core principles.

To assess whether open-source models are up to the task of filtering and summarizing open-ended responses, LLM-generated output for two leading open-source models was compared to OpenAI's GPT models: Mistral's *Mixtral*, a 46.7 billion parameter large language model with a unique Mixture of Expert (MOE) architecture, and Meta's Llama 3.1 405B, a 405 billion parameter model representing the latest iteration in the Llama series. Both Mixtral and Llama 3.1 405B have shown strong performance on various benchmarks, with Mixtral surpassing Meta's 70 billion parameter Llama model on Huggingface's Open LLM Leaderboard (Beeching et al., 2023) and competing favorably against GPT-3.5 on thousands of blind assessments of chat output (Zheng et al., 2023). Recent estimates place Llama 3.1 405B at the upper end of open-source models, with performance mirroring the latest class of proprietary models such as GPT-4o and Anthropic's Claude Sonnet 3.5.

The performance of OpenAI's GPT models, Mistral's Mixtral, and Meta's Llama 3.1 405B were directly compared using the same open-ended responses and prompts from both stud-

[3] In this paper, I use the term "open-source" to refer to models that share their weights publicly. Mixtral is licensed under Apache 2.0, which is a common open-source license. However, Meta's Llama 2 has some limitations on commercial use, which contradicts the "Open Source Definition" that states that open-source software should not impose any restrictions on its users (Perens et al., 1999).

[4] Of course, stochastic outputs from large language models are largely unavoidable due to their probabilistic sampling of tokens. Even with zero temperature, a parameter that minimizes the randomness of outputs, stochastic outputs in large language models can also emerge due to technical factors such as floating-point errors and parallel core sequencing.

[5] Both proprietary and "open-source" large language models rely on Common Crawl, a publicly available dataset of web crawl data that includes copyrighted material alongside openly licensed content. Initiatives like OLMo (Open Language Model) that aim to replicate the full training process may enable future LLMs to fully rely on permissively licensed content, potentially addressing some of these ethical concerns.

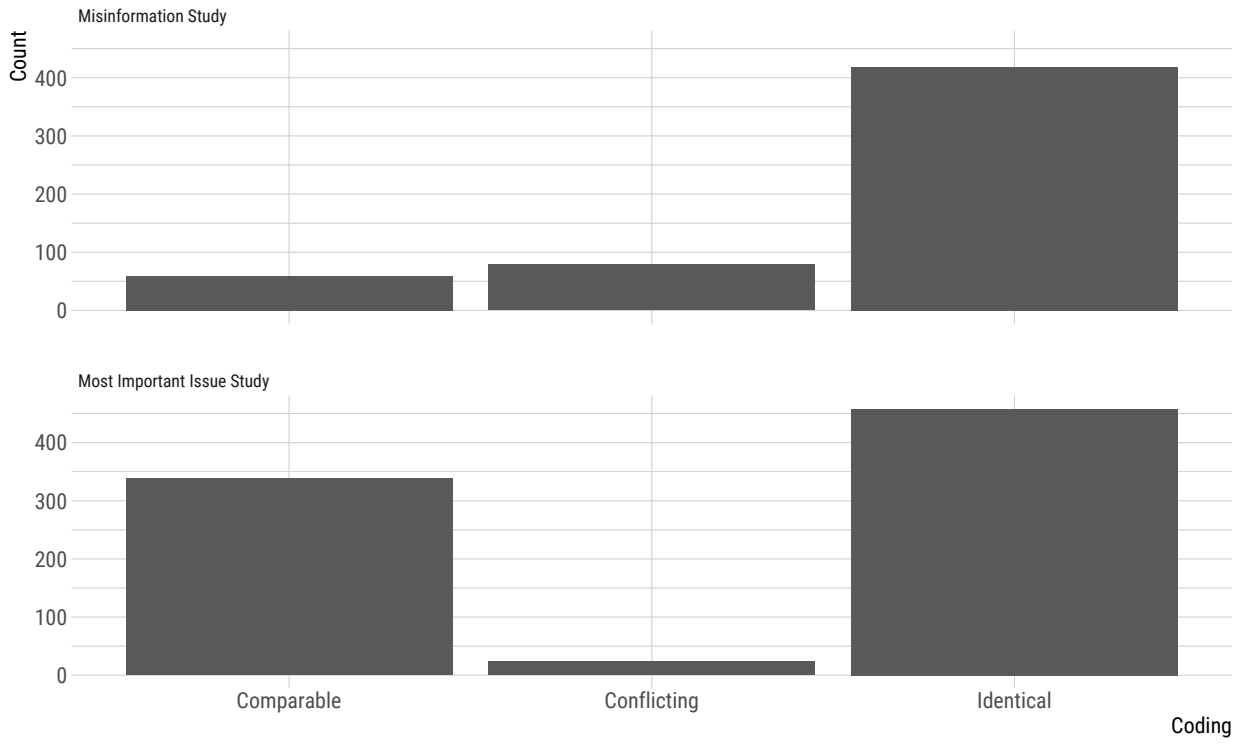

Figure E1: Comparison between Mixtral (Mistral 8x7B Instruct 0.1) and GPT-3.5/4

ies.[6] Completions were classified as identical, comparable/similar, or conflicting/different. Completions were marked as comparable when they returned outputs that were similar in meaning (e.g., categorizing an open-ended response about universal healthcare as healthcare versus universal healthcare), whereas conflicting outputs consisted of entirely different concepts (e.g., classifying a response about a cryptocurrency investor who was arrested as "corruption" versus cryptocurrency).

The models agreed the vast majority of the time. For Mixtral, in the misinformation study, models produced identical or comparable output 86% of the time; whereas this number was 97% in the most important issue study. For Llama 3.1 405B, in the misinformation study, models produced identical or similar output 84.4% of the time (73.5% identical, 10.9% similar); whereas this number was 99% in the most important issue study (38% identical, 61% similar). The higher share of "comparable" vs. identical outputs in the most important issue study for both open-source models (41% for Mixtral and 61% for Llama 3.1 405B, versus 11% and 10.9% respectively in the misinformation study) is mostly a function of how these models handled specificity versus generality when summarizing issue positions. For example, the Mixtral had a tendency to classify any immigration-related response as "Immigration," whereas GPT-4 was more likely to distinguish between different kinds of immigration responses (i.e.,

[6] As mentioned in the manuscript, the more capable model, GPT-4, was used to summarize issue positions, whereas GPT-3 was used to summarize negative political claims.

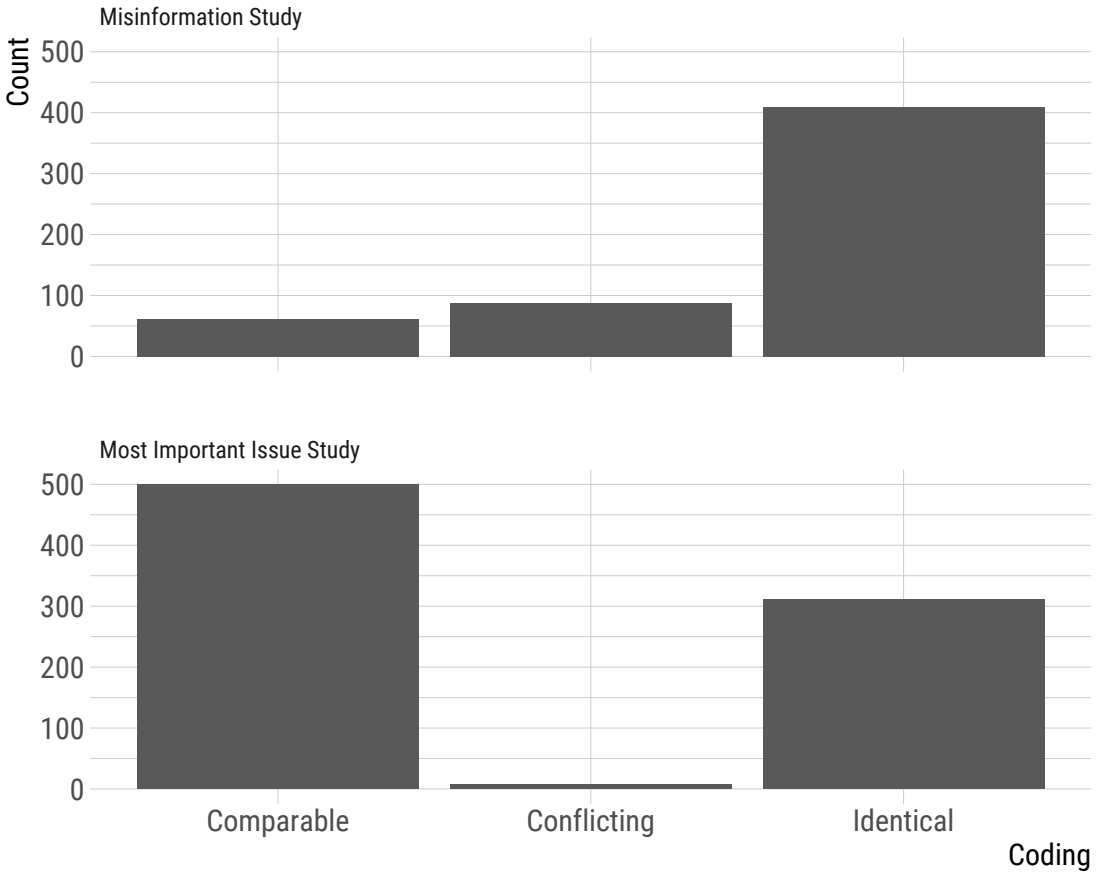

Figure E2: Comparison between Llama 3.1 (Llama 3.1 405B Instruct) and GPT-3.5/4

"Legal Immigration," "Border Security") and Llama 3.1 was more likely to use "Border Security" as a catch-all for immigration-related concerns.[7]

When classifying the output into better or worse categories, I coded cases where the summary accurately reflected the user response (e.g., "Class Exploitation" when a user specifically describes the exploitation of lower classes, versus "Inequality") as the "better" completion. For Mixtral, in the misinformation study, the OpenAI model produced "better" output 57% of the time, whereas this number was 65% in the issue importance study. For Llama 3.1 405B, in the misinformation study, the OpenAI model produced "better" output 57.8% of the time. In the issue importance study, Llama 3.1 405B performed produced more satisfactory summaries 43.7% of the time compared to OpenAI's models 56.3%.

These results suggest that while there are still some differences between proprietary and open-source models, the gap is narrowing. Open-source models like Mixtral and Llama 3.1 405B are increasingly capable of performing complex tasks such as filtering and summarizing open-ended responses with high accuracy. This progress in open-source models offers researchers more options for implementing methods like CSAS while potentially addressing some of the ethical and reproducibility concerns associated with proprietary models.

E.1 Toxicity Detection across Model Types

Overall, the performance of Mistral AI's Mixtral and Meta's Llama was comparable to OpenAI's models, but OpenAI generally had an edge in terms of producing higher quality completions. It is possible that prompt modifications could yield even higher quality completions, given that Mixtral and Llama produced output that matched or resembled the output of the proprietary model. This suggests Mixtral and Llama 3.1 could potentially be used as "drop-in replacements" for GPT-3.5 or GPT-4 when implementing the CSAS method.

Still, toxicity filtering remains an area where open-source models may lag behind proprietary models like OpenAI's GPT-3 and GPT-4. Proprietary models often benefit from extensive tuning and "red-teaming" efforts where toxic content is deliberately produced to improve detection of harmful outputs, leading to more sophisticated toxicity filters. This is crucial, especially when dealing with sensitive topics such as misinformation where exposure to harmful content may be a risk. However, this is a quickly evolving area. Toxicity filters using the Llama model have been recently developed, and could serve as a potential open-source replacement. For example, Inan et al. (2023) use the 7-billion parameter version of Llama 2 to train a moderation model, and find that its classification accuracy rivals OpenAI's moderation endpoint. These moderation models are publicly accessible using various inference APIs and allow scholars to build fully open-source routines that manage completions and toxicity filters.

Despite the potential risks of harmful content, it is important to acknowledge that instances

[7] It is important to note that classifications need not be viewed as 'right' or 'wrong,' as this could depend on research objectives. If the goal is to extract general issue topics, open-source models may be preferred due to their tendency to err on the side of broader themes, whereas if more niche issue positions are desired, GPT-3.5 or GPT-4 may be ideal.

of toxic *output* are relatively rare in most applications. For instance, across the studies, there were zero instances of LLM completions involving derogatory terms or slurs, and risk of harmful completions may be relatively low in settings such as gauging issue importance. In such contexts, open-source models may still be suitable even without sophisticated toxicity filters, given their strong performance in classifying and summarizing text. Nevertheless, when it comes to areas such as misinformation, the use of open-source models might carry higher risks. The lack of advanced toxicity filters could potentially allow harmful content to slip through, making these models less ideal. Testing and refining open-source moderation tools may be a necessary step before these alternatives are implemented. Appendix G describes how to deploy the CSAS method, and a GUI has been developed that allows scholars to carry out their own "red teaming" efforts by assessing how models handle different types of adversarial inputs.

E.2 Testing Proprietary and Open-Source Models in Real-Time

A study with 200 participants was conducted on CloudResearch Connect to estimate costs across different API providers and assess the real-time performance of open-source models on January 6, 2024. 100 participants completed a CSAS survey that summarized their open-ended responses using GPT-4, wheres another 100 participants completed a CSAS survey using Mixtral. As mentioned in the manuscript, the inference cost per participant was approximately .005 cents for Mixtral using AnyScale's ChatCompletion endpoint and .01 cents for GPT-4. Overall, GPT-4 and Mixtral produced similar completions. Though, Mixtral was slightly more prone to completion errors. Whereas all of GPT-4 summaries involved a single issue, there were four instances of completions using Mixtral that included multiple issues (e.g., "Abortion and vaccine choice") and three instances of completions that did not return an issue, but instead continued a conversation (e.g., "In your examples, you have provided specific political issues...").

In sum, while open-source models like Mixtral exhibit commendable performance in many areas, their application in contexts requiring stringent content moderation should be approached with caution. Until these models achieve parity with proprietary models in toxicity filtering, their use should be carefully considered, especially in settings where misinformation or sensitive content is a concern.

F Toxicity Filters

Toxicity filters prevent harmful content from reaching online participants, but they may also filter out claims that could reveal important aspects of public opinion or misinformation. Zero instances of toxic content involving slurs or hateful content in the open-ended responses or the completions were observed. However, some items that were flagged as toxic by OpenAI's moderation endpoint were not added to the question bank, even though they could have been relevant for studying misinformation. These items included topics such as child abuse, drug use, and violence (e.g., "Ted Cruz is the Zodiac killer").

Given that viral misinformation often dwells on these themes, and tracking the nature of false claims across groups and time is an important objective of much of this research, toxicity filters that are overly sensitive to certain concepts could limit what we learn from the CSAS method. That being said, if there are promising items that are rejected due to toxicity filters, one could incorporate these items into future surveys, much like one would learn from open-ended responses in traditional surveys.

Though the possibility that toxicity filters may produce "false negatives" is concerning when studying topics such as misinformation, it may not be as significant a problem for other applications such as issue importance. In these applications, the primary focus is not to delve into the specifics of controversial or potentially harmful claims about politics, but to reveal policies or issue areas that are important to participants. As such, the likelihood of encountering content that would be flagged by toxicity filters is inherently lower in these contexts.

G Implementing the CSAS Method

Application of the CSAS method requires a database. The implementation in the manuscript uses an external SQL database hosted by a web provider, Web Hosting Hub, that stores the items, along with the number of ratings and average rating for each question. The latter two statistics are sufficient for implementing Thompson sampling, since the variance is calculated as the inverse of the total number of ratings. PHP is used to send information between the SQL database and Qualtrics. Given that one may want to implement CSAS without using a long-term hosting provider, it is worth exploring alternatives that can provide flexible and affordable services on a project-by-project basis. To that end, a Django implementation of the input, update, and sampling routines has been developed that can be self-hosted or hosted on popular cloud providers such as Replit.

Below, I walk through the steps of setting up a CSAS survey using the cloud hosting provider, Replit, as well as a self-hosted version. Though the latter is cost effective and does not require an external database, it may not be feasible to host locally due to computational resources. Before launching the server, pilot tests are recommended to simulate the expected traffic. In Qualtrics, the "Generate Test Responses" function can be used to run these simulations. Running CSAS locally also carries risks, since it makes local servers publicly accessible. Thus, external hosting may be preferable from a security vantage point.

The CSAS application currently includes the option to use OpenAI's GPT-3.5 turbo and Mistral AI's Mixtral models. The former uses OpenAI's API to generate embeddings and flag content as toxic as part of the "redundancy" and "toxicity" filters described in the paper, whereas the latter uses Anyscale, an LLM hosting provider, to carry out the filtering process using open-source alternatives. These open-source models are Llama Guard for moderation, and General Text Embeddings (gte-large) for redundancy. Therefore, one can choose to either implement a fully proprietary or open-source model "stack."

G.1 "Easy CSAS" Replit Example

- Sign up for a free Replit account.
- Visit https://replit.com/@democsas/CSAS-Helper
- Fork the repository by clicking "Fork" on the top right.
- Update secrets

  SECRET_KEY: This can be anything, but preferably a strong and random string that is hard to guess.

  OPENAI_API_KEY: To use OpenAI's models, an API key is needed. This can be obtained by following this guide.

  ANYSCALE_API_KEY: AnyScale is a provider that hosts open-source models such as Llama 2 and Mixtral. One can sign up for an account at this page.
- After updating secrets, clicking Run will enable the Django web server and open the "CSAS Helper."
- Save the relevant parameters (e.g., minimum scale value, maximum scale value, API) and

add seed items. **Note: The number of seed items must be equal to or greater than the number of dynamic items. Otherwise, there will not be enough items to sample from in the database, and an error will be produced.**

- Copy and paste the HTML from the "Easy CSAS" section into the HTML view of a Text/Graphic Question Type in Qualtrics.
- Launch the survey.

G.2 More Customizable Replit Example

- Sign up for a free Replit account.
- Visit https://replit.com/@democsas/CSAS-Helper
- Fork the repository by clicking "Fork" on the top right.
- Update secrets

  SECRET_KEY: This can be anything, but preferably a strong and random string that is hard to guess.

  OPENAI_API_KEY: To use OpenAI's models, an API key is needed. This can be obtained by following this guide.

  ANYSCALE_API_KEY: AnyScale is a provider that hosts open-source models such as Llama 2 and Mixtral. One can sign up for an account at this page.
- After updating secrets, clicking Run will enable the Django web server and open the "CSAS Helper."
- The "CSAS Helper" allows users to test how input is processed by different LLMs, revise prompts, set default APIs, and the number of dynamic items. It also provides a set of Qualtrics links that can be used to implement the input, update, and sampling routines of the CSAS method.
- In Qualtrics, the sampling step can be implemented by creating a Web Service that points to the sampling URL. Clicking "Test" will produce the relevant fields. To quickly save these fields, one should click Select All and Add Embedded Data.
- Before the input block, an open-ended question should query the user about a topic. Another Web Service should be created after this question that points to the input URL. The query parameter should be input, and it should be set equal to the internal question ID.
- Ratings should be provided to participants by piping in embedded data fields (e.g., ${e://Field/q_1}). The user-submitted question is saved in Qualtrics as ${e://Field/completion}.
- To implement the update routine, the update URL should be used with a list of query parameters (${e://Field/q_1}) and ratings using Qualtrics' Recode format (e.g., {q://QID3/SelectedAnswerRecode/2}).

  A Qualtrics template for a "three dynamic item" survey is provided here to facilitate setup.

G.3 More Customizable Local Hosting Example

- Install GitHub (https://github.com/git-guides/install-git), Python 3.0 (https://www.python.org/), and Ngrok (https://ngrok.com/download)

- Open Terminal/Shell
- Clone the GitHub repo (https://github.com/scholar-anonymous/csas.git)
- Create a virtual environment: virtualenv env
- Active the virtual environment: venv source/bin/activate
- Install requirements.txt (pip install -r requirements.txt)
- Change directory into csas_files
- Update API keys by using the export command in Terminal (e.g., export SECRET_KEY=foobar).

  SECRET_KEY: This can be anything, but preferably a strong and random string that is hard to guess.

  OPENAI_API_KEY: To use OpenAI's models, an API key is needed. This can be obtained by following this guide.

  ANYSCALE_API_KEY: AnyScale is a provider that hosts open-source models such as Llama 2 and Mixtral. One can sign up for an account at this page.
- Check for changes to models (databases) and create migration files: python manage.py makemigrations
- Apply changes to database: python manage.py migrate
- Launch Django app: python manage.py runserver localhost:8080
- Launch ngrok: ngrok http 8080
- Click "Visit Site" after ngrok disclaimer
- The "CSAS Helper" allows users to test how input is processed by different LLMs, revise prompts, set default APIs, and the number of dynamic items. It also provides a set of Qualtrics links that can be used to implement the input, update, and sampling routines of the CSAS method.
- In Qualtrics, the sampling step can be implemented by creating a Web Service that points to the sampling URL. Clicking "Test" will produce the relevant fields. To quickly save these fields, one should click Select All and Add Embedded Data.
- Before the input block, an open-ended question should query the user about a topic. Another Web Service should be created after this question that points to the input URL. The query parametxer should be input, and it should be set equal to the internal question ID.
- Ratings should be provided to participants by piping in embedded data fields (e.g., ${e://Field/q_1}). The user-submitted question is saved in Qualtrics as ${e://Field/completion}.
- To implement the update routine, the update URL should be used with a list of query parameters (${e://Field/q_1}) and ratings using Qualtrics' Recode format (e.g., {q://QID3/SelectedAnswerRecode/2}).

  A Qualtrics template for a "three dynamic item" survey is provided here to facilitate setup.

H Reducing Redundant Items

Document embeddings are used in the CSAS method to minimize redundant items, given that survey questions submitted to question banks may still possess similar meanings even after exact duplicates are removed. In the issue importance study, the five most similar questions from the question bank were retrieved using cosine similarity and GPT-4 was prompted to return "the same issue" if a related issue of theme had already been mentioned. In the misinformation study, a precise threshold of .90 was used to determine which items were included in the question bank. Only items with cosine similarity scores below .90 were included, with the assumption that questions scoring above these thresholds possessed similar meanings.

An advantage of precise thresholds is greater transparency over how items are included, rather than relying on a "black box" decision made by a large langage model. However, they introduce an element of arbitrariness. To assess how the inclusion of items varies as a function of similarity thresholds, I take the open-ended responses from both studies and assess the number of topics that are produced when different "uniqueness" thresholds. I process each submitted item by comparing it to all of the items that were submitted before using cosine similarity. The item is added to the bank if and only if there is no match for which the cosine similarity score is k or lower, where k varies from .65 to .99 by increments of .5.

In the misinformation study, the number of "unique" items is minuscule when smaller thresholds are used, but this quickly picks up at thresholds around .8. The same goes for issues. Thresholds just above .9 and below .99 appear to be a "sweet spot" for diverse, but non-redundant issues. For example, in the misinformation study, four different variations of Joe Biden having "dementia" are present in the question bank if a threshold of .99 is used, whereas two are produced with a threshold of .9. As a general recommendation, maintaining thresholds above .9 and below .99 strikes a good balance between item diversity and minimizing redundancy.

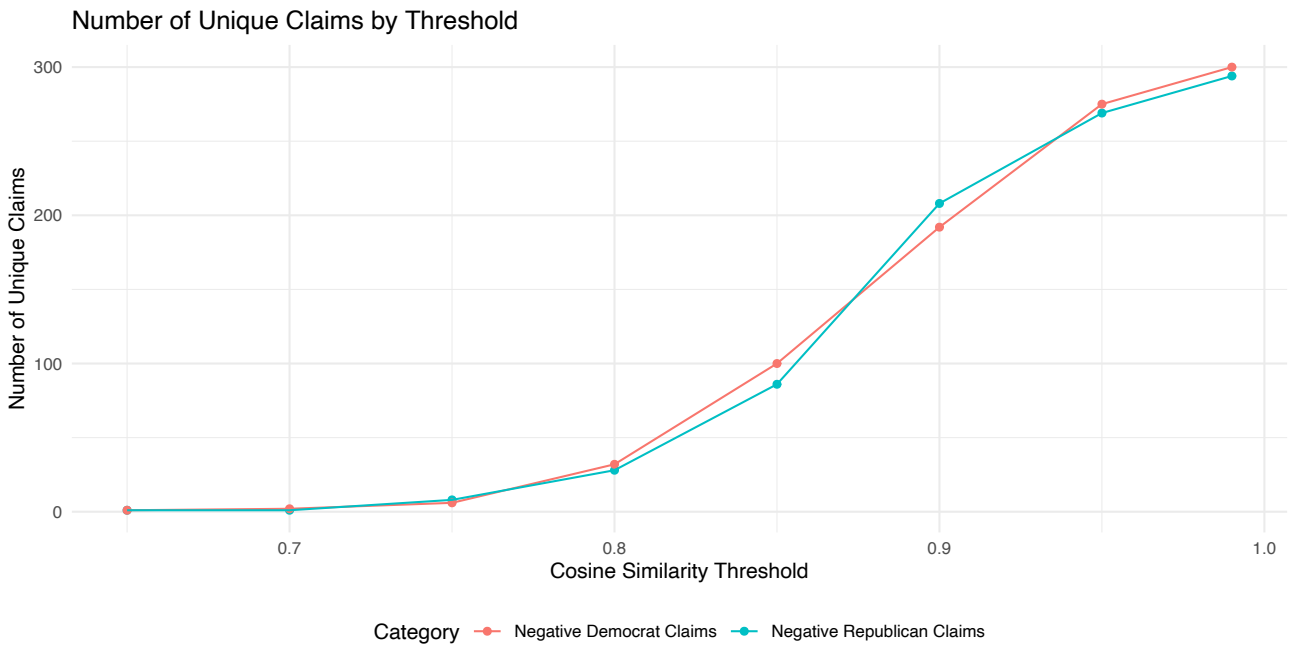

Figure H1: Sensitivity to Different Cosine Similarity Thresholds in the Misinformation Study

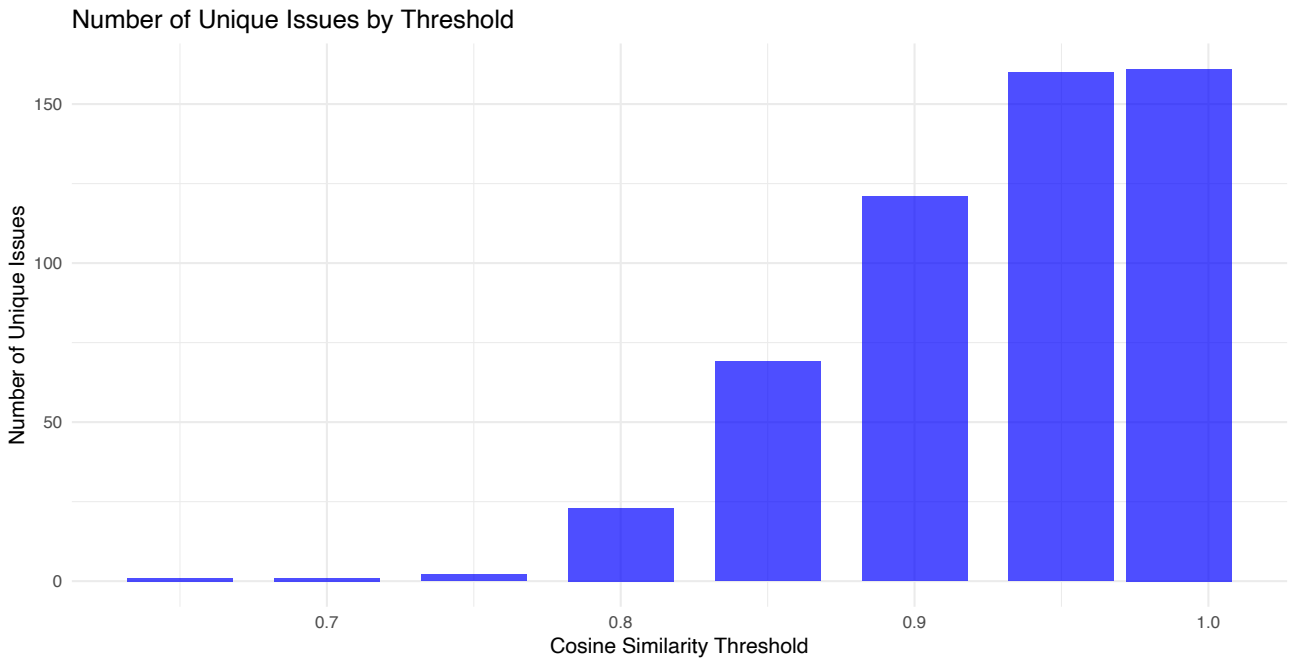

Figure H2: Sensitivity to Different Cosine Similarity Thresholds in the Issue Importance Study

I Demographic Bias in CSAS Estimates

Given the iterative updating of probabilities over time when using multi-arm bandits, one concern is that certain types of participants may respond to surveys toward the beginning of data collection, potentially altering the items that are presented to future participants. Though the presentation of items is random and this should mitigate biases in estimates, there is still the possibility that different kinds of items are rated due to sample characteristics. For example, if background characteristics like partisanship influence how quickly participants complete surveys, items could reflect this, with more liberal-leaning items appearing in the question bank as a function of more Democrats participating early.

To investigate this issue, I conducted an analysis in Study 2, which has a larger sample for detecting potential differences in demographics. Specifically, I examined whether late survey completers differed significantly from early completers. A regression analysis of completion time (which is rescaled to range from a minimum of 0 to a maximum of 1) on age, education, income, partisanship, ideology, and gender found no evidence that standard demographics explain the time of completion. Specifically, coefficients are small and none reach statistical significance at conventional levels (all $p > 0.1$). This suggests that "earlier" respondents tend to resemble "late" respondents, at least in this study.

That being said, if demographic bias is expected to be a significant concern, use of a more specialized algorithm may help address this issue. For example, Qin and Russo (2022) propose a method called deconfounded Thompson Sampling (DTS), which adjusts for covariates when estimating the reward distributions for different arms. In contrast to contextual bandits, which specialize in retrieving optimal arms for specific contexts or subgroups, DTS aims to estimate the effect of each arm while adjusting for confounding variables. Future work could implement this approach in settings where sample imbalances that unfold over time are suspected.

	Model 1
(Intercept)	0.478***
	(0.070)
Gender	0.027
	(0.017)
Partisanship (7-point)	0.012
	(0.007)
Education	0.005
	(0.010)
Income	-0.005
	(0.003)
Ideology	-0.012
	(0.009)
Race (Non-White)	0.021
	(0.022)
Age (Years)	0.00007
	(0.0007)
N	789
R^2	0.011

+ p $<$ 0.1, * p $<$ 0.05, ** p $<$ 0.01, *** p $<$ 0.001

J Evaluation of Question Selection Methods and Algorithms: A Toy Example

Here, I consider three different methods discussed in the paper: Expert Selection, Uniform Random Sampling, and Gaussian Thompson Sampling (GTS). Each method presents distinct advantages and limitations that are important to consider when designing surveys and interpreting their results. To fix ideas, I present an illustrative example involving ten participants and ten items. Table 1 presents the full universe of items and participants, representing the ideal scenario where all possible data could be collected. The uniform and GTS examples mimic how items are revealed over the course of the study, where participants submit new items for rating and item selection probabilities depend on the set of submitted items. This serves as a benchmark against which we can use to examine how the different methods recover quantities of interest. In practice, collecting such data is impractical or impossible due to resource constraints or the size of the population of items.

Table J1: Full Universe of Items and Participants

Participant	1	2	3	4	5	6	7	8	9	10
1	3.2	4.1	2.8	3.5	2.9	3.7	4.0	3.3	3.8	3.6
2	3.0	4.3	3.1	3.7	3.2	3.5	3.9	3.4	3.6	3.8
3	3.3	4.0	2.7	3.6	3.1	3.8	4.1	3.5	3.9	3.7
4	3.1	4.2	3.0	3.4	3.0	3.6	3.8	3.2	3.7	3.5
5	3.4	4.0	2.9	3.8	3.3	3.9	4.2	3.6	4.0	3.9
6	3.2	4.1	2.8	3.5	2.8	3.7	3.9	3.3	3.8	3.6
7	3.3	4.2	3.0	3.7	3.1	3.8	4.0	3.4	3.9	3.7
8	3.1	4.0	2.9	3.6	3.0	3.6	3.9	3.3	3.7	3.5
9	3.2	4.3	2.8	3.5	3.2	3.7	4.1	3.5	3.8	3.8
10	3.3	4.1	3.0	3.8	3.1	3.9	4.0	3.4	3.9	3.7
Mean	3.21	4.13	2.90	3.61	3.07	3.72	3.99	3.39	3.81	3.68
SE	0.04	0.04	0.04	0.05	0.05	0.04	0.04	0.04	0.04	0.04

J.1 Expert Selection

Expert selection involves specialists choosing a fixed set of items that all participants rate. In our illustrative example, experts choose Items 1, 2, and 3 for evaluation. This approach offers several advantages. By ensuring all participants provide feedback on the same set of items, it facilitates direct comparison of responses across the sample. Furthermore, this method leverages expert knowledge to focus on the most relevant or important items, which can be more efficient than the exploratory approaches described below.

However, the expert selection method may overlook valuable information from items not selected by experts, potentially missing items that could be critical to a more comprehensive understanding. This limitation is particularly salient in rapidly evolving fields or when studying phenomena that are not yet well understood. Additionally, this method may inadvertently reflect the biases and assumptions of the experts, which may not align with the broader population's perspectives. The fixed nature of the item set also precludes adaptive learning

during the survey process, potentially missing opportunities to explore unexpected items that emerge during data collection.

Table J2: Expert Selection

Participant	1	2	3	4	5	6	7	8	9	10
1	3.2	4.1	2.8	-	-	-	-	-	-	-
2	3.0	4.3	3.1	-	-	-	-	-	-	-
3	3.3	4.0	2.7	-	-	-	-	-	-	-
4	3.1	4.2	3.0	-	-	-	-	-	-	-
5	3.4	4.0	2.9	-	-	-	-	-	-	-
6	3.2	4.1	2.8	-	-	-	-	-	-	-
7	3.3	4.2	3.0	-	-	-	-	-	-	-
8	3.1	4.0	2.9	-	-	-	-	-	-	-
9	3.2	4.3	2.8	-	-	-	-	-	-	-
10	3.3	4.1	3.0	-	-	-	-	-	-	-
Mean	3.21	4.13	2.90	-	-	-	-	-	-	-
SE	0.04	0.04	0.04	-	-	-	-	-	-	-

J.2 Uniform Random Sampling

Uniform random sampling offers an alternative approach by selecting items for each participant with equal probability. This method ensures comprehensive coverage of the item space, as all items in the question bank at a given point have an equal chance of being evaluated. Formally, for the uniform probability method, the probability of observing item i is $\frac{1}{u}$, where u represents the total number of revealed items for participant p. For example, in our toy dataset, u is 4 for participant 2, whereas it is 10 for participant 10. This approach increases the "spread" of ratings across items, which can be particularly valuable in exploratory research where the goal is to canvas the full range of potential items, including those that might be less immediately salient.

However, one of the primary challenges with uniform sampling is the potential for sparse data on individual items, as the random selection process may result in some items being sampled infrequently or not at all, especially in scenarios with a large question bank and relatively few participants. This sparsity can make it difficult to draw robust conclusions about any specific item via lower precision. Furthermore, while the equal probability of selection for all items is simple and transparent, it may lead to inefficiencies in practice if resources are devoted to "low-performing" items in domains where it is important to recover salient issues or claims. In such cases, the uniform approach may allocate scarce survey resources to items of limited value, potentially at the expense of gathering more data on better-performing items.

Table J3: Uniform Random Sampling

Participant	1	2	3	4	5	6	7	8	9	10
1	3.2	4.1	2.8	-	-	-	-	-	-	-
2	3.0	4.3	-	3.7	-	-	-	-	-	-
3	3.3	-	2.7	-	3.1	-	-	-	-	-
4	-	4.2	3.0	-	-	3.6	-	-	-	-
5	3.4	-	-	3.8	-	-	4.2	-	-	-
6	-	4.1	-	-	2.8	-	-	3.3	-	-
7	-	-	3.0	-	-	3.8	-	-	3.9	-
8	-	4.0	-	3.6	-	-	-	-	-	3.5
9	3.2	-	-	-	-	-	4.1	-	3.8	-
10	-	-	3.0	3.8	-	-	-	3.4	-	-
Mean	3.22	4.14	2.90	3.73	2.95	3.70	4.15	3.35	3.85	3.50
SE	0.07	0.06	0.07	0.05	0.15	0.10	0.05	0.05	0.05	-

J.3 Gaussian Thompson Sampling (GTS)

Gaussian Thompson Sampling (GTS) represents a more sophisticated approach to item selection, adaptively choosing items based on their performance in previous ratings. This method is designed to balance the competing demands of exploration (gathering information about lesser-known items) and exploitation (focusing on items that have shown to be informative or important). The adaptive nature of GTS offers several advantages. By continuously updating item selection probabilities based on previous ratings, GTS can efficiently identify and focus on high-performing items, providing more precision for this subset of items relative to what would be obtained with uniform sampling.

However, GTS also has its own share of limitations. One primary concern is the potential for premature convergence on items that perform well initially, which may lead to overlooking items that could be valuable but require more ratings. This bias towards early performers could be particularly problematic in studies where item relevance may evolve over time or across different subgroups of the population.

The GTS method's focus on high-performing items is evident in the concentration of responses for Items 2, 4, 7, and 9, which consistently received high ratings. This approach yields more precise estimates for these items compared to uniform sampling. For instance, Item 2 has a standard error of 0.04 under GTS versus 0.06 under uniform sampling. This reduction in standard errors for frequently sampled items illustrates a key advantage of GTS: it can provide more precise estimates for items deemed important by the algorithm, effectively allocating more resources to these high-performing items.

It is important to note that the toy example presents a somewhat extreme case of GTS's item selection behavior. In practice, as implemented in the paper, probability floors are applied to ensure that the probability for any item in the question bank remains above .01. This modification mitigates the complete avoidance of low-rated items, maintaining a balance

Table J4: Gaussian Thompson Sampling (k=3)

Participant	1	2	3	4	5	6	7	8	9	10
1	3.2	4.1	2.8	-	-	-	-	-	-	-
2	3.0	4.3	-	3.7	-	-	-	-	-	-
3	-	4.0	-	3.6	-	-	4.1	-	-	-
4	-	4.2	-	3.4	-	-	3.8	-	3.7	-
5	-	4.0	-	3.8	-	-	4.2	-	4.0	-
6	-	4.1	-	3.5	-	-	3.9	-	3.8	-
7	-	4.2	-	3.7	-	-	4.0	-	3.9	-
8	-	4.0	-	3.6	-	-	3.9	-	3.7	-
9	-	4.3	-	3.5	-	-	4.1	-	3.8	-
10	-	4.1	-	3.8	-	-	4.0	-	3.9	-
Mean	3.10	4.13	2.80	3.62	-	-	4.00	-	3.83	-
SE	0.10	0.04	-	0.05	-	-	0.05	-	0.04	-

between focusing on high-performers and exploring the full range of options. While this practical implementation is more complex, the toy example effectively highlights one of the key principles of GTS: the up-weighting of better-performing items to improve estimation precision for the most salient or important options in the question bank.

K An Application to Local Political Concerns

100 participants were recruited on CloudResearch Connect to participate in a study of local politics. The "Easy CSAS" method with GPT-3.5 (Appendix G) was used to convert concerns that should be addressed by "city or town officials" into policy items, and participants were asked to rate the importance of five dynamic items, along with their own, on a 1-5 importance scale. The five seed items were the following: "Housing Affordability," "Traffic and Commutes," "Homelessness," "Neighborhood Development," and "Urban Infrastructure." As shown in Figure 1, housing affordability was rated as the most important issue by the sample. However, user-submitted items such as "Economic Stability," "Palestinian Conflict," "Immigration Reform," "Property Taxes," "Animal Control," and "Education Funding" ranked among the highest in importance. Though the sample was national in scope, CSAS can just as easily be used to develop items in local or state-level surveys. For example, when subsetting on New York residents, issues such as "Housing Standards" and "Property Taxes" rise to the top of issue importance, whereas these items are rated as less important in the full sample. This illustrates the potential for the CSAS method to be used as a tool in domains such as local and state politics where polling is irregular and priors about quality survey items are uncertain.

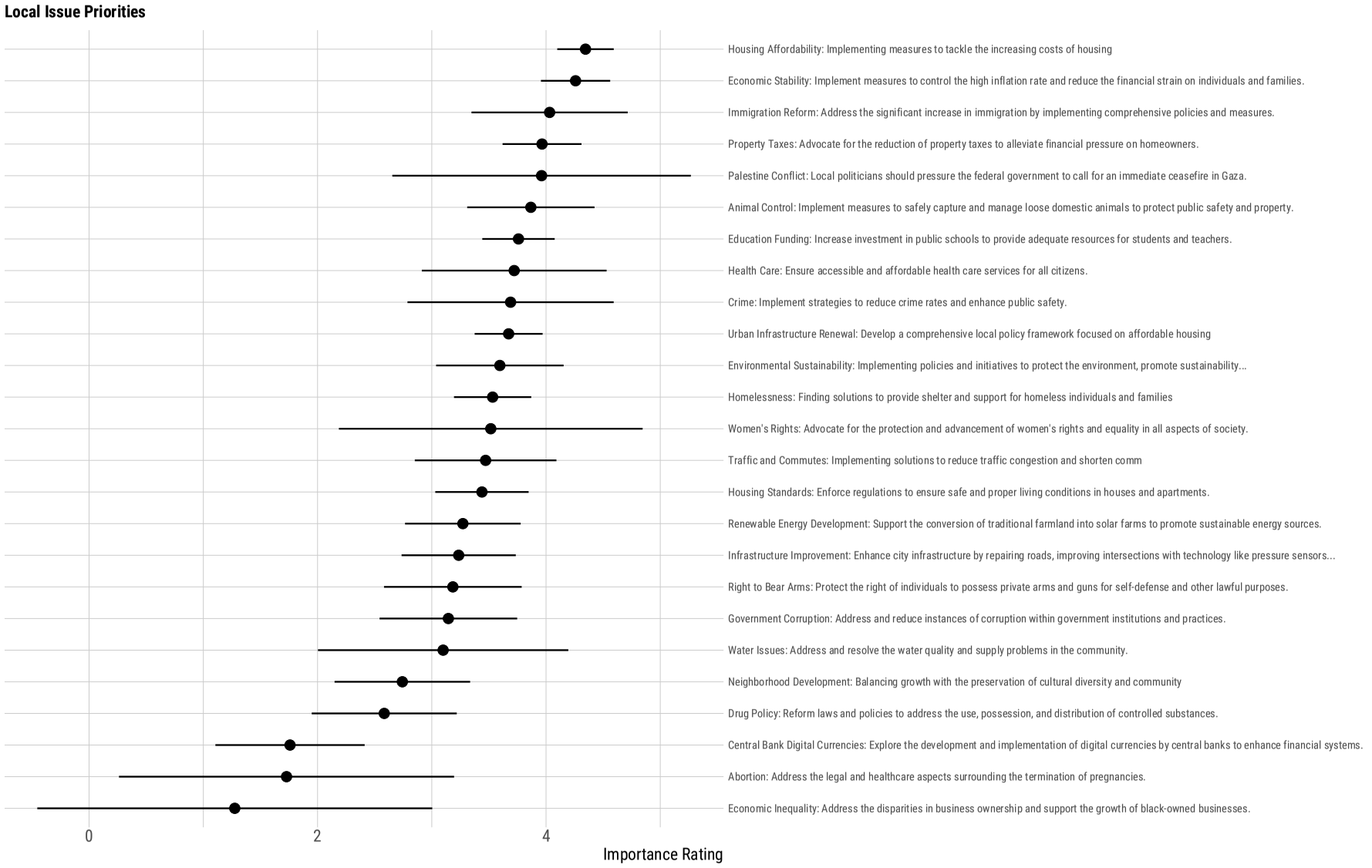

Figure K1: IPW-weighted estimates of survey questions measuring issue importance with corresponding 95% confidence intervals. Statistically significant (insignificant) differences at conventional levels of significance (α = .05; two-tailed) are depicted using solid (dashed) lines.

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. (2023). Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Agrawal, S. and Goyal, N. (2017). Near-optimal regret bounds for thompson sampling. *Journal of the ACM (JACM)*, 64(5):1–24.

Beeching, E., Fourrier, C., Habib, N., Han, S., Lambert, N., Rajani, N., Sanseviero, O., Tunstall, L., and Wolf, T. (2023). Open llm leaderboard.

Carrasco-Farré, C. (2022). The fingerprints of misinformation: how deceptive content differs from reliable sources in terms of cognitive effort and appeal to emotions. *Humanities and Social Sciences Communications*, 9(1):1–18.

Hadad, V., Hirshberg, D. A., Zhan, R., Wager, S., and Athey, S. (2021). Confidence intervals for policy evaluation in adaptive experiments. *Proceedings of the national academy of sciences*, 118(15):e2014602118.

Inan, H., Upasani, K., Chi, J., Rungta, R., Iyer, K., Mao, Y., Tontchev, M., Hu, Q., Fuller, B., Testuggine, D., et al. (2023). Llama guard: Llm-based input-output safeguard for human-ai conversations. *arXiv preprint arXiv:2312.06674*.

Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D. S., de las Casas, D., Hanna, E. B., Bressand, F., Lengyel, G., Bour, G., Lample, G., Lavaud, L. R., Saulnier, L., Lachaux, M.-A., Stock, P., Subramanian, S., Yang, S., Antoniak, S., Scao, T. L., Gervet, T., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. (2024). Mixtral of experts.

Leech, B. L. (2002). Interview methods in political science. *PS-WASHINGTON-*, 35(4):663–664.

Mellon, J., Bailey, J., Scott, R., Breckwoldt, J., and Miori, M. (2022). Does gpt-3 know what the most important issue is? using large language models to code open-text social survey responses at scale. *Using Large Language Models to Code Open-Text Social Survey Responses At Scale (December 22, 2022)*.

Offer-Westort, M., Coppock, A., and Green, D. P. (2021). Adaptive experimental design: Prospects and applications in political science. *American Journal of Political Science*, 65(4):826–844.

Palmer, A., Smith, N. A., and Spirling, A. (2023). Using proprietary language models in academic research requires explicit justification. *Nature Computational Science*, pages 1–2.

Perens, B. et al. (1999). The open source definition. *Open sources: voices from the open source revolution*, 1:171–188.

Qin, C. and Russo, D. (2022). Adaptive experimentation in the presence of exogenous nonstationary variation. *arXiv preprint arXiv:2202.09036*.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. (2023). Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Treier, S. and Jackman, S. (2008). Democracy as a latent variable. *American Journal of Political Science*, 52(1):201–217.

Velez, Y. R., Porter, E., and Wood, T. J. (2023). Latino-targeted misinformation and the power of factual corrections. *The Journal of Politics*, 85(2):789–794.

Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E. P., Zhang, H., Gonzalez, J. E., and Stoica, I. (2023). Judging llm-as-a-judge with mt-bench and chatbot arena.