

THE UNIVERSITY OF CHICAGO

DEFINING AND MEASURING QUALITIES OF LANGUAGE FOR
HUMAN-CENTERED AI

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE DIVISION OF THE PHYSICAL SCIENCES
IN CANDIDACY FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE

BY
KAREN ZHOU

CHICAGO, ILLINOIS
AUGUST 2026

Copyright © 2026 by Karen Zhou
All Rights Reserved

TABLE OF CONTENTS

LIST OF FIGURES	vi
LIST OF TABLES	ix
ACKNOWLEDGMENTS	xi
ABSTRACT	xiii
1 INTRODUCTION	1
2 QUANTIFYING THE UNIQUENESS AND DIVISIVENESS OF PRESIDENTIAL DISCOURSE	5
2.1 Introduction	5
2.2 Data	6
2.2.1 Data collection	8
2.2.2 Opponent mention sentence tagging	9
2.3 Uniqueness and Divisiveness Metrics	10
2.3.1 LLM-based uniqueness	11
2.3.2 Divisive speech lexicon	14
2.3.3 References to political opponents	18
2.4 Results	19
2.4.1 Donald Trump is unique among all presidential candidates in all types of speech	19
2.4.2 Trump speaks most divisively	23
2.4.3 References to political opponents in speech	25
2.4.4 Relating uniqueness, divisiveness, and opponent mentions	29
2.5 Discussion	29
3 FROM FEEDBACK TO CHECKLISTS: GROUNDED EVALUATION OF AI-GENERATED CLINICAL NOTES	33
3.1 Introduction	33
3.2 Related Work	35
3.3 Data	37
3.3.1 User Feedback	37
3.3.2 Human Preference Ratings	38
3.3.3 Reference Notes: ACI-bench	38
3.4 Pipeline & Implementation	39
3.4.1 Checklist Generation Methods	40
3.4.2 Checklist Refinement	41
3.5 Results	45
3.5.1 Generation and Refinement	45
3.5.2 Checklist Questions	46

3.6	Evaluation	49
3.6.1	Feedback Coverage & Diversity and LLM Enforceability	49
3.6.2	Predictive Power	50
3.6.3	Robustness against Perturbations	51
3.6.4	Correlation with Human Preference Ratings	54
3.7	Conclusion	55
3.8	Ethical Considerations	58
4	AUTOCHECKLIST: COMPOSABLE PIPELINES FOR CHECKLIST GENERATION AND SCORING WITH LLM-AS-A-JUDGE	60
4.1	Introduction	60
4.2	Related Work	62
4.3	System Details	63
4.4	Deployment and Usage	67
4.5	Validation	72
4.6	Case Study: Author Rebuttal Checklists	75
4.7	Conclusion and Future Work	77
5	CIVICCHATS: DESIGNING AI ASSISTANTS FOR CIVIC DECISION-MAKING	79
5.1	Introduction	79
5.2	Methods	81
5.2.1	Study design	81
5.2.2	Experiment Flow	82
5.2.3	Evaluating Vote Rationales	84
5.2.4	Participants and Deployment	87
5.3	Results	88
5.3.1	Engagement	89
5.3.2	Voting Behavior	89
5.3.3	Quality of Vote Rationale	90
5.4	Conclusion	93
6	CONCLUSION	95
	REFERENCES	99
A	SUPPLEMENTARY MATERIAL FOR CHAPTER 3	112
A.1	LLM-based Uniqueness Robustness Checks	112
A.1.1	BPC	112
A.1.2	Correlation with Readability Metrics	112
A.1.3	Uniqueness scores over time	112
A.1.4	Top %-ile uniqueness	113
A.1.5	LLM-based uniqueness results with unmasked model	113
A.2	Additional Divisive Word Lexicon Results	113
A.2.1	Divisive Word Lexicon	113

A.2.2	Usage over time	114
A.2.3	Usage by candidate	114
A.2.4	Utterances Using Divisive Words	114
A.2.5	Annotator Agreement	114
A.3	Additional Opponent Mention Results	115
A.3.1	Rate of opponent mentions	115
A.3.2	Fightin’ Words overlap	116
B	LLM PROMPTS FOR FROM FEEDBACK TO CHECKLISTS	129
B.1	LLM Prompts	129
B.1.1	Baselines	129
B.1.2	Feedback Checklists	129
B.1.3	Unit Tests	131
B.1.4	Feedback Coverage	132
C	CIVICCHATS SUPPLEMENTARY MATERIAL	133
C.1	Additional Methodology Details	133
C.2	Chatbot Development and Evaluation	135
C.3	Public Website	136

LIST OF FIGURES

2.1	Distribution of sentence lengths among all speakers, across debates, SOTU, and campaign data (error bars represent 95% CI). Donald Trump tends to use the shortest sentences on average.	7
2.2	Conceptual example of how the FW overlap metric is calculated for $N = 3$	17
2.3	Average sentence uniqueness for each speaker, across all data types. Higher bars indicate greater uniqueness, i.e., that speaker’s speech is less likely to be uttered by other candidates. Trump is the most distinctive speaker among these candidates for debates and SOTU speeches. Error bars correspond to the 95% CI. . .	20
2.4	Trump’s average sentence uniqueness scores compared to the others aggregated by party. The number of sentences is denoted on each bar (with error bars representing 95% CI). Trump is significantly distinct from other Republicans. The uniqueness scores of the remaining Republicans are closer to those of Democrats than with Trump. For SOTU and campaign speeches, Republicans (without Trump) are less unique than Democrats on average.	21
2.5	Sentence uniqueness across different sentence lengths, for Trump, all other Republicans, and Democrats. For debates and SOTU, Trump is consistently more distinctive across all sentence lengths.	21
2.6	Overall percentage of words used that are divisive. Trump uses the most words from our divisive lexicon, in all types of speech.	22
2.7	Validation of the LLM-based uniqueness metric with alternative LLMs. The resulting trends are consistent with original findings that Trump’s speech is more unique than other Democratic and Republican presidential candidates.	23
2.8	Uniqueness of speech broken down by opponent mentions, for debates (the error bars represent 95% CI). Trump is most distinctive regardless of whether he references an opponent or not.	26
2.9	FW overlap scores across different top- n thresholds, comparing Trump and the other candidates aggregated by party.	28
2.10	Top-15 FW overlap score in debates for each candidate. Trump’s FW associated with opponent mentions generally have the lowest overlap in adjective usage compared to other candidates, which is another indication that his language is distinctive.	28
2.11	Pairwise analysis of our three proposed metrics, aggregated globally across all speakers. Figure 2.11a shows that in debates and campaign speeches, sentences that use divisive language tend to be more unique as well. For sentences containing opponent mentions in debates, Figure 2.11b shows that such utterances tend to be more distinctive and Figure 2.11c shows that they tend to have higher divisive word usage as well.	30
3.1	Example checklist questions for the Assessment and Plan section of a clinical note. The checklist score consists of the proportion of satisfied questions. . . .	34
3.2	Proposed end-to-end pipeline.	39

3.3	α values for the objective function, where α is the weight of the coverage term. The α value is set to 0.5 for the final checklists, since it provides balance between coverage and diversity.	44
3.4	Objective score (coverage and diversity) vs. length of checklist. $k = 20$ is the optimal number of questions.	46
3.5	The enforceability score of each checklist question, indicating how LLM enforceable each one is. The feedback checklist has both a greater quantity of perfectly enforceable questions and a higher average rate of enforceability.	50
3.6	The feedback checklist has a higher perturbation Δ than the baseline checklist. It is more robust against perturbations, particularly for missing information, organization, and redundancy/hallucination.	52
3.7	The feedback checklist has a higher average score for the reference notes than the baseline checklist.	54
3.8	Correlation with human preference ratings is significant for our checklist ($p \leq 0.05$ from a paired t -test, Cohen’s $d = 0.28$), but not for the baseline.	55
4.1	In the <i>Compare</i> page of AutoChecklist’s UI, users select methods and enter an input to generate side-by-side checklists, enabling direct comparison of generation strategies on the same task.	61
4.2	<i>Instance-level</i> generators produce one checklist per input; criteria are tailored to each specific task (target). <i>Corpus-level</i> generators produce one checklist for an entire dataset of targets, capturing general quality patterns derived from higher-level signals.	65
4.4	The Batch results page showing per-item score breakdowns and aggregate statistics for a dataset evaluation run.	68
4.5	Additional UI pages to support prompt iteration and dataset evaluation.	69
4.8	Validation of built-in pipelines on RewardBench and SummEval.	73
4.9	Author rebuttal checklists generated from ICLR 2019 reviews.	76
5.1	UI for treatment conditions. Control group sees only the official text of the ballot measure. Ballotpedia sees the official text and Ballotpedia sections. Chatbot conditions see the official text and Ballotpedia sections, plus the chatbot interface.	82
5.2	UI for post-treatment questions.	83
5.3	Pairwise annotation UI for rationales, inspired by Chatbot Arena, to develop Elo-style ratings of different study conditions.	84
5.4	Tradeoff examples as identified with an LLM tagger, displaying tradeoffs (A and B) that are resolved (chose A or B) vs. not resolved.	85
5.5	Official texts of 2026 ballot measures for Colorado and Florida used in user studies.	87
5.6	Engagement metrics for different treatment conditions, measured as active minutes on treatment page (excluding AI response time) and user chat message lengths.	88
5.7	Lengths of vote rationales from different study conditions, with darker bars indicating significantly longer lengths than the control.	88
5.8	Vote distribution for different treatment conditions, for each ballot measure.	89
5.9	Elo scores and corresponding significance for the five study conditions.	90

5.10	Rates of resolved vs. unresolved tradeoffs across conditions.	91
5.11	Overlap scores with official ballot text for different treatment conditions.	92
5.12	Overlap scores for different treatment conditions.	93
A.1	Average sentence BPC across each data type. In campaigns, Trump is the most unpredictable among those candidates; however, he is ranks closer to the middle for predictability in debates and SOTU	116
A.2	Heatmaps of Spearman Correlation Coefficients between our uniqueness metric and existing readability scores (FKGL, Gunning Fog, Dale Chall, and ARI). There is little to no correlation between our score and readability; in contrast, there is strong correlation between the different readability indexes.	117
A.3	Uniqueness score over time, aggregated by political party.	118
A.4	For debates, Trump is consistently most unique for each term that he runs for president (2016, 2020). He is slightly more unique in his first election cycle, in 2016. The error bars represent 95%-confidence intervals.	118
A.5	For SOTU, Trump is consistently most unique throughout all years of his presidency. His speech distinctiveness increases after the first year of his term. The error bars represent 95%-confidence intervals.	119
A.6	For campaign speeches, Trump is consistently most unique for each term that he runs for president (2016, 2020). His campaign speech in his second election cycle is slightly more unique than his first time. The error bars represent 95%-confidence intervals.	119
A.7	Uniqueness score by candidate, of sentences within the top 10-ile of scores for each candidate (the error bars represent 95%-confidence intervals). Note that Trump continues to be most unique in all datasets.	120
A.8	Average sentence uniqueness for each speaker, across all data types, using the UNMASKED model (the error bars represent 95%-confidence intervals). Trump is still the most unique speaker in all types of speeches.	121
A.9	Frequency of usage over time	122
A.10	Heatmaps showing divisive word utterance frequency, normalized over all words spoken, by speaker.	123
A.11	Distribution of sentences containing opponent mentions among all speakers, across the data types. Note the different rates for debates vs. SOTU. Trump has the highest rate of opponent mentions in debates.	127
A.12	Fightin' Words overlap results from debates. Trump's FW associated with opponent mentions generally have the lowest overlap in adjective usage compared to other candidates, which is another indication that his language is distinctive. . .	128
C.1	CivicChats.org.	136

LIST OF TABLES

2.1	Overall statistics of the three data types we use to present our main findings.	8
2.2	The 178 words in our divisive word lexicon, which are verified as being divisive by the majority of 4 annotators. Our original ten seed words are denoted with a [*]	16
2.3	Percentage of majority class annotator agreement on binary labels (divisive? “yes” or “no”). 58.9% of the 360 initial terms had full annotator agreement, and 86.7% of these 360 have at least 3/4 annotators agreeing on the label. When limiting to the 178 terms that at least 3/4 annotators agree are divisive, 69.1% have full agreement on the label.	17
2.4	Top 10 divisive words used by Trump in each type of speech. Raw counts of usage are denoted in parentheses next to each word.	24
2.5	Example sentences from debates and campaigns, with preceding context, spoken by Trump that use words from the divisive lexicon. Bolded words are matched with our divisiveness lexicon.	25
2.6	Top-25 FW adjectives said by Donald Trump in debates, either in reference to opponents or not. Z-scores are provided in parentheses.	29
3.1	Example synthetic user feedback and star ratings.	37
3.2	Evaluation of the final checklists, with respect to feedback coverage, diversity, and LLM enforceability. Diversity is additionally evaluated by intra-cluster and inter-cluster distances. The feedback checklist beats the baseline checklist in each category.	45
3.3	Descriptions of the perturbed notes for evaluation. The number of notes is the number of reference notes that were perturbed with each perturbation. Perturbations beginning with <code>prm</code> are from Wang et al. [2025b]	51
4.1	Five checklist generation abstractions in <code>AutoChecklist</code> . Each represents a distinct strategy for deriving evaluation criteria. The rightmost column lists built-in pipelines that instantiate each generator with paper-specific prompt templates and scoring strategies.	64
4.2	Built-in pipelines. Each pipeline pairs a generator with a default <code>ChecklistScorer</code> configuration and paper-specific prompt templates. Score metrics: pass rate, normalized, or weighted (see §4.3).	66
4.3	Preference discrimination on RewardBench pairs using instance-level pipelines, under both the original <code>gpt-5-mini</code> generator and a <code>claude-sonnet-4.6</code> (via OpenRouter) ablation. The <code>gpt-5</code> scorer is held fixed across runs. W/L/T = Win/Loss/Tie counts; Δ = mean (chosen – rejected); d = Cohen’s d . All Wilcoxon p -values are $< 10^{-8}$	72
4.4	Corpus-level checklist validation on SummEval under both the original <code>gpt-5-mini</code> generator and a <code>sonnet-4.6</code> (via OpenRouter) ablation; the <code>gpt-5</code> scorer is held fixed. ρ_s : Spearman vs. human scores; MAE on rescaled 1–5 scores. Pearson and Kendall correlations tracked Spearman closely ($ \Delta < 0.06$) and are omitted for space. All ρ_s are significant ($p < .001$).	74

4.5	Evaluation of checklist-based rebuttal scoring on ICLR 2019. RATING: Spearman r_s between pass rate and reviewer rating. ACCEPTANCE: rank-biserial effect size $ d $ and pass rate difference $\Delta\bar{x} = \bar{x}_{acc} - \bar{x}_{rej}$. RATING CHANGE: point-biserial r_{pb} and ROC-AUC for predicting whether the reviewer changed their rating post-rebuttal. $*p < .05$, $**p < .01$, $***p < .001$	76
A.1	[DEBATES] Example usages of words from our divisive lexicon throughout debates.	124
A.2	[SOTU] Example usages of words from our divisive lexicon throughout SOTU addresses. In general, there are fewer uses of divisive words during SOTU speeches.	125
A.3	[CAMPAIGN] Example usages of words from our divisive lexicon throughout campaigns.	126

ACKNOWLEDGMENTS

First, I thank my advisor, Chenhao Tan, for his support and mentorship during the past five years. I am grateful for his guidance and patience, as well as our many candid conversations, which have been pivotal to my growth into a more independent and confident researcher. It has been a privilege to work in the Chicago Human+AI lab with him on interesting and meaningful research.

I am also grateful for the support of my dissertation committee members, Mina Lee and Lexing Xie. Their encouragement and suggestions have helped me refine my thinking and communication, turning my dissertation into a stronger body of work. Lexing’s invitation to present at the Australian National University was especially meaningful; visiting her lab was very special and memorable. Ben Zhao’s insights during my MS defense have continued to be thought-provoking in the later stages of this PhD as well. My studies were also supported by a GFSD Fellowship — thank you to Joretta Joseph and my sponsors at NIST for their investment and support of my work.

This PhD would not have happened without Lillian Lee and Ana Smith. Our collaborations at Cornell University nurtured my interest in NLP research, which led me to pursue this journey. I have also benefitted from the guidance of mentors and collaborators from my internships: Gurudutt Hosangadi, Gargi Vyas, John Giorgi, Pranav Mani, Peng Xu, and Davis Liang.

My labmates and friends have been a significant highlight of my time in the program. During my tenure in CHAI, I have been fortunate to work alongside Vivian Lai, Shi Feng, Han Liu, Chao-Chun (Joe) Hsu, Chacha Chen, Rosa Zhou, Mourad Heddaya, Haokun Liu, Dang Nguyen, Xiaoyan Bai, Qirun Dai, Julia Mendelsohn, Cristina Garbacea, Xiao Liu, Chenghao Yang, Jiamin Yang, Zander Meitus, Milo Chase, and others. I especially thank Cristina and Chacha for their advice and support as senior labmates. Chacha is also a confidante whose humor and perspective have carried me through several professional and

personal challenges. Mourad has been a great collaborator, and the camaraderie from him and Haokun made the final stretch of this PhD more manageable. Xiaoyan and Jiamin have enriched my time in Chicago through our many shared adventures. I'm grateful to have met Iris Lin, who has always been generous with her encouragement and wisdom. Debasmita Bhattacharya has been a dear friend from our undergraduate days to our respective graduate programs, and I am glad that we can always reconnect as if no time has passed. Also since college, I've been fortunate to have Max Chen as a friend and supporter whose work ethic and research accomplishments have been inspiring to follow. Beyond the academic realm, Emily, Irene, Eli and the State Place gamers, Suzanne and Aldo, Lahari, Jose, and Mandii have shared many hours of fun, laughter, and companionship with me (and provided occasional cat-sitting services). I have been proud to call Chicago my home alongside many of them.

I owe special acknowledgements to the following people who have sustained me and kept me grounded over the years. To Rishi, thank you for your steady presence and unwavering support. I am indebted to the warmth and generosity of the Advanis and Panjwanis. To Chloe and Rachel, thank you for navigating the highs and lows of life with me since grade school. It is the greatest privilege to feel wholly seen and to know I can rely on your unconditional support and acceptance. And to Josh, thank you for the new experiences and cherished memories we (and Ruth) have shared; I'm grateful to have been able to depend on you. Each of you believed in me wholeheartedly, even when I doubted myself. I could not have made it this far without you.

Lastly, I thank my parents for their sacrifices and support. They have trusted in and accepted my life decisions. I deeply appreciate the opportunities their hard work has afforded me; their investment in my education gave me a strong foundation and made it possible for me to pursue this PhD.

To all of you and everyone else who has been a part of this journey, I extend my deepest gratitude. Thank you.

ABSTRACT

Natural language is a fundamental medium of communication, across humans and machines. It is often subjective and contextual, meaning that many language-based tasks lack singular ground truth answers. However, reliable analysis and evaluation of such data and tasks are still crucial, especially in high-stakes settings with direct impact on humans. This thesis approaches these challenges by grounding the measurement of subjective concepts in human signals of domain knowledge and building frameworks that leverage multiple, complementary dimensions for evaluation. We first present a suite of metrics to quantify rhetorical distinctiveness and divisiveness in political discourse. By combining large language models, lexical resources, and graph-based analyses, we characterize how U.S. presidents differ in linguistic style and use of antagonistic language in their public speeches. The second study introduces an approach for distilling large-scale clinician feedback into structured, LLM-enforceable checklist items, in order to define criteria for clinical note quality. Used with LLM-as-a-Judge, these checklists yield interpretable and auditable criteria that demonstrate alignment with expert judgment and outperform a zero-shot baseline in coverage, diversity, and predictive power. We incorporate this pipeline into an open-source library that unifies multiple approaches for automatic checklist generation and scoring from the literature, to support extensibility of rubric-based evaluation. Finally, we assess the impact of human-AI conversation on voter engagement and reasoning around ballot measures. Through controlled user studies, we observe considerable differences in vote distributions across treatment groups, with discourse-oriented chatbot conditions yielding user-written rationales that are preferred under Elo-style ratings, deliberative of tradeoffs, and self-articulated. Together, these studies demonstrate how subjective and abstract qualities can be quantified for natural language tasks, by incorporating the domain expertise of humans, scalability of NLP tools and LLMs, and multi-faceted interpretations of quality.

CHAPTER 1

INTRODUCTION

Natural language has been used for communication and decision-making by humans and increasingly also by machines (e.g., language models). Since language can shape real outcomes in high-stakes settings like medicine and political communication [Shulman et al., 2022, Ong et al., 2025, Flusberg et al., 2024], measuring and understanding it reliably and at scale is important. Traditionally, evaluation has been done with labeled data and gold references that assume some notion of correctness, but language is subjective and contextual, which means there may not be a single correct answer or ground truth. Gold references and labels aren’t always aligned with real user intuitions; static benchmarks can diverge from human preferences [Aristidou, 2026]. Human judgment is typically the gold standard, but even experts can disagree on interpretations of language or even what qualities are important to capture, which is reflected in low inter-rater agreement [Arora et al., 2025].

To support human-centered AI (i.e., attuned to the needs and preferences of human users), reliable analysis and evaluation of language are crucial. Recent advances in natural language processing (NLP) and large language models (LLMs) offer scalable capabilities to do so, but their reliability on subjective tasks depends on how we define and measure the qualities of interest. This thesis approaches these challenges by leveraging human domain expertise to quantify subjective concepts and by building frameworks that leverage multiple, complementary dimensions for evaluation.

In the first part of this thesis, we measure rhetorical distinctiveness and divisiveness to investigate the following questions: Do American presidents speak discernibly differently from each other? If so, in what ways? And are these differences confined to any single medium of communication? To answer these questions, we introduce a novel metric of uniqueness based on large language models, develop a new lexicon for divisive speech, and present a framework for assessing the distinctive ways in which presidents speak about their

political opponents. Applying these tools to a variety of corpora of presidential speeches, we find considerable evidence that Donald Trump’s speech patterns diverge from those of all major party nominees for the presidency in recent history. Trump is significantly more distinctive than his fellow Republicans, whose uniqueness values appear closer to those of the Democrats. Contributing to these differences is Trump’s employment of divisive and antagonistic language, particularly when targeting his political opponents. These differences hold across a variety of measurement strategies, arise on both the campaign trail and in official presidential addresses, and do not appear to be an artifact of secular changes in presidential communications.

Then, moving from human-generated to machine-generated language, we develop a framework for evaluating the quality of AI-generated clinical notes by distilling large-scale clinician feedback into structured, LLM-enforceable checklist items. AI-generated clinical notes are increasingly used in healthcare, but evaluating their quality remains a challenge due to high subjectivity and limited scalability of expert review. Existing automated metrics often fail to align with real-world physician preferences. To address this, we propose a pipeline that systematically distills real user feedback into structured checklists for note evaluation. Used with LLM-as-a-Judge, these checklists yield interpretable and auditable criteria, grounded in human feedback and enforceable by LLM-based evaluators. Using deidentified data from over 21,000 clinical encounters (prepared in accordance with the HIPAA safe harbor standard) from a deployed AI medical scribe system, we show that our feedback-derived checklist outperforms a baseline approach in offline evaluations on coverage, diversity, and predictive power for human ratings. Extensive experiments confirm the checklist’s robustness to quality-degrading perturbations, significant alignment with clinician preferences, and practical value as an evaluation methodology. In offline research settings, our checklist offers a practical tool for flagging notes that may fall short of our defined quality standards.

More broadly in the literature, checklists have emerged as a popular approach for inter-

pretable and fine-grained evaluation, particularly with LLM-as-a-Judge. Beyond evaluation, these structured criteria can serve as signals for model alignment, reinforcement learning, and self-correction. Thus to better integrate our previous pipeline into the broader field, we develop an open-source library, **AutoChecklist**, that unifies multiple approaches from the literature for automatic checklist generation and scoring, supporting the extensibility of rubric-based evaluation across domains. The library unifies checklist-based evaluation into composable pipelines, with built-in support for open-source and proprietary LLM providers. Validation experiments confirm that these checklist methods significantly align with human preferences and quality ratings, and a case study on ICLR peer review rebuttals demonstrates flexible domain adaptation.

Finally, we assess the impact of human–AI conversation on voter engagement and reasoning around ballot measures. We develop the CivicChats platform, which hosts three chatbot personas for different modes of deliberation, and conduct controlled user studies to evaluate the impact of these conversation modes on voter engagement and reasoning. We observe considerable differences in vote distributions across treatment groups, with deliberation-oriented chatbot conditions yielding user-written rationales that are preferred under Elo-style ratings, decisive around identified tradeoffs, and re-articulated from their chat interactions.

In the following chapters, we:

- Introduce a suite of metrics for quantifying the uniqueness and divisiveness of political rhetoric
- Develop a framework for evaluating the quality of AI-generated clinical notes with checklists grounded in physician feedback
- Release an open-source library that unifies LLM-based checklist generation and scoring methods for downstream applications
- Conduct controlled user studies and develop methods to assess the impacts of human–AI conversation on civic decision-making around ballot measures

Together, these studies demonstrate how abstract qualities of language can still be measured in the absence of ground truth. We end with a discussion of the implications of this thesis and its potential future directions.

CHAPTER 2

QUANTIFYING THE UNIQUENESS AND DIVISIVENESS OF PRESIDENTIAL DISCOURSE

2.1 Introduction

The rise of the modern presidency is defined, in no small measure, by the chief executive’s changing relationship to the American public. In what Jeffrey Tulis [Tulis, 1987] calls the “rhetorical presidency,” modern presidents are expected to routinely stand before the public in order to explain, persuade, inform, and instruct. How precisely they do so, though, is open to interpretation. While certain norms of communication govern their behavior, presidents have a fair measure of discretion to speak as they choose.

When fulfilling their oratory duties, do modern presidents adhere to a common script? Or do some presidents defy rhetorical conventions and speak in ways that, at least among themselves, are novel and surprising? Leveraging recent advances in large language models (LLMs) [Radford et al., 2018], we develop a new quantifier of uniqueness by directly measuring the unpredictability of language patterns. In addition, we incorporate more standard lexical techniques to examine a prevalent yet understudied construct, divisiveness [Cinar et al., 2020]. We operationalize *divisiveness* as language that is intended to impugn and delegitimize the speaker’s target, and we develop a new lexicon for such speech. Furthermore, we introduce a comparative framework for assessing mentions of opponents, which are especially prevalent in presidential debates.

We use our proposed tools to analyze large and diverse corpora of presidential speech. In doing so, we are able to both characterize the overall distinctiveness of presidents’ speech patterns and examine specific qualities that previous research has overlooked. Moreover, we can distinguish general speech patterns from those directed towards one’s political opponents.

In nearly all of our analyses, Donald Trump appears as a clear outlier. On the campaign

trail, in presidential debates, and in official presidential addresses, we find, Trump’s speech patterns routinely differ from those of all recent presidents — lending credence to Kurt Anderson’s observation that, “The version of English [Trump] speaks amounts to its own patois, with a special vocabulary and syntax and psychological substrate” [Anderson, 2018].

Like previous scholars [Kayam, 2018, Frischling, 2019, Jordan et al., 2019], we find that Trump tends to communicate in shorter, more simplistic sentences. But whereas previous quantitative research on presidential rhetoric relied exclusively on lexicons (e.g., [Jordan et al., 2019]) and sentiment classification (e.g., [Zhong, 2016]), which necessarily disregard contextual information included in the body of speeches, our methods are able to show that Trump speaks in ways that are holistically different from all modern presidents. These differences are pervasive and large — so much so, in fact, that the observed differences between Trump and his fellow Republicans exceed those between Republicans and Democratic presidents. Contributing to these differences, we show, is Trump’s tendency to speak in ways that are especially divisive, particularly when focusing on his political opponents.¹ These findings, moreover, are robust to a variety of measurement strategies, arise across rich and diverse corpora of texts, and do not appear to be an artifact of secular time trends.

2.2 Data

Our research investigates three genres of political speech: presidential debates in general elections since 1960, State of the Union (SOTU) speeches since 1961, and a sample of campaign speeches assembled by the American Presidency Project [Woolley and Peters, 1999]. Debates and SOTU speeches are generally standard across presidents and time, so all avail-

1. This finding is consistent with previous research showing that Trump uses language that is explicitly intended to evoke an emotional response from his revision audience [Hart, 2020], and that Trump’s speech patterns reveal his autocratic ambitions [Körner et al., 2022], anti-democratic views [Jamieson and Taussig, 2017], a willingness to condone political violence [Ntontis et al.], a commitment to populism [Cinar et al., 2020], and low levels of analytic thinking [Jordan et al., 2019]. None of these scholars, however, have developed a lexicon of divisiveness, as we do in this paper.

able documents are included in our main analyses.² Publicly available campaign documents, however, are imbalanced and not comprehensive over the same time frame, so these corpora are limited to speeches delivered within one month of Election Day in every presidential election since 2008. Table 2.1 provides summary statistics of the final datasets for which we present results.³

Figure 2.1 shows the distribution of sentence lengths from each speaker across the datasets. From the outset, we note that Trump tends to speak in markedly shorter sentences than do other presidents. Whereas Trump’s sentences range from 10.4 to 14.5 words in the three data sources, the overall averages range from 17.6 to 24.4 words. And when comparing presidents within each data source, Trump registered the single lowest number of average words per sentence among all presidents within debates and campaigns, and the second lowest number within SOTU addresses.

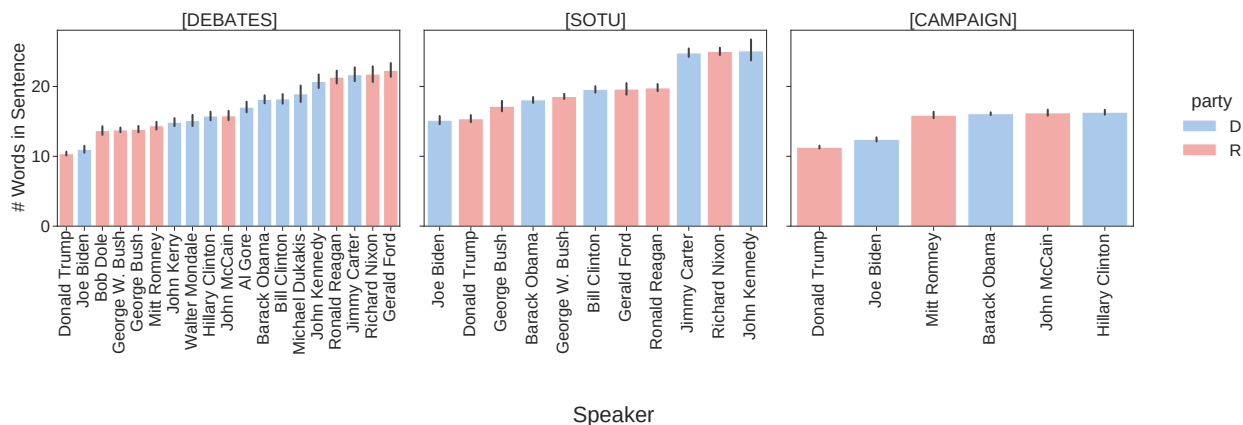


Figure 2.1: Distribution of sentence lengths among all speakers, across debates, SOTU, and campaign data (error bars represent 95% CI). Donald Trump tends to use the shortest sentences on average.

2. Additional data are used for model training. See §2.3.1 for details.

3. Within these data, Donald Trump spoke in five debates (3,610 sentences), four SOTU addresses (1,471 sentences), and 28 campaign speeches (7,488 sentences) between the years of 2016-2020.

Table 2.1: Overall statistics of the three data types we use to present our main findings.

	DEBATES	SOTU	CAMPAIGN
# speeches	35	67	187
# sentences	35,096	22,775	36,295
date range	1960-2020	1961-2022	2008-2020
# candidates compared	19	11	6
party ratio (Dem:Rep)	10:9	5:6	3:3
avg. sents/candidate	1,449	2,070	5,701
avg. candidate sent len.	18.76	24.42	17.62
avg. speeches/candidate	3.6	6.1	31.2

2.2.1 Data collection

Each dataset is scraped from the American Presidents Project (APP) database [Woolley and Peters, 1999]. In general, all text for each speech is scraped and tagged with its speaker. In addition, some metadata are collected, like the date and title of the speech or document. In order to properly annotate speakers, particularly in the debate speeches, the speaker’s name is identified based on a variety of factors like special text styling, string matching, and html formatting. In addition, audience annotations, like “laughter” and “applause”, are filtered out from the data to the best of our ability. After scraping, the data are subsampled and reviewed to ensure quality and correct speaker annotations.

Debates data The Presidential debates data include General Election debates from 1960 to the present day and Primary Election debates since 2000. For the purposes of this analysis, we exclude primary debates and any general election debates that featured vice-presidential candidates. Sentences spoken by candidates who were not the nominee for either the Republican or Democratic party are also excluded (e.g., Ross Perot in 1992, moderators, and audience members).

SOTU data Because we focus on modern presidents, State of the Union addresses from 1961 onwards are used in deriving our main results. The SOTU dataset includes a few

speeches that are not officially considered State of the Union addresses but functionally operate as such. Beginning with Reagan, recent presidents have started addressing a joint session of Congress shortly after their inaugurations. According to the American Presidency Project, “it is probably harmless to categorize these as State of the Union messages (as we do). The impact of such a speech on public, media, and congressional perceptions of presidential leadership and power should be the same as if the address was an official State of the Union”.⁴

Campaigns data All campaign documents from 1932 to 2020 are collected initially. We manually identify keywords that indicate campaign speeches of interest and filter speech titles based on those keywords. Keywords selected are: “remarks”, “speech”, and “address”. Manually removed documents include: speeches with reporters, question/answer format speeches, press releases, town halls, press releases, co-appearances with other politicians and spouses, etc.

Subsequently, we apply clique filtering to ensure that any duplicate speeches are removed. It is common that a candidate has a stump speech that is delivered at multiple events, and to diversify and balance the dataset, we remove such duplicates. These de-duplicated data are used for model training.

After filtering, our final campaign dataset includes only speeches delivered within the month before election day and only from candidates since 2008. This smaller subset is used to generate the results presented in the main paper.

2.2.2 Opponent mention sentence tagging

After each dataset is filtered, sentences are tagged as mentioning an opponent or not through an automated process. Sentences are classified as definitely including an opponent mention,

4. <https://www.presidency.ucsb.edu/documents/presidential-documents-archive-guidebook/annual-messages-congress-the-state-the-union>

possibly including an opponent mention, or not including an opponent mention based on keywords and the presence of parts of speech. For debates, “opponents” are identified by the name(s) of the debate partner or their party. After automatically tagging the debates dataset, the sentences that are labeled as possibly including an opponent mention are manually reviewed by an expert team of four researchers. The team compares a subset of pairwise overlapping ratings for consistency. The Cohen’s κ for inter-coder agreement is roughly 0.8, which indicates substantial agreement.

For SOTU, opponent mentions only include those of other presidential candidates; names of candidates who never held presidential office are not categorized as mentions of opponents, in contrast to the debates dataset. Following a manual review of selected speeches from the SOTU dataset, it is concluded that references to nonpresidential figures are much more neutral and considerably less frequent in State of the Union speeches than they are in debate contexts.

For campaign data, we automatically tag opponents using the names of the final party candidates from the opposing party of the speaker. Opposing-party candidates from the primaries are not automatically tagged.

The strict guidelines for automatic tagging ensure high precision of the labels. Limitations of this automatic tagging method include lack of coreference resolution and difficulty of measuring recall.

2.3 Uniqueness and Divisiveness Metrics

We quantify the uniqueness of political speeches using three complementary approaches: (i) a novel metric based on LLMs; (ii) a new resource of lexicons for divisive speech; and (iii) a comparative framework for the portrayal of political opponents involving lexical features. Together, these three approaches allow for a robust and multifaceted comparison of speech patterns that involve global assessments beyond lexical comparisons and specific evaluations

of divisiveness. We introduce the intuition behind each approach below.

2.3.1 *LLM-based uniqueness*

Large language models, like the GPT family of models, have received widespread attention for their abilities to statistically characterize the complex structures of natural language text. LLMs can measure the predictability of text by calculating the likelihood of the next word or sequence of words in a given context, to produce a measure known as “perplexity,” which is typically used to evaluate the quality of LLMs [Melis et al., 2018, Chen et al., 1998]. Perplexity metrics have also been used in applications such as modeling psycholinguistic characteristics, predicting health states from transcripts, and predicting clinician fatigue [Hao et al., 2020, Colla et al., 2022, Hsu et al., 2025]. To control for tokenization methods across models, standard perplexity measures can be supplemented with measures of “bits-per-character” (BPC) [Liang et al., 2022].

Contributing to this literature, we propose a metric of “uniqueness” based on the ability of LLMs to estimate the probabilities of word sequences, and we then use these estimates to compare political speech from various presidents and presidential candidates, whether delivered from the White House or on the campaign trail. Specifically, from a pool of presidential candidates, we determine how likely the speech of one speaker is to be produced by the others. A positive value for one candidate’s sentence uniqueness, therefore, suggests that other candidates are unlikely to say it. The larger this value, the less likely other candidates are to do so.

While this definition of uniqueness may have similarity with authorship attribution, authorship attribution is setup as classification task, not the unsupervised modeling used in uniqueness [Huang et al., 2025, He et al., 2024]. Stylometry is a related task, involving quantitative assessment of linguistic features [Lagutina et al., 2019]; uniqueness differs in not specifying such features but rather learning holistic patterns.

The advantage of this metric is its consideration of the preceding context of given speech. Rather than examine words or phrases in isolation, this metric considers the order in which they appear, and thereby provides a much more nuanced characterization of speech patterns. The precise meaning of the scores, however, may be difficult to interpret because they do not reveal the exact features that make a speech distinctive, and because they only allow for comparisons within a specified pool of speakers.

Language model training and analysis We fine-tune a pretrained, 124M parameter GPT-2 model using the huggingface [Radford et al., 2019] and PyTorch Lightning [Falcon and The PyTorch Lightning team, 2019] libraries for each of the three data types, resulting in three different language models:

- LM_{DEBATES}, trained on 35 debates (35,096 sentences) from 1960 to 2020
- LM_{SOTU}, trained on 246 speeches (69,630 sentences) from 1790 to 2022
- LM_{CAMPAIGN}, trained on 640 documents (83,038 sentences) from 1932 to 2020

To preprocess data for training, we parsed each speech into sentences, prefixed each sentence with the speaker prompt (e.g., “Donald Trump:”, and masked any named entities (as identified by spaCy NER tagger) with a <ENT> mask.⁵) We fine-tune each model for 10 epochs on all available corresponding data with a learning rate of $5e-5$. Validation with larger, more recent LLMs is shared in §2.4.1.

BPC/predictability and uniqueness scores Consider a set of presidents or candidates $C = \{c_1, \dots, c_n\}$ who each have a corresponding set of sentences $\mathcal{S} = \{S_1, \dots, S_n\}$ where each set $S_i = [s_i^1, \dots, s_i^m]$, for each of the three data types.

5. Versions of each model were also trained on unmasked data. The results from these models are consistent with our main findings and are presented in §A.1.5.

We use bits-per-character (BPC), also known as bits-per-byte, as a proxy for “predictability” of a sentence. Lower BPC values correspond to higher predictability. We use BPC instead of perplexity or loss directly to account for variation in tokenization techniques.

We calculate BPC as follows. From our fine-tuned models, we are able to obtain *loss* values, $\mathcal{L}(t_i)$ for each token t_i in an input. For some sentence of tokens $s = (t_1, t_2, \dots, t_k)$ where $\text{len}(s)$ denotes the number of characters in s :

$$\mathcal{BPC}(s) = \frac{1}{\text{len}(s)} \sum_{i=1}^k \mathcal{L}(t_i)$$

In other words, $\mathcal{BPC}(s)$ is the sum of cross-entropy losses of each token in a sentence s , divided by the number of characters (bytes) in the s . We calculate the $\mathcal{BPC}(s)$ of a sentence s with a context window size of 512 tokens. That is, preceding sentences of the one in question are provided to obtain the most representative score of its predictability.

We first obtain the BPC of a sentence s_i^j with its original speaker c_i as its speaker prompt. For example, “Donald Trump: Make America great again” is denoted by $\mathcal{BPC}_{c_i}(s_i^j)$. The BPC of the same sentence s_i^j with an alternative speaker prompt c_k , e.g., “Hillary Clinton: Make America great again” is denoted by $\mathcal{BPC}_{c_k}(s_i^j)$. Now, we define a “uniqueness” score for s_i^j sentence as follows:

$$\text{SENTUNIQ}(s_i^j) = \left(\frac{1}{|C| - 1} \sum_{c_k \in C \setminus \{c_i\}} \mathcal{BPC}_{c_k}(s_i^j) \right) - \mathcal{BPC}_{c_i}(s_i^j)$$

Intuitively, this means that the uniqueness of a sentence is defined as the difference between its BPC with its original speaker prompt and the average of its BPC scores with each of the $|C| - 1$ alternative candidates as a speaker prompt. The greater this difference is, the higher the $\text{SENTUNIQ}(s_i^j)$ score is, suggesting that sentence s_i^j is most likely to be said by the original speaker c_i and not by the other candidates. Since debates include an

opponent speaker in the transcript (e.g., Trump vs Biden), we exclude the opponent from the replacement candidates for sentences from that particular debate (i.e., $C \setminus \{\text{Trump, Biden}\}$), to avoid simulating unrealistic debates with only one speaker.

Then, the overall “uniqueness” score for a candidate c_i is the average of the SENTUNIQ scores for each of c_i ’s sentences in S_i :

$$\text{UNIQ}(c_i) = \frac{1}{|S_i|} \sum_{s_i^j \in S_i} \text{SENTUNIQ}(s_i^j)$$

Again, the intuition here is that the larger this score, the greater the average difference is for sentence uniqueness, i.e., overall, speaker c_i ’s sentences are not likely to be said by another candidate.

2.3.2 *Divisive speech lexicon*

To analyze the actual content of language used by presidents and presidential candidates, a “divisiveness” lexicon is created and applied to each dataset. We define language as “divisive” if it intends to impugn and delegitimize the speaker’s target, e.g., by attacking their intelligence, integrity, or intentions. Examples of divisive accusations include “racist”, “dishonest”, “corrupt”, or “ridiculous”. Such labels are expressly designed to put the target on defense and accentuate differences and distance between parties.

Our definition of divisive is distinct from other commonly analyzed constructs such as political “polarization”, which encompasses language that is associated more with one side than the other but is otherwise agnostic about its valence [Simchon et al., 2022]. Meanwhile, speech may be divisive without being “toxic,” which contains hateful, abusive, or offensive content [Sap et al., 2021]. Personal attacks can also be categorized as a form of toxic language, which prior work has examined primarily in the context of online, written communication [Wulczyn et al., 2017]. Nor is our measure of divisiveness the simple antonym of traditional

notions of politeness [Brown and Levinson, 1987], which usually involve honoring social conventions, showing gratitude, paying compliments, avoiding complaints and curses, and respecting the listener’s autonomy with the use of softening statements and hedges. More than just impolite or insulting, divisive speech, as we conceive it, is explicitly intended to serve the political purposes of delegitimization, marginalization, and distancing between speaker and target.

To the best of our knowledge, ours is the first divisive speech lexicon. This lexicon consists of 178 words that four researchers independently reviewed to be qualitatively “divisive” in political speech. A strength of this lexicon-based analysis is its easy interpretability and applicability. Divisive words may be used by candidates from any political party in a wide variety of settings, are readily identified, and, as we subsequently show, are broadly agreed upon by coders. As with all lexical approaches, however, the measure is inherently limited by the subjective nature of lexical evaluations and the lack of contextual consideration. Our resource, like all lexicons, by itself does not account for surrounding context like negation or valence. Strategies like pairing our lexicon with part-of-speech taggers may be leveraged for downstream tasks.

Implementation details First, a vector space word model is used to populate a list of candidate divisive words. Then, we review and refine the candidate list through researcher annotations. The word vector model used is Gensim’s glove-wiki-gigaword-300 [Řehůřek and Sojka, 2010, Pennington et al., 2014].

Ten seed “divisive” terms are chosen manually by NLP and political science experts as common politically divisive English terms, before seeing the actual speech data. The ten seed terms used are: “stupid”, “dishonest”, “unamerican”, “idiot”, “deplorable”, “pathetic”, “immoral”, “disgrace”, “incompetent”, “foolish”. With these seed terms, 350 additional terms with the highest cosine similarity in the vector space model are added to the lexicon. Each of these initial 360 terms is analyzed by four researchers as “divisive” or “not divisive” based

on the following criteria:

- No modifiers, e.g., “utterly”, “extremely”
- Should not include words that can often be used both divisively and nondivisively
- Should only include words that would be considered divisive in most political contexts

Only words that receive a majority of votes by the annotators are included in the final lexicon. The final size of the lexicon is 178 words (see Table 2.2).

stupid*, dishonest*, unamerican*, idiot*, deplorable*, pathetic*, immoral*, disgrace*, incompetent*, foolish*, irresponsible, shameful, disgraceful, disgusting, hypocritical, idiotic, unprofessional, ridiculous, despicable, unethical, outrageous, arrogant, inexcusable, ignorant, shameless, disrespectful, absurd, ludicrous, reprehensible, unconscionable, scandalous, contemptible, inept, unworthy, appalling, laughable, disingenuous, cowardly, callous, unjust, indefensible, foolhardy, selfish, preposterous, deceitful, disloyal, ashamed, illogical, distasteful, naive, heartless, crass, repugnant, sinful, unpatriotic, abhorrent, obnoxious, childish, thoughtless, stupidity, demeaning, untrustworthy, awful, egotistical, deluded, dumb, greedy, boorish, barbaric, tasteless, foolishness, cruel, amoral, pretentious, hypocrite, revolting, reckless, manipulative, nonsensical, idiots, sexist, intolerable, uncivilized, egregious, duplicitous, undignified, atrocious, corrupt, contemptuous, vile, liar, vindictive, careless, wretched, hateful, bigoted, bastards, undemocratic, feckless, cowardice, pompous, mindless, spineless, arrogance, rude, treasonous, vulgar, disgusted, insufferable, sleazy, fool, perverse, scurrilous, insolent, grotesque, devious, incompetence, buffoon, unintelligent, undisciplined, odious, insane, liars, depraved, crazy, moronic, uncouth, petulant, ugly, elitist, frivolous, hypocrisy, dastardly, incapable, tyrannical, fools, heinous, thuggish, impulsive, detestable, demented, traitor, hypocrites, filthy, conceited, traitors, loathsome, barbarous, irrational, savages, gullible, obscene, repulsive, delusional, insult, scoundrel, abominable, deplore, mockery, perversion, bumbling, dopey, inane, nauseating, brainless, incorrigible, exploitive, gutless, dishonesty, unqualified, conniving, promiscuous, degrading, racist, ruthless, perverted, diabolical, betrayal

Table 2.2: The 178 words in our divisive word lexicon, which are verified as being divisive by the majority of 4 annotators. Our original ten seed words are denoted with a *.

Annotator agreement Following Mohammad and Turney [2010], in order to analyze how often the annotators agree with each other, we additionally calculate the percentage of times

the majority class has size 4 (all annotators agree), size 3 (all but one agree), and size 2 (even split). Table 2.3 shows these agreement values: 58.9% of the 360 initial terms had full annotator agreement, and 86.7% of these 360 have at least 3/4 annotators agreeing on the label. When limiting to the 178 terms that at least 3 out of 4 annotators agree are divisive, 69.1% have full agreement on the label (and 100% have majority agreement, by definition). So, the final terms include primarily instances that have high agreement scores.

Table 2.3: Percentage of majority class annotator agreement on binary labels (divisive? “yes” or “no”). 58.9% of the 360 initial terms had full annotator agreement, and 86.7% of these 360 have at least 3/4 annotators agreeing on the label. When limiting to the 178 terms that at least 3/4 annotators agree are divisive, 69.1% have full agreement on the label.

Terms	Majority class size (%)		
	four	three	two
360 initial	58.9	27.8	13.3
178 final	69.1	30.9	0.0

Since there are four annotators, we calculate Fleiss’ κ for interannotator agreement, as opposed to Cohen’s κ which is designed for two raters. For the initial 360 terms, we obtain a Fleiss’ κ of 0.54, which indicates a moderate level of agreement. Discussion of this score and why it may underestimate true agreement is included in §A.2.5.

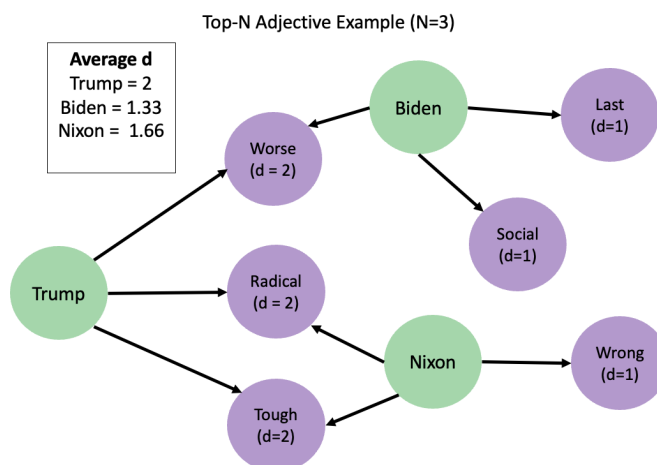


Figure 2.2: Conceptual example of how the FW overlap metric is calculated for $N = 3$.

2.3.3 *References to political opponents*

Our analysis further expands upon prior work by culling the subset of sentences that explicitly refer to political opponents. We specifically examine presidential debates, in which we define “opponents” here to be either the debate partner or their party. The methodology of tagging opponent mentions is described in §2.2.2.

Once speech referring to opponents is distinguished, we employ the Fightin’ Words (FW) method [Monroe et al., 2009] to identify words more strongly associated with opponent mentions; we extend this comparison across multiple candidates by calculating the overlap of each entity’s corresponding word sets with those of all presidents and presidential candidates. Intuitively, a greater overlap indicates that other candidates use similar rhetoric in opponent mentions, while a smaller overlap indicates that other candidates do not use similar rhetoric in opponent mentions. This metric thus provides a novel measure for quantifying a candidate’s distinctiveness with respect to their portrayal of opponents by combining lexical and graph analysis. The results are readily interpretable and clarify the distinctive qualities of language used to characterize one’s opponent. The metric is similarly constrained by the pool of candidates and the finite selection of descriptors.

Fightin’ Words overlap metric Monroe et al. [2009] introduce a methodology for lexical feature selection that calculates odds ratios between word probabilities between two related corpora. We use this method with the informative Dirichlet prior to obtain the Fightin’ Words (FW) in a data type for each candidate.

For each candidate $s \in C$, we specifically get the set of FW between the words associated with their opponent mentions ($FW_Y(s)$) versus the set of those that are not ($FW_N(s)$). We then examine the top- n of these sets respectively, $FW_Y^n(s)$ and $FW_N^n(s)$. For Y opponent mentions, we create a graph representation as follows (see Figure 2.2):

- speaker nodes S , where each node corresponds to a candidate

- word nodes W , where W corresponds to the union of each candidate’s top- n ($FW_Y^n(s)$), i.e., $W = \bigcup_{s \in C} FW_Y^n(s)$
- edges $E(S, W)$, where edge e is added between $s \in S$ and $w \in W$ if w is in $FW_Y^n(s)$

The top- n FW overlap metric ($OM_Y^n(s)$) is then calculated as the

$$OM_Y^n(s) = \frac{1}{n} \sum_{\forall w \in FW_Y^n(s)} \deg(w),$$

where $\deg(w)$ corresponds to the degree of the word node ($\#$ edges entering w). Likewise, for the set of words not associated with opponent mentions, we have

$$OM_N^n(s) = \frac{1}{n} \sum_{\forall w \in FW_N^n(s)} \deg(w).$$

Equivalently, the overlap metric for s can be thought of as the average number of speaker sets $FW_Y^n(c_i)$, $\forall c_i \in C$ that the top- n words of s appear in. A low $OM_Y^n(s)$ indicates that speaker s uses more distinct language to refer to opponents, while a higher score corresponds to opponent-referring language that is similar to that of other candidates.

2.4 Results

2.4.1 *Donald Trump is unique among all presidential candidates in all types of speech*

LLM-based uniqueness. We start by presenting results based on the LLM-based uniqueness metric. We find that Trump is the most distinctive speaker in debates and SOTU speeches (see Figs. 2.3 and 2.4). Additionally, among campaign speeches analyzed from 2008 onwards, Trump speaks in ways that stand apart from all other candidates. By comparing candidate speech patterns aggregated by party, Figure 2.4 shows that Trump is more

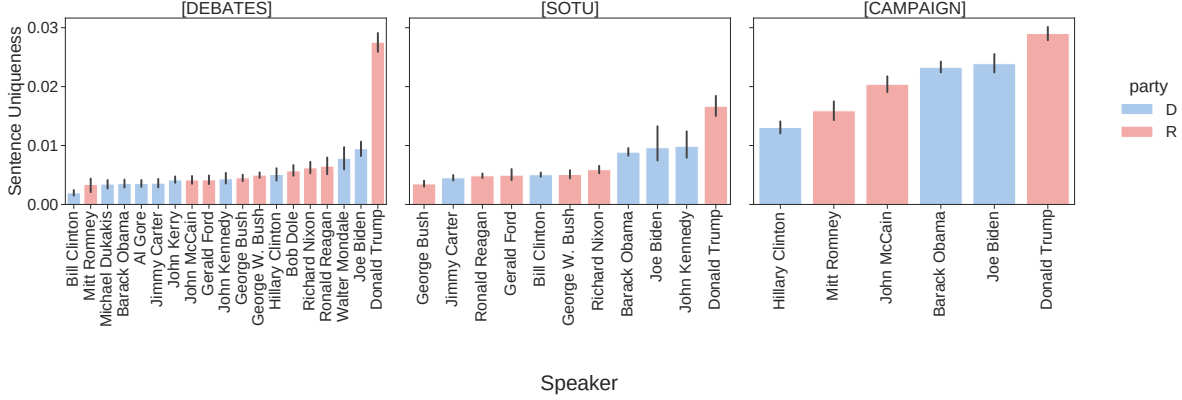


Figure 2.3: Average sentence uniqueness for each speaker, across all data types. Higher bars indicate greater uniqueness, i.e., that speaker’s speech is less likely to be uttered by other candidates. Trump is the most distinctive speaker among these candidates for debates and SOTU speeches. Error bars correspond to the 95% CI.

distinctive than his fellow Republicans for all types of speech. Indeed, the observed difference between Democratic and Republican candidates is minor compared to the gap observed between Trump and everyone else.

While Trump’s speech is characterized by shorter sentences (Figure 2.1), we confirm that Trump’s uniqueness is consistent across sentences of all lengths in Figure 2.5. We also note that Biden has similarly short sentences on average as Trump, but his uniqueness scores are close in magnitude to those of the other candidates, particularly in debates and SOTU addresses.

Furthermore, we find that there is minimal correlation between our uniqueness metric and standard simplicity scores, suggesting that the language model is not conflating uniqueness with language complexity (see §A.1.2).

Further extensions and robustness checks. When aggregating over presidential terms and years, Trump appears as slightly more distinctive in 2016 than in 2020 for debates, while his SOTU and campaign speeches increase in uniqueness over the years. Overall, he remains consistently more distinctive in both election cycles than the other candidates under consideration (see §A.1.3). Prior to Trump’s first election cycle, there are no clear temporal

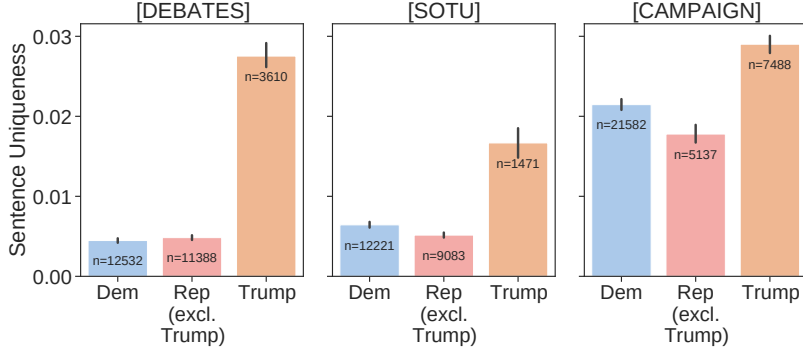


Figure 2.4: Trump’s average sentence uniqueness scores compared to the others aggregated by party. The number of sentences is denoted on each bar (with error bars representing 95% CI). Trump is significantly distinct from other Republicans. The uniqueness scores of the remaining Republicans are closer to those of Democrats than with Trump. For SOTU and campaign speeches, Republicans (without Trump) are less unique than Democrats on average.

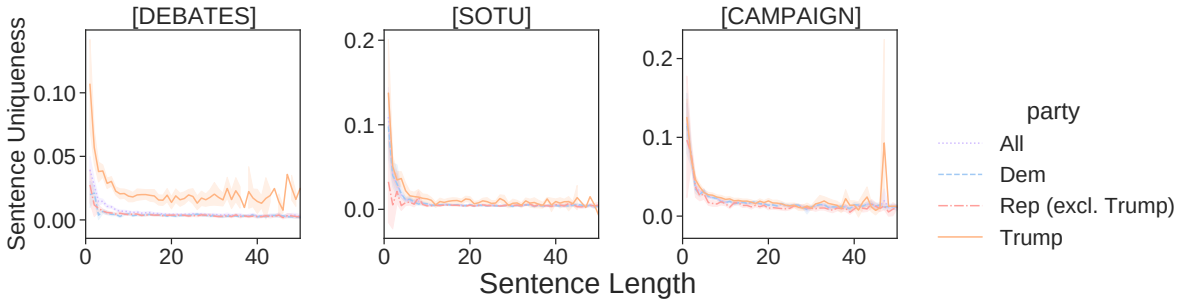


Figure 2.5: Sentence uniqueness across different sentence lengths, for Trump, all other Republicans, and Democrats. For debates and SOTU, Trump is consistently more distinctive across all sentence lengths.

trends to suggest that uniqueness has been increasing over time. Moreover, when comparing the uniqueness scores of the top decile of unique sentences for each speaker, we again find that Trump is the most distinctive speaker in all our samples of political speech (see §A.1.4).

Score validation. The results presented in the main paper are based on the OpenAI GPT-2 model, which was the state-of-the-art open-source causal language model at the start of this project [Radford et al., 2019]. We are interested in causal language modeling, i.e., the predictability of text in a *forward sequential manner*. Causal language models are more

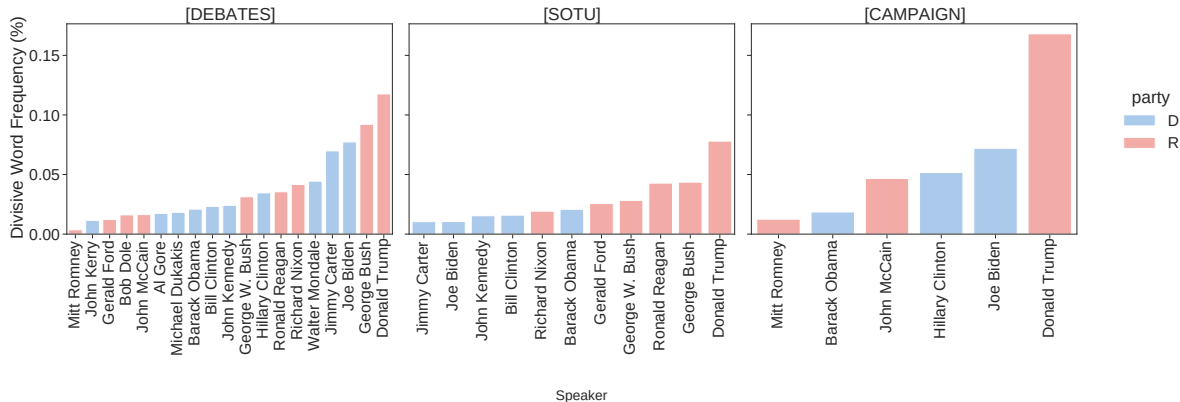
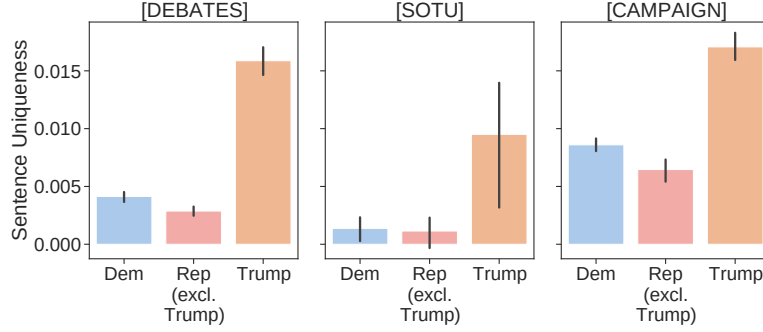
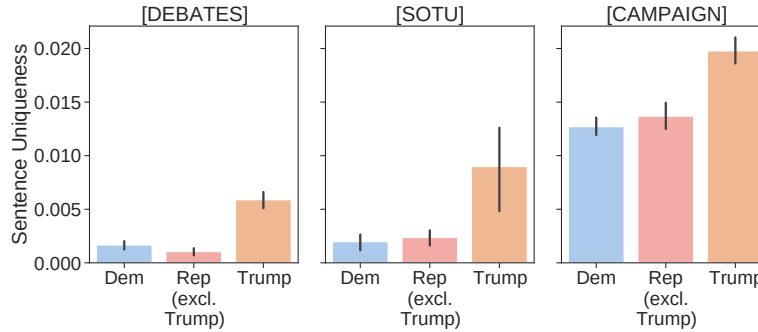


Figure 2.6: Overall percentage of words used that are divisive. Trump uses the most words from our divisive lexicon, in all types of speech.

appropriate for our task of scoring uniqueness, as opposed to masked language models (e.g., BERT, RoBERTa) which are designed to predict missing tokens within a sequence based on *bidirectional* context (and thus excel at text classification). We validate our uniqueness metric implementation with other LLMs. Specifically, we recalculate scores with the Gemma-2B and Phi-1.5 (1.3B) base models [Team et al., 2024, Li et al., 2023]. These models are considerably larger than the 124M parameters of GPT-2 [Radford et al., 2019]. Like our original setup up for GPT-2, we further train each model on all available corresponding data with a learning rate of $5e-5$. Each model is trained with LoRA tuning for 15 epochs, using LLaMA-Factory [Zheng et al., 2024]. As shown in Figure 2.7, while the exact scores differ, the trends are consistent with the main results: that is, Trump’s speech is the most unique among the selection of Democratic and Republican presidential candidates.



(a) Gemma 2B



(b) Phi-1.5

Figure 2.7: Validation of the LLM-based uniqueness metric with alternative LLMs. The resulting trends are consistent with original findings that Trump’s speech is more unique than other Democratic and Republican presidential candidates.

2.4.2 Trump speaks most divisively

Divisive word lexicon. Leveraging our divisive word lexicon, we measure its usage across the different types of presidential speech. The frequency of lexicon word appearances is calculated for each candidate across each dataset and is shown in Figure 2.6. Word frequencies are calculated as the number of times a divisive word from the lexicon is spoken by a speaker in a given dataset (debates, SOTU, campaign speeches) divided by the total number of words spoken by the speaker in that in dataset. Both the lexicon words and the speech data are processed to remove contractions and punctuation. Frequency trends provide a macroscopic look at the usage of divisive language over time. In §A.2, temporal plots and heatmaps of

Table 2.4: Top 10 divisive words used by Trump in each type of speech. Raw counts of usage are denoted in parentheses next to each word.

DEBATES	SOTU	CAMPAIGN
stupid (14),	cruel (3),	crazy (135),
racist (14),	vile (3),	corrupt (111),
disgrace (12),	ruthless (2),	stupid (69),
corrupt (8),	foolish (2),	dishonest (53),
disgraceful (8),	corrupt (2),	disgrace (45),
ridiculous (6),	reckless (2),	ridiculous (31),
ashamed (6),	savages (1),	racist (27),
stupid (6),	ugly (1),	incompetent (25),
filthy (6),	outrageous (1),	stupidity (22),
dishonest (6)	ridiculous (1)	ashamed (22)

divisive word usage provide a more granular look and do not show strong trends over time.

One common trend shown by the frequency plots for each dataset is Donald Trump’s relatively high usage of words from the divisiveness lexicon compared to other speakers. In all three datasets, Donald Trump’s speeches rank highest in divisive word usage compared to the other candidates analyzed. Examples of Trump’s most frequently used words from this lexicon usage are shown in Table 2.4. He is most verbosely divisive in debates and campaign speeches, uttering terms like “crazy”, “corrupt”, and “stupid” with high frequency. The contexts of debates and campaigns are more combative than SOTU addresses, which is reflected in the higher levels of divisive language in those two mediums. Table 2.5 contains a selection of Trump’s divisiveness in context; additional example sentences from various speakers can be found in the §A.2.

Across all debates and campaign speeches, which are the two corpora with greatest prevalence of divisive language, sentences that use divisive words are more unique (see §2.4.4). The Spearman correlation coefficient between uniqueness and divisive word usage is 0.01 in debates and campaigns ($p < 0.05$ in campaigns and $p = 0.30$ in debates). The weak, positive correlation suggests that divisiveness amounts to only a small part of the overall uniqueness of any speech pattern.

Table 2.5: Example sentences from debates and campaigns, with preceding context, spoken by Trump that use words from the divisive lexicon. Bolded words are matched with our divisiveness lexicon.

DEBATES:

Trump: Let me tell you something.

Trump: You take a look at Mosul.

Trump: The biggest problem I have with the **stupidity** of our foreign policy, we have Mosul.

Trump: I was at a little Haiti the other day in Florida.

Trump: And I want to tell you, they hate the Clintons, because what’s happened in Haiti with the Clinton Foundation is a **disgrace**.

CAMPAIGN:

Trump: We’re going to bring back the miners and the factory workers and the steel workers.

Trump: We’re going to put them back to work.

Trump: The economic policies of Bill and Hillary Clinton have destroyed manufacturing in your state and throughout the entire country.

Trump: The **corrupt** Clintons gave us NAFTA.

Trump: The fact is, this is the single most important election in the history of our country.

Trump: And sleepy Joe Biden’s made a corrupt bargain.

Trump: You saw the bargain he made, in exchange for his party’s nomination, which he shouldn’t have gotten because if Pocahontas got out one day early, I’d be running against **Crazy** Bernie, which would have been okay, too.

2.4.3 References to political opponents in speech

In addition to examining overall speech patterns, we can also extract the portions that reference political opponents. Next, we show that Trump is more likely to mention his political opponents than are other presidential candidates and that when doing so, Trump uses particularly distinctive language.

Rates of opponent mentions. As one might expect, candidates routinely mention their opponents in debates and only rarely in SOTU addresses. Overall, rates of sentences that mention opponents for debates, SOTU, and campaign speeches are 20.60%, 0.83%, and

6.95%, respectively (see §A.3). Trump has the highest rate of opponent mentions in debates.

Since the opponent mentions are more common in presidential debates, we revisit our LLM-based measure for the debates to confirm that Trump is not unique simply because he is more likely to mention opponents. Figure 2.8 shows that the LLM-based uniqueness score rankings are relatively consistent across opponent and nonopponent mentions. Moreover, Trump’s language is consistently unique in these debates, whether or not he calls out an opponent. Concurrently, sentences that refer to an opponent tend to be more distinct than those that do not (see §2.4.4). And in debates, we additionally find that sentences that mention opponents contain significantly higher frequencies of divisive words. If you are looking to isolate the distinguishing characteristics of a candidate’s speech patterns, therefore, you would do well to focus on how they talk about their political opponents.

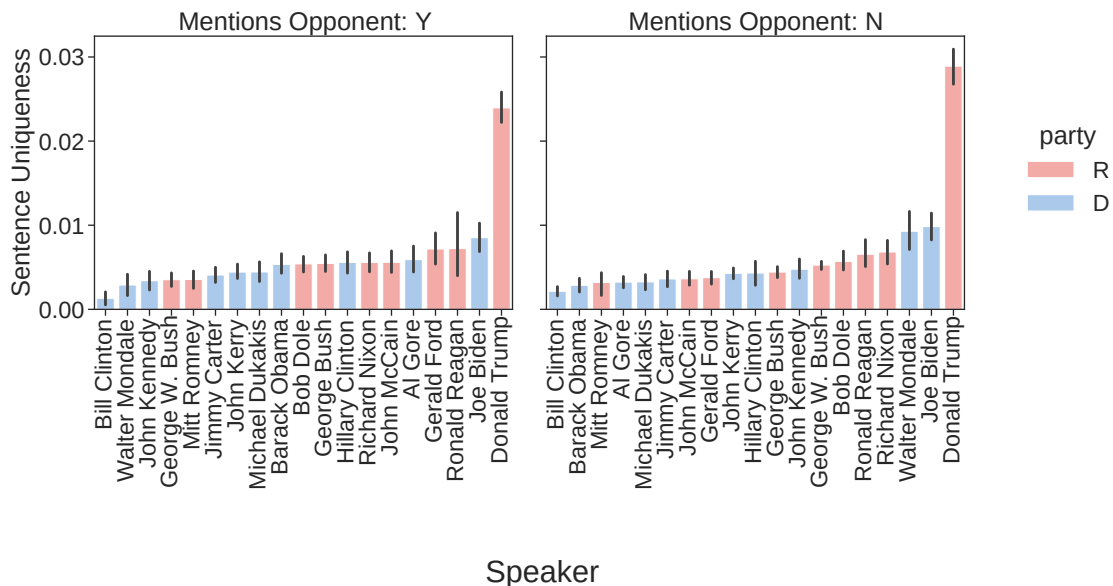


Figure 2.8: Uniqueness of speech broken down by opponent mentions, for debates (the error bars represent 95% CI). Trump is most distinctive regardless of whether he references an opponent or not.

Fightin’ Words adjective overlap. To identify which *adjectives* are most strongly associated with describing opponents for each candidate, we calculate the odds ratios between

opponent mention sentences and nonopponent mention sentences, i.e., Fightin’ Words (FW) [Monroe et al., 2009]. After these odds ratios are calculated, we compare the words most commonly associated with opponent mentions across candidates and calculate an overlap score. This *FW overlap metric* for each candidate is calculated as the average number of speakers who share each of the candidate’s top- n FW adjectives.

Intuitively, the lower the FW overlap score for a candidate, the fewer adjectives that candidate uses in common with others when describing their political opponents. For example, a low FW overlap metric indicates that the words a politician uses most frequently to describe an opponent are distinct from those of other candidates. From Figure A.12a, we see that Trump has the lowest FW overlap in debates on average across different top- n thresholds. Among adjectives that Trump uses in references to opponents, his top- n FW are consistently distinct for $n = 5$ to $n = 25$.⁶ Figure A.12c ranks the speakers by their FW overlap score at the top-15 threshold and shows that Trump has the lowest FW overlap scores when mentioning opponents (additional top- n plots can be found in the §A.3.2). Trump has a higher overlap of FW descriptors when not referencing opponents, indicating that he is more distinctive for how he *describes* his opponents.

Examples of Trump’s top-25 FW adjectives can be seen in Table 2.6; the left panel includes language used to describe his political opposition, while the right panel includes references to his own party and allies. In addition to attacking opponents (e.g., “disgraceful”), Trump tends to use fairly simplistic adjectives like “massive” and “super”.⁷

6. There are considerably more sentences of nonopponent mentions, but only looking at top- n FW (across different thresholds) helps control for that difference.

7. Some of these words, like “radical” and “liberal,” appear in the “polarization” dictionary of [Simchon et al., 2022]. Taken as a whole though, our divisive word lexicon contains very little overlap with that dictionary.

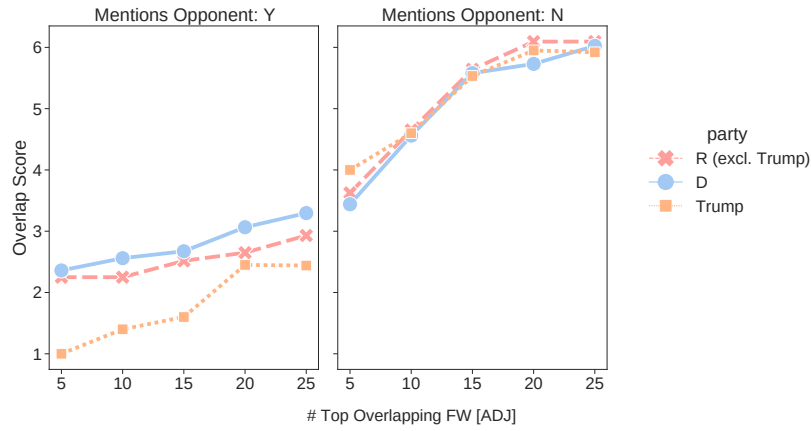


Figure 2.9: FW overlap scores across different top- n thresholds, comparing Trump and the other candidates aggregated by party.

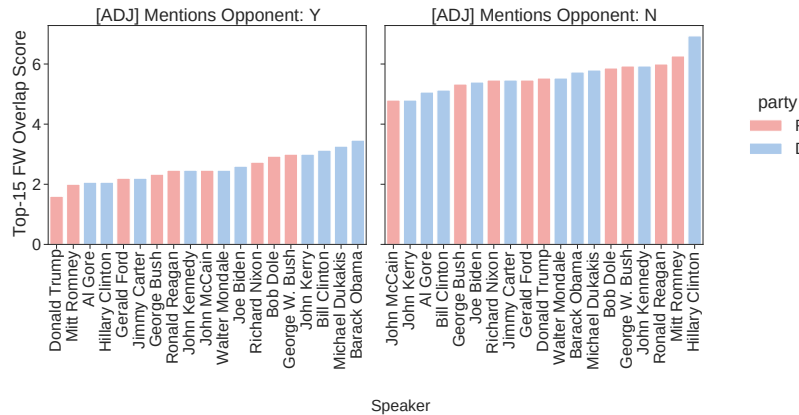


Figure 2.10: Top-15 FW overlap score in debates for each candidate. Trump's FW associated with opponent mentions generally have the lowest overlap in adjective usage compared to other candidates, which is another indication that his language is distinctive.

Table 2.6: Top-25 FW adjectives said by Donald Trump in debates, either in reference to opponents or not. Z-scores are provided in parentheses.

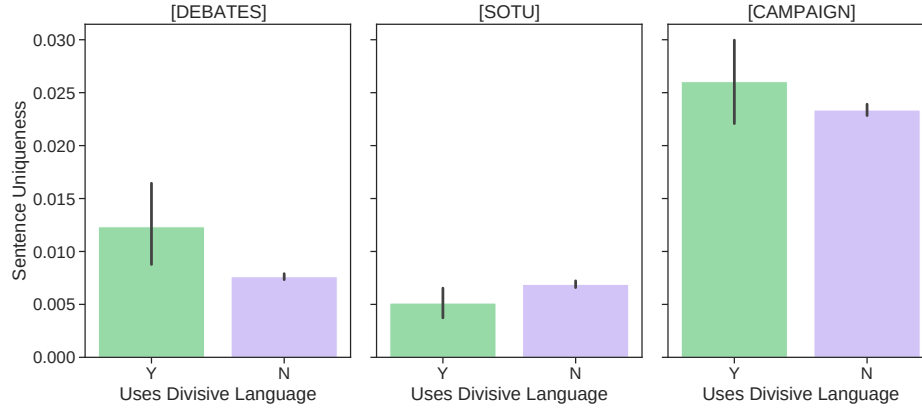
Mentions Opponents: Y	Mentions Opponents: N
left (2.07), long (1.85), tough (1.80), own (1.79), ok (1.79), bigger (1.79), worse (1.66), radical (1.57), super (1.50), real (1.50), effective (1.47), xenophobic (1.46), liberal (1.46), disgraceful (1.46), massive (1.33), single (1.33), political (1.09), last (1.09), economic (1.08), short (1.08), huge (1.04), various (1.04), red (1.04), upset (1.04), back (1.04)	good (-3.63), great (-3.25), important (-2.36), more (-2.08), inner (-2.00), strong (-1.99), right (-1.84), proud (-1.80), other (-1.79), expensive (-1.64), old (-1.53), big (-1.53), better (-1.50), greatest (-1.44), young (-1.41), sad (-1.41), able (-1.31), much (-1.31), african (-1.27), fine (-1.27), beautiful (-1.27), tougher (-1.18), nice (-1.18), least (-1.12), sure (-1.12)

2.4.4 Relating uniqueness, divisiveness, and opponent mentions

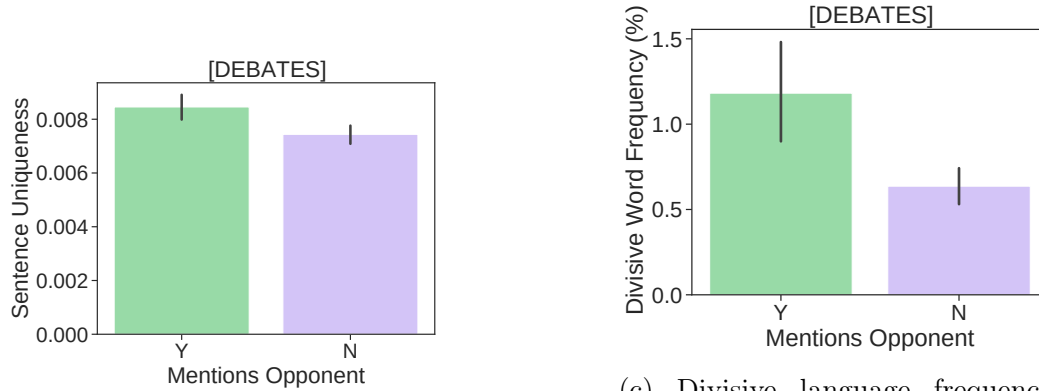
Figure 2.11a shows that in debates and campaign speeches, sentences that use divisive language tend to be more unique as well. The Spearman correlation coefficient between divisiveness and uniqueness is 0.01 for debates and campaigns ($p < 0.05$ in campaigns and $p = 0.30$ in debates). For sentences containing opponent mentions in debates, Figure 2.11b shows that such utterances tend to be more distinctive and Figure 2.11c shows that they tend to have higher divisive word usage as well. The Spearman correlation for opponent mentions is 0.05 and 0.02 ($p < 0.05$) between uniqueness and divisiveness, respectively.

2.5 Discussion

In this work, we proposed novel approaches to quantify the uniqueness and divisiveness of presidential discourse. Our results show that whether on the campaign trail, the debate stage, or the official dais of the House of Representatives, Trump speaks differently from all modern presidents and presidential candidates. We confirm his distinctiveness for speaking in shorter, simpler, and more repetitive sentences. We further quantify the uniqueness of his overall speech compared to that of other presidential candidates. Finally, we demonstrate that Trump uses language that is more divisive, antagonistic, and explicitly focused on his



(a) Sentence uniqueness across divisive language usage, for each type of speech.



(b) Sentence uniqueness across opponent mentions, in debates.

(c) Divisive language frequency across opponent mentions, in debates

Figure 2.11: Pairwise analysis of our three proposed metrics, aggregated globally across all speakers. Figure 2.11a shows that in debates and campaign speeches, sentences that use divisive language tend to be more unique as well. For sentences containing opponent mentions in debates, Figure 2.11b shows that such utterances tend to be more distinctive and Figure 2.11c shows that they tend to have higher divisive word usage as well.

political opponents.

The differences between Trump and other candidates do not appear to be an artifact of secular communication trends (see §A.1.3), whether in the general coarsening of public discourse or the tendency to speak in simpler sentences. Across multiple points of comparison, observed differences between Trump and other contemporary candidates appear every bit as large as those between Trump and candidates from the 1960s and 1970s.

Our findings, of course, come with a variety of limitations. The campaign data, for instance, include samples from only the most recent candidates and exclude certain communication formats, such as interviews. None of our datasets cover instances when surrogates (family members, vice presidents, etc.) speak on behalf of a candidate. While other works have examined Trump’s tweets [Lewandowsky et al., 2020], we do not assess any of his social media postings for lack of comparable data for other candidates, especially older ones. Furthermore, our data collection ends at 2022, excluding more recent remarks from Trump that have been flagged in the media for their escalated divisiveness [Kurtzleben, 2023].

Our research invites a variety of extensions. Given that values of the LLM-based uniqueness metric can only be understood relative to the elements of a selected sample, future work would do well to compare Trump to other speakers, such as populist leaders in other countries. Just as we investigate divisiveness as a new dimension of political speech with particular resonance for Trump’s rise to power, other scholars might evaluate other dimensions of political speech, such as its analytical sophistication or propensity to endorse racial, gender, or sexual stereotypes. Additionally, future research might explore the possibility of contagion effects from Trump’s speech patterns to other political figures, as politicians periodically emulate and influence one another.

We also recognize that all of our metrics come with tradeoffs, which we have elucidated throughout this essay. Still, the tools we develop can be applied to a wide variety of settings. While we use them to compare speech patterns among presidents and presidential candidates,

they can just as easily be deployed to analyze the language used by any public officials. And in addition to characterizing overall differences in speech patterns, we illuminate ways of assessing particular qualities that appear in either stand-alone speeches or interactive exchanges.

The substantive findings presented herein, moreover, establish the uniqueness of Trump’s speech patterns, just as they reveal particular qualities that distinguish Trump from all modern US presidential candidates. More research, of course, is required to map these speech patterns into larger political strategy. Nonetheless, we conjecture that these qualities broadly contribute to Trump’s enduring appeal as a populist who unabashedly denounces established political enemies in a historical period of acute polarization, distrust, and division.

CHAPTER 3

FROM FEEDBACK TO CHECKLISTS: GROUNDED EVALUATION OF AI-GENERATED CLINICAL NOTES

3.1 Introduction

Evaluating the quality of AI-generated text is challenging, especially in domains where the text is highly specialized and requires expert knowledge. This is particularly true in the medical domain, specifically for AI-generated clinical notes. Automated metrics are scalable but often misaligned with clinical needs, relying on limited reference notes and penalizing harmless stylistic differences. Reference-free methods tend to focus narrowly on factuality [Xie et al., 2024]. Meanwhile, human evaluation by clinical experts is generally high-quality but costly, inconsistent, and difficult to scale due to subjective preferences and varying documentation standards across specialties [Wang et al., 2025a, Hanson et al., 2012]. The diversity of clinical practice makes defining a single set of evaluation criteria (e.g., a rubric) that applies across all notes difficult [Yim et al., 2023, Wang et al., 2024, Croxford et al., 2025a].

Checklists are commonly employed in aviation and healthcare, used to improve safety and quality [Reijers et al., 2017, Chaparro et al., 2019, Brykczynski, 1999, Degani and Wiener, 1993, Haynes et al., 2009] and to guide human- or LLM-evaluators in complex tasks. So, to address the challenge of evaluating AI-generated clinical notes, we propose leveraging real user feedback to generate grounded checklists.

For each encounter, clinicians may provide free-form written feedback for the corresponding note. Such user feedback reflects real issues clinicians face with AI-generated notes, providing valuable insights into the qualities that make a clinical note effective or lacking. By analyzing this feedback, we can automatically uncover the attributes associated with highly-rated notes. These qualities can be compiled into structured checklists to effectively evaluate

clinical notes. The checklist can be used with LLMs-as-a-Judge [Zheng et al., 2023], to automatically flag notes that are lacking, at scale for checklists that are too long for humans to evaluate, or to support human evaluation by providing a rubric for the human evaluator to follow. Thus, we tackle the following research question: can user feedback be systematically distilled into a note quality checklist?

Checklist Score: 7 / 10

- ☒ Does the section include a clear and concise diagnosis or differential diagnosis for each problem?
- ☒ Are the rationale and evidence for each diagnosis clearly documented?
- ☒ Does the section provide a detailed plan for addressing each identified problem, including specific treatments and interventions?
- ☐ Are follow-up plans and timelines specified for each problem?
- ☒ Does the section mention any need for additional diagnostic testing or referrals?
- ☒ Is patient education or counseling regarding the diagnosis and treatment plan documented?
- ☒ Is there documentation of the patient's or caregiver's understanding and agreement with the plan?
- ☐ Does the section include considerations for any comorbid conditions that may affect the management of the patient's problems?
- ☒ Is there documentation of any medication changes, with reasons provided for each change?

Figure 3.1: Example checklist questions for the Assessment and Plan section of a clinical note. The checklist score consists of the proportion of satisfied questions.

To achieve these goals, we propose a method to automatically generate a checklist, given user feedback. The pipeline involves leveraging user feedback as context, prompting an LLM for candidate checklist questions, and then applying a series of refinement steps to select for the most salient questions. Finally, we conduct automatic evaluation of the final checklists,

with plans for human validation and checking for generalization¹. An example of how a checklist would be used for scoring a note can be seen in Figure 3.1.

Our contributions are as follows:

- We propose a systematic pipeline for generating and refining checklists from real user feedback, with clear evaluation metrics.
- In our internal offline evaluation, the checklist appears more comprehensive, diverse, and predictive of human ratings than a baseline (zero-shot) checklist.
- We demonstrate the checklist’s robustness to various quality-degrading perturbations (e.g., missing information, poor writing flow and organization, and redundancy and hallucinations) as well as significant alignment with expert preferences.

3.2 Related Work

Clinical note evaluation Guideline or rubric-based evaluation of clinical notes has been explored in several works [Yim et al., 2019]. Stetson et al. [2012] is a 9-item rubric for medical documentation quality, a shorter version of the Physician Documentation Quality Instrument (PDQI) [Stetson et al., 2008]. It is scored on a 5-point Likert scale and has been used to evaluate AI-generated clinical notes [Amenyo et al., 2025, Croxford et al., 2025b]. Eng et al. [2024] creates a high-level guideline for medical documentation best practices. Burke et al. [2014] is another physician-validated medical note rubric. These rubrics often have Likert scales as answer options, which introduces additional complexity and ambiguity for human and LLM raters. Rubrics are also difficult to develop and score, due to subjectivity and variability of practice [Croxford et al., 2025a, Oleson, 2024]. While these existing rubrics are static, we introduce an approach that is comprehensive of large-scale user concerns and dynamic to new feedback.

1. Our checklists are not intended for clinical deployment or for altering patient care without clinician review.

Other note evaluation works include training reward models [Wang et al., 2025b], or proposing reference-free evaluation on pre-defined desired attributes (typically factuality-based) [Kanithi et al., 2024, Xie et al., 2024]. We are the first to propose a checklist-based evaluation approach grounded in real human feedback.

Checklist generation With the utility of checklists being proven across high-stakes domains [Reijers et al., 2017, Chaparro et al., 2019, Brykczynski, 1999, Degani and Wiener, 1993, Haynes et al., 2009], there is a growing body of work on generating checklists automatically with LLMs [Lee et al., 2025, Cook et al., 2024, Pereira et al., 2024, Wei et al., 2025, Li et al., 2025]. These works focus on generating checklists specific to individual instructions or data samples (e.g., a summary or user conversation), rather than generalizable checklists that can be applied across a corpus of data.

For other clinical applications, Nan et al. [2017] propose a rule-based method for generating dynamic clinical safety checklists and Zhang et al. [2021] learns optimal checklists for clinical decision support from a set of predefined binarized features. Checklists have also been manually developed by experts, for individual medical notes [Savkov et al., 2022] and generative medical tasks in Chinese [Zhang et al., 2025]. Our work aims to automatically generate checklists for clinical notes that apply across encounters, checking for items beyond facts.

Checklist evaluation Wang et al. [2023] propose a method of generating natural language cluster labels based on a goal; we similarly want to generate checklist questions that cover a corpus of feedback, based on the goal of evaluating note quality. Our feedback coverage methodology is loosely inspired by their Propose-Assign-Select framework. Guo et al. [2024] define a testbed of global perturbations to assess metrics for plain language summarization; we draw from these criteria to form our own perturbation benchmark.

Checklist-based evaluation with LLM-as-a-Judge LLMs are increasingly used as automatic evaluators, due to their ability to handle nuance and scale. In particular, LLM-evaluators (or LLMs-as-a-Judge) are used to evaluate the quality of another LLM’s generations. However, LLM-evaluators have been shown to contain biases like increased position bias with more answer candidates [Ye et al., 2024], leading us to choose a binary checklist-based evaluation approach.

Works like Goldberg et al. [2024] use LLMs to evaluate compliance with a pre-existing checklist, and many others have explored checklist-based evaluation with LLM-as-a-Judge for a variety of tasks [Lin et al., 2024, Que et al., 2024, Chu et al., 2025]. It has been extensively demonstrated that checklists can be used to improve the reliability of LLM-evaluators and agreement with human judgments [Lee et al., 2025, Pereira et al., 2024, Cook et al., 2024, Wei et al., 2025, Li et al., 2025].

3.3 Data

We leverage a mix of proprietary data (user feedback and note ratings) from a commercial Ambient AI company and open-source datasets (clinical notes).

Feedback	Rating
“Does not capture all the key clinical details discussed during the visit”	2
“Prefer past tense”	4
“Could benefit from clearer summary and clinical reasoning”	3
“Covers everything I would have wanted to see”	5

Table 3.1: Example synthetic user feedback and star ratings.

3.3.1 User Feedback

We retrieve approximately 22,000 encounters with user feedback of > 2 words, non-null (1-5) star ratings. Table 3.1 provides some synthetic examples. The user feedback is free-text,

and the star ratings are on a scale of 1-5, with 1 being the worst and 5 being the best; these are obtained from real users of an Ambient AI company that scribes clinical encounters and generates clinical notes. We specifically filter for feedback that is relevant to note quality and content, as opposed to other issues like recording or EHR integration. We use an internal tool to determine the relevant note section (e.g. ASSESSMENT AND PLAN) for each piece of feedback and only keep the relevant feedback for each section.

3.3.2 Human Preference Ratings

We also have access to 506 pairs of note sections with internal preference ratings, which are rated by clinical experts within the company. These pairs of notes have at least two raters agreeing on the preferred note in a pairwise comparison. Each sample of this dataset contains the transcript of the encounter and the pair of notes for the section being rated, with the preferred note being marked.

3.3.3 Reference Notes: ACI-bench

We use 207 reference notes and transcripts from ACI-bench [Yim et al., 2023]. Each note contains four sections, corresponding to the sections our user feedback is tagged with in parentheses:

- SUBJECTIVE (*HPI_Subjective*) includes items taken during verbal exam and typically written in the chief complaint, history of present illness, and past social history.
- OBJECTIVE EXAM (Physical Exam or *PE*) includes content from the physical examination on the day of the visit.
- OBJECTIVE RESULTS (*Results*) includes diagnostics taken prior to the visit, including laboratory or imaging results.

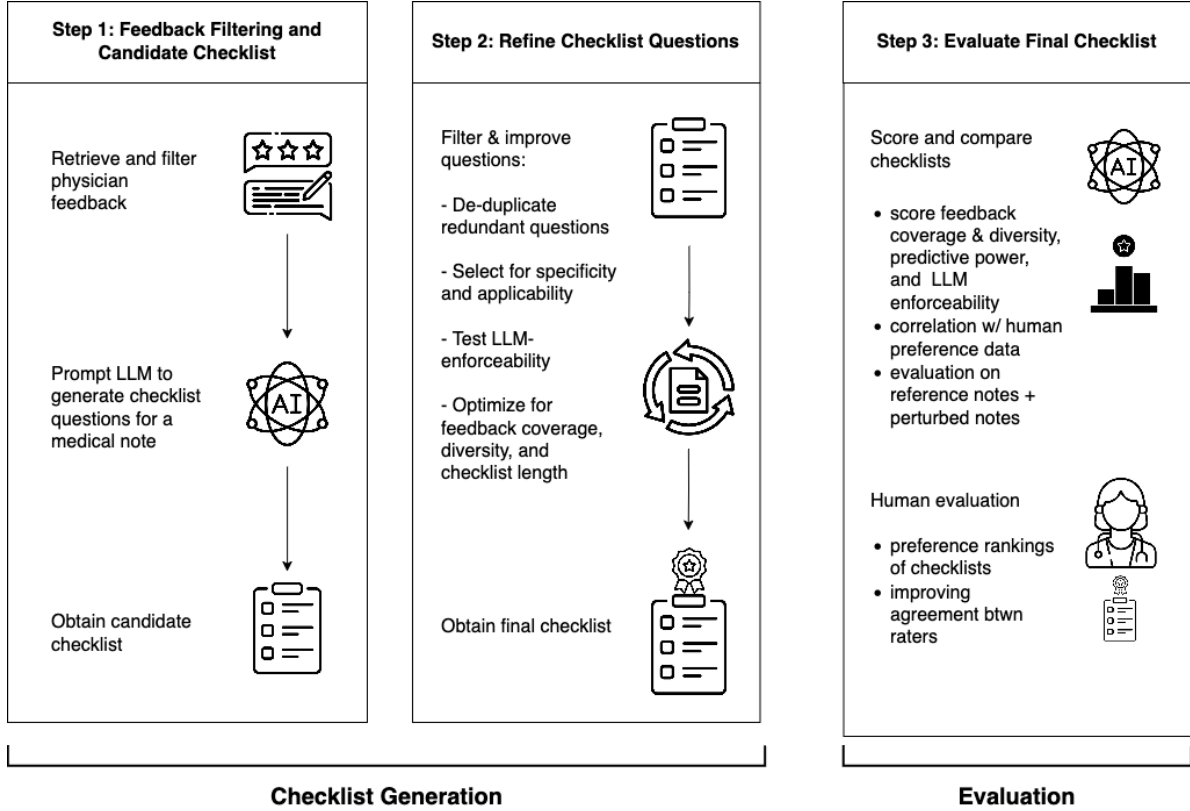


Figure 3.2: Proposed end-to-end pipeline.

- **ASSESSMENT AND PLAN (AP)** includes the doctor’s diagnosis and planned tests and treatments.

In cases where certain sections are missing, an EMPTY flag is used as the content. We utilize these reference notes to calculate pass rates/checklist scores, generate perturbed notes, and create unit tests.

3.4 Pipeline & Implementation

We start with some definitions:

- **Checklist question:** a question that can be answered with “Yes” or “No”; specifically, the question is designed to be answered with “Yes” if the note meets a desired criteria of good clinical note.

- **Checklist:** a series of checklist questions that can be used to evaluate a clinical note.
- **Passing (a question):** if a note scores “Yes” for the checklist question.
- **Checklist score:** for a note and a checklist, the proportion of the checklist questions that pass.

To streamline the evaluation process and make checklists more easily comparable, we make several simplifying assumptions:

- We focus on the Assessment and Plan (*AP*) section of the notes, since it is a key component of clinical notes and all encounters should have this section. However, our proposed methods can be extended to other note sections.
- Checklist questions must be answerable w/ “Yes” or “No” for all encounters, i.e., no “N/A” answers allowed.
- Questions should be answerable given only transcript and the specified note section.
- Checklists should be concise, to be easily processable by humans.

Figure 3.2 shows the proposed pipeline². We will go over generation (§3.4.1) and refinement (§3.4.2) of the checklist questions below. See §3.5 for details of the final checklists, and §3.6 for evaluation.

3.4.1 Checklist Generation Methods

We generate candidate checklist questions for each note section separately. Appendix §B.1 contains the prompts used for generating checklists.

2. This pipeline is illustrative, uses de-identified data, and remains subject to further validation.

Baseline: We prompt a LLM with the instruction to generate candidate checklist questions, without any feedback or specified guidelines (zero-shot). This baseline is designed to isolate the contribution of clinician feedback. We chose not to adapt prior rubrics such as Stetson et al. [2012] or Burke et al. [2014] since they are Likert-style and not directly comparable to binary checklists without careful redesign, and having a clinical expert draft checklist questions from scratch has proven non-trivial and time-consuming, due to the subjectivity and exhaustiveness of the task.

With Feedback: We prompt a LLM with the feedback corpus and the instruction to generate candidate checklist questions. Because there is more feedback than can fit in the model’s context window, we split the feedback into batches. Questions are confirmed to be written such that a “Yes” answer corresponds to a good clinical note; if not, the LLM rewrites them in the correct direction.

3.4.2 Checklist Refinement

Once we have candidate checklist questions, we conduct several steps to filter out undesirable questions, based on our criteria in §3.4. The refinement steps, which are detailed below, include:

1. De-duplicating redundant questions.
2. Tagging for generally applicable and section specific questions.
3. Dropping questions that are not LLM enforceable.
4. Selecting a final subset optimized for feedback coverage and diversity.

Redundant Questions We obtain embeddings of each question, using `text-embedding-3-large`, and then calculate the cosine similarity between each pair of questions. We build a graph

where each question is a node, and an edge exists between two questions if their cosine similarity is ≥ 0.85 (this threshold is chosen from manual inspection of clusters). Each connected component of the graph is a cluster of similar questions, while isolated nodes are unique questions. For each cluster, we prompt `gpt-4o` to generate a single question consolidating the similar questions.

Tagging for applicability and specificity We keep only questions that are applicable to general encounters, ensuring that they are answerable with a simple “Yes” or “No” for all encounters, without allowing for “N/A” responses. Additionally, we flag questions that reference other note sections, retaining only section-specific questions. This ensures that questions are answerable using only the transcript and the specified note section. For example, when tagging for applicability, we drop questions like “Is there a comprehensive assessment and plan for all chronic conditions discussed, ensuring nothing important is omitted?” For section-specificity, we flag questions like “Is there a clear separation between the history of present illness (HPI) and the action plan to ensure there is no overlap?” since they would require access to the *HPI* section to answer. To implement this, we prompt the `o3-mini` model to tag each question using a zero-shot, chain-of-thought (CoT) approach.

LLM Enforceability We want to ensure that the questions are answerable by LLM-evaluators. Thus, we define **enforceability unit tests**: mini-benchmarks of 10 reference notes per checklist question that pass the question criteria, where each reference note is then rewritten to fail the criteria. A unit test is “passed” if the rewritten note receives a score of “No” for the question. By doing this, we can select the checklist questions that are actually enforceable by LLMs.

We measure the enforceability rate for each checklist question, which is the proportion of rewritten notes that correctly fail the criteria, which is then averaged across all questions into a single **enforceability score** for the checklist. The 10 reference notes in each question’s

mini-benchmark are selected randomly among all the ones that pass the question. We prompt `gpt-4.1-mini` to rewrite the reference notes to fail; see §B.1.3 for the prompt used to rewrite the reference notes to fail the checklist questions. For example, if the checklist question is “Does the note use the correct pronouns?” and the reference note uses “she”, then the note could be rewritten to use “he” to fail the criteria. Post-processing is done to ensure only the rewritten note is produced, i.e., no explanations³. We discard questions with an enforceability score below a threshold of 0.7.

Optimal subset of questions for feedback coverage and diversity The final refinement step selects an optimal subset of checklist questions that maximizes coverage of user feedback while minimizing redundancy and checklist length. *Coverage* is defined as whether each feedback is supported by each question, and it is measured by the assigner LLM, with the prompt defined in §B.1.4. We obtain a “coverage matrix”, where (i, j) value is 1 if question i covers feedback j , and 0 otherwise. Total coverage rate C , is the sum of the coverage matrix divided by the number of feedback items.

To avoid redundancy, we also want feedback *diversity* among the checklist questions’ coverage sets. For questions i and j with covered sets of feedback F_i and F_j , we define the *similarity* S_{ij} of the feedback items covered by questions i and j as their Jaccard similarity, i.e., $S_{ij} = |F_i \cap F_j| / |F_i \cup F_j|$. Then for a checklist of n questions, we can define the diversity of question i as one minus its average similarity with all other questions:

$$D_i = 1 - \frac{1}{n-1} \sum_{j=1, j \neq i}^n S_{ij}$$

Total diversity D is the average of the diversity of all questions.

Since we also want to avoid overly length checklists, we also add a weight λ for penalizing

3. Manual verification of 50 pairs of notes spanning 27 different checklist questions has yielded a 90% validation rate (rewritten notes that correctly fail); the remaining 10% of rewritten notes are still of inferior quality (missing/inaccurate details), but do not necessarily fail the criteria of the question.

length. Finally, for a checklist of k questions, we can define a objective score to balance coverage and diversity of a checklist, given α as the weight:

$$Score(k) = \alpha \cdot C + (1 - \alpha) \cdot D - \lambda \cdot k$$

We can then use beam search to select the optimal subset of k questions.

Figure 3.3 shows the α values for the objective function, where α is the weight of the coverage term. We select a final α value of 0.5, since it provides balance between coverage and diversity.

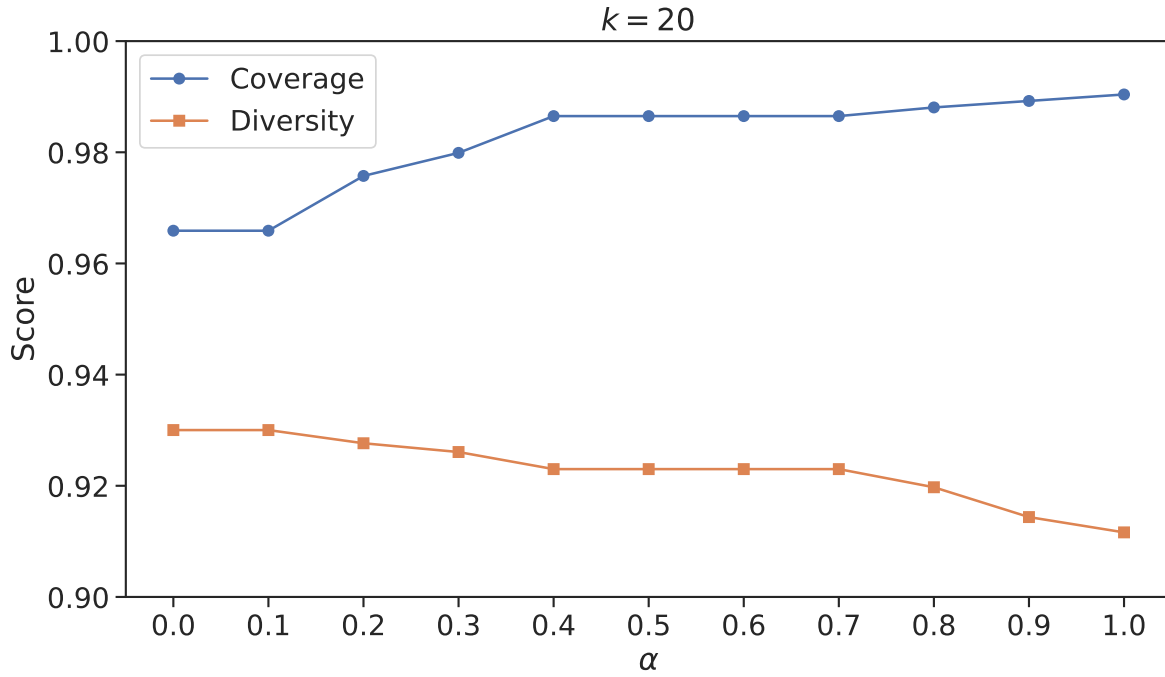


Figure 3.3: α values for the objective function, where α is the weight of the coverage term. The α value is set to 0.5 for the final checklists, since it provides balance between coverage and diversity.

Metric	Baseline ($n = 10$)	Feedback ($n = 20$)
Coverage $\uparrow$	0.978	0.987
Diversity $\uparrow$	0.897	0.923
Intra-cluster Dist $\downarrow$	0.670	0.667
Inter-cluster Dist $\uparrow$	0.061	0.080
LLM enforceability $\uparrow$	0.82	0.89

Table 3.2: Evaluation of the final checklists, with respect to feedback coverage, diversity, and LLM enforceability. Diversity is additionally evaluated by intra-cluster and inter-cluster distances. The feedback checklist beats the baseline checklist in each category.

3.5 Results

3.5.1 Generation and Refinement

The ~7700 feedback items for the *AP* section are randomly split into 8 batches of approximately 900-1000 items each. 97 candidate checklist questions are generated from this set of feedback.

Filtering out redundant questions and tag for applicability and specificity results in 50 questions. For example, the questions “Is the assessment and plan section well-organized and clearly separates different medical issues or diagnoses?” and “Is the assessment and plan organized in a way that is easy to read, with distinct separation between different patient problems?” are clustered and merged into a single question: “Is the assessment and plan section well-organized with clear separation of medical issues or diagnoses?”

These 50 questions are then evaluated for LLM enforceability. We find that 36 questions have a unit test threshold of at least 0.7, and we discard the remaining 14 questions. Such discarded questions include: “Is unnecessary and speculative language avoided in the assessment and plan section?” The average enforceability score for the remaining questions is 0.9.

Finally, we select a subset of k questions that maximize coverage and diversity. Setting

$\alpha = 0.5$ and $\lambda = 0.0005$, we find that $k = 20$ provides the optimal balance of coverage and diversity with length of the checklist (see Figure 3.4).

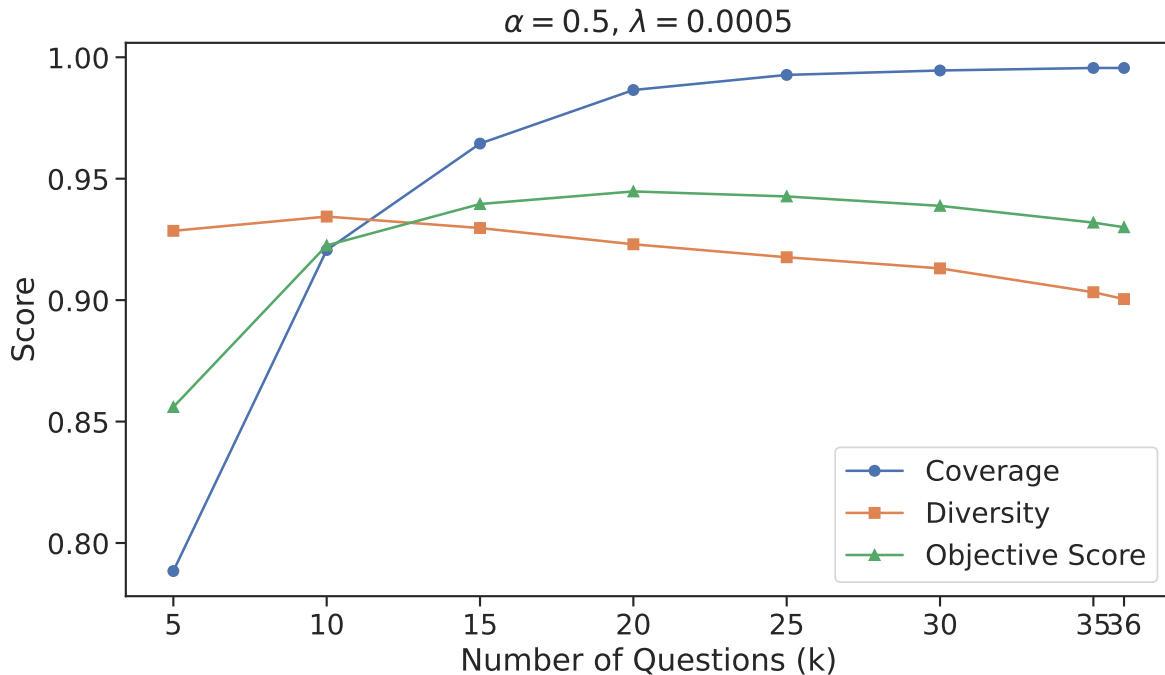


Figure 3.4: Objective score (coverage and diversity) vs. length of checklist. $k = 20$ is the optimal number of questions.

3.5.2 Checklist Questions

Following our refinement steps, we obtain a final checklist of 20 questions. Our checklist is compared against a baseline checklist of 10 questions. These checklists are intended for quality assessment, not reimbursement or coding optimization; additional audit measures are required to prevent misuse.

Baseline Checklist for *AP* ($n = 10$)

1. Does the section include a clear and concise diagnosis or differential diagnosis for each problem?
2. Are the rationale and evidence for each diagnosis clearly documented?

3. Does the section provide a detailed plan for addressing each identified problem, including specific treatments and interventions?
4. Are follow-up plans and timelines specified for each problem?
5. Does the section mention any need for additional diagnostic testing or referrals?
6. Is patient education or counseling regarding the diagnosis and treatment plan documented?
7. Is there documentation of the patient's or caregiver's understanding and agreement with the plan
8. Does the section include considerations for any comorbid conditions that may affect the management of the patient's problems?
9. Is there documentation of any medication changes, with reasons provided for each change?
10. Are potential risks and benefits of the treatment plans discussed?

Feedback Checklist for *AP* ($n = 20$)

1. Is the assessment and plan section organized in a clear and concise manner, such as using bullet points instead of paragraphs where appropriate?
2. Is the assessment and plan section comprehensive, covering all relevant patient issues discussed during the visit?
3. Are diagnoses and plans consistent with the patient's reported issues and clinical findings?
4. Does the assessment and plan section include the main diagnosis and relevant additional diagnoses?
5. Is there sufficient detail in the assessment and plan to reflect patient-specific nuances discussed during the visit?

6. Does the assessment and plan section include only relevant clinical reasoning and plan details, without repeating history of present illness (HPI) information or subjective patient statements?
7. Are diagnoses prioritized correctly according to the main reason for the visit?
8. Is the assessment and plan based on information that is discussed or agreed upon with the patient?
9. Are all relevant diagnostic and management details, such as medication names, doses, and follow-up plans, accurately captured in the assessment and plan?
10. Does the assessment and plan include a summary of the patient's current clinical status, such as stable or unstable?
11. Is the assessment and plan section free from incorrectly attributing patient statements to physician recommendations or vice versa?
12. Does the assessment and plan section include follow-up actions or recommendations discussed during the patient encounter?
13. Is the assessment and plan free of extraneous information, ensuring that only details relevant to the current diagnosis and treatment plan are included?
14. Are diagnoses in the assessment and plan section consistent with the clinical discussion and relevant to the specialty?
15. Is there a clear distinction between the patient's reported symptoms and the physician's assessments or recommendations?
16. Are the goals of treatment and expected outcomes clearly stated in the assessment and plan?
17. Is the assessment and plan section clear, concise, and free of redundancy?

18. Does the assessment and plan document treatment options and decisions discussed during the visit?
19. Is all relevant medication information, including names, dosages, and frequency, accurately documented in the assessment and plan?
20. Is medical terminology correctly and consistently used in the assessment and plan section?

3.6 Evaluation

We then evaluate the utility of the final checklist, using the following suite of evaluation metrics:

- Scores for feedback coverage, feedback diversity, and LLM enforceability.
- Predictive power of the checklist questions.
- Robustness against perturbations.
- Correlation with human preference ratings.

3.6.1 Feedback Coverage & Diversity and LLM Enforceability

Following the definitions given in §3.4.2, we can measure the coverage, diversity, and LLM enforceability of our checklist questions compared to a baseline checklist. We additionally evaluate diversity by embedding the feedback covered by each question (using `text-embedding-3-large`), and then minimizing intra-cluster distances among questions and maximizing inter-cluster distances between question clusters. Intra-cluster distance is defined as the average distance between all feedback items covered by a question, and inter-cluster distance is defined as the average distance between the centroids of the clusters of feedback items covered by each question.

Our checklist has better feedback coverage and diversity and is more LLM enforceable than the baseline checklist, as shown in Table 3.2. The feedback checklist has a coverage of 0.987 and a diversity of 0.923, compared to the baseline’s coverage of 0.978 and diversity of 0.897. The feedback checklist also has a lower intra-cluster distance and a higher inter-cluster distance than the baseline checklist, indicating that it covers a wider range of feedback items while avoiding redundancy. Our feedback checklist also obtains a higher enforceability score, i.e., it is more answerable by LLMs. From Figure 3.5, we see that the baseline checklist has an average unit test scores of 0.82 over 10 questions, while the feedback checklist has an average unit test score of 0.89 over 20 questions.

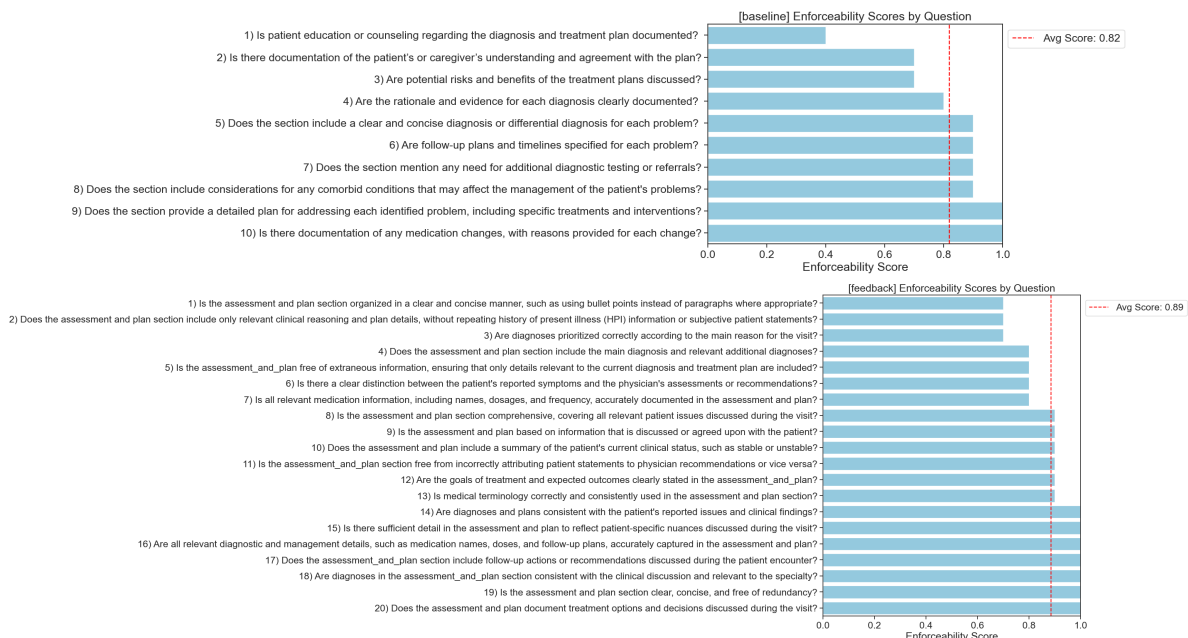


Figure 3.5: The enforceability score of each checklist question, indicating how LLM enforceable each one is. The feedback checklist has both a greater quantity of perfectly enforceable questions and a higher average rate of enforceability.

3.6.2 Predictive Power

We measure the predictive power of the checklist questions by using them as features to predict the star ratings of the notes. We can do this by training a classifier on the feedback

data, where the input is the feedback and the output is the star rating of the note. The checklist questions are used as features for this task, with the mapping of the input to features as the coverage matrix from §3.4.2. For simplicity, we set up the task as a binary classification task, by using data with star ratings of 1 as negative and 5 as positive (1113 and 680 samples respectively). We then train a Logistic Regression model with 5-fold cross-validation on 80% of the feedback and test with the remaining 20%.

Our checklist has better predictive power than the baseline checklist as well. Our feedback checklist yields a higher accuracy of 0.70 for predicting the star ratings of the notes, compared to 0.62 for the baseline checklist. We also find that our feedback checklist has a higher macro F1 score of 0.63, compared to 0.43 for the baseline checklist. This indicates that the feedback checklist is better aligned with the star ratings of the notes.

Name	Description	Section	# Notes
<code>delete_sentence</code>	delete random sentences from each section	All	207
<code>delete_section</code>	remove a whole section of the note	All	207
<code>repeat_sentence</code>	repeat random sentences in each section	All	207
<code>coherence_shuffle</code>	shuffle sentences within each section	All	207
<code>section_shuffle</code>	shuffle sentences across sections	All	207
<code>irrelevant_sentence</code>	insert irrelevant sentences (e.g., from other notes)	All	207
<code>prm_inaccuracy</code>	errors involving incorrect info or unsupported topics	AP	67
<code>prm_hallucination</code>	introduce unrelated subject entities	AP	67
<code>prm_unhelpful</code>	introduce vague, confusing, or incomplete expressions	AP	67
<code>prm_incomplete_step</code>	randomly removing specific steps from A&P	AP	67
<code>prm_paraphrase</code>	Gemini Pro 1.5 paraphrased note sections	AP	67

Table 3.3: Descriptions of the perturbed notes for evaluation. The number of notes is the number of reference notes that were perturbed with each perturbation. Perturbations beginning with `prm` are from Wang et al. [2025b]

3.6.3 Robustness against Perturbations

As described in §3.3.3, we leverage ACI-bench notes as a set of **reference notes** that are of good quality. We then define a set of **perturbations**, i.e., transformations, that can be applied to the reference notes, to simulate poor quality notes. We are inspired by the

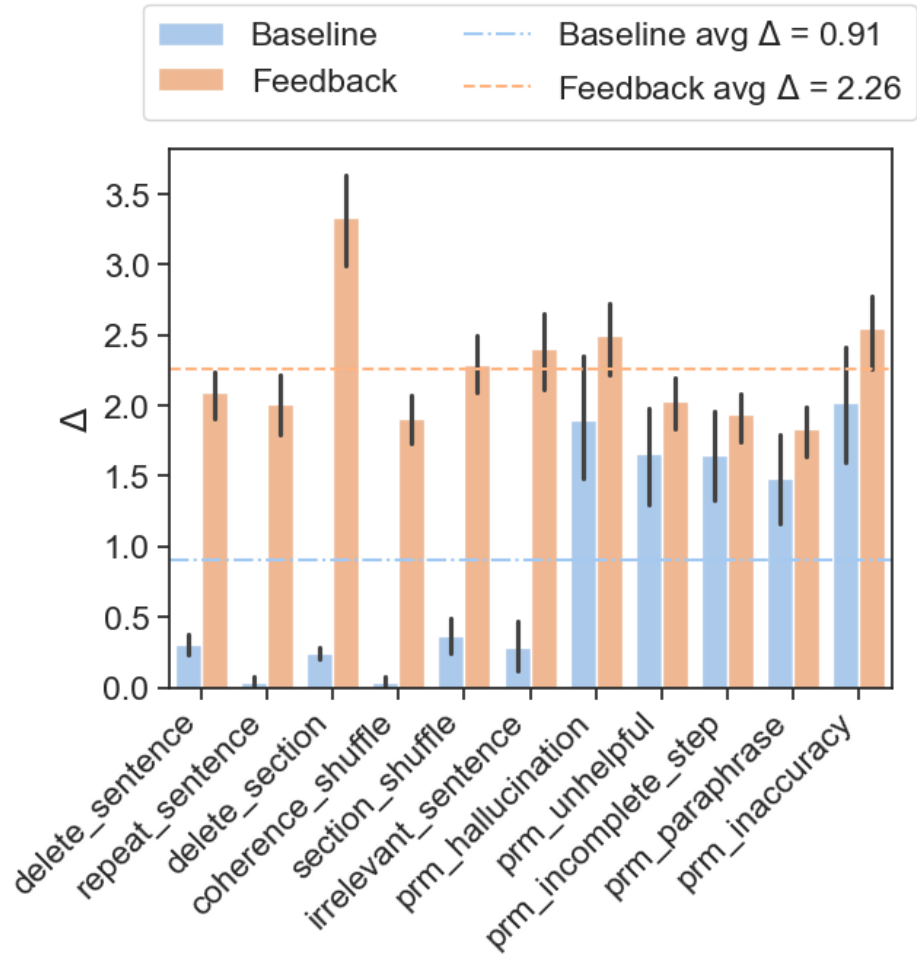


Figure 3.6: The feedback checklist has a higher perturbation Δ than the baseline checklist. It is more robust against perturbations, particularly for missing information, organization, and redundancy/hallucination.

perturbations defined by Guo et al. [2024] for our use case, though we implement our own methods. We also use data from Wang et al. [2025b] as perturbed versions of *AP* note sections. Specifically, we apply each of the following perturbations in Table 3.3 to the reference notes. With this benchmark, we can measure the robustness of our final checklists against changes in informativeness, coherence, and factuality. After scoring the reference notes and the perturbed notes with the checklist, we can measure the robustness of the checklist against these perturbations.

For a checklist question, we define its **Perturbation Delta** (Δ) as the score of the perturbed note subtracted from the score of the reference note, averaged over all notes. Since a perturbed note should score lower than the reference note, Δ should be larger for more robust checklist questions. We can then measure the robustness of the full checklist by measuring the average Δ across all questions.

Our checklist is significantly more robust against perturbations than the baseline checklist. As shown in Figure 3.6, the feedback checklist has an average perturbation Δ of 2.26, compared to 0.91 for the baseline checklist. The Δ ’s are significantly greater than 0 for all perturbations ($p < 0.001$, one-sample t -test). In contrast, the baseline shows no significant change for the `repeat_sentence` and `coherence_shuffle` perturbations. We also confirm that most Δ ’s are significantly larger for our feedback checklist than the baseline, with $p < 0.05$ for a two-sample t -test and a large effect size (Cohen’s $d = 2.94$).

Our checklist is particularly robust to perturbations like `delete_sentence` and `delete_section`, which are designed to simulate missing salient information. Furthermore, our checklist is more sensitive to perturbations like `section_shuffle` and `coherence_shuffle`, which are designed to simulate poor writing flow and organization. `repeat_sentence` and `irrelevant_sentence` are also detected by our feedback checklist, but not by the baseline checklist, suggesting that our feedback checklist is better at detecting redundancy and hallucination errors in the notes.

We also see that our feedback checklist produces a higher average score for the reference

notes than the baseline checklist (Figure 3.7).

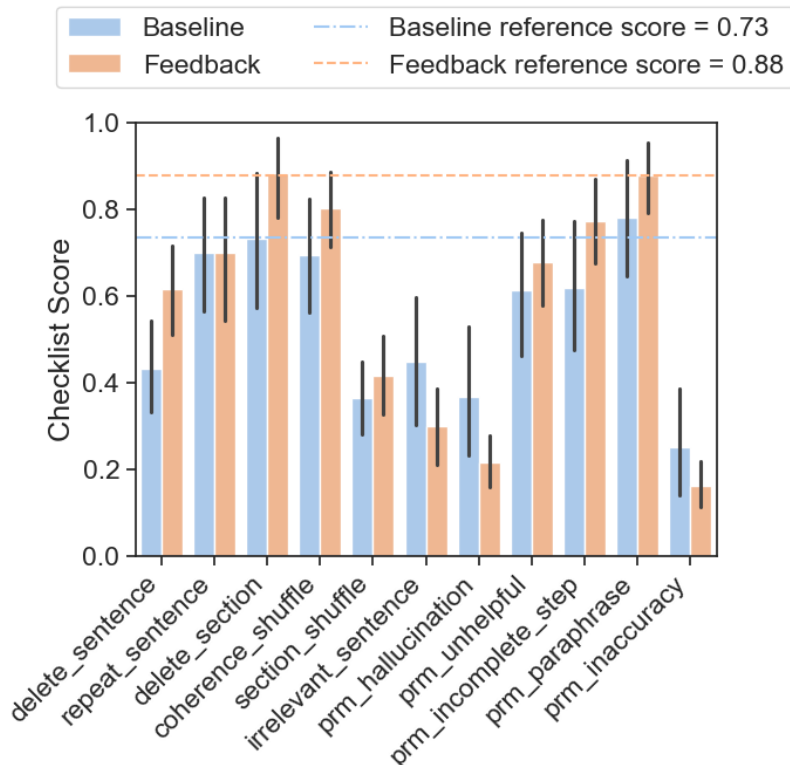


Figure 3.7: The feedback checklist has a higher average score for the reference notes than the baseline checklist.

3.6.4 Correlation with Human Preference Ratings

We use 109 unique pairs of preference ratings for the *AP* section for our evaluation. For each note pair, we compute the checklist score for each note. We then aggregate the scores for the ranked notes and check for statistical significance. Ideally, the preferred note should have a higher checklist score than the non-preferred note.

Our checklist is significantly correlated with human preferences compared to the baseline checklist (see Figure 3.8). That is, our checklist produces a significant difference ($p \leq 0.05$ from a paired t -test) in scores between the preferred and non-preferred notes, with preferred notes having the higher score. The baseline checklist does not produce a significant

difference in scores between the preferred vs. non-preferred notes.

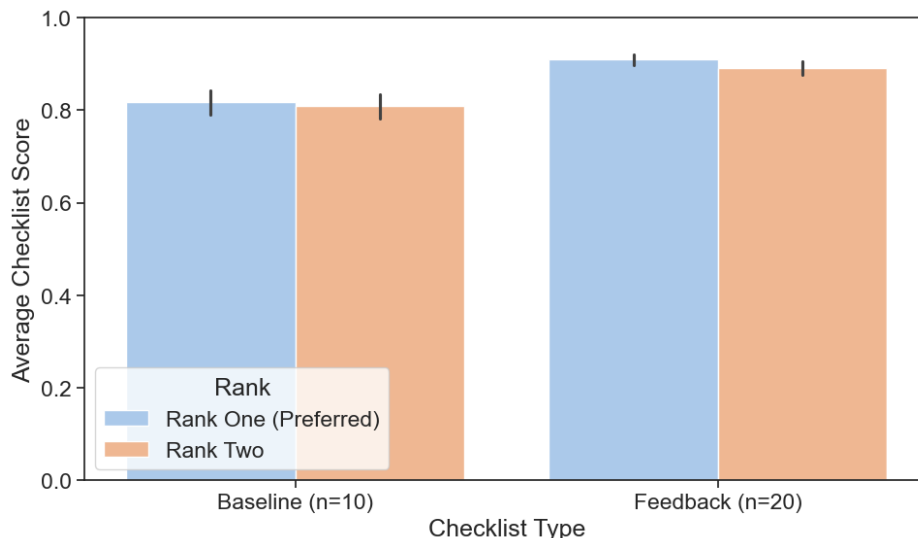


Figure 3.8: Correlation with human preference ratings is significant for our checklist ($p \leq 0.05$ from a paired t -test, Cohen’s $d = 0.28$), but not for the baseline.

3.7 Conclusion

In this work, we propose a method to automatically generate a checklist for evaluating clinical notes, based on real user feedback for notes produced by a commercial Ambient AI company. We show that our generated checklists have more desirable properties than a baseline checklist, and we define a series of evaluation metrics to assess the quality of the checklists. In particular, we show that our checklists are comprehensive, diverse, predictive of human ratings, and aligned with expert preferences.

Our approach addresses a critical need in healthcare AI for scalable, clinically-grounded evaluation. By systematically distilling feedback from over 21,000 real clinical encounters, we demonstrate that user concerns can be transformed into actionable evaluation criteria. The resulting checklist serves as an interpretable, enforceable tool that can flag potentially problematic notes while maintaining transparency about evaluation criteria.

This work demonstrates the practical value of leveraging large-scale user feedback to

create evaluation tools for deployed AI systems. Our pipeline provides a systematic approach to convert real user feedback into structured evaluation criteria, offering a scalable alternative to purely manual review while preserving clinical input and oversight.

Future Work

Future work includes scaling the pipeline to additional note sections and domains, implementing dynamic and more robust feedback filtering, and incorporating advanced evaluation methods such as feature importance analysis, human studies, and improved LLM-evaluator reasoning to further refine and validate the checklists. Some evaluation metrics, like feedback coverage/diversity and LLM enforceability, can be incorporated into the checklist refinement pipeline, to help select for the most salient questions. This would allow the refinement process to be more iterative.

Scaling the Pipeline We would like to scale the pipeline to generate checklists for other note sections. We would also like to implement a dynamic pipeline, where the checklist is updated as new feedback comes in. This would allow us to continuously improve the checklist and adapt it to changing user needs.

Additional Feedback Filtering The feedback we receive is often unstructured and noisy. There is a lot of redundancy in the feedback, and not all of it is actionable. So, we can apply additional filtering steps to the feedback, such as grouping by specialty, filtering out unactionable feedback, and clustering similar feedback.

Question Importance for Predicting Human Ratings We can further identify questions that have the greatest influence on performance, using feature importance measures like SHAP values [Lundberg and Lee, 2017]. These values can also help assign priority to

the order of questions in the checklist, so that the most important questions are asked first or given more weight in the final checklist score.

Human Evaluation To validate the quality of our checklists, we will run human evaluation. Some potential user studies include:

- Pairwise preference of checklists (e.g., rate baseline vs. feedback checklist questions)
- Have physicians conduct evaluation of real deployed notes using checklists, and see if there is higher agreement between raters compared to evaluating without a checklist

Generalizability To show the generalizability of our checklist generation method, we would like to apply it to some alternate domain. We are still brainstorming what dataset(s) we could use for this. Requirements of the data include user feedback and ratings for some product that should fulfill certain standards.

Tuning a LLM-evaluator We can further try tuning LLM-as-a-Judge specifically for our task of checklist scoring. While the current setup simply has the LLM-evaluator produce a “Yes”/“No” answer for each checklist question, we can also allow the LLM to reason about the note and checklist question and provide a more detailed explanation of its score. Allowing the evaluator to cross-validate the note with structured EHR data could enhance accuracy by catching factual inconsistencies that text-only evaluation misses.

Limitations

Our pipeline makes several simplifying assumptions, such as excluding "N/A" answers and cross-section comparisons, which may limit the scope and nuance of the resulting checklists. Checklist coverage is inherently limited by the content of user feedback; e.g., if there are no complaints about pronoun usage, there may not be a checklist question about using correct

pronouns. Checklist questions that favor comprehensive notes may penalize concise ones that are still clinically sufficient, such as in time-sensitive settings like emergency care. Our methodology could be adapted for these specialties by filtering for specialty-specific feedback, as well as for other contexts (e.g., multi-provider notes).

Our current setup treats all checklist questions with equal weight, and though this assessment is inherently subjective, we acknowledge that clinical significance varies across checklist items. Weighted scoring could be implemented automatically (e.g., Wei et al. [2025]) or added during clinical review of the rubric as a post-hoc step, without redesigning our existing methodology.

We rely heavily on LLMs in our pipeline and evaluation, which introduces its own set of challenges. When using LLMs-as-a-Judge, there is a lack of ground truth labels to determine whether clinical notes satisfy checklist questions, such as for ACI-bench notes. Exhaustive evaluation is difficult, since each checklist question can be a substantial problem on its own. As such, our current metrics are not exhaustive. Finally, we rely on proxies for positive signals, such as internal pairwise preference data, though these proxies also have their own limitations (e.g., poor agreement between raters, though we mitigate this by selecting pairs with at least two raters in agreement).

3.8 Ethical Considerations

Our work involves the use of clinical note data and user feedback for the purpose of building automated evaluation pipelines. All proprietary data used in this work, such as user feedback, star ratings, and clinical note content, is handled in compliance with applicable data privacy policies and HIPAA regulations. All PHI is de-identified in accordance with HIPAA safe harbor standard before model training and evaluation; no direct identifiers are transmitted outside the covered environment.

Our checklist is designed as a scalable triage tool, not a replacement for clinical judg-

ment. Our generated checklists are designed to be interpretable and transparent, but further validation with practicing clinicians is necessary before clinical deployment. Clinicians can review the finite set of criteria, after which the checklist can be applied at scale to flag notes for expert attention. We acknowledge that LLM outputs may introduce subtle errors or biases, and we flag this as an area for continued validation. In production, the checklist will be part of a much larger evaluation suite that involves human and clinician oversight at different stages, not to be used as a single criterion for decision making.

CHAPTER 4

AUTOCHECKLIST: COMPOSABLE PIPELINES FOR CHECKLIST GENERATION AND SCORING WITH LLM-AS-A-JUDGE

4.1 Introduction

Checklists decompose quality into individually verifiable criteria, with simple yes/no answers that bypass position bias of pairwise comparisons and subjectivity of scalar metrics with LLM-as-a-Judge [Ye et al., 2024]. As a result, checklists offer interpretable and fine-grained evaluation of text quality, making them an effective tool for evaluation with LLM evaluators [Cook et al., 2024, Viswanathan et al., 2025, Wei et al., 2025, Zhou et al., 2025].

Several checklist generation methods have been proposed in recent work (see §4.2), each with distinct codebases, prompting strategies, and scoring mechanisms. These approaches differ along several axes: scope of checklist questions, inclusion of a reference response in checklist generation, unweighted vs. weighted scoring, and question refinement steps, if any. No existing toolkit provides a unified interface across these methods, making it difficult to compare or extend them for new tasks without significant re-implementation.

To address this, we develop **AutoChecklist**, an open-source Python library that consolidates common checklist approaches into composable pipelines. At its core is a taxonomy of five checklist generation abstractions, each encoding a distinct strategy for deriving evaluation criteria. A modular *Generator* $\rightarrow$ *Refiner* $\rightarrow$ *Scorer* pipeline connects any generator with a unified scorer, and new configurations can be registered via prompt templates alone. We additionally validate the utility of the library (§4.5) and demonstrate its effectiveness in a new domain (§4.6). Our contributions are as follows:

- A **taxonomy of five generator abstractions** (direct, contrastive, inductive, deductive, and interactive) that organizes checklist methods by their strategies for producing evaluation criteria.

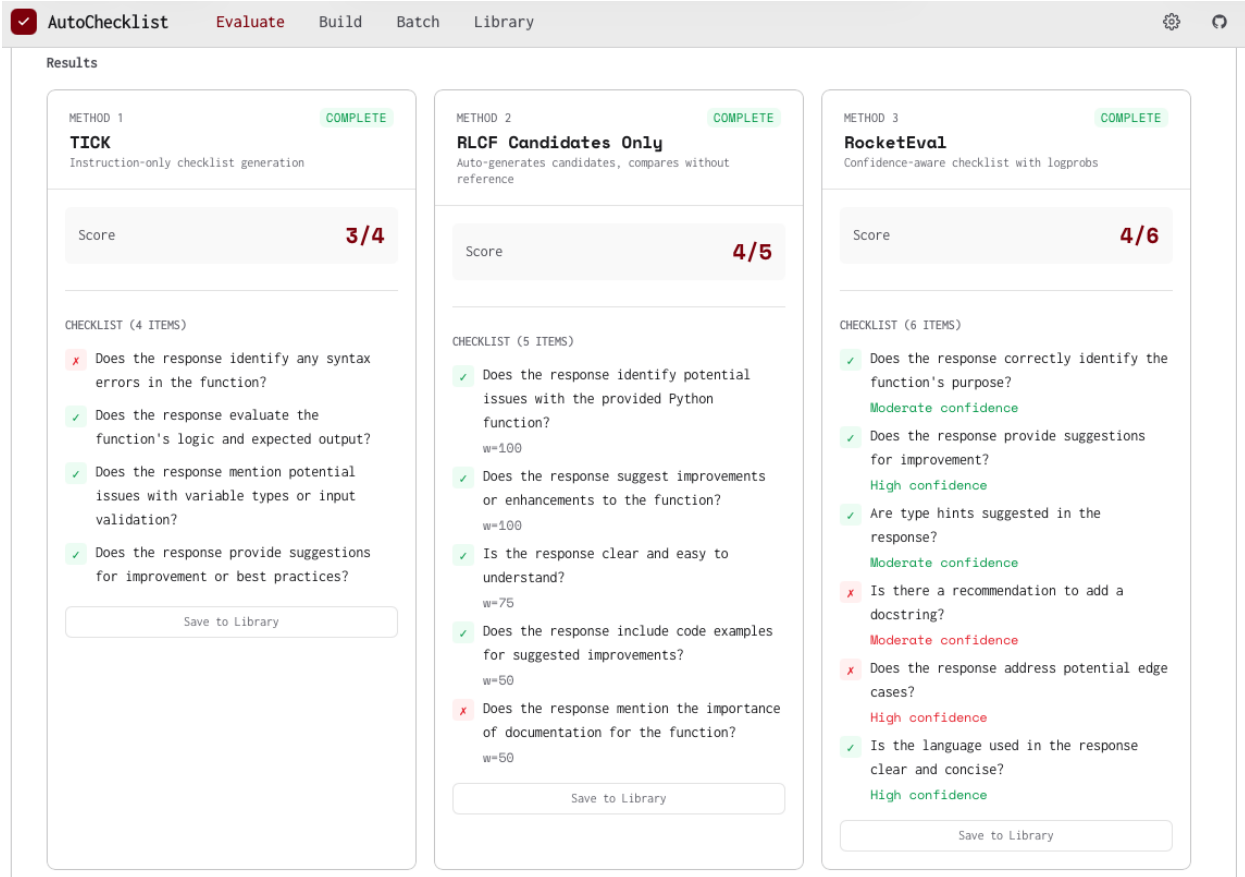


Figure 4.1: In the *Compare* page of AutoChecklist’s UI, users select methods and enter an input to generate side-by-side checklists, enabling direct comparison of generation strategies on the same task.

- A framework of **composable pipelines**: ten built-in configurations implementing published methods, compatible with a unified scorer that consolidates three scoring strategies from the literature. Users can customize these pipelines to new tasks via Markdown prompt templates alone.
- A **pip-installable Python package** with multiple usage modes, including a CLI for off-the-shelf evaluation with pre-defined pipelines.
- **Multi-provider LLM backend** support for OpenAI, OpenRouter, and vLLM (including VLLMOfflineClient for local GPU inference), and automatic handling of structured output.
- A **locally hosted user interface** supporting most package functionality, allowing for

interactive prompt customization, pipeline configuration and comparison (e.g., Figure 4.1), and batch evaluation.

`AutoChecklist` is publicly available at <https://github.com/ChicagoHAI/AutoChecklist>.

4.2 Related Work

Checklist Generation & Evaluation. Checklist-based evaluation with LLM-as-a-Judge has been leveraged for a variety of tasks and benchmarks [Lin et al., 2024, Que et al., 2024, Qin et al., 2024]. A common evaluation setup is to generate a checklist per input-output pair, such as for a user query and LLM response. Checklist generation can either use zero- to few-shot prompting to generate yes/no questions from the input alone, or by incorporating a reference and/or alternative response samples Cook et al. [2024], Wei et al. [2025], Viswanathan et al. [2025]. Other methods aim to derive a shared checklist from corpus-wide signals. These signals are either pre-defined by humans and then processed into granular criteria, or extracted from brainstorming or feedback items into structured checklists Lee et al. [2025], Chu et al. [2025], Zhou et al. [2025]. These methods often involve multiple steps to refine the initial checklist questions, such as through deduplication, filtering, or elaboration.

Despite their differences, all of these methods share a common structure: generate criteria from an input, optionally refine them, then score a target response. `AutoChecklist` builds on this shared structure. We identify five generator abstractions, each encoding a distinct reasoning strategy, and factor them into interchangeable generators, refiners, and scorers within a single library. To our knowledge, this is the first unified toolkit for LLM-based checklist generation.

Utility of Checklists. Beyond evaluation, structured criteria serve as signals for alignment and reinforcement learning [Viswanathan et al., 2025, Siro et al., 2026, Dineen et al., 2025, Yang et al., 2026] and self-correction [Wan et al., 2026]. However, Furuhashi et al. [2025] find that checklists are not consistently useful and highlight the need to more clearly define objective evaluation criteria. **AutoChecklist** aims to lower the barrier for both human and LLM judges to define and select such criteria.

Unified Frameworks. `j judges` is a similar library to use and create LLM-as-a-Judge evaluators, curating a set of LLM evaluators in a low-friction format that can be used off-the-shelf or customized. Numerous other open-source packages exist for evaluating and testing LLM systems (DeepEval, RAGAS, OpenAI Evals). While the generated checklists from **AutoChecklist** are primarily used with LLM-as-a-Judge, their usefulness is not limited to automatic evaluation; there is also potential for such checklists to help further align human annotators in subjective tasks (e.g., essay grading, clinical note assessment). Furthermore, **AutoChecklist** features unified criteria *generation*; most other libraries focus on scoring.

4.3 System Details

A *checklist* $C = \{q_1, \dots, q_n\}$ is a set of yes/no questions designed to assess a target text. Each question q_i may carry an optional importance weight $w_i \in [0, 100]$. Given a target response, a *scorer* assigns each item an answer $a_i \in \{\text{YES}, \text{NO}\}$ and optionally a confidence $c_i \in [0, 1]$. The *pass rate* for a single input is $|\{a_i = \text{YES}\}|/|C|$, or $\sum w_i \cdot \mathbf{1}[a_i = \text{YES}] / \sum w_i$ when importance weights are available.

Since different methods are originally implemented with specific tasks in mind (e.g., evaluating LLM instruction-following or quality of a summary), we further define the following abstractions: a checklist *input* is the instruction, query, or task being evaluated (e.g., “Write a haiku about autumn” or “Summarize: {article}”); the checklist or evaluation *target* is the

Generator	Level	Strategy	Pipelines
DIRECT	Instance	Single-step generation: given the input (and optionally a reference), directly prompt the LLM to produce a checklist	TICK [Cook et al., 2024], RLCF Direct [Viswanathan et al., 2025], RocketEval [Wei et al., 2025]
CONTRASTIVE	Instance	Counterfactual reasoning: generate candidate responses of varying quality, then derive criteria by contrasting good vs. bad responses	RLCF Candidate-based [Viswanathan et al., 2025], CRG [Liu et al., 2025]
INDUCTIVE	Corpus	Bottom-up induction: convert reviewer feedback or observations into general evaluation criteria, with built-in deduplication and selection	Zhou et al. [2025]
DEDUCTIVE	Corpus	Top-down decomposition: convert expert-defined evaluation dimensions into specific checklist questions, with optional augmentation	Lee et al. [2025]
INTERACTIVE	Corpus	Starting with defined dimensions, extract criteria from human and LLM think-aloud evaluation sessions via a multi-stage pipeline of clustering and question generation	Chu et al. [2025]

Table 4.1: Five checklist generation abstractions in **AutoChecklist**. Each represents a distinct strategy for deriving evaluation criteria. The rightmost column lists built-in pipelines that instantiate each generator with paper-specific prompt templates and scoring strategies.

output being scored against the checklist (e.g., the haiku or summary).

Design and Architecture. As depicted in Figure 4.2, generators fall into two categories. *Instance-level generators* produce a checklist per input (e.g., per prompt-response pair). **DirectGenerator** (DIRECT) implements direct inference: a single LLM call that converts the input into checklist questions. **ContrastiveGenerator** (CONTRASTIVE) implements counterfactual reasoning; it first generates candidate responses of varying quality (usually from weaker models), then derives discriminative criteria by contrasting them. If the user provides preference pairs, the generator can skip candidate generation and directly derive criteria by contrasting chosen and rejected responses. *Corpus-level generators* derive a shared checklist from corpus-wide signals. **InductiveGenerator** (INDUCTIVE) creates checklists “bottom-up” from unstructured observations, such as user feedback or reviews. **DeductiveGenerator**

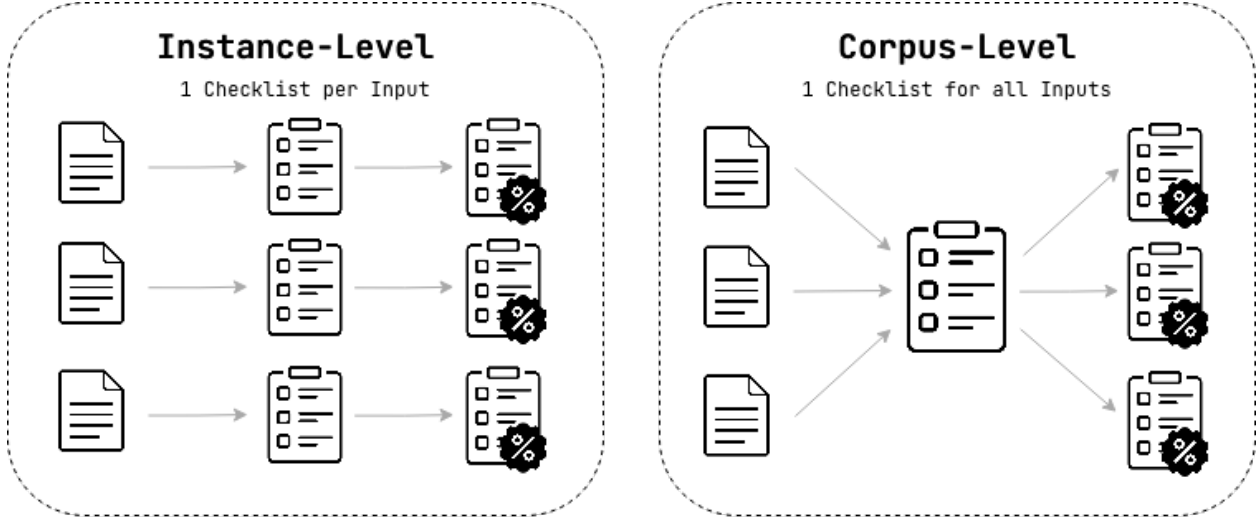


Figure 4.2: *Instance-level* generators produce one checklist per input; criteria are tailored to each specific task (target). *Corpus-level* generators produce one checklist for an entire dataset of targets, capturing general quality patterns derived from higher-level signals.

(DEDUCTIVE) performs top-down decomposition from expert-defined evaluation dimensions. **InteractiveGenerator** (INTERACTIVE) extracts criteria from simulated think-aloud protocols, wherein participants verbalize their thoughts during tasks. Table 4.1 shows how several papers ultimately share similar underlying steps for checklist generation.

Refiners. Refiners are optional post-processing steps composed in sequence before scoring. The **Deduplicator** merges semantically redundant questions via embedding similarity and LLM-based cluster merging. The **Tagger** applies additional filtering of items by a specified quality (e.g., generality, specificity). The **UnitTester** validates that each item is “LLM enforceable” [Zhou et al., 2025]. The **Selector** uses beam search to optimize an objective function against checklist length. For most of the corpus-level generators, refinement steps are built into the generator classes; future work can refactor them into more modular steps.

Scorer. A unified **ChecklistScorer** class consolidates scoring strategies from three different papers into a single configurable interface. It operates in two modes: *batch* mode scores all checklist questions in a single LLM call, while *item* mode scores one question per

Pipeline	Generator	Score	Ref?
<code>tick</code>	DIRECT	pass	No
<code>rocketeval</code>	DIRECT	norm.	Yes
<code>rlcf_direct</code>	DIRECT	weight.	Yes
<code>rlcf_cand.</code>	CONTRASTIVE	weight.	Yes
<code>rlcf_cand._only</code>	CONTRASTIVE	weight.	No
<code>or_pairwise</code>	CONTRASTIVE	pass	No
<code>or_listwise</code>	CONTRASTIVE	pass	No
<code>checkeval</code>	DEDUCTIVE	pass	—
<code>interacteval</code>	INTERACTIVE	pass	—
<code>feedback</code>	INDUCTIVE	pass	—

Table 4.2: Built-in pipelines. Each pipeline pairs a generator with a default `ChecklistScorer` configuration and paper-specific prompt templates. Score metrics: pass rate, normalized, or weighted (see §4.3).

call. It can configure use of chain-of-thought reasoning (`capture_reasoning`) and logprob-derived confidence scores (`use_logprobs`). The scorer computes three aggregate metrics: `pass_rate` (fraction of YES answers), `weighted_score` ($\sum w_i \cdot \mathbf{1}[a_i=\text{YES}] / \sum w_i$, using importance weights as in Viswanathan et al. [2025]), and `normalized_score` (calibrated from logprob confidence as in Wei et al. [2025]). A `primary_metric` parameter designates which metric the `primary_score` attribute returns; each built-in pipeline sets this to match its original paper’s scoring method. For a corpus of targets, both macro-average scores and Decomposed Requirements Following Ratio (DRFR), i.e., micro-average pass rate [Qin et al., 2024], are calculated.

Pipelines. Each stage in a pipeline is its own unit; any generator can be paired with any scorer, and refiners compose in sequence. For instance, a user can generate checklists with the INTERACTIVE strategy but score them with the normalized scorer, a previously unsupported combination.

Table 4.2 shows how the five generator abstractions are instantiated as ten built-in pipelines, i.e., pre-set configurations. Each row pairs a generator with a default scorer configuration and paper-specific prompt templates, but any generator can be combined with

any scorer. Users can add new pipelines by registering custom prompt templates, without modifying library code. Base classes are also available for ease of developing more advanced custom components.

4.4 Deployment and Usage

`AutoChecklist` is `pip`-installable and supports three usage modes with increasing customization: a CLI for off-the-shelf evaluation, a web interface for interactive exploration and prompt iteration, and a Python API for full pipeline control. The library supports multiple LLM backends via a shared `LLMClient` protocol: OpenAI API, OpenRouter, and vLLM (both as an online server and via `VLLMOfflineClient` for local GPU inference). Structured output is handled via provider-enforced JSON schema (`response_format`), or appended format instructions and parsing as a fallback.

```
autochecklist run --pipeline tick \  
  --data eval_data.jsonl \  
  --output results.jsonl \  
  --generator-model openai/gpt-5-mini \  
  --scorer-model openai/gpt-5-mini
```

Figure 4.3: CLI: end-to-end evaluation with a built-in pipeline.

Command-Line Interface. The CLI provides three subcommands for use with registered components: `run` (generate checklists and score in one step), `generate` (produce checklists without scoring), and `score` (score targets against a pre-existing checklist). A fourth subcommand, `list`, discovers available generators, scorers, and refiners. Users specify a built-in pipeline (e.g., `--pipeline tick`), a custom generator prompt file (`--generator-prompt my_eval.md`), or a saved pipeline config JSON. Runs are resumable after interruption, with incremental saving of results.

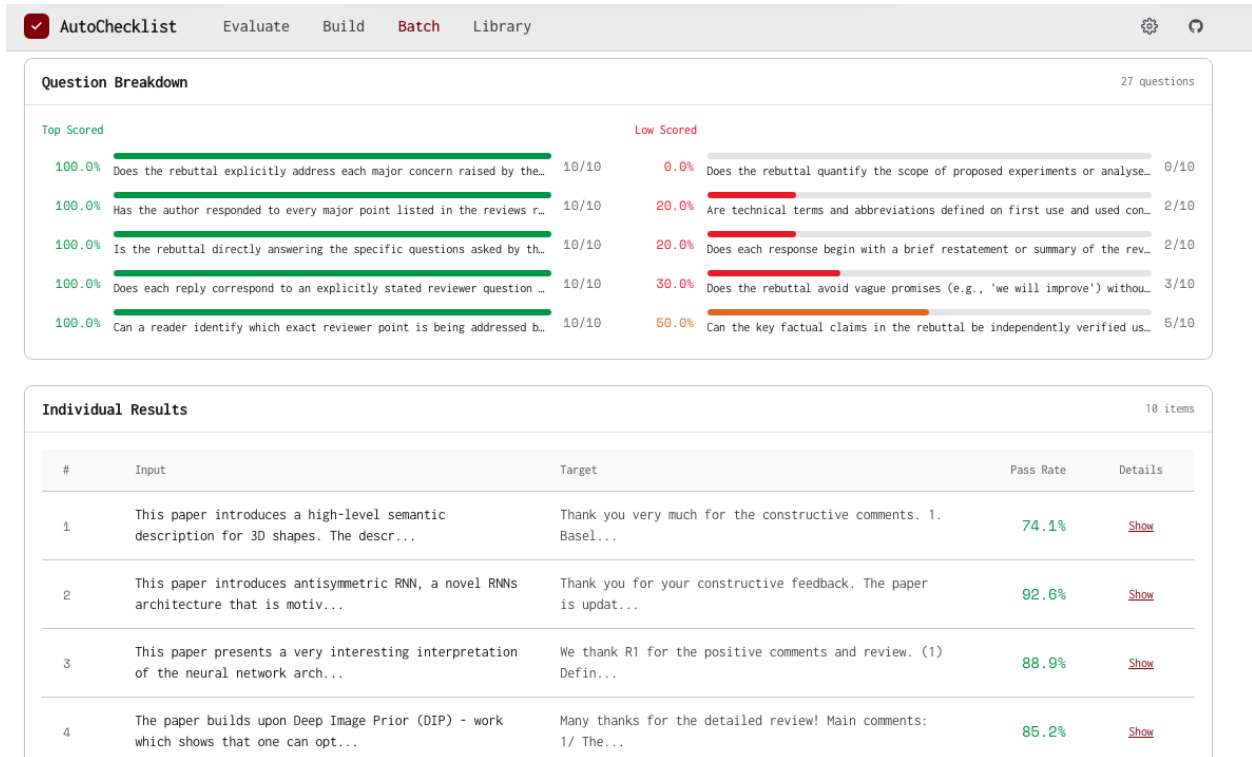


Figure 4.4: The Batch results page showing per-item score breakdowns and aggregate statistics for a dataset evaluation run.

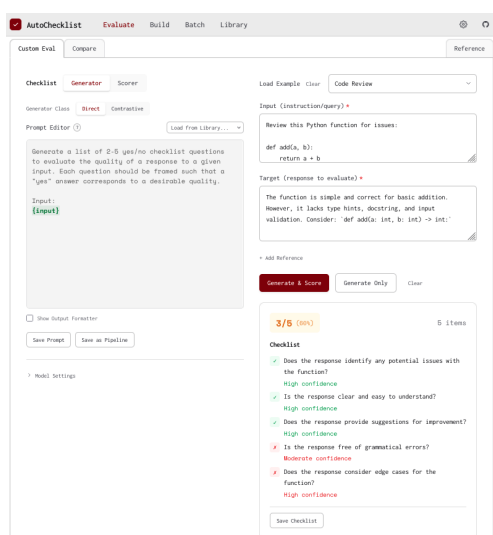
User Interface. A locally hosted interface built with FastAPI and Next.js provides interactive access to most library functionality without writing code, easily launched with `autochecklist ui`.

The main page **Evaluate** offers three tabs. *Custom Eval* provides full prompt editing: users choose a generator class, edit generator and scorer prompts in tabbed editors, load from saved prompt templates, select an output format and scorer type, then run an evaluation and view checklist and score results. The *Compare* tab enables side-by-side comparison of built-in methods and custom pipelines on the same input, with results displayed as horizontally scrollable method cards (see Figure 4.1). *Reference* includes descriptions for all generators, scorer options, and pipelines. The **Build** interface supports **Deductive** corpus-level checklist generation for downstream use.

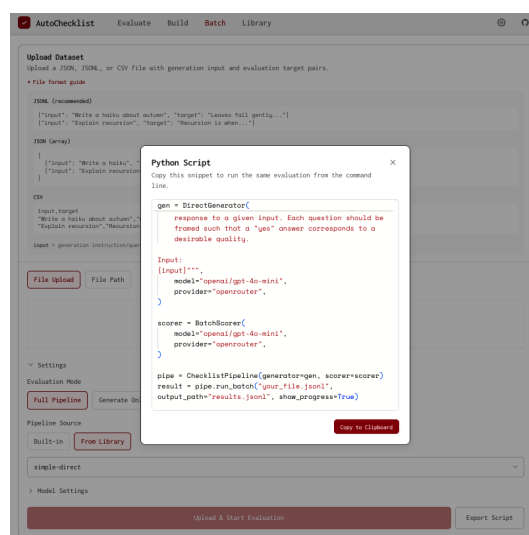
Configurations from these workflows can be saved to the **Library**, which stores *Checklists*

for saved checklists (clicking in allows for edits), *Prompt Templates* for reusable prompts with placeholders, and *Pipelines* for saving complete evaluation configurations that bundle generator class, prompts, scorer settings, and output format.

To process more data samples at once, the **Batch** page supports CSV/JSON data upload, evaluation mode with saved Library items, progress tracking, and a results page with basic aggregate statistics and per-item score breakdowns (such as in Figure 4.4). There is also the option to export the experiment in Python code to run the same pipeline through a script, which is preferred for very large datasets and for more fine-grained customization. See Figure 4.5 for examples.



(a) The *Custom Eval* tab, where users edit generator and scorer prompts, select an output format, and run single-input evaluations.



(b) The *Batch* page, where users can use built-in and custom pipelines to generate checklists and/or score a dataset, along with an option to export the pipeline as a script.

Figure 4.5: Additional UI pages to support prompt iteration and dataset evaluation.

Python API. The Python API offers full control over the pipeline, ideal for large-scale checklist evaluation with finalized generator and scorer configurations. The `pipeline()` factory is the recommended entry point: it resolves a pipeline name to a pre-configured generator and scorer, accepts optional refiners, and returns a callable that generates and scores

in one step. For finer-grained control, `ChecklistPipeline` allows explicit composition of independently configured generators, refiners, and scorers. Components can also be used outside of any pipeline (e.g., calling a generator and scorer directly). Batch results provide both macro-average scores and Decomposed Requirements Following Ratio (DRFR, i.e., a micro-average pass rate) [Qin et al., 2024], and can be exported as DataFrames or JSONL files. Users can register custom pipelines via `register_custom_pipeline()` and share configurations as JSON files, making new evaluation setups reusable without modifying library code.

```
from autochecklist import pipeline

# Built-in pipeline: generate checklist + score in one call
pipe = pipeline("tick", generator_model="openai/gpt-5-mini",
               scorer_model="openai/gpt-5-mini")
result = pipe(input="Write a haiku about autumn.", target="Leaves
               fall gently down...")
print(f"Pass rate: {result.pass_rate:.0%}")

# Evaluation over a dataset
data = [{"input": "Write a haiku", "target": "Leaves fall..."},
        ...]
result = pipe.run_batch(data, show_progress=True)
print(f"Macro pass rate: {result.macro_pass_rate:.0%}")
result.to_dataframe() # or result.to_jsonl("results.jsonl")
```

Figure 4.6: Python API: basic usage

```
from autochecklist import DirectGenerator, ChecklistScorer,
    pipeline
from autochecklist.refiners import Deduplicator, Tagger
```

```

from pathlib import Path

# Custom generator prompt (inline or from a Markdown file)
gen = DirectGenerator(custom_prompt=Path("prompts/my_eval.md"),
    model="openai/gpt-5-mini")
checklist = gen.generate(input="Write a haiku about autumn.")

# Mix and match: chain refiners, then score with a different
    scorer
checklist = Deduplicator(model="openai/gpt-5-mini").refine(
    checklist)
checklist = Tagger(model="openai/gpt-5-mini").refine(checklist)
scorer = ChecklistScorer(mode="item", primary_metric="weighted",
    model="openai/gpt-5-mini")
score = scorer.score(checklist, target="Leaves fall gently down
    ...")

# Or compose everything through the pipeline factory
pipe = pipeline("tick", generator_model="openai/gpt-5-mini",
    refiners=["deduplicator", "tagger"], scorer="
        weighted")

# Switch LLM provider: use a local vLLM server instead of
    OpenRouter
pipe = pipeline("tick", provider="vllm", base_url="http://
    localhost:8000/v1",
    generator_model="meta-llama/Llama-3-8B")

```

	Pipeline Generator	W/L/T	Δ	d	t -test p
tick	gpt-5-mini	75/10/15	+0.286	0.919	6.3e-15
	sonnet-4.6	67/11/22	+0.259	0.846	3.0e-13
rlcf	gpt-5-mini	70/16/14	+0.228	0.785	4.9e-12
	sonnet-4.6	54/11/35	+0.203	0.696	4.4e-10

Table 4.3: Preference discrimination on RewardBench pairs using instance-level pipelines, under both the original `gpt-5-mini` generator and a `claude-sonnet-4.6` (via OpenRouter) ablation. The `gpt-5` scorer is held fixed across runs. W/L/T = Win/Loss/Tie counts; Δ = mean (chosen – rejected); d = Cohen’s d . All Wilcoxon p -values are $< 10^{-8}$.

```
# Register and save custom pipelines for reuse

from autochecklist import register_custom_pipeline,
    save_pipeline_config

register_custom_pipeline("my_eval", generator_prompt=Path("
    prompts/my_eval.md"), scorer="weighted")

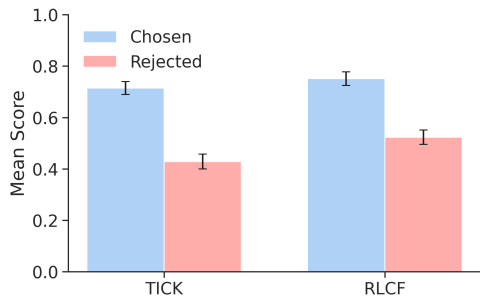
save_pipeline_config("my_eval", "my_eval.json") # share as a
    portable JSON config

pipe = pipeline("my_eval", generator_model="openai/gpt-5-mini",
    scorer_model="openai/gpt-5-mini")
```

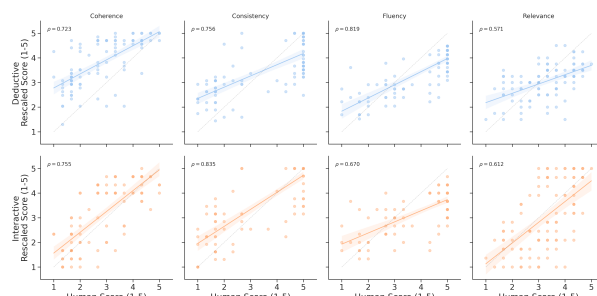
Figure 4.7: Python API: custom prompts, providers, and composition

4.5 Validation

We validate that our pipeline implementations produce useful and effective checklists for two benchmarks: RewardBench Lambert et al. [2025] for instance-level methods and SummEval Fabbri et al. [2021] for corpus-level methods. All experiments use `gpt-5` as the scorer; to test robustness of the pipelines with different checklist generators, we use both `gpt-5-mini` and `claude-sonnet-4.6`.



(a) Mean checklist pass rates from `tick` and `rlcf` on RewardBench pairs.



(b) Rescaled checklist pass rates against human assigned scores on SummEval, across four dimensions.

Figure 4.8: Validation of built-in pipelines on RewardBench and SummEval.

Instance-Level. We test whether instance-level checklist scores can discriminate between preferred and rejected responses. From RewardBench, we sample 100 preference pairs and generate a checklist for each pair’s input prompt using the `tick` (DIRECT, reference-free) and `rlcf_candidate_only` (CONTRASTIVE, reference-free) pipelines. Each method scores both the chosen and rejected response, and we measure win ratio (fraction where the chosen response scores higher), mean score delta, and effect size.

Table 4.3 shows that both pipelines significantly discriminate preferred responses (also in Figure 4.8a). For `gpt-5-mini`, `tick` achieves a 75% win rate with a large effect size ($d = 0.919$, $p < .001$), and `rlcf_candidate_only` achieves 70% ($d = 0.785$, $p < .001$). `claude-sonnet-4.6` has a smaller win to tie ratio, but still produces checklists that give significantly scores to preferred responses. These results confirm that the library’s instance-level pipelines produce checklist criteria that align with human preference judgments of quality.

Corpus-Level. We validate corpus-level methods by measuring correlation with expert quality judgments on the SummEval benchmark, which provides human scores on a 1–5 scale across four dimensions: coherence, consistency, fluency, and relevance. We generate corpus-level checklists using `checkeval` (DEDUCTIVE) and `interacteval` (INTERACTIVE),

Pipeline	Dimension	<u>gpt-5-mini</u>		<u>sonnet-4.6</u>	
		ρ_s	MAE	ρ_s	MAE
checkeval	Coherence	0.723	1.104	0.733	1.041
	Consistency	0.756	0.925	0.804	0.627
	Fluency	0.819	0.829	0.805	0.863
	Relevance	0.571	0.719	0.639	0.652
interacteval	Coherence	0.755	0.703	0.775	0.639
	Consistency	0.835	0.565	0.808	0.627
	Fluency	0.670	0.980	0.759	1.047
	Relevance	0.612	0.910	0.587	0.838

Table 4.4: Corpus-level checklist validation on SummEval under both the original **gpt-5-mini** generator and a **sonnet-4.6** (via OpenRouter) ablation; the **gpt-5** scorer is held fixed. ρ_s : Spearman vs. human scores; MAE on rescaled 1–5 scores. Pearson and Kendall correlations tracked Spearman closely ($|\Delta| < 0.06$) and are omitted for space. All ρ_s are significant ($p < .001$).

then score 100 summaries against each checklist. We map checklist scores to the 1–5 scale ($\text{pass rate} \times 4 + 1$).

Both methods achieve strong correlations across all four dimensions, across generator models (Table 4.4). All correlations are significant at $p < .001$ (Figure 4.8b). **interacteval** achieves the highest dimensional correlations on coherence ($\rho = 0.775$ with **claude-sonnet-4.6**) and consistency ($\rho = 0.835$ with **gpt-5-mini**), outperforming **checkeval** on both attributes in Mean Absolute Error (MAE) from human ratings as well. **checkeval** beats **interacteval** on correlation with fluency ($\rho > 0.805$), also with lowest MAE on this dimension. MAE for relevance is lowest for **checkeval** methods, though correlation strength is more mixed depending on generator model. These results confirm that the library’s corpus-level pipelines produce checklists whose pass rates align with expert judgments, a conclusion that holds across different generator models/providers.

4.6 Case Study: Author Rebuttal Checklists

To demonstrate how `AutoChecklist`’s composable design enables application to new domains, we apply checklist-based evaluation to peer review rebuttals, a setting where no existing checklist method has been tested. The adaptation required only prompt modifications and the library’s existing generator and scorer infrastructure.

Review-rebuttal dynamics of peer review have been studied extensively, uncovering their effects on final scores and successful rebuttal strategies [Gao et al., 2019, Dershowitz and Verma, 2023, He et al., 2026, Ma et al., 2026]. While there are efforts to evaluate review quality and usefulness [Ryu et al., 2025, Sadallah et al., 2025], no prior study has applied checklist-based evaluation to automatically assess *rebuttal* quality. Checklists offer a natural fit: each item can target a specific concern raised by the reviewer, yielding interpretable, fine-grained assessment of rebuttal responses.

Setup. We collect 100 review–rebuttal pairs via the OpenReview API and evaluate them with four generator configurations spanning both instance-level and corpus-level approaches. We select pairs from ICLR 2019, the latest year with post-rebuttal rating change data.

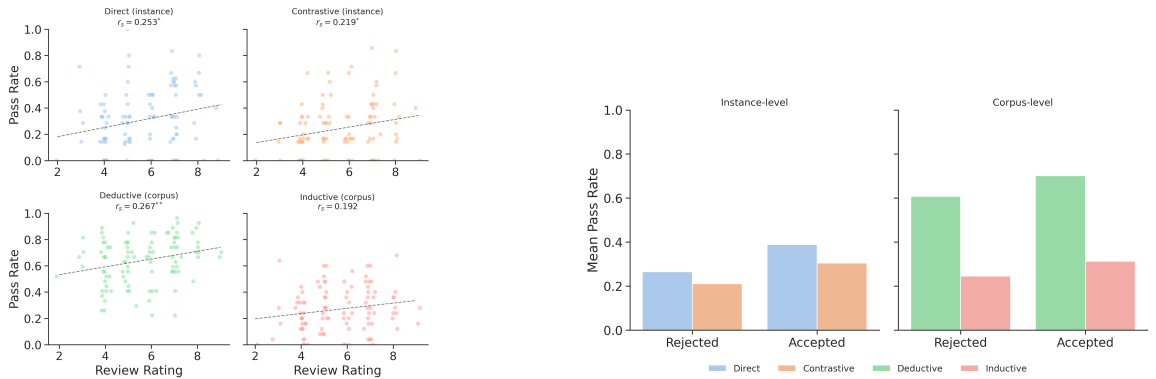
We register custom generator prompts specific to the peer review context. Additionally, `DEDUCTIVE` uses three dimensions from Ma et al. [2026]—Relevance, Argumentation Quality, and Communication Quality—each with sub-dimensions (e.g., Semantic Alignment, Evidence Support, Constructiveness). `INDUCTIVE` takes 1,000 labeled reviewer sentences related to “weaknesses” as input, generates candidate checklist items that capture common patterns of concern across the corpus, and then refines the checklist with deduplication, tagging, and optimal selection.

Results. We evaluate each pipeline’s checklist pass rates against three external signals of rebuttal quality: (1) `RATING`: Spearman correlation between pass rate and the reviewer’s numeric rating; (2) `ACCEPTANCE`: rank-biserial effect size between accepted and rejected

Generator	RATING	ACCEPTANCE		RATING CHANGE	
	r_s	$ d $	$\Delta\bar{x}$	r_{pb}	AUC
DIRECT	0.253*	0.296*	+0.124	0.143	0.587
CONTRASTIVE	0.219*	0.270*	+0.094	0.084	0.531
DEDUCTIVE	0.267**	0.282*	+0.094	0.242*	0.668
INDUCTIVE	0.192	0.240*	+0.068	0.172	0.630

Table 4.5: Evaluation of checklist-based rebuttal scoring on ICLR 2019. RATING: Spearman r_s between pass rate and reviewer rating. ACCEPTANCE: rank-biserial effect size $|d|$ and pass rate difference $\Delta\bar{x} = \bar{x}_{acc} - \bar{x}_{rej}$. RATING CHANGE: point-biserial r_{pb} and ROC-AUC for predicting whether the reviewer changed their rating post-rebuttal. * $p < .05$, ** $p < .01$, *** $p < .001$.

papers; and (3) RATING CHANGE: point-biserial correlation and ROC-AUC for predicting whether the reviewer changed their rating post-rebuttal.



(a) Checklist pass rate vs. reviewer rating on review–rebuttal pairs; dashed lines show linear fits.

(b) All four generators assign higher pass rates to rebuttals for accepted papers vs. rejected papers.

Figure 4.9: Author rebuttal checklists generated from ICLR 2019 reviews.

Table 4.5 reveals the patterns across generator types. DIRECT shows the strongest acceptance discrimination and moderate rating correlation. CONTRASTIVE shows similar patterns with slightly lower effect sizes. DEDUCTIVE achieves the highest rating correlation, and both DEDUCTIVE and INDUCTIVE are the only methods with significant rating-change prediction, i.e., corpus-level checklists may better capture the signals in rebuttals that convince reviewers to update their score.

This case study illustrates two features of the library’s design. First, the custom prompt API allows adapting any pipeline to a new domain without modifying library code, as the existing generator and scorer infrastructure is reusable with custom prompts. Second, the composable pipeline makes it straightforward to compare instance-level and corpus-level approaches on the same data, revealing their complementary strengths.

4.7 Conclusion and Future Work

We present `AutoChecklist`, a composable Python library that unifies LLM-based checklist evaluation. Beyond consolidating existing methods, the library contributes a taxonomy of five generator abstractions that organizes the design space of checklist generation by reasoning strategy, as well as a unified `ChecklistScorer` that consolidates three scoring strategies. These utilities, combined with the composable *Generator* $\rightarrow$ *Refiner* $\rightarrow$ *Scorer* architecture, enable systematic comparison and rapid extension of checklist evaluation methods. Overall, `AutoChecklist` is a practical contribution around aggregating methods from original, specific tasks into a resource that supports further exploration and extensions into new domains.

Validation on RewardBench and SummEval confirms that library-generated checklists align with human judgments. The peer review rebuttal case study (§4.6) demonstrates that adapting to a new domain requires only new custom prompt templates and no library code changes. Future directions include integrating new checklist methods as they emerge, supporting human-in-the-loop generation and refinement for corpus-level checklists, and providing more library features through the UI.

Limitations. We aim to faithfully implement the built-in pipelines based on their papers and code, but we make some changes for standardization (e.g., enforcing JSON structured outputs) and generalization (e.g., modifying original prompts). We provide default prompts calibrated to the original papers, but users adapting to new domains should expect to iterate

on prompt templates. Using OpenAI or OpenRouter providers also incurs API costs that users should account for. Finally, there is not yet full end-to-end support for corpus-level pipelines that require multi-stage human-in-the-loop workflows (e.g., INDUCTIVE’s refinement pipeline and an interface to collect human and LLM data for INTERACTIVE’s input attributes).

Ethical Considerations Automatic checklist-based evaluation is imperfect, especially with multiple LLM-based steps. We caution that checklist scores should not replace human judgment in high-stakes settings without careful validation.

The library is released under the Apache 2.0 license. Built-in pipelines are adapted based on publicly available papers and code. Validation experiments use open-source benchmarks (RewardBench, SummEval) and publicly accessible OpenReview data. No private or personally identifiable data is used. Users do not expose API keys or data through using the package or UI.

CHAPTER 5

CIVICCHATS: DESIGNING AI ASSISTANTS FOR CIVIC DECISION-MAKING

5.1 Introduction

In the United States, ballot measures—proposed laws, policies, or constitutional amendments up for public vote—are common and instrumental to democracy. Questions about housing, criminal justice, labor rights, and changes to state constitutions are put to voters to decide directly. However, political issues and ballot measures can be confusing, obscuring the effects of one outcome versus another, and most voters aren’t especially familiar with the measures they encounter. Ballot measures with more complex language lead to more roll-off (i.e., not voting), indicating that accessibility is a key factor in democratic participation [Reilly and Richey, 2011]. Consequently, voters often vote based on informational shortcuts, status quo bias, or abstain entirely [Lupia, 1994, Hessami and Resnjanskij, 2019].

People are also frequently unfamiliar with upcoming ballot measures, even on high-profile issues or days before the election, though political environment can influence awareness [Barth et al., 2020, Nicholson, 2003]. When voters have minimal prior knowledge about a ballot initiative, they are more likely to vote “yes” when it is written in simple, easy-to-understand language; conversely, they are more likely to vote “no” or not vote at all when it is written in complex language [Shulman et al., 2022].

Political knowledge increases one’s likelihood of political participation and voting Reichert [2016]. Technology can help surface information and support accessibility, but not without its own challenges. Chat interfaces with large language models (LLMs) have been increasingly used for persuasion and deliberation [Costello et al., 2024, 2025], as well as for information access in voting advice applications [Zhu et al., 2025]. While chatbots have been used to lower ballot measure information barriers [Ash et al., 2025], more deliberative interactions have not

been explored for this domain. A large-scale study assessing the effects of deliberation and increased political knowledge on vote choice has found that participating in a deliberative event is related to an increased likelihood of vote switching, though changes in party choices are not shown to be related to the observed increase in political knowledge [Bernhagen and Schmitt, 2014].

As mainstream chatbots tend to match user beliefs rather than clarifying or challenging them, i.e., sycophancy [Sharma et al., 2024, Cheng et al., 2025], we aim to develop a platform that supports more meaningful deliberation. In this work, we develop the CivicChats platform that hosts three chatbot personas for different modes of deliberation: a Q&A mode for understanding what a measure does and what’s at stake, an argumentative mode that presents views competing with the user’s own, and a reflective mode that helps users examine and develop their thinking. Through controlled user studies, we evaluate the impact of these conversation modes on voter engagement, decisions, and reasoning around ballot measures.

While we collect categorical vote choices and Likert-scale ratings, we also collect free-form responses from participants to explain their vote decision. To measure the quality of user reasoning from the free-form responses, we construct the following methods of evaluation: an Elo-style rating system for comparing impact of treatment conditions on user rationales, a tradeoff resolution measure for assessing the ability to identify and resolve tradeoffs in the rationale, and a transcript overlap measure for capturing what users take away from chatbot interactions.

When deployed in two user studies on 2026 ballot measures in Colorado and Florida ($N = 600$ total), we find that significant differences arise depending on the form of the ballot context and mode of AI interaction provided to the users, specifically around engagement, vote distributions, and user-written vote justifications.

5.2 Methods

This work has received IRB approval (IRB25-038) from the University of Chicago. Informed consent is obtained from participants before each study begins. We exclude participants for inattentiveness: with an explicit check before beginning the survey and through filtering on response time (less than 2 minutes total duration). All data is collected on a secured departmental computer server.

5.2.1 *Study design*

We design a between-subjects study with five treatment conditions: **Official_Text**, **Ballotpedia**, and the three chatbot conditions of **QA**, **Argue**, and **Reflect**. In the online user study, all participants are randomly assigned to a condition for a given ballot measure:

1. **Official_Text**: users only see the ballot measure’s official text, which serves as a control.
2. **Ballotpedia**: users are additionally provided sections of the human-curated Ballotpedia entry for the ballot measure, as a static, structured resource.
3. **QA**: along with the Ballotpedia sections, users are provided an interactive chatbot with RAG and web search capabilities, designed to be neutral and factual. Retrieval database includes Ballotpedia data as well as the full text of the measure.
4. **Argue**: includes the UI and capabilities of **QA**, plus the ability to argue for and against the measure, as push back on the user’s provided stance.
5. **Reflect**: includes the UI and capabilities of **QA**, plus the ability to ask probing questions about the user’s provided stance.

The treatment conditions are designed with progressive disclosure, starting with the **Official_Text** condition and gradually adding more content and/or AI capabilities.

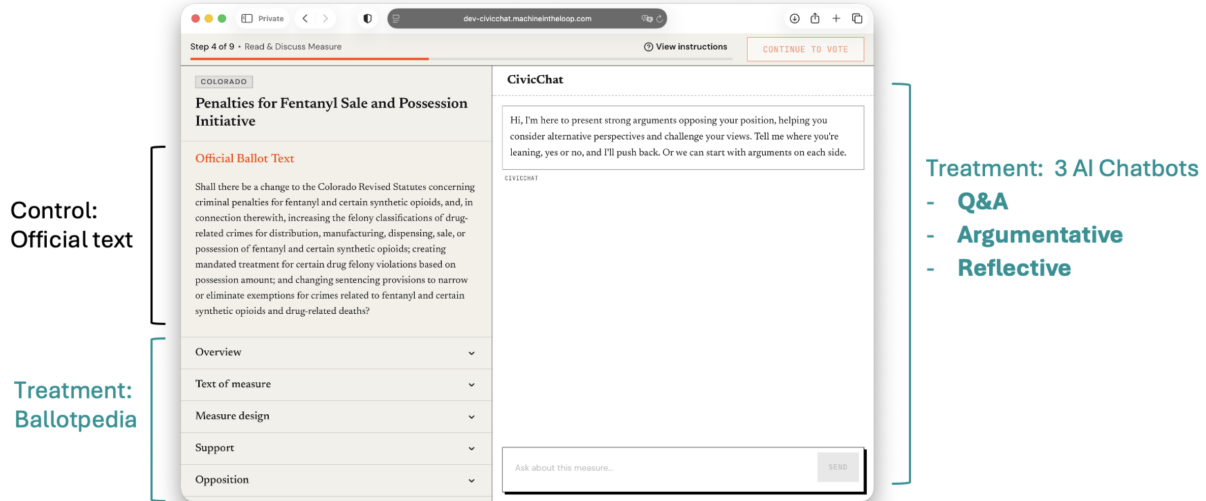


Figure 5.1: UI for treatment conditions. Control group sees only the official text of the ballot measure. Ballotpedia sees the official text and Ballotpedia sections. Chatbot conditions see the official text and Ballotpedia sections, plus the chatbot interface.

5.2.2 Experiment Flow

The entry point is a landing page that explains the study, provides the consent form (confidentiality and risks), and outlines the compensation of \$4 for completing the study (estimated to take approximately 15 minutes). Upon agreement, participants are given an overview of ballot measures and the study structure. Following this page is an explicit attention check, with a multiple choice question to confirm understanding of the purpose of the study. Users must answer correctly to proceed to the next page.

Next, the users are asked pre-treatment questions: “How familiar are you with this ballot measure?” and “How likely are you to vote on this ballot measure in the upcoming election?” with five-point Likert scale responses.

Study flow diverges here depending on the treatment condition. For the **Official_Text** and **Ballotpedia** conditions, their ballot measure content is presented on the left side of the screen. For the **QA**, **Argue**, and **Reflect** conditions, the ballot measure content is on the left and the chatbot interface is on the right. Users must send at least five messages to the

chatbot to proceed to the next page. The UI for the treatment pages is shown in Figure 5.1.

The post-treatment questions are presented in the right side of the ballot info, including: vote choice (Yes/No/Undecided), confidence in vote and understanding of measure (five-point Likert scale), and voting intent (five-point likelihood Likert scale). All conditions are then asked for a free-form explanation of their vote decision: “Please explain your reasoning for your voting decision. Try your best to address opposing viewpoints or arguments”. Users must write at least 30 words and spend at least 2 minutes of active time on the page (see Figure 5.2).

Following those steps, users are asked to provide an email address for the follow-up survey, as well as some demographic questions around education level, political affiliation, voting frequency, trust in AI, and media habits. Finally, users are asked to provide general feedback and, if in a chat condition, share if they would be interested in using such a tool in the future.

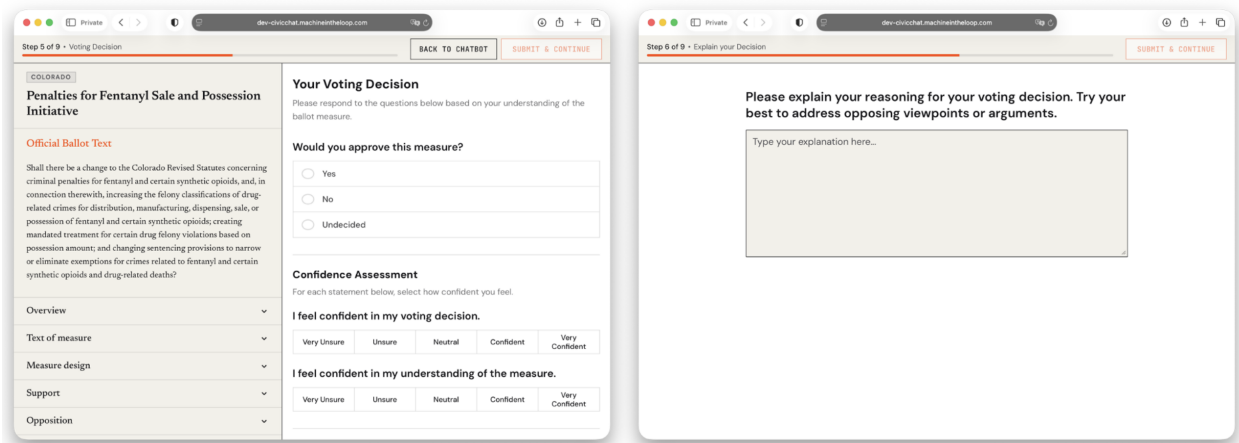


Figure 5.2: UI for post-treatment questions.

The format of mixing multiple-choice and free-form responses balances the standardization and subjective quality, a methodology supported by prior work in survey design and cognitive assessment [Conrad et al., 2025]. We measure participants’ voting confidence and understanding through their self-ratings and grading of their free-responses. We also track behavioral engagement signals like time spent on the survey.

FLORIDA

Changes to Budget Stabilization Fund Amendment

Official Ballot Text

CONSTITUTIONAL AMENDMENT

ARTICLE III, SECTION 19

BUDGET STABILIZATION FUND.—Proposing an amendment to the State Constitution to increase the amount of funds that may be retained in the budget stabilization fund from 10% to 25% of general revenue collections, require the legislature to transfer the lesser of \$750 million or the amount required to reach 25% of the general revenue collections each year unless certain conditions are met, and allow the legislature to withdraw funds for critical state needs.

Overview

Text of measure

Measure design

Support

Rate the following explanations. Participants were asked: "Please explain your reasoning for your voting decision. Try your best to address opposing viewpoints or arguments."

364/1000 (done by me: 174, anchors 30/30)

Only pick "Both are X" if they are truly indistinguishable; pick whichever is relatively better, even if both are generally good/bad.

YES

pair-00365

EXPLANATION A

I would vote in favor of it because it sets aside money in case there is another major recession like it said that Florida was not ready for the 2000 one. The opponents say that it's too much and more has to be done first, but the Republicans never help people they are just there to hurt people and punch down with their green, racism, bigotry and hatred.

So all in all it seems it would be good and if needed we would have that money set aside when we need it

On the other hand there are so many crooks in Republican politics that they may pilfer current money since there is money to fall back on.

EXPLANATION B

I was opposed more in the beginning since it is supported by Republicans. But the bot helped me come to a conclusion that I cannot trust legislature to do the right thing without some rules. Also, being that we are a hurricane state it makes sense to have that reserve. Also having a set amount of years instead of yearly gives stabilization.

PREFERENCE

1 / 2 / 3 / 4

1 A IS BETTER

2 BOTH GOOD

3 BOTH BAD

4 B IS BETTER

Figure 5.3: Pairwise annotation UI for rationales, inspired by Chatbot Arena, to develop Elo-style ratings of different study conditions.

5.2.3 Evaluating Vote Rationales

Vote rationales are free-form responses written by participants to explain their vote decision. To determine how good a rationale is and the treatment conditions' impact on them, we develop the following measures:

- **Elo-style rating:** relative rankings based on human preferences of treatment conditions with respect to the quality of user rationales.
- **Tradeoff resolution:** rate of resolving identified tradeoffs in the rationale, where a *tradeoff* is a passage where the participant weighs two competing considerations and resolves or accepts the tension.
- **Transcript takeaways:** rate of rationale content words that appear in the transcript, from the user and/or the chatbot.

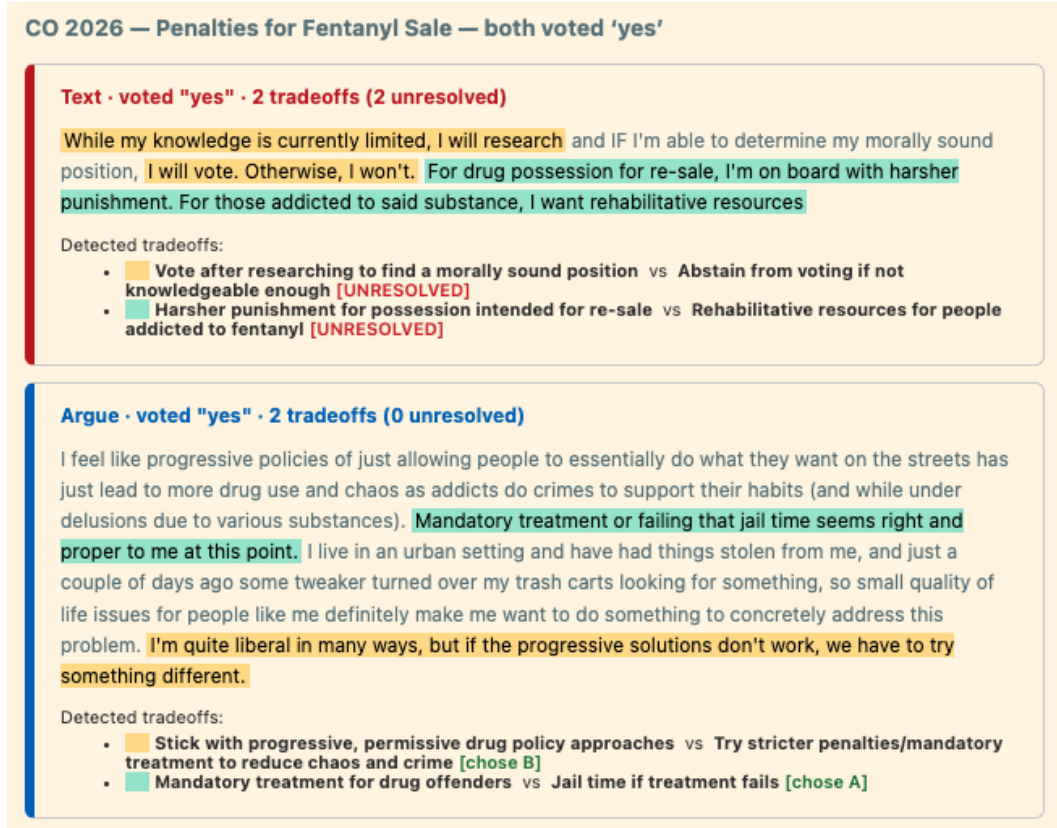


Figure 5.4: Tradeoff examples as identified with an LLM tagger, displaying tradeoffs (A and B) that are resolved (chose A or B) vs. not resolved.

Elo-style rating We use the Elo-style rating system to compare the rationales of different treatment conditions. In chess, the Elo rating system is a method for calculating the relative skill levels of players, that adapts to outcomes of matches. It is designed so that the probability of player A beating player B depends only on the gap between their Elo scores.

Elo scoring has been adopted for use in comparing LLMs in leaderboards like Chatbot Arena [Chiang et al., 2024], where human users indicate the winning response to the same input from a pair of LLM responses. Specifically, users are asked to choose one of: “A is better”, “Both are good”, “Both are bad”, and “B is better”.

By establishing treatment conditions as “players” and rationales as “matches”, we adapt this head-to-head setup to measure relative strengths of the treatments on rationale quality. We set up pairs of rationales that have the same vote direction on the same ballot measure,

with word count ratios under 2.0 to account for extreme length differences. Pairs are primarily across all five treatment conditions, with some pairs within treatment conditions to measure intra-treatment variation.

Like Chatbot Arena, we give half credit to ties. Three annotators all score a validation set of 30 pairs to measure inter-annotator agreement. After annotating enough pairs to ensure statistical power in a UI (shown in Figure 5.3), we fit a Bradley-Terry model (the underlying Maximum Likelihood Estimation model of Elo scoring) on the preference labels. The scores are rescaled to 1500 for the `Official_Text` control condition for easier interpretability, as that is the default score for new chess players.

Tradeoff resolution Part of explanation quality for a ballot measure vote also involves deliberation of tradeoffs. Tradeoffs may be within-stance (e.g., voting yes despite worrying about cost) or between-stance (weighing an opposing argument). We prompt an LLM to identify these semantically rather than by lexical markers like “but” or “on the other hand.”

An LLM scores each participant’s rationale with strict JSON-schema output for distinct tradeoffs—passages where the participant weighs two or more considerations against each other. For each tradeoff the LLM returns two competing sides A and B, and a resolution: user chose A, B, or neither, with some examples seen here in Figure 5.4. See §C.1 for the tradeoff identification system prompt, user prompt, and structured response format.

Transcript takeaways To further isolate effects of each kind of treatment on voter explanations, we also try to measure what they take away from their AI interactions. We measure this through content word overlap of rationales with users’ own messages, with AI messages, and with AI-unique words (those said only by the AI, not by the user or in the ballot text). Explanations are preprocessed by lowercasing and removing stopwords and punctuation.

The measure **Penalties for Fentanyl Sale and Possession Initiative** is on the ballot in **Colorado** on **November 3, 2026**.

Ballot Title: *Shall there be a change to the Colorado Revised Statutes concerning criminal penalties for fentanyl and certain synthetic opioids, and, in connection therewith, increasing the felony classifications of drug-related crimes for distribution, manufacturing, dispensing, sale, or possession of fentanyl and certain synthetic opioids; creating mandated treatment for certain drug felony violations based on possession amount; and changing sentencing provisions to narrow or eliminate exemptions for crimes related to fentanyl and certain synthetic opioids and drug-related deaths?* [Show less](#)

(a) Colorado.

The measure **Changes to Budget Stabilization Fund Amendment** is on the ballot in **Florida** on **November 3, 2026**.

Ballot Title: *CONSTITUTIONAL AMENDMENT*

ARTICLE III, SECTION 19

BUDGET STABILIZATION FUND.—Proposing an amendment to the State Constitution to increase the amount of funds that may be retained in the budget stabilization fund from 10% to 25% of general revenue collections, require the legislature to transfer the lesser of \$750 million or the amount required to reach 25% of the general revenue collections each year unless certain conditions are met, and allow the legislature to withdraw funds for critical state needs. [Show less](#)

(b) Florida.

Figure 5.5: Official texts of 2026 ballot measures for Colorado and Florida used in user studies.

5.2.4 Participants and Deployment

We run the study on two live ballot measures to appear on the November 2026 ballot: Colorado’s Penalty for Fentanyl Sales and Possession Initiative and Florida’s Changes to Budget Stabilization Fund Amendment (Figure 5.5). Both are selected to cover different topics and content. The Colorado measure aims to increase penalties for fentanyl-related offenses and require treatment for certain felony possession cases. The Florida amendment aims to increase the state’s "rainy-day" fund cap and require regular deposits, with flexibility to pause them in certain situations. Participants are recruited via Prolific and selected to

be residents of the US state for which the ballot measure applies.

5.3 Results

We recruit $N = 200$ participants from Colorado and $N = 400$ participants from Florida. Mean age of users is 40.2 (SD = 14.2), with 52.8% female and 47.0% male. 74.5% of participants are white, 9.5% are Black, 5.2% are Asian, 7.4% are mixed race, and 3.3% are other or prefer not to say. Participants complete the survey in an average of 15.3 minutes (SD = 10.3).

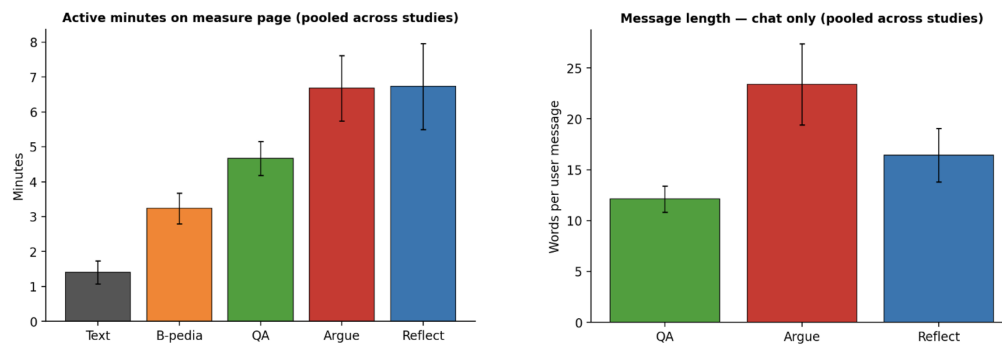


Figure 5.6: Engagement metrics for different treatment conditions, measured as active minutes on treatment page (excluding AI response time) and user chat message lengths.

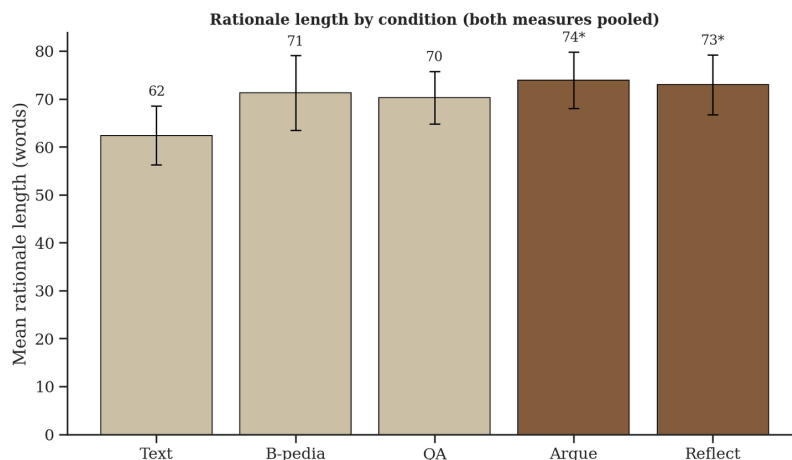


Figure 5.7: Lengths of vote rationales from different study conditions, with darker bars indicating significantly longer lengths than the control.

5.3.1 Engagement

We measure engagement through time spent on treatment pages (excluding AI response time), message lengths, and rationale lengths. In Figure 5.6, participants in **Ballotpedia** spend more time than the **Official_Text** condition, which is expected given the additional content provided. The AI conditions spend more time than both **Official_Text** and **Ballotpedia** users. Among the AI conditions, participants in the **Reflect** and **Argue** conditions spend more time conversing with the chatbot than participants in the **QA** condition. Additionally, participants in the **Reflect** and **Argue** conditions write longer messages than participants in the **QA** condition. We also see that **Reflect** and **Argue** conditions produce longer rationales that are significantly longer than the **Official_Text** control group, though not against other treatment conditions (Figure 5.7). So although the **Reflect** and **Argue** conditions spend the most time in the treatment page, that does not translate to longer rationales than the ones from the other treatment conditions.

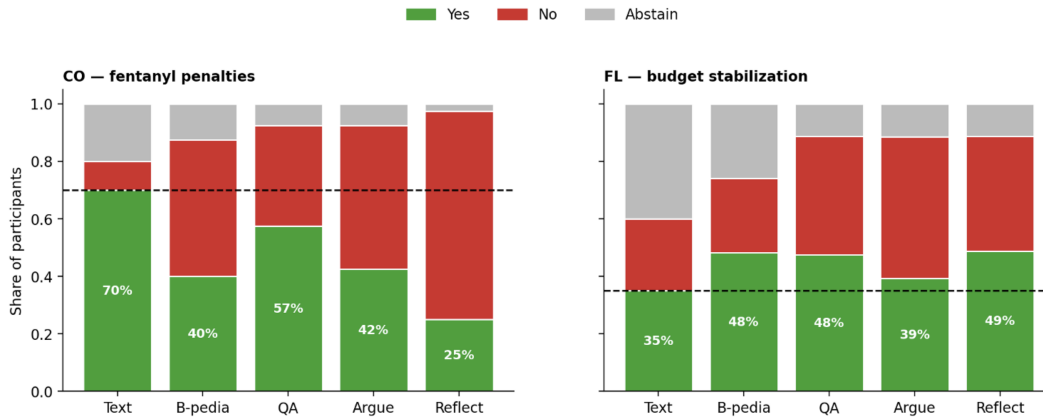


Figure 5.8: Vote distribution for different treatment conditions, for each ballot measure.

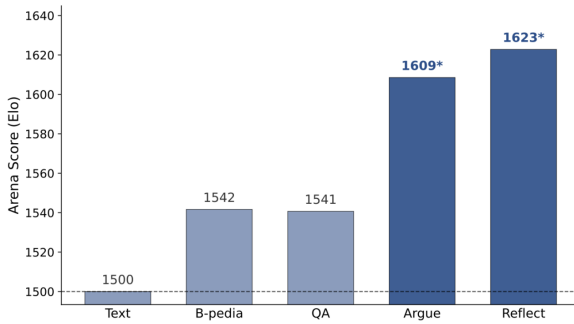
5.3.2 Voting Behavior

Voting “Yes” or “No” have different implications depending on ballot context, but the **Official_Text** condition distributions may be reflective of how users in the wild would vote on election day

if they do not research beforehand. We expect that the treatment conditions will lead to different vote distributions compared to the **Official_Text** control, as users are given more context. We do see this play out across both state measures (Figure 5.8).

In both states, exposure to treatments reduces the abstention rate (by 28% in Florida and 12% in Colorado), suggesting that the interventions help people be more decisive. **Ballotpedia** drops Colorado’s approval rate by 30% but raises Florida’s by 13%. As ballot measures are decided by majority vote (excluding abstentions), these different vote distributions would indicate different election outcomes, highlighting the potential real-world impact of these interventions.

5.3.3 Quality of Vote Rationale



(a) Elo scores for each condition, relative to the **Official_Text** control.

	Text	B-pedia	QA	Argue	Reflect
Text	—	+27	+46	+120	+111
B-pedia	-27	—	+19	+93	+84
QA	-46	-19	—	+74	+65
Argue	-120	-93	-74	—	-9
Reflect	-111	-84	-65	+9	—

Column – Row Elo gap (green = column significantly higher than row, red = column lower, grey = not significant)

(b) Significance of Elo scores between each pair of treatment conditions.

Figure 5.9: Elo scores and corresponding significance for the five study conditions.

We hypothesize that users in the **Reflect** and **Argue** conditions will have higher quality vote rationales than participants in the **Official_Text**, **QA**, and **Ballotpedia** conditions. As noted before, **Reflect** and **Argue** conditions produce longer rationales that are significantly longer than the control. But verbosity is not necessarily a sign of quality, so we also measure quality through Elo-style ratings, tradeoff resolution, and transcript overlap as described in §5.2.3.

To obtain relative rankings, we annotate 424 total pairs of rationales across conditions, with moderate to substantial agreement on the validation set (Fleiss’s $\kappa = 0.539$). We confirm no significant position bias. We find that **Argue** and **Reflect** produce significantly preferred explanations over the control across both measures (Figure 5.9a). The interpretation of this score is that rationales from those users beat out the ones from **Official_Text** around 67% of the time, and beat out **Ballotpedia** and **QA** around 61% of the time. There is some significant correlation with length, but while **Argue** and **Reflect** rationale *lengths* are not different from **Ballotpedia** and **QA**, in Elo ratings they are significantly preferred against **Ballotpedia** (Figure 5.9b). **Argue** is also significantly preferred against **QA**. These scores indicate that human preferences do extend beyond verbosity, and that the content of the rationales do differ even between AI conditions.

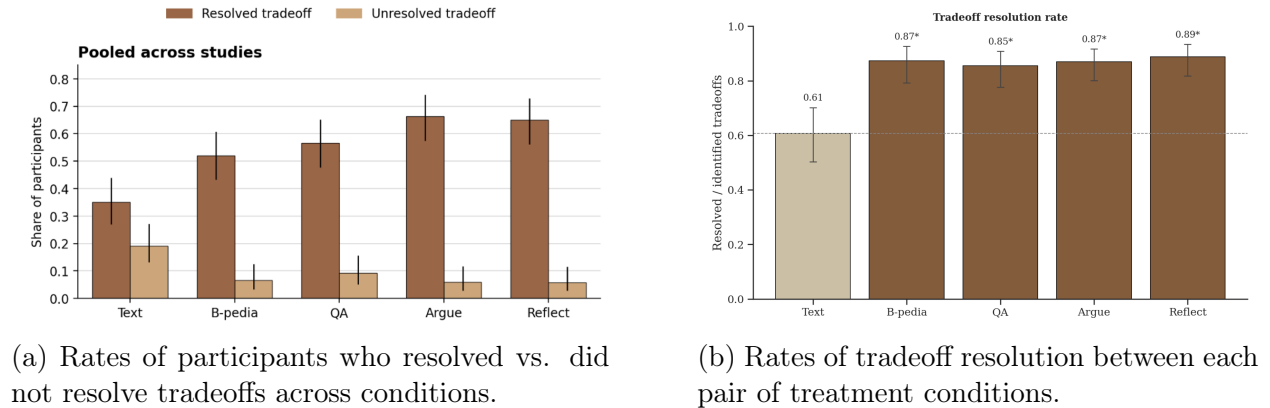


Figure 5.10: Rates of resolved vs. unresolved tradeoffs across conditions.

Another measure of quality involves deliberation of tradeoffs, which we measure through rate of tradeoff resolution. Treatment groups have a larger proportion of participants who resolve tradeoffs in their rationales than the **Official_Text** users (Figure 5.10a). **Official_Text** users have the highest rate of users who do not resolve any tradeoffs at all. At the identified tradeoff level, treatment groups’ rationales also have higher rates of resolution than those of **Official_Text** users (Figure 5.10b), which indicates that these users are engaging more with the nuances of the measure in their written explanations. Since these

users have access to the Support/Opposition arguments from Ballotpedia, and conditionally chatbots to synthesize and summarize such arguments, the higher rates of resolution suggest that the treatment users are able to be more decisive around conflicting considerations when justifying their vote.

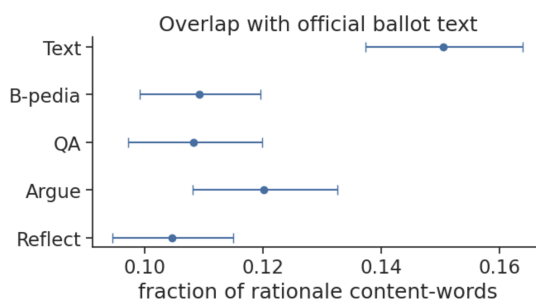


Figure 5.11: Overlap scores with official ballot text for different treatment conditions.

Finally, we further isolate the effects of each intervention on voter explanation by measuring what they take away from their specific treatments. As a baseline, we look at the extent to which rationales echo language from the official ballot text. From Figure 5.11, all treatment groups use significantly less official language than the `Official_Text` control (not unexpected given that they have access to extra context).

We then look at takeaways from the chat transcript for AI conditions. From Figure 5.12, `Argue` and `Reflect` users synthesize their own messages into their rationales, meaning a higher portion of their explanation is echoing or re-articulating their own chat messages. This is likely reflective of the chatbot personas themselves, which are designed specifically to elicit user stances. The `QA` condition not showing this outcome further supports that the more deliberative chatbots perform as intended. Another finding is that users in `Reflect` take up the most AI messaging, even when limited to AI unique words. This may also be due to the `Reflect` condition being designed to probe and surface unrealized assumptions.

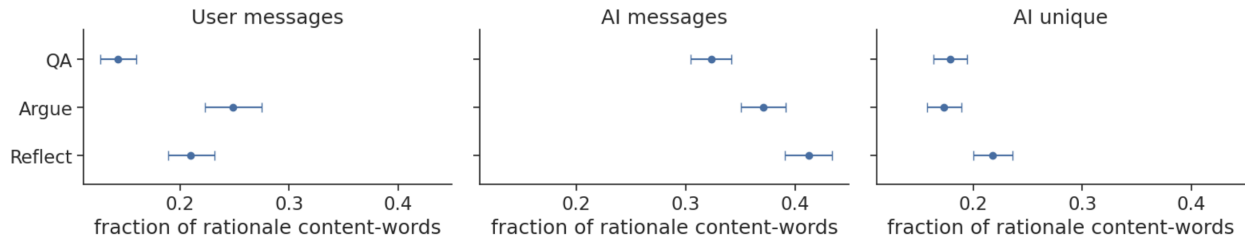


Figure 5.12: Overlap scores for different treatment conditions.

5.4 Conclusion

In this chapter, we conduct a user study to evaluate the effectiveness of an AI-powered chatbot in helping voters understand and make informed decisions about ballot measures. We design three chatbot personas, two of which are designed to elicit and engage directly with user stances. These chatbot treatments result in higher engagement through time spent and writing verbosity.

We find that treatment conditions lead to different vote distributions, with access beyond official ballot text leading to lower abstention rates. This supports previous findings that low information voters are more likely to roll off [Reilly and Richey, 2011], and the significant shifts in vote distribution suggest that these interventions could impact real ballot outcomes.

User-written justifications for their votes also differ across treatment conditions. In particular, the deliberation-oriented chatbots (**Argue** and **Reflect**) yield more preferred and nuanced explanations than the standard Q&A chatbot (**QA**), as measured by Elo-style ratings and tradeoff resolution. These two treatments also draw out more reasoning during the chat interaction itself, which users then carry over into their final rationale

A key methodological contribution is a set of measures for assessing free-form vote rationales beyond surface verbosity, which let us isolate how each conversation mode shapes voter reasoning. Informed and active voting is critical to the success of democracy. As AI chatbots become more prevalent in civic decision-making, our system and findings suggest that AI interactions, particularly those that elicit and engage directly with user stances, can

have significant and meaningful effects on voter behavior.

Future work. There are many opportunities for future work to support civic decision-making with AI. One avenue is to measure longer-term effects of the chatbot treatments on voter behavior, through follow-up surveys or longitudinal studies. Our platform at CivicChats.org (see §C.3) has collected over 900 chats, and future work involves analyzing these chat transcripts to understand how users engage with the chatbot personas in the wild. Additional discourse modes could be explored, such as multi-AI advisors or debate components to help participants assess the tradeoffs of a measure [Rahman et al., 2025]. There are also many opportunities to extend the evaluation methods we develop in this work, such as by defining or generating rubrics to grade the quality of user responses.

Limitations. Tradeoffs are identified through LLM-as-a-Judge, which has not been substantially validated by humans, though manual inspection does support the judge’s labels. The statistical power of the Elo ratings to further distinguish treatment conditions is also limited by the number of pairs annotated. An increase in the number of pairs would allow for more precise and sensitive comparisons.

CHAPTER 6

CONCLUSION

In this thesis, we have demonstrated various frameworks for measuring abstract qualities of language in human-centered domains, where subjective and contextual judgments may not have singular ground truth answers. Specifically, we have introduced a suite of metrics for quantifying the uniqueness and divisiveness of presidential speech, developed a framework for evaluating the quality of AI-generated clinical notes with feedback-grounded checklists, released an open-source library that unifies LLM-based checklist generation and scoring methods for rubric-based evaluation, and assessed the impact of human–AI conversation on voter engagement and reasoning around ballot measures. These frameworks span the high-stakes domains of politics, healthcare, and civic decision-making, where the problems we address have real-world consequences.

As discussed in Chapter 2, presidential discourse attracts considerable attention from popular media and scholars alike. In a time when efforts to computationally compare such rhetoric had been limited to lexical methods, we propose a novel suite of metrics to advance such analyses and identify new findings. In particular, we leverage large language models to establish an original metric of uniqueness, develop a new lexicon for divisive speech, and introduce a comparative framework for the portrayal of political opponents. We then apply these methods to a rich assembly of campaign speeches, presidential debates, and official States of the Union addresses. Across all these datasets, we find that Donald Trump’s political rhetoric is unique among modern presidents, and is defined, in part, by his use of antagonistic language, particularly when directed at political opponents. While this work was completed before the 2024 presidential election, the results remain relevant today as Trump continues to be a polarizing figure in American and global politics.

Established Ambient AI companies like Abridge provide hospitals with AI-generated clinical notes that are used for real patient care. The stakes of this domain are unquestionably

high, and the quality of the AI-generated notes is critical to not only the success of the company but also the well-being of patients. Since evaluating clinical note quality remains a challenge due to high subjectivity and limited scalability of expert review, we have developed a pipeline that systematically distills large-scale clinician feedback into structured, LLM-enforceable checklist items for note evaluation. Used with LLM-as-a-Judge, these checklists yield interpretable and auditable criteria, grounded in human feedback and enforceable by LLM-based evaluators. Using deidentified data from over 21,000 clinical encounters from a deployed AI medical scribe system, we show that our feedback-derived checklist outperforms a baseline approach in offline evaluations on coverage, diversity, and predictive power for human ratings. Extensive experiments have shown the checklist’s robustness to quality-degrading perturbations, significant alignment with clinician preferences, and practical value as an evaluation methodology. Our checklist offers a practical tool with the potential to reduce clinician effort in reviewing notes, without replacing expert oversight.

The checklist generation pipeline from Chapter 3 is incorporated into **AutoChecklist**, a library that consolidates multiple approaches from the literature for automatic checklist generation and scoring. By defining a common taxonomy of checklist generation abstractions, this toolkit supports the extensibility of rubric-based evaluation across domains. Specifically, we show that existing methods fall under two types of checklists, instance-level and corpus-level, and that five distinct generation patterns can be used to implement ten built-in pipelines. We demonstrate that the built-in pipelines generate checklists that align with human preferences and quality ratings, and also show how the library supports straightforward domain adaptation to a novel task of peer review rebuttal evaluation. **AutoChecklist** has been released as a public resource for the community¹, to support adaptation of these methods to future applications and domains.

Finally, we assess the impact of human–AI conversation on voter engagement and reason-

1. available at <https://github.com/ChicagoHAI/AutoChecklist>

ing around ballot measures in Chapter 5. We develop chatbot personas for different modes of conversation, and conduct controlled user studies to evaluate their effects. Considerable differences in engagement and vote distributions arise across treatment groups, where the interventions reduce the abstention rate across two ballot measures from different states. We also find that deliberation-oriented chatbot conditions yield user-written rationales that are preferred under Elo-style ratings, decisive around identified tradeoffs, and re-articulated from their chat interactions. These findings suggest that AI interactions, particularly those that elicit and engage directly with user stances, can have significant and meaningful effects on voter behavior.

There are ample opportunities for future work in this area, as the need for evaluation of language outputs (from humans and LLMs) at scale continues to grow and evolve. Expert intuitions and taste continue to be difficult to operationalize, especially in specialized domains like scientific research [Tong et al., 2026, Ho et al., 2026]. Specifically around the pattern of developing checklists to define structured criteria, decomposing quality into atomic, verifiable properties is also a growing paradigm for inference-time model improvement [Wan et al., 2026]. Checklist-based methods have proven to be useful for several applications (as described in Chapter 4), but limitations have been identified to address: checklists can have mixed effectiveness for evaluation [Furuhashi et al., 2025] and model-generated rubrics lag behind human-annotated criteria on hard instances [Zhang et al., 2026]. Closing the discriminative gap is an active area of research [Sun et al., 2026].

To conclude, the studies in this dissertation demonstrate how abstract qualities of language can still be measured in the absence of ground truth in various human-centered domains, by being thoughtful and intentional about what and how we measure. By leveraging human domain expertise to scope the evaluation criteria, NLP tools and LLMs for scalability, and multi-faceted interpretations of quality to provide multiple points of evidence, we treat evaluation as its own nontrivial task, not just a downstream application. This thesis

documents a foundation for further exploration of how we can build AI systems that are more grounded, interpretable, and aligned with human values.

REFERENCES

- Solomon Amenyo, Maura R. Grossman, Daniel G. Brown, and Brendan Wylie-Toal. Assessment of ai-generated pediatric rehabilitation soap-note quality, 2025. URL <https://arxiv.org/abs/2503.15526>.
- Kurt Anderson. How to talk trump. *The Atlantic*, 2018.
- Angela Aristidou. AI benchmarks are broken. Here’s what we need instead. — technologyreview.com. <https://www.technologyreview.com/2026/03/31/1134833/ai-benchmark-s-are-broken-heres-what-we-need-instead/>, March 2026. [Accessed 08-06-2026].
- Rahul K Arora, Jason Wei, Rebecca Soskin Hicks, Preston Bowman, Joaquin Quiñonero-Candela, Foivos Tsimpourlas, Michael Sharman, Meghan Shah, Andrea Vallone, Alex Beutel, et al. Healthbench: Evaluating large language models towards improved human health. *arXiv preprint arXiv:2505.08775*, 2025.
- Elliott Ash, Sergio Galletta, and Giacomo Opocher. Ballotbot: Can ai chatbots lower voter-information barriers? *Available at SSRN*, 2025.
- Jay Barth, Craig M. Burnett, and Janine Parry. Direct democracy, educative effects, and the (mis) measurement of ballot measure awareness. *Political Behavior*, 42(4):pp. 1015–1034, 2020. ISSN 01909320, 15736687. URL <https://www.jstor.org/stable/48694279>.
- Patrick Bernhagen and Hermann Schmitt. Deliberation, political knowledge and vote choice: Results from an experiment with second-order elections. *European Union Politics*, 15(3): 352–371, 2014. doi:10.1177/1465116514531506. URL <https://doi.org/10.1177/1465116514531506>.
- Robert L. Brennan and Dale J. Prediger. Coefficient kappa: Some uses, misuses, and alternatives. *Educational and Psychological Measurement*, 41(3):687–699, 1981. ISSN 1552-3888. doi:10.1177/001316448104100307.
- Penelope Brown and Stephen C Levinson. *Politeness: Some universals in language usage*, volume 4. Cambridge university press, 1987.
- Bill Brykczynski. A survey of software inspection checklists. *SIGSOFT Softw. Eng. Notes*, 24(1):82, January 1999. ISSN 0163-5948. doi:10.1145/308769.308798. URL <https://doi.org/10.1145/308769.308798>.
- Harry B Burke, Albert Hoang, Dorothy Becher, Paul Fontelo, Fang Liu, Mark Stephens, Louis N Pangaro, Laura L Sessums, Patrick O’Malley, Nancy S Baxi, Christopher W Bunt, Vincent F Capaldi, Julie M Chen, Barbara A Cooper, David A Djuric, Joshua A Hodge, Shawn Kane, Charles Magee, Zizette R Makary, Renee M Mallory, Thomas Miller, Adam Saperstein, Jessica Servey, and Ronald W Gimbel. Qnote: an instrument for measuring the quality of ehr clinical notes. *Journal of the American Medical Informatics Association*, 21(5):910–916, 01 2014. ISSN 1067-5027. doi:10.1136/amiajnl-2013-002321. URL <https://doi.org/10.1136/amiajnl-2013-002321>.

- Alex Chaparro, Joseph Roland Keebler, Elizabeth H. Lazzara, and Anastasia Diamond. Checklists: A review of their origins, benefits, and current uses as a cognitive aid in medicine. *Ergonomics in Design: The Quarterly of Human Factors Applications*, 27:21 – 26, 2019. URL <https://journals.sagepub.com/doi/epdf/10.1177/1064804618819181>.
- Stanley F Chen, Douglas Beeferman, and Roni Rosenfeld. Evaluation metrics for language models, 1998.
- Myra Cheng, Sunny Yu, Cinoo Lee, Pranav Khadpe, Lujain Ibrahim, and Dan Jurafsky. Elephant: Measuring and understanding social sycophancy in llms. *arXiv preprint arXiv:2505.13995*, 2025.
- Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Hao Zhang, Banghua Zhu, Michael Jordan, Joseph E. Gonzalez, and Ion Stoica. Chatbot arena: An open platform for evaluating llms by human preference, 2024. URL <https://arxiv.org/abs/2403.04132>.
- Seong Yeub Chu, Jong Woo Kim, and Mun Yong Yi. Think together and work better: Combining humans’ and llms’ think-aloud outcomes for effective text evaluation. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, pages 1–23, 2025. doi:10.1145/3706598.3713181. URL <https://dl.acm.org/doi/10.1145/3706598.3713181>.
- Ipek Cinar, Susan Stokes, and Andres Uribe. Presidential rhetoric and populism. *Presidential Studies Quarterly*, 50(2):240–264, 2020.
- Davide Colla, Matteo Delsanto, Marco Agosto, Benedetto Vitiello, and Daniele P. Radicioni. Semantic coherence markers: The contribution of perplexity metrics. *Artificial Intelligence in Medicine*, 134:102393, 2022. ISSN 0933-3657. doi:<https://doi.org/10.1016/j.artmed.2022.102393>. URL <https://www.sciencedirect.com/science/article/pii/S0933365722001440>.
- Frederick G. Conrad, Rachel E. Davis, Carolyn Lau, Melissa Armendáriz, Stephanie Morales, Timothy P. Johnson, and Johnny Blair. Probing in cognitive interviews can promote acquiescence. *Methods, Data, Analyses*, 2025. doi:10.12758/mda.2025.02. Online ISSN: 2190-4936.
- Jonathan Cook, Tim Rocktäschel, Jakob Foerster, Dennis Aumiller, and Alex Wang. Ticking all the boxes: Generated checklists improve llm evaluation and generation. *arXiv preprint arXiv:2410.03608*, 2024.
- Thomas H. Costello, Gordon Pennycook, and David G. Rand. Durably reducing conspiracy beliefs through dialogues with ai. *Science*, 385(6714):eadq1814, 2024. doi:10.1126/science.adq1814.

- Thomas H Costello, Gordon Pennycook, and David G Rand. Just the facts: How dialogues with ai reduce conspiracy beliefs, Feb 2025. URL osf.io/preprints/psyarxiv/h7n8u_v1.
- Emma Croxford, Yanjun Gao, Nicholas Pellegrino, Karen K. Wong, Graham Wills, Elliot First, Frank J Liao, Cherodeep Goswami, Brian W. Patterson, and Majid Afshar. Current and future state of evaluation of large language models for medical summarization tasks. *Npj health systems*, 2, 2025a. URL <https://api.semanticscholar.org/CorpusID:276140921>.
- Emma Croxford, Yanjun Gao, Nicholas Pellegrino, Karen K. Wong, Graham Wills, Elliot First, Miranda Schnier, Kyle Burton, Cris G. Ebby, Jillian Gorskic, Matthew Kalscheur, Samy Khalil, Marie Pisani, Tyler Rubeor, Peter Stetson, Frank Liao, Cherodeep Goswami, Brian Patterson, and Majid Afshar. Development and validation of the provider documentation summarization quality instrument for large language models, 2025b. URL <https://arxiv.org/abs/2501.08977>.
- Asaf Degani and Earl L. Wiener. Cockpit checklists: Concepts, design, and use. *Human Factors*, 35(2):345–359, 1993. doi:10.1177/001872089303500209. URL <https://doi.org/10.1177/001872089303500209>.
- Nachum Dershowitz and Rakesh M. Verma. Rebutting rebuttals. *Commun. ACM*, 66(9): 35–41, August 2023. ISSN 0001-0782. doi:10.1145/3584664. URL <https://doi.org/10.1145/3584664>.
- Jacob Dineen, Aswin Rrv, Qin Liu, Zhikun Xu, Xiao Ye, Ming Shen, Zhaonan Li, Shijie Lu, Chitta Baral, Muhao Chen, and Ben Zhou. QA-LIGN: Aligning LLMs through constitutionally decomposed QA. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng, editors, *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 20619–20642, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-335-7. doi:10.18653/v1/2025.findings-emnlp.1123. URL <https://aclanthology.org/2025.findings-emnlp.1123/>.
- Kathleen Eng, Katherine T Johnston, Ivo H. Cerda, Kushal T. Kadakia, Alison Mosier-Mills, and Anita Vanka. A patient-centered documentation skills curriculum for preclerkship medical students in an open notes era. *MedEdPORTAL : the Journal of Teaching and Learning Resources*, 20, 2024.
- Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. SummEval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409, 2021. doi:10.1162/tacl_a_00373. URL <https://aclanthology.org/2021.tacl-1.24/>.
- William Falcon and The PyTorch Lightning team. PyTorch Lightning, March 2019. URL <https://github.com/Lightning-AI/lightning>.

- Stephen J. Flusberg, Kevin J. Holmes, Paul H. Thibodeau, Robin L. Nabi, and Tee-nie Matlock. The psychology of framing: How everyday language shapes the way we think, feel, and act. *Psychological Science in the Public Interest*, 25(3):105–161, 2024. doi:10.1177/15291006241246966. URL <https://doi.org/10.1177/15291006241246966>. PMID: 39704149.
- Bill Frischling. Not “stable genius” again, or please stop making us run this analysis, Oct 2019. URL <https://blog.factba.se/2019/10/03/not-stable-genius-again-or-please-stop-making-us-run-this-analysis/>.
- Momoka Furuhashi, Kouta Nakayama, Takashi Kodama, and Saku Sugawara. Are checklists really useful for automatic evaluation of generative tasks? In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 10641–10664, 2025. URL <https://aclanthology.org/2025.emnlp-main.538/>.
- Yang Gao, Steffen Eger, Ines Hanselowski, and Iryna Gurevych. Does my rebuttal matter? insights from a major NLP conference. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1274–1290. Association for Computational Linguistics, 2019.
- Alexander Goldberg, Ihsan Ullah, Thanh Gia Hieu Khuong, Benedictus Kent Rachmat, Zhen Xu, Isabelle Guyon, and Nihar B Shah. Usefulness of llms as an author checklist assistant for scientific papers: Neurips’24 experiment. *arXiv preprint arXiv:2411.03417*, 2024.
- Yue Guo, Tal August, Gondy Leroy, Trevor Cohen, and Lucy Lu Wang. Appls: Evaluating evaluation metrics for plain language summarization, 2024. URL <https://arxiv.org/abs/2305.14341>.
- Janice L. Hanson, Mark B. Stephens, Louis N. Pangaro, and Ronald W. Gimbel. Quality of outpatient clinical notes: a stakeholder definition derived through qualitative research. *BMC Health Services Research*, 12:407 – 407, 2012.
- Yiding Hao, Simon Mendelsohn, Rachel Sterneck, Randi Martinez, and Robert Frank. Probabilistic predictions of people perusing: Evaluating metrics of language model performance for psycholinguistic modeling. In Emmanuele Chersoni, Cassandra Jacobs, Yohei Oseki, Laurent Prévot, and Enrico Santus, editors, *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 75–86, Online, November 2020. Association for Computational Linguistics. doi:10.18653/v1/2020.cmcl-1.10. URL <https://aclanthology.org/2020.cmcl-1.10/>.
- Roderick Hart. *Trump and Us: What He Says and Why People Listen*. Cambridge University Press, Cambridge, 2020.
- Alex B. Haynes, Thomas G. Weiser, William R. Berry, Stuart R. Lipsitz, Abdel-Hadi S. Breizat, E. Patchen Dellinger, Teodoro Herbosa, Sudhir Joseph, Pascience L. Kibatala, Marie Carmela M. Lapitan, Alan F. Merry, Krishna Moorthy, Richard K. Reznick, Bryce

- Taylor, and Atul A. Gawande. A surgical safety checklist to reduce morbidity and mortality in a global population. *New England Journal of Medicine*, 360(5):491–499, 2009. doi:10.1056/NEJMsa0810119. URL <https://www.nejm.org/doi/full/10.1056/NEJMsa0810119>.
- Xie He, Arash Habibi Lashkari, Nikhill Vombatkere, and Dilli Prasad Sharma. Authorship attribution methods, challenges, and future research directions: A comprehensive survey. *Information*, 15(3), 2024. ISSN 2078-2489. doi:10.3390/info15030131. URL <https://www.mdpi.com/2078-2489/15/3/131>.
- Zhitao He, Yiran Lyu, and Pascale Fung. Dancing in chains: Strategic persuasion in academic rebuttal via theory of mind. In *The Fourteenth International Conference on Learning Representations*, 2026. URL <https://arxiv.org/abs/2601.15715>.
- Zohal Hessami and Sven Resnjanskij. Complex ballot propositions, individual voting behavior, and status quo bias. *European Journal of Political Economy*, 58(C):82–101, None 2019. doi:10.1016/j.ejpoleco.2018.10.004. URL <https://ideas.repec.org/a/eee/poleco/v58y2019icp82-101.html>.
- Sy-Tuyen Ho, Minghui Liu, Huy Nghiem, and Furong Huang. Soundnessbench: Can your ai scientist really tell good research ideas from bad ones?, 2026. URL <https://arxiv.org/abs/2605.30329>.
- Chao-Chun Hsu, Ziad Obermeyer, and Chenhao Tan. A machine learning model using clinical notes to identify physician fatigue. *Nature Communications*, 16(1):5791, July 2025. ISSN 2041-1723. doi:10.1038/s41467-025-60865-4. URL <https://doi.org/10.1038/s41467-025-60865-4>.
- Baixiang Huang, Canyu Chen, and Kai Shu. Authorship attribution in the era of llms: Problems, methodologies, and challenges. *ACM SIGKDD Explorations Newsletter*, 26(2): 21–43, 2025.
- Kathleen Hall Jamieson and Doron Taussig. Disruption, demonization, deliverance, and norm destruction: The rhetorical signature of donald j. trump. *Political Science Quarterly*, 132(4):619–50, 2017.
- Kayla N. Jordan, Joanna Sterling, James W. Pennebaker, and Ryan L. Boyd. Examining long-term trends in politics and culture through language of political leaders and cultural institutions. *Proceedings of the National Academy of Sciences*, 116(9):3476–3481, 2019. doi:10.1073/pnas.1811987116. URL <https://www.pnas.org/doi/abs/10.1073/pnas.1811987116>.
- Praveen K Kanithi, Clément Christophe, Marco AF Pimentel, Tathagata Raha, Nada Saadi, Hamza Javed, Svetlana Maslenkova, Nasir Hayat, Ronnie Rajan, and Shadab Khan. Medic: Towards a comprehensive framework for evaluating llms in clinical applications, 2024. URL <https://arxiv.org/abs/2409.07314>.

- Orly Kayam. The readability and simplicity of donald trump’s language. *Political Studies Review*, 16(1):73–88, 2018. doi:10.1177/1478929917706844. URL <https://doi.org/10.1177/1478929917706844>.
- Robert Körner, Jennifer R. Overbeck, Erik Körner, and Astrid Schütz. How the linguistic styles of donald trump and joe Biden reflect different forms of power. *Journal of Language and Social Psychology*, 41(6):631–658, 2022. doi:10.1177/0261927X221085309. URL <https://doi.org/10.1177/0261927X221085309>.
- Danielle Kurtzleben. Why trump’s authoritarian language about “vermin” matters, Nov 2023. URL <https://www.npr.org/2023/11/17/1213746885/trump-vermin-hitler-immigration-authoritarian-republican-primary>.
- Ksenia Lagutina, Nadezhda Lagutina, Elena Boychuk, Inna Vorontsova, Elena Shliakhtina, Olga Belyaeva, Ilya Paramonov, and P.G. Demidov. A survey on stylometric text features. In *2019 25th Conference of Open Innovations Association (FRUCT)*, pages 184–195, 2019. doi:10.23919/FRUCT48121.2019.8981504.
- Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. RewardBench: Evaluating reward models for language modeling. In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 1755–1797, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-195-7. doi:10.18653/v1/2025.findings-naacl.96. URL <https://aclanthology.org/2025.findings-naacl.96/>.
- Yukyung Lee, JoongHoon Kim, Jaehee Kim, Hyowon Cho, Jaewook Kang, Pilsung Kang, and Najoung Kim. CheckEval: A reliable LLM-as-a-judge framework for evaluating text generation using checklists. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng, editors, *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, November 2025. ISBN 979-8-89176-332-6. doi:10.18653/v1/2025.emnlp-main.796. URL <https://aclanthology.org/2025.emnlp-main.796/>.
- Stephan Lewandowsky, Michael Jetter, and Ullrich K. H. Ecker. Using the president’s tweets to understand political diversion in the age of social media. *Nature Communications*, 11(1):5764, November 2020. ISSN 2041-1723. doi:10.1038/s41467-020-19644-6.
- Mingxuan Li, Hanchen Li, and Chenhao Tan. Hypoeval: Hypothesis-guided evaluation for natural language generation, 2025. URL <https://arxiv.org/abs/2504.07174>.
- Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. Textbooks are all you need ii: **phi-1.5** technical report. *arXiv preprint arXiv:2309.05463*, 2023.

- Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*, 2022.
- Bill Yuchen Lin, Yuntian Deng, Khyathi Chandu, Faeze Brahman, Abhilasha Ravichander, Valentina Pyatkin, Nouha Dziri, Ronan Le Bras, and Yejin Choi. Wildbench: Benchmarking llms with challenging tasks from real users in the wild, 2024. URL <https://arxiv.org/abs/2406.04770>.
- Tianci Liu, Ran Xu, Tony Yu, Ilgee Hong, Carl Yang, Tuo Zhao, and Haoyu Wang. Openrubrics: Towards scalable synthetic rubric generation for reward modeling and llm alignment, 2025. URL <https://arxiv.org/abs/2510.07743>.
- Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc., 2017. URL <http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf>.
- Arthur Lupia. Shortcuts versus encyclopedias: Information and voting behavior in california insurance reform elections. *American Political Science Review*, 88(1):63–76, March 1994. doi:None. URL https://ideas.repec.org/a/cup/apsrev/v88y1994i01p63-76_09.html.
- Qianli Ma, Chang Guo, Zhiheng Tian, Siyu Wang, Jipeng Xiao, Yuanhao Yue, and Zhipeng Zhang. Paper2rebuttal: A multi-agent framework for transparent author response assistance, 2026. URL <https://arxiv.org/abs/2601.14171>.
- Gábor Melis, Chris Dyer, and Phil Blunsom. On the state of the art of evaluation in neural language models. 2018.
- Saif Mohammad and Peter Turney. Emotions evoked by common words and phrases: Using Mechanical Turk to create an emotion lexicon. In Diana Inkpen and Carlo Strapparava, editors, *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 26–34, Los Angeles, CA, June 2010. Association for Computational Linguistics. URL <https://aclanthology.org/W10-0204>.
- Saif M. Mohammad and Peter D. Turney. Crowdsourcing a word-emotion association lexicon, 2013.
- Burt Monroe, Michael Colaresi, and Kevin Quinn. Fightin’ Words: Lexical Feature Selection and Evaluation for Identifying the Content of Political Conflict. *Political Analysis*, 16, August 2009. doi:10.1093/pan/mpn018.
- Shan Nan, Pieter van Gorp, Xudong Lu, Uzay Kaymak, Hendrikus H. M. Korsten, Richard Vdovjak, and Huilong Duan. A meta-model for computer executable dynamic clinical safety checklists. *BMC Medical Informatics and Decision Making*, 17, 2017. URL <https://bmcmmedinformdecismak.biomedcentral.com/articles/10.1186/s12911-017-0551-0>.

- Stephen P. Nicholson. The political environment and ballot proposition awareness. *American Journal of Political Science*, 47(3):403–410, 2003. doi:<https://doi.org.proxy.uchicago.edu/10.1111/1540-5907.00029>. URL <https://onlinelibrary-wiley-com.proxy.uchicago.edu/doi/abs/10.1111/1540-5907.00029>.
- Evangelos Ntontis, Klara Jurstakova, Fergus Neville, S. Alexander Haslam, and Stephen Reicher. A warrant for violence? an analysis of donald trump’s speech before the us capitol attack. *British Journal of Social Psychology*, n/a(n/a). doi:<https://doi.org/10.1111/bjso.12679>. URL <https://bpspsychub.onlinelibrary.wiley.com/doi/abs/10.1111/bjso.12679>.
- Jon Oleson. Deepscore: A comprehensive approach to measuring quality in ai-generated clinical documentation, 2024. URL <https://arxiv.org/abs/2409.16307>.
- Jasmine Chiat Ling Ong, Liyuan Jin, Kabilan Elangovan, Gilbert Yong San Lim, Daniel Yan Zheng Lim, Gerald Gui Ren Sng, Yu He Ke, Joshua Yi Min Tung, Ryan Jian Zhong, Christopher Ming Yao Koh, Keane Zhi Hao Lee, Xiang Chen, Jack Kian Ch’ng, Aung Than, Ken Junyang Goh, Chuan Poh Lim, Tat Ming Ng, Nan Liu, and Daniel Shu Wei Ting. Large language model as clinical decision support system augments medication safety in 16 clinical specialties. *Cell Reports Medicine*, 6(10):102323, 2025. ISSN 2666-3791. doi:<https://doi.org/10.1016/j.xcrm.2025.102323>. URL <https://www.sciencedirect.com/science/article/pii/S2666379125003969>.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi:10.3115/v1/D14-1162. URL <https://aclanthology.org/D14-1162>.
- Jayr Pereira, Andre Assumpcao, and Roberto Lotufo. Check-eval: A checklist-based approach for evaluating text quality, 2024. URL <https://arxiv.org/abs/2407.14467>.
- William D. Perreault and Laurence E. Leigh. Reliability of nominal data based on qualitative judgments. *Journal of Marketing Research*, 26(2):135–148, 1989. ISSN 00222437. URL <http://www.jstor.org/stable/3172601>.
- Yiwei Qin, Kaiqiang Song, Yebowen Hu, Wenlin Yao, Sangwoo Cho, Xiaoyang Wang, Xuansheng Wu, Fei Liu, Pengfei Liu, and Dong Yu. InFoBench: Evaluating instruction following ability in large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13025–13048, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi:10.18653/v1/2024.findings-acl.772. URL <https://aclanthology.org/2024.findings-acl.772/>.
- Haoran Que, Feiyu Duan, Liqun He, Yutao Mou, Wangchunshu Zhou, Jiaheng Liu, Wenge Rong, Zekun Moore Wang, Jian Yang, Ge Zhang, Junran Peng, Zhaoxiang Zhang,

- Songyang Zhang, and Kai Chen. Hellobench: Evaluating long text generation capabilities of large language models, 2024. URL <https://arxiv.org/abs/2409.16191>.
- Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training, 2018.
- Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners, 2019.
- Salman Rahman, Sheriff Issaka, Ashima Suvarna, Genglin Liu, James Shiffer, Jaeyoung Lee, Md Rizwan Parvez, Hamid Palangi, Shi Feng, Nanyun Peng, et al. Ai debate aids assessment of controversial claims. *arXiv preprint arXiv:2506.02175*, 2025.
- Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA.
- Frank Reichert. How internal political efficacy translates political knowledge into political participation. *Europe’s Journal of Psychology*, 12(2):221–241, May 2016. doi:10.5964/ejop.v12i2.1095.
- Hajo Alexander Reijers, Henrik Leopold, and Jan Recker. Towards a science of checklists. In *Hawaii International Conference on System Sciences*, 2017.
- Shauna Reilly and Sean Richey. Ballot question readability and roll-off: The impact of language complexity. *Political Research Quarterly*, 64(1):59–67, 2011. doi:10.1177/1065912909349629. URL <https://doi.org/10.1177/1065912909349629>.
- Hyun Ryu, Doohyuk Jang, Hyemin S. Lee, Joonhyun Jeong, Gyeongman Kim, Donghyeon Cho, Gyouk Chu, Minyeong Hwang, Hyeongwon Jang, Changhun Kim, Haechan Kim, Jina Kim, Joowon Kim, Yoonjeon Kim, Kwanhyung Lee, Chanjae Park, Heecheol Yun, Gregor Betz, and Eunho Yang. Reviewscore: Misinformed peer review detection with large language models, 2025. URL <https://arxiv.org/abs/2509.21679>.
- Abdelrahman Sadallah, Tim Baumgärtner, Iryna Gurevych, and Ted Briscoe. The good, the bad and the constructive: Automatically measuring peer review’s utility for authors. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng, editors, *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 28991–29021, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi:10.18653/v1/2025.emnlp-main.1476. URL <https://aclanthology.org/2025.emnlp-main.1476/>.
- Maarten Sap, Swabha Swayamdipta, Laura Vianna, Xuhui Zhou, Yejin Choi, and Noah A. Smith. Annotators with attitudes: How annotator beliefs and identities bias toxic language detection. In *North American Chapter of the Association for Computational Linguistics*, 2021. URL <https://api.semanticscholar.org/CorpusID:244117167>.

- Aleksandar Savkov, Francesco Moramarco, Alex Papadopoulos Korfiatis, Mark Perera, Anya Belz, and Ehud Reiter. Consultation checklists: Standardising the human evaluation of medical note generation. In Yunyao Li and Angeliki Lazaridou, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 111–120, Abu Dhabi, UAE, December 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.emnlp-industry.10. URL <https://aclanthology.org/2022.emnlp-industry.10/>.
- Mrinank Sharma, Meg Tong, Tomek Korbak, David Duvenaud, Amanda Asbell, Sam Bowman, Esin Durmus, Zac Hatfield-Dodds, Scott Johnston, Shauna Kravec, et al. Towards understanding sycophancy in language models. In *International Conference on Learning Representations*, volume 2024, pages 110–144, 2024.
- Hillary C. Shulman, Matthew D. Sweitzer, Olivia M. Bullock, Jason C. Coronel, Robert M. Bond, and Shannon Poulsen. Predicting vote choice and election outcomes from ballot wording: The role of processing fluency in low information direct democracy elections. *Political Communication*, 39(5):652–673, 2022. doi:10.1080/10584609.2022.2092920. URL <https://doi.org/10.1080/10584609.2022.2092920>.
- Almog Simchon, William J Brady, and Jay J Van Bavel. Troll and divide: the language of online polarization. *PNAS Nexus*, 1(1):pgac019, 03 2022. ISSN 2752-6542. doi:10.1093/pnasnexus/pgac019. URL <https://doi.org/10.1093/pnasnexus/pgac019>.
- Clemencia Siro, Pourya Aliannejadi, and Mohammad Aliannejadi. Learning to judge: LLMs designing and applying evaluation rubrics. In *Findings of the European Chapter of the Association for Computational Linguistics*, 2026. URL <https://arxiv.org/abs/2602.08672>.
- Peter D. Stetson, Frances P. Morrison, Suzanne Bakken, and Stephen B. Johnson. Preliminary development of the physician documentation quality instrument. *Journal of the American Medical Informatics Association*, 15(4):534–541, 2008. ISSN 1067-5027. doi:<https://doi.org/10.1197/jamia.M2404>. URL <https://www.sciencedirect.com/science/article/pii/S1067502708000728>.
- Peter D Stetson, Suzanne Bakken, Jesse O Wrenn, and Eugenia L Siegler. Assessing electronic note quality using the physician documentation quality instrument (pdqi-9). *Applied clinical informatics*, 3(02):164–174, 2012.
- Mengyuan Sun, Yu Li, Zhuohao Yu, Shikun Zhang, and Wei Ye. Support vector rubrics: Closing the gap between self-generated and human rubrics, 2026. URL <https://arxiv.org/abs/2606.08077>.
- Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

- Jingqi Tong, Mingzhe Li, Hangcheng Li, Yongzhuo Yang, Yurong Mou, Weijie Ma, Zhiheng Xi, Hongji Chen, Xiaoran Liu, Qinyuan Cheng, Ming Zhang, Qiguang Chen, Weifeng Ge, Qipeng Guo, Tianlei Ying, Tianxiang Sun, Yining Zheng, Xinchu Chen, Jun Zhao, Ning Ding, Xuanjing Huang, Yugang Jiang, and Xipeng Qiu. Ai can learn scientific taste, 2026. URL <https://arxiv.org/abs/2603.14473>.
- Jeffrey Tulis. *The Rhetorical Presidency*. Princeton University Press, 1987.
- Vijay Viswanathan, Yanchao Sun, Xiang Kong, Meng Cao, Graham Neubig, and Tongshuang Wu. Checklists are better than reward models for aligning language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL <https://openreview.net/forum?id=RPRqKhjrr6>.
- Yuxuan Wan, Tianqing Fang, Zaitang Li, Yintong Huo, Wenxuan Wang, Haitao Mi, Dong Yu, and Michael R. Lyu. Inference-time scaling of verification: Self-evolving deep research agents via test-time rubric-guided verification. *arXiv preprint arXiv:2601.15808*, 2026. URL <https://arxiv.org/abs/2601.15808>.
- Hanyin Wang, Chufan Gao, Bolun Liu, Qiping Xu, Guleid Hussein, Mohamad El Labban, Kingsley Iheasirim, Hariprasad Korsapati, Chuck Outcalt, and Jimeng Sun. Towards adapting open-source large language models for expert-level clinical note generation, 2025a. URL <https://arxiv.org/abs/2405.00715>.
- Hanyin Wang, Chufan Gao, Qiping Xu, Bolun Liu, Guleid Hussein, Hariprasad Korsapati, Mohamad El Labban, Kingsley Iheasirim, Mohamed Hassan, Gokhan Anil, Brian Bartlett, and Jimeng Sun. Process-supervised reward models for verifying clinical note generation: A scalable approach guided by domain expertise, 2025b. URL <https://arxiv.org/abs/2412.12583>.
- Zifeng Wang, Hanyin Wang, Benjamin Danek, Ying Li, Christina Mack, Hoifung Poon, Yajuan Wang, Pranav Rajpurkar, and Jimeng Sun. A perspective for adapting generalist ai to specialized medical ai applications and their challenges, 2024. URL <https://arxiv.org/abs/2411.00024>.
- Zihan Wang, Jingbo Shang, and Ruiqi Zhong. Goal-driven explainable clustering via language descriptions. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10626–10649, Singapore, December 2023. Association for Computational Linguistics. doi:10.18653/v1/2023.emnlp-main.657. URL <https://aclanthology.org/2023.emnlp-main.657>.
- Tianjun Wei, Wei Wen, Ruizhi Qiao, Xing Sun, and Jianghong Ma. Rocketeval: Efficient automated LLM evaluation via grading checklist. In *The Thirteenth International Conference on Learning Representations*, 2025. URL <https://openreview.net/forum?id=zJjzNj6QUe>.

- [dataset] John T Woolley and Gerhard Peters. The american presidency project. *Santa Barbara, CA. Available from World Wide Web: <http://www.presidency.ucsb.edu/ws>*, 1999.
- Ellery Wulczyn, Nithum Thain, and Lucas Dixon. Ex machina: Personal attacks seen at scale. In *Proceedings of the 26th international conference on world wide web*, pages 1391–1399, 2017.
- Yiqing Xie, Sheng Zhang, Hao Cheng, Pengfei Liu, Zelalem Gero, Cliff Wong, Tristan Naumann, Hoifung Poon, and Carolyn Rose. DocLens: Multi-aspect fine-grained evaluation for medical text generation. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 649–679, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi:10.18653/v1/2024.acl-long.39. URL <https://aclanthology.org/2024.acl-long.39/>.
- Zhichao Yang, Sepehr Janghorbani, Dongxu Zhang, Jun Han, Qian Qian, Andrew Ressler II, Gregory D. Lyng, Sanjit Singh Batra, and Robert E. Tillman. Health-SCORE: Towards scalable rubrics for improving health-LLMs. *arXiv preprint arXiv:2601.18706*, 2026. URL <https://arxiv.org/abs/2601.18706>.
- Jiayi Ye, Yanbo Wang, Yue Huang, Dongping Chen, Qihui Zhang, Nuno Moniz, Tian Gao, Werner Geyer, Chao Huang, Pin-Yu Chen, et al. Justice or prejudice? quantifying biases in llm-as-a-judge. *arXiv preprint arXiv:2410.02736*, 2024.
- Wen-wai Yim, Ashley Mills, Harold Chun, Teresa Hashiguchi, Justin Yew, and Bryan Lu. Automatic rubric-based content grading for clinical notes. In Eben Holderness, Antonio Jimeno Yepes, Alberto Lavelli, Anne-Lyse Minard, James Pustejovsky, and Fabio Rinaldi, editors, *Proceedings of the Tenth International Workshop on Health Text Mining and Information Analysis (LOUHI 2019)*, pages 126–135, Hong Kong, November 2019. Association for Computational Linguistics. doi:10.18653/v1/D19-6216. URL <https://aclanthology.org/D19-6216/>.
- Wen-wai Yim, Yujuan Fu, Asma Ben Abacha, Neal Snider, Thomas Lin, and Meliha Yetisgen. Aci-bench: a novel ambient clinical intelligence dataset for benchmarking automatic visit note generation. *Nature Scientific Data*, 2023.
- Haoran Zhang, Quaid Morris, Berk Ustun, and Maryzeh Ghassemi. Learning optimal predictive checklists. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS ’21, Red Hook, NY, USA, 2021. Curran Associates Inc. ISBN 9781713845393.
- Ming Zhang, Yujiong Shen, Zelin Li, Huayu Sha, Binze Hu, Yuhui Wang, Chenhao Huang, Shichun Liu, Jingqi Tong, Changhao Jiang, Mingxu Chai, Zhiheng Xi, Shihan Dou, Tao Gui, Qi Zhang, and Xuanjing Huang. Llmeval-med: A real-world clinical benchmark for medical llms with physician validation, 2025. URL <https://arxiv.org/abs/2506.04078>.

- Qiyuan Zhang, Junyi Zhou, Yufei Wang, Fuyuan Lyu, Yidong Ming, Can Xu, Qingfeng Sun, Kai Zheng, Peng Kang, Xue Liu, and Chen Ma. Rubricbench: Aligning model-generated rubrics with human standards, 2026. URL <https://arxiv.org/abs/2603.01562>.
- Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36: 46595–46623, 2023.
- Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyang Luo, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. *arXiv preprint arXiv:2403.13372*, 2024. URL <http://arxiv.org/abs/2403.13372>.
- Weifeng Zhong. The candidates in their own words: A textual analysis of 2016 president primary debates. 2016. URL <https://www.aei.org/wp-content/uploads/2016/04/The-candidates-in-their-own-words.pdf>.
- Karen Zhou, John Michael Giorgi, Pranav Mani, Peng Xu, Davis Liang, and Chenhao Tan. From feedback to checklists: Grounded evaluation of AI-generated clinical notes. In Saloni Potdar, Lina Rojas-Barahona, and Sebastien Montella, editors, *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*. Association for Computational Linguistics, November 2025. ISBN 979-8-89176-333-3. doi:10.18653/v1/2025.emnlp-industry.104. URL <https://aclanthology.org/2025.emnlp-industry.104/>.
- Jianlong Zhu, Manon Kempermann, Vikram Kamath Cannanure, Alexander Hartland, Rosa M. Navarrete, Giuseppe Carteny, Daniela Braun, and Ingmar Weber. Learn, explore and reflect by chatting: Understanding the value of an llm-based voting advice application chatbot. In *Proceedings of the 7th ACM Conference on Conversational User Interfaces*, CUI ’25. Association for Computing Machinery, 2025. doi:10.1145/3719160.3736611.

APPENDIX A

SUPPLEMENTARY MATERIAL FOR CHAPTER 3

A.1 LLM-based Uniqueness Robustness Checks

A.1.1 BPC

Recall that we employ bits-per-character (BPC), also known as bits-per-byte, as a measure of “predictability” of a sentence. Lower BPC values correspond to higher predictability. We use BPC instead of perplexity or loss directly to account for variation in tokenization techniques. BPC is the sum of cross-entropy losses of each token in a sentence, divided by the number of characters (bytes) in the sentences. Figure A.1 shows average sentence BPC per candidate (Figure A.1a) and aggregated by party over sentence length (Figure A.1b). While in campaigns, Trump is the most unpredictable among those candidates, he is ranks closer to the middle for predictability in debates and SOTU.

A.1.2 Correlation with Readability Metrics

We calculate the Spearman correlation coefficients for our uniqueness metric with existing readability scores. Our metric has little to no correlation with readability, while different readability indexes show strong correlation with each other (Figure A.2). Figure A.2 also confirms that our metric is distinct from sentence length.

A.1.3 Uniqueness scores over time

Figure A.3 shows the uniqueness scores over time; in particular, the scores increased in recent years. We also present the uniqueness scores by year/term for debates (Figure A.4), SOTU (Figure A.5), and campaigns (Figure A.6). Overall, Trump is still consistently the most unique every year and term, compared to those of the other presidential candidates. For

debates, he is slightly more unique in his first election cycle in 2016, whereas in campaigns, his second election cycle involves more distinctive speech than his first. For SOTU addresses, Trump’s speech becomes more distinct in his second year onwards.

A.1.4 Top %-ile uniqueness

Figure A.7 shows that, when comparing the uniqueness scores of the top decile of unique sentences for each speaker, we again find that Trump is the most unique speaker in all our samples of political speech.

A.1.5 LLM-based uniqueness results with unmasked model

In the main paper, we present results using language models that are trained on data with named entities masked out with a <ENT> token, as identified by the spaCy NER tagger. This decision is made to prevent the model from learning patterns like “Only Trump mentions Biden’s name during debates”.

We also fine-tune each dataset’s model on the data without masking named entities, with results presented in Figure A.8. These results are overall consistent with those of the masked model. Figure A.8a shows that Trump remains the most distinctive speaker in all data types, while Figure A.8b confirms still that the distinctiveness holds across all sentence lengths. Trump is still more similar in uniqueness to Democrats than his fellow Republicans (Figure A.8c).

A.2 Additional Divisive Word Lexicon Results

A.2.1 Divisive Word Lexicon

Table 2.2 contains the 178 words in our proposed divisive word lexicon.

A.2.2 Usage over time

In debates and campaigns, divisive language usage increased after 2012 (Figure A.9), which corresponds to the onset of Trump’s candidacy. Indeed, we find that Trump uses the most divisive language of all candidates.

A.2.3 Usage by candidate

Heatmaps of divisive word usage can provide a more granular look at the specific divisive language used by each candidate (Figure A.10). For example, in the debates dataset, it is notable that the frequency of use of the word “racist” has become used more by recent candidates like Trump and Biden. In the debates dataset some of Donald Trump’s most frequently used words in this lexicon are those like “disgrace”, “stupid”, “filthy”, “hate”, and “racist” (Figure A.10a). In Donald Trump’s SOTU addresses, the most frequently used divisive words are those like “corrupt”, “vile”, “foolish”, “cruel”, and “savage” (Figure A.10b). In campaign speeches, Donald Trump’s most frequently used divisive words are those like “corrupt”, “crazy”, “stupid”, and “dishonest” (Figure A.10c).

A.2.4 Utterances Using Divisive Words

Excerpts of speech using divisive words can be found in Table A.1, Table A.2, and Table A.3 for debates, SOTU, and campaign speeches respectively.

A.2.5 Annotator Agreement

For the initial 360 terms, we obtain a Fleiss’ κ of 0.54, which indicates a moderate level of agreement. Since divisiveness is a somewhat subjective phenomenon, this moderate agreement is similar to that obtained on other subjective lexicons, such as the word-emotion association (average Fleiss’ κ of 0.29) [Mohammad and Turney, 2013] and the initial terms

in the polarization dictionary (Cohen’s κ of 0.61) [Simchon et al., 2022]. Our final set of 178 terms has full agreement from the majority of annotators, but we still discuss the implications of our Fleiss’ κ below.

Agreement is affected by several characteristics of divisiveness. Firstly, divisiveness is a spectrum and terms can be “a little” to “extremely” divisive. Annotators only give binary labels, so disagreement may be higher due to lack of nuance. Relatedly, the words are presented out of context; they may be more or less divisive depending on usage. However, identifying all possible contexts of our lexicon usage is costly and nontrivial. Finally, like the word-emotion association lexicon [Mohammad and Turney, 2013], our initial data is biased, with more “not divisive” terms than “divisive” terms (based on only 31.6% of the 360 terms receiving unanimous agreement). Both Cohen’s and Fleiss’ κ are considered conservative in such settings with label imbalance [Mohammad and Turney, 2013, Brennan and Prediger, 1981, Perreault and Leigh, 1989]. As such, these κ values may underestimate the true rater agreement for these cases.

Ultimately, we introduce the first such lexicon of divisiveness for text analysis. Usage of our lexicon may be augmented with additional strategies like dependency parsing and leveraging surrounding context.

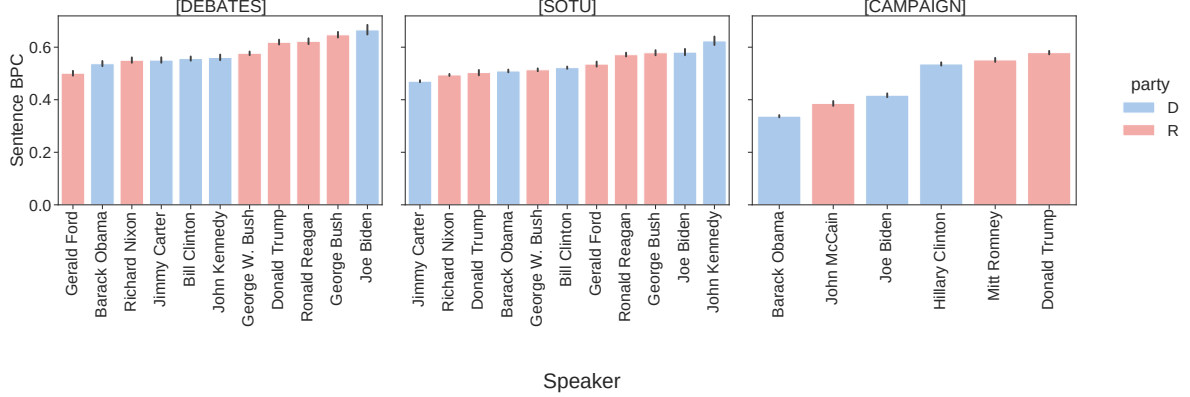
A.3 Additional Opponent Mention Results

A.3.1 *Rate of opponent mentions*

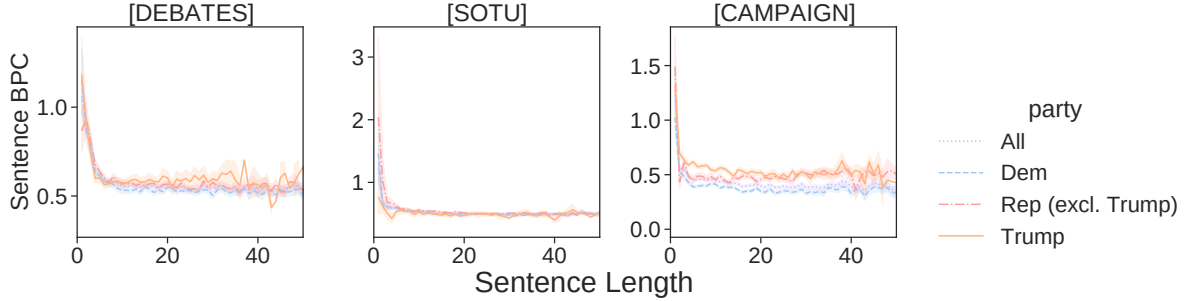
The overall rates of sentences that mention opponents for debates, SOTU, and campaign speeches are 20.60%, 0.83%, and 6.95% respectively. Figure A.11 shows the distribution of sentences containing opponent mentions among the candidate speakers. Trump has the highest rate of opponent mentions in debates.

A.3.2 *Fightin' Words overlap*

We present additional plots for the Fightin' Words overlap metric, for different top- n adjectives in debates (see Figure A.12). In particular, for top-5, 10, and 25 adjective Fightin' Words, Trump has the lowest overlap in adjectives he uses in association with opponent mentions. These trends are consistent with those presented in the main paper.

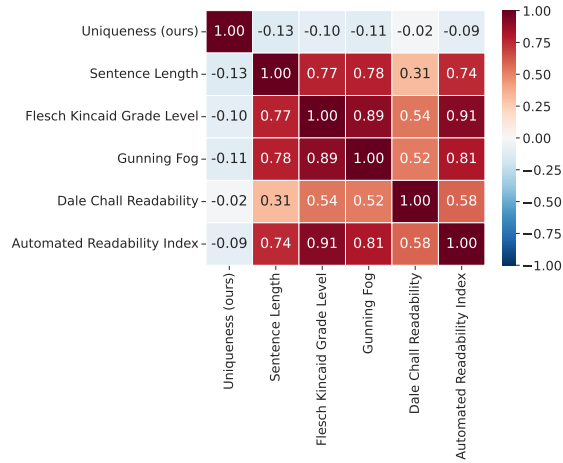


(a) Average sentence BPC per candidate (error bars represent 95%-confidence intervals)

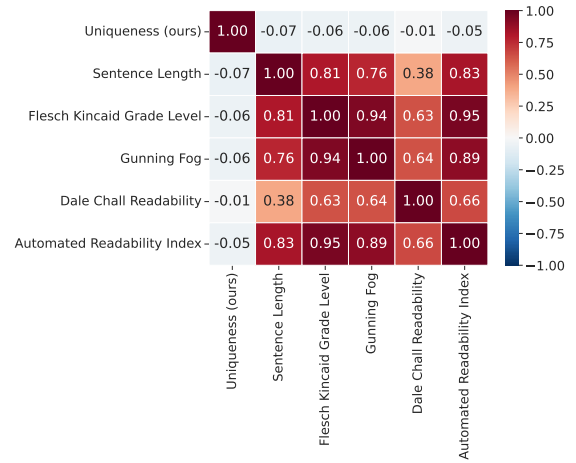


(b) Sentence BPC across sentence length

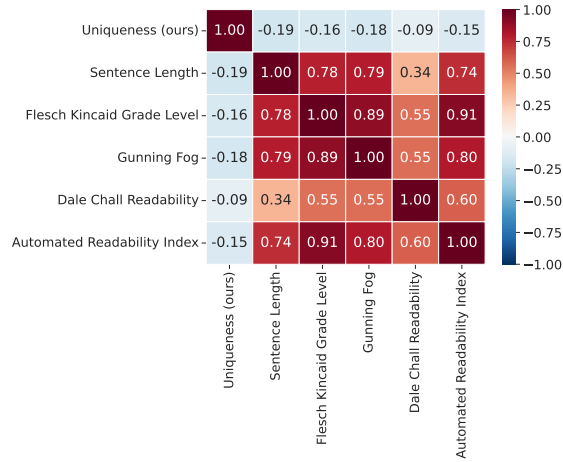
Figure A.1: Average sentence BPC across each data type. In campaigns, Trump is the most unpredictable among those candidates; however, he is ranks closer to the middle for predictability in debates and SOTU



(a) Debates



(b) SOTU



(c) Campaign

Figure A.2: Heatmaps of Spearman Correlation Coefficients between our uniqueness metric and existing readability scores (FKGL, Gunning Fog, Dale Chall, and ARI). There is little to no correlation between our score and readability; in contrast, there is strong correlation between the different readability indexes.

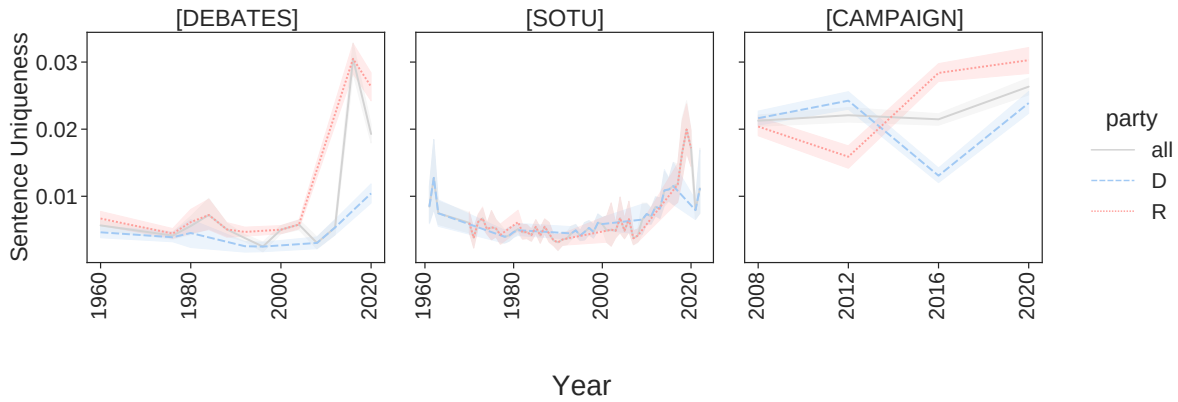
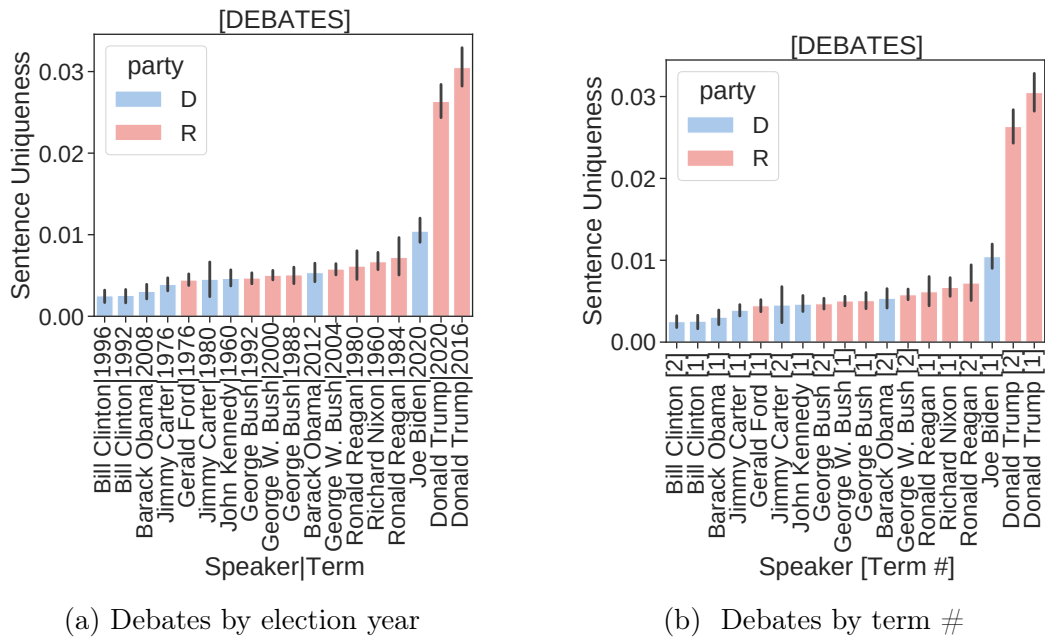


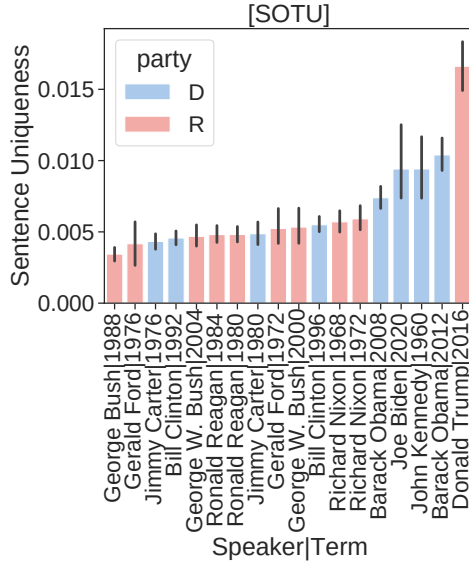
Figure A.3: Uniqueness score over time, aggregated by political party.



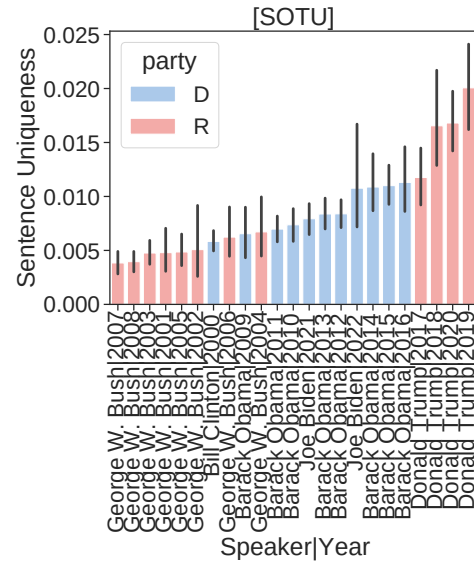
(a) Debates by election year

(b) Debates by term #

Figure A.4: For debates, Trump is consistently most unique for each term that he runs for president (2016, 2020). He is slightly more unique in his first election cycle, in 2016. The error bars represent 95%-confidence intervals.

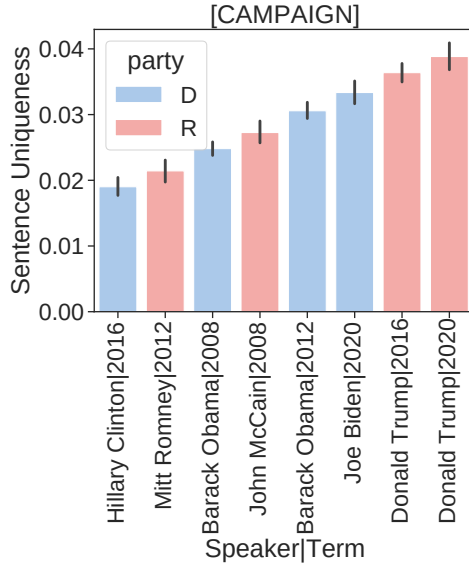


(a) SOTU by term

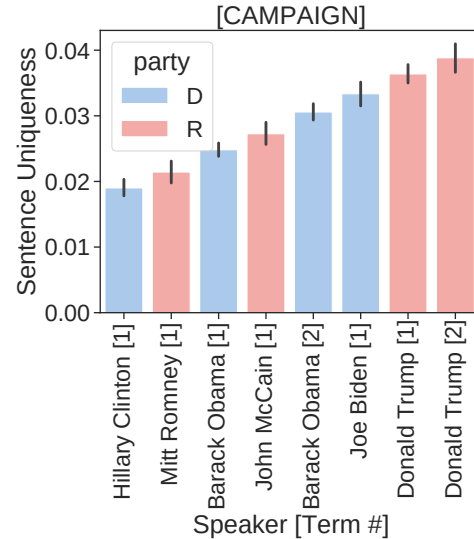


(b) SOTU by year

Figure A.5: For SOTU, Trump is consistently most unique throughout all years of his presidency. His speech distinctiveness increases after the first year of his term. The error bars represent 95%-confidence intervals.



(a) Campaign by election year



(b) Campaign by term #

Figure A.6: For campaign speeches, Trump is consistently most unique for each term that he runs for president (2016, 2020). His campaign speech in his second election cycle is slightly more unique than his first time. The error bars represent 95%-confidence intervals.

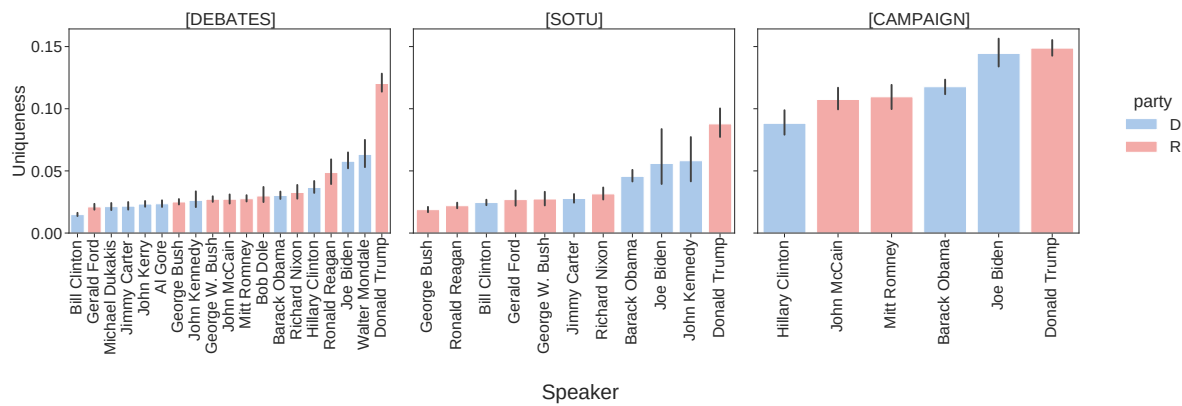
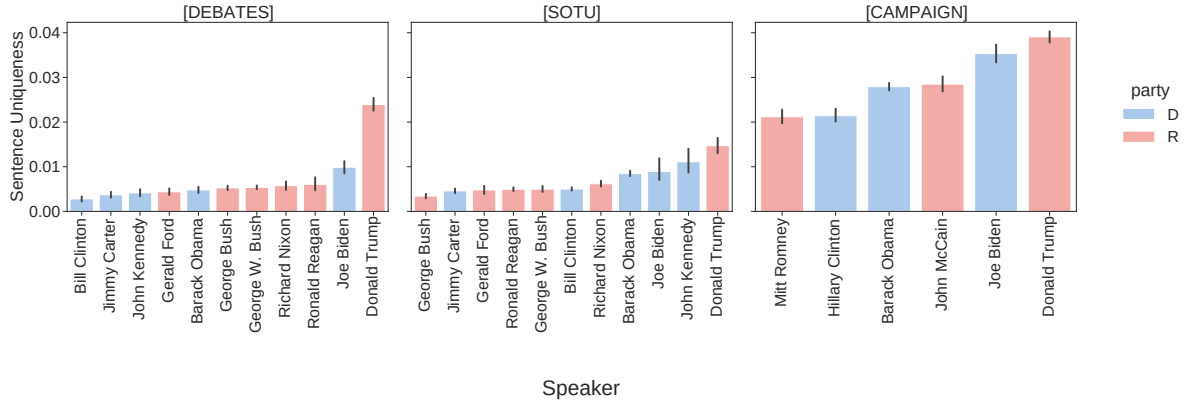
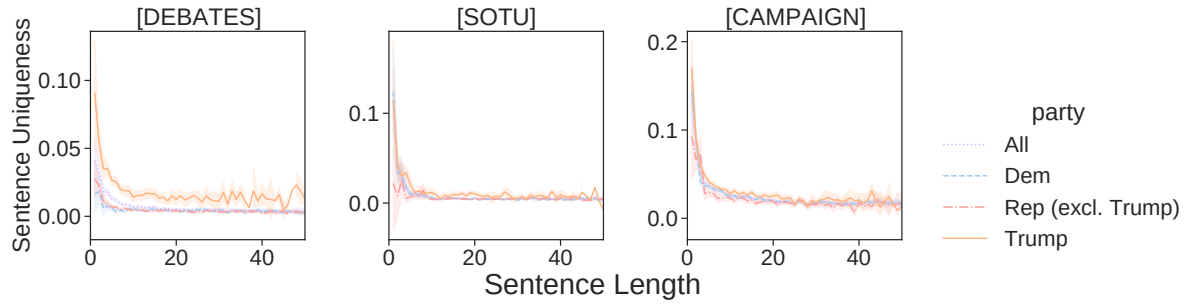


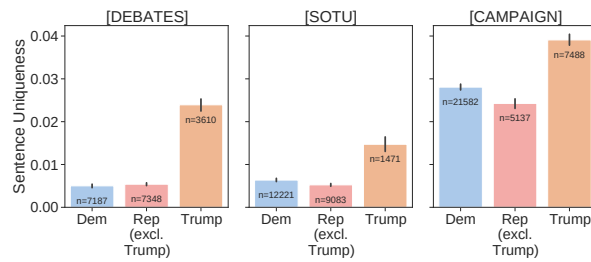
Figure A.7: Uniqueness score by candidate, of sentences within the top 10-ile of scores for each candidate (the error bars represent 95%-confidence intervals). Note that Trump continues to be most unique in all datasets.



(a) Overall score by candidate



(b) Scores by sentence length



(c) Uniqueness by party

Figure A.8: Average sentence uniqueness for each speaker, across all data types, using the UNMASKED model (the error bars represent 95%-confidence intervals). Trump is still the most unique speaker in all types of speeches.

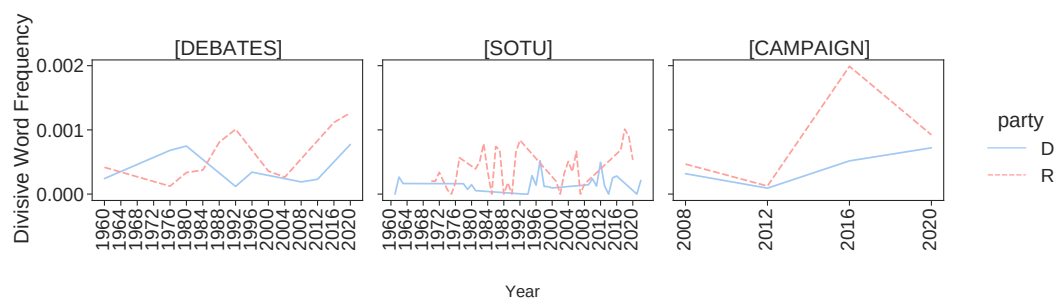
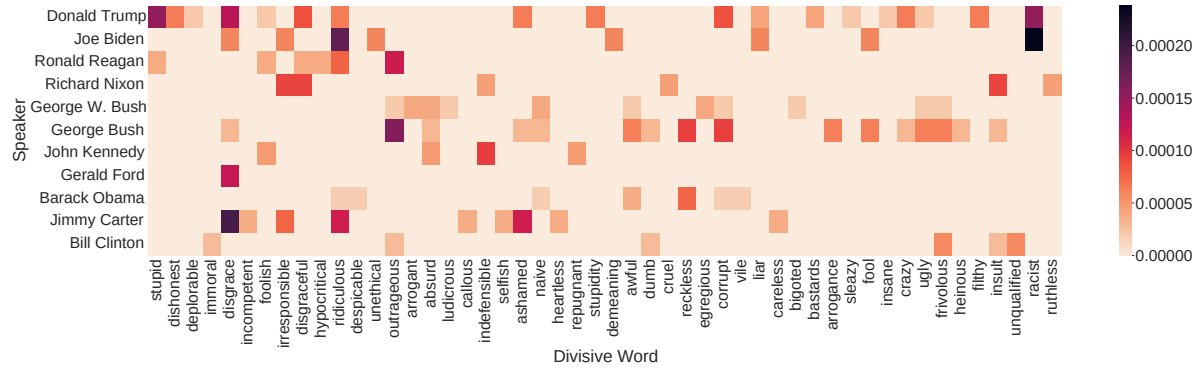
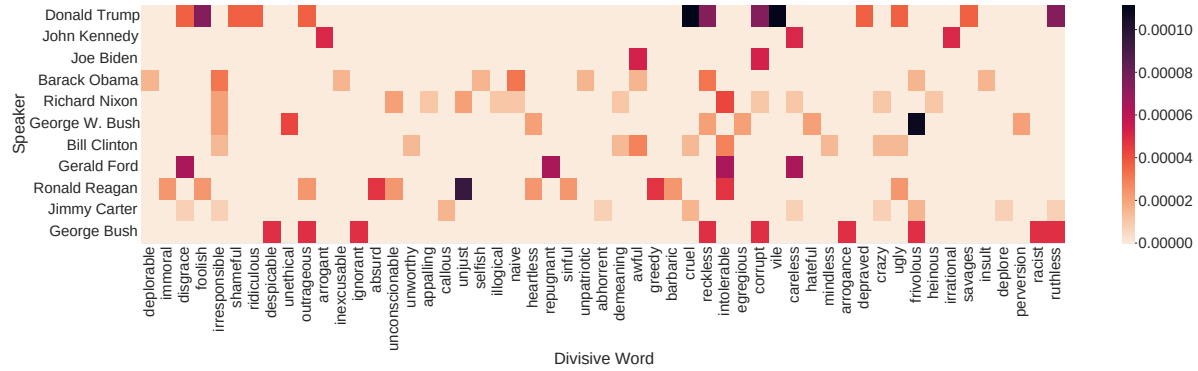


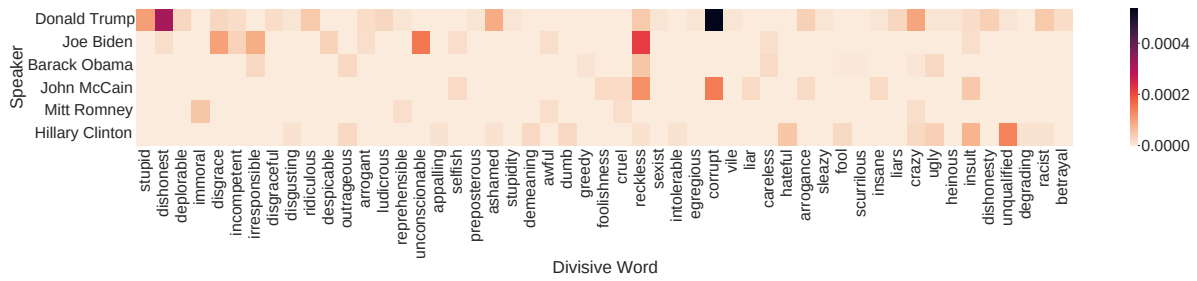
Figure A.9: Frequency of usage over time



(a) Debates



(b) SOTU



(c) Campaign

Figure A.10: Heatmaps showing divisive word utterance frequency, normalized over all words spoken, by speaker.

Year	Example Sentence(s) w/ Context
2020	Other: [to Biden] Let him go for a minute, and then you can go. Donald Trump: And rebuild the building. Joe Biden: No. Donald Trump: It's the dumbest- Joe Biden: That is not, that is not. . . Donald Trump: ... most ridiculous. . .
2020	Joe Biden: Russia is paying you a lot. Joe Biden: China is paying a lot. Joe Biden: And your hotels and all your businesses all around the country, all around the world. Joe Biden: And China's building a new road to a new gas a golf course you have overseas. Joe Biden: So what's going on here? Joe Biden: Why don't you release your tax return or stop talking about corruption?
2020	Donald Trump: I mean, they can say anything. Donald Trump: It's a very- it makes me sad because I am the least racist person. Donald Trump: I can't even see the audience because it's so dark, but I don't care who's in the audience. Donald Trump: I'm the least racist person in this room. Other: OK. Other: Vice President Biden, let me ask you, very quickly, and then I have a follow up question for you. Joe Biden: Abraham Lincoln. Joe Biden: Here is one of the most racist presidents we've had in modern history. Joe Biden: He pours fuel on every single racist fire, every single one.
2016	Donald Trump: They say, what do we need this for? Donald Trump: So Mosul is going to be a wonderful thing. Donald Trump: And Iran should write us a letter of thank you, just like the really stupid the stupidest deal of all time, a deal that's going to give Iran absolutely nuclear weapons.
2016	Hillary Clinton: And we should demand that Donald release all of his tax returns so that people can see what are the entanglements and the financial relationships that he has... Other: We're going to get to that later. Other: Secretary Clinton, you're out of time. Hillary Clinton: ... with the Russians and other foreign powers. Other: Mr. Trump? Donald Trump: Well, I think I should respond, because so ridiculous.
1976	Gerald Ford: On the other hand, when you have a bill of that magnitude, with those many provisions, a President has to sit and decide if there is more good than bad. Gerald Ford: And from the analysis that I have made so far, it seems to me that that tax bill does justify my signature and my approval. Other: Governor Carter, your response. Jimmy Carter: Well, Mr. Ford is changing considerably his previous philosophy. Jimmy Carter: The present tax structure is a disgrace to this country.
1988	George Bush: In terms of negative campaigning, you know, I don't want to sound like a kid in the schoolyard: he started it. George Bush: But take a look at the Democratic convention take a look at it. George Bush: Do you remember the Senator from Boston chanting out there and the ridicule factor from that lady from Texas that was on there; I mean, come on, this was just outrageous.

Table A.1: [DEBATES] Example usages of words from our divisive lexicon throughout debates.

Year	Example Sentence(s) w/ Context
2022	Joe Biden: We’re choking Russia’s access to technology that will sap its economic strength and weaken its military for years to come. Joe Biden: Tonight I say to the Russian oligarchs and the corrupt leaders who have bilked billions of dollars off this violent regime: No more.
2020	Donald Trump: The terrorist responsible for killing Sergeant Hake was Qasem Soleimani, who provided the deadly roadside bomb that took Chris’s life. Donald Trump: Soleimani was the Iranian regime’s most ruthless butcher, a monster who murdered or wounded thousands of American servicemembers in Iraq.
2018	Donald Trump: I am asking Congress to address the fundamental flaws in the terrible Iran nuclear deal. Donald Trump: My administration has also imposed tough sanctions on the communist and socialist dictatorships in Cuba and Venezuela. Donald Trump: But no regime has oppressed its own citizens more totally or brutally than the cruel dictatorship in North Korea.
2018	Donald Trump: In April, this will be the last time you will ever file under the old and very broken system, and millions of Americans will have more take-home pay starting next month—a lot more. Donald Trump: We eliminated an especially cruel tax that fell mostly on Americans making less than \$50,000 a year, forcing them to pay tremendous penalties simply because they couldn’t afford Government-ordered health plans.
2017	Donald Trump: As promised, I directed the Department of Defense to develop a plan to demolish and destroy ISIS, a network of lawless savages that have slaughtered Muslims and Christians, and men and women and children of all faiths and all beliefs. Donald Trump: We will work with our allies, including our friends and allies in the Muslim world, to extinguish this vile enemy from our planet.
2016	Barack Obama: But after years now of record corporate profits, working families won’t get more opportunity or bigger paychecks just by letting big banks or big oil or hedge funds make their own rules at everybody else’s expense. Barack Obama: Middle class families are not going to feel more secure because we allowed attacks on collective bargaining to go unanswered. Barack Obama: Food stamp recipients did not cause the financial crisis; recklessness on Wall Street did.
1991	George Bush: But the world has to wonder what the dictator of Iraq is thinking. George Bush: If he thinks that by targeting innocent civilians in Israel and Saudi Arabia, that he will gain advantage, he is dead wrong. George Bush: If he thinks that he will advance his cause through tragic and despicable environmental terrorism, he is dead wrong.
1982	Ronald Reagan: Contrary to some of the wild charges you may have heard, this administration has not and will not turn its back on America’s elderly or America’s poor. Ronald Reagan: Under the new budget, funding for social insurance programs will be more than double the amount spent only 6 years ago. Ronald Reagan: But it would be foolish to pretend that these or any programs cannot be made more efficient and economical.

Table A.2: [SOTU] Example usages of words from our divisive lexicon throughout SOTU addresses. In general, there are fewer uses of divisive words during SOTU speeches.

Year	Example Sentence(s) w/ Context
2020	Joe Biden: The last thing you need is a President who ignores you, looks down at you, who just doesn't understand you. Joe Biden: Like President Trump. Joe Biden: His reckless personal conduct since his diagnosis, the destabilizing effect it's having on our government, is unconscionable.
2020	Donald Trump: The fact is, this is the single most important election in the history of our country. Donald Trump: And sleepy Joe Biden's made a corrupt bargain.
2016	Donald Trump: Now, Bernie Sanders should be angry right? Donald Trump: Shouldn't he be angry? Donald Trump: Now, I'll tell you what. Donald Trump: The system is rigged. Donald Trump: The system is rigged. Donald Trump: I've been saying it for a—it's rigged, and we're gonna straighten it out. Donald Trump: But the system is rigged. Donald Trump: Hillary is not the victim; the American people are the victims of this system. Donald Trump: So corrupt in so many ways.
2016	Donald Trump: When we win on November 8th, we are going to Washington, D.C. and we are going to drain the swamp. Donald Trump: Gonna drain the swamp. Donald Trump: We're gonna drain the swamp, folks. Donald Trump: We're gonna drain that swamp. Donald Trump: Another important issue for Americans is integrity in journalism. Donald Trump: These people are among the most dishonest people I've ever met, spoken to, done business with. Donald Trump: These are the most dishonest people.
2016	Donald Trump: Our trade deals, we lose \$800 billion a year on trade. Donald Trump: We have trade deficits. Donald Trump: Think of that. Donald Trump: Who negotiates these deals? Donald Trump: You know who does? Donald Trump: Stupid people. Donald Trump: Stupid people. Donald Trump: With very stupid leadership.
2016	Hillary Clinton: We should honor the men and women in uniform who fight for our country. Hillary Clinton: That's why I was so appalled when Donald Trump tweeted that the new effort underway to push the terrorists out of the key city of Mosul is already, and I quote him, "a total disaster" and that our country is, again a quote, "looking dumb." Hillary Clinton: Really? Hillary Clinton: He's declaring defeat before the battle has even started. Hillary Clinton: He's proving once again he is unqualified to be commander in chief of our military.

Table A.3: [CAMPAIGN] Example usages of words from our divisive lexicon throughout campaigns.

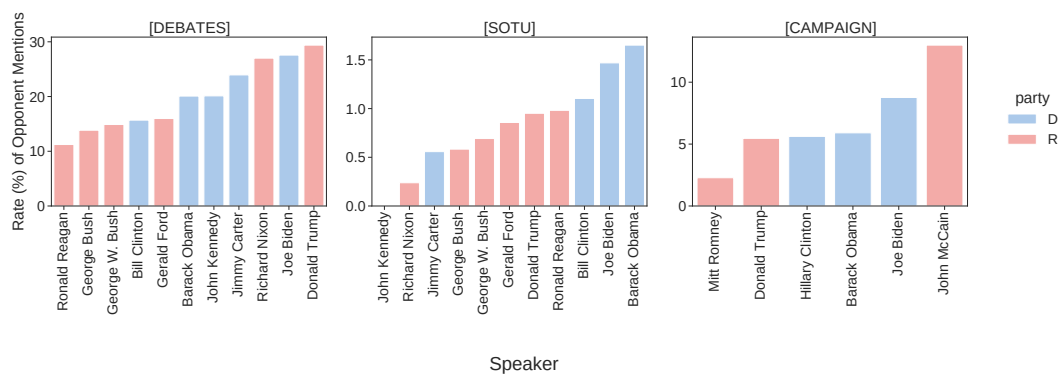
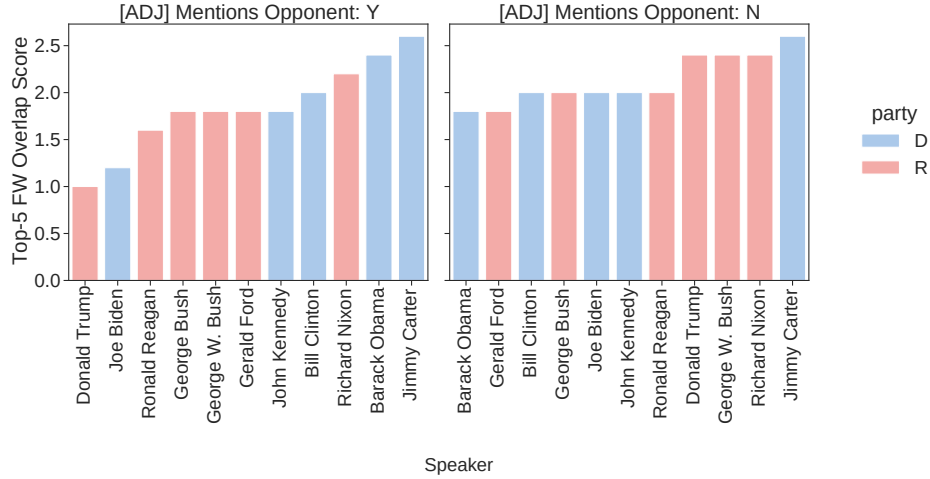
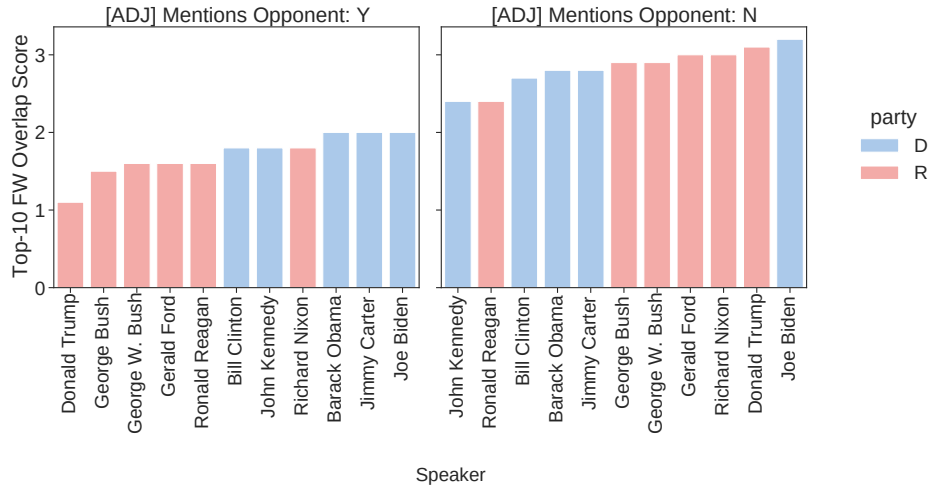


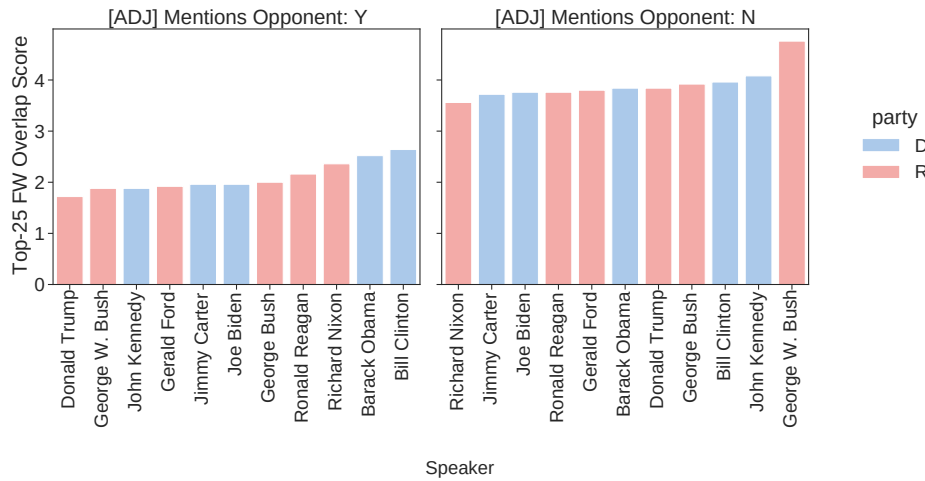
Figure A.11: Distribution of sentences containing opponent mentions among all speakers, across the data types. Note the different rates for debates vs. SOTU. Trump has the highest rate of opponent mentions in debates.



(a) Top-5 FW overlap score in debates for each candidate



(b) Top-10 FW overlap score in debates for each candidate



(c) Top-25 FW overlap score in debates for each candidate

Figure A.12: Fightin' Words overlap results from debates. Trump's FW associated with opponent mentions generally have the lowest overlap in adjective usage compared to other candidates, which is another indication that his language is distinctive.

APPENDIX B

LLM PROMPTS FOR FROM FEEDBACK TO CHECKLISTS

B.1 LLM Prompts

gpt-4o is used for checklist generation.

B.1.1 Baselines

System message:

```
1 You are an expert in clinical documentation. You want to evaluate the quality of a
   clinical note. You will generate a list of simple yes/no questions, such that
   the "Yes" answer corresponds to a good clinical note.
2 You may optionally specify the specific note section that each question pertains to
   . The sections are as follows:
3 - "subjective" includes items taken during verbal exam and typically written in the
   chief complaint, history of present illness, and past social history
4 - "objective_exam" includes content from the physical examination on the day of the
   visit
5 - "objective_results" includes diagnostics taken prior to the visit, including
   laboratory or imaging results
6 - "assessment_and_plan" includes the doctor's diagnosis and planned tests and
   treatments
7 If the question is applicable to the full note, then denote "full" as the pertinent
   section.
```

B.1.2 Feedback Checklists

Proposer's system message for a given section:

1 You are an expert in clinical documentation.

2 You will be provided with an itemized list of physician feedback on the {section}
section of medical notes.

3 Your task is to generate a set of yes/no questions that comprehensively reflect
this feedback.

4

5 The questions should aim to identify the presence or absence of good documentation
practices in the {section} section.

6 The questions should be specific to the {section} section and should not reference
or depend on the content of other sections of the medical note.

7 If a feedback item pertains to multiple note sections, focus only on the parts that
apply to {section} content.

8 Each question should ideally address multiple feedback items rather than targeting
only one.

9 Keep questions concise, atomic, and objective; avoid complex sentence structure.

10 For each question, list the corresponding INDICES of the feedback item that are
addressed.

11

12 Avoid overly specific questions; instead, favor general questions that apply across
a variety of medical encounters.

13 Do not include protected health information (PHI) in any questions.

14

15 Use third-person language when referring to the note.

16 A "Yes" answer should indicate that the note meets good clinical documentation
standards.

17

18 Incorporate both positive and negative feedback:

19 - For negative feedback, generate questions that would flag a note if it exhibited
similar issues.

20 - For positive feedback, generate questions that promote those qualities in all
notes.

21

22 Generate as many questions as necessary to cover the feedback comprehensively.

Assigner's system message:

1 You will be given an itemized checklist of questions and a user feedback.

2 Indicate which questions cover the feedback. Respond as a list of question item
numbers.

3

4 Questions:

5 {itemized checklist questions}

B.1.3 Unit Tests

System message for re-writing reference notes to fail checklist criteria:

1 You are an expert in clinical documentation. You will be given a checklist question
and a medical note that fulfills the criteria, with respect to the given
clinical transcript.

2 Your task is to rewrite the note so that it fails the criteria with respect to the
transcript.

User message for re-writing reference notes to fail checklist criteria:

1 Transcript: {reference transcript}

2

3 Note: {reference note}

4

5 Question: {question}

B.1.4 Feedback Coverage

Assigner's system message:

1 You will be given an itemized checklist of questions and a user feedback.

2 Indicate which questions cover the feedback. Respond as a list of question item
numbers.

3

4 Questions:

5 {itemized checklist questions}

APPENDIX C

CIVICCHATS SUPPLEMENTARY MATERIAL

C.1 Additional Methodology Details

Tradeoff identification system prompt:

1 You are a content analysis assistant. Identify each distinct TRADEOFF in a
participant's short written explanation of how they would vote on a ballot
measure.

2

3 A TRADEOFF is a passage in which the participant weighs two or more considerations
against each other when justifying their vote - acknowledging that a
consideration cuts in a different direction from their conclusion (or that two
considerations supporting their conclusion exist in tension), and resolving or
accepting that tension.

4

5 Tradeoffs may be:

6 - WITHIN-STANCE: the participant votes yes but acknowledges a cost or risk of
voting yes (e.g., 'I support stricter penalties even though they may worsen
prison overcrowding').

7 - BETWEEN-STANCE: the participant weighs an opposing argument against their own (e.
g., 'critics worry about racial disparities, but I think public safety matters
more here').

8

9 Both count.

10

11 Critical guidance:

12 - Identify tradeoffs SEMANTICALLY. Do NOT require lexical markers like 'but', '

however', 'on the other hand', or 'while'. A tradeoff can be expressed without any of those connectives.

- 13 - Conversely, the presence of 'but' or 'however' does NOT automatically make a tradeoff. Only count it if two genuinely competing considerations are being weighed.
- 14 - Pure stance-only justifications do NOT count. If the participant lists reasons X and Y that both pull the same direction without tension, that is NOT a tradeoff.
- 15 - Mere expression of uncertainty ('I'm not sure', 'I don't know enough') does NOT count unless it weighs two specific considerations.
- 16 - A single explanation may contain 0, 1, or several tradeoffs. Only include genuinely distinct tradeoffs (do not split one weighing into two).

17

18 For each tradeoff, return:

- 19 - consideration_a: short paraphrase of one side (<=15 words)
- 20 - consideration_b: short paraphrase of the other side (<=15 words)
- 21 - evidence_span: a verbatim span from the explanation anchoring the tradeoff (<=25 words)
- 22 - resolution: 'chose_a' if the participant ultimately favored consideration_a, 'chose_b' if they favored consideration_b, 'unresolved' if they accepted the tension without picking a side.

23

24 If the explanation contains no tradeoffs, return an empty list.

Tradeoff identification user prompt:

```
1 ## Ballot Measure
2 {measure_context}
3
```

```
4  ## Participant's Decision Explanation
5  {explanation_text}
```

Structured esponse format:

```
1  RESPONSE FORMAT (strict JSON schema, name="tradeoff_framing")
2  {
3    "tradeoffs": [
4      {
5        "consideration_a": "<string>",
6        "consideration_b": "<string>",
7        "evidence_span": "<string>",
8        "resolution": "chose_a" | "chose_b" | "unresolved"
9      }
10   ]
11 }
12 All four properties are required; additionalProperties = false.
13
```

C.2 Chatbot Development and Evaluation

LLM-as-a-Judge is a standard approach for evaluating AI assistant behavior. As such, we evaluate the chatbot’s conversational behavior using quantitative metrics and structured, LLM-based rubric scoring. The judge model reviews full conversations against checklists that capture desired behaviors of the system. These criteria cover factual grounding, tone, conversation flow, and appropriate levels of informativeness. For the Q&A mode, we measure evenhandedness around presenting relevant considerations without favoring one side. For the more discursive modes, we check for appropriate levels of push back of and probing on user positions while being forthright when it disagrees and avoiding manipulative or

leading language. We also directly track response latency, and length and FKGL of chatbot responses compared to user inputs, to emphasize conciseness and accessibility. The deployed CivicChats personas are based on iterative improvement from rounds of this evaluation.

Chatbot Factuality We conduct factuality evaluation of the AI chatbots, using LLM-as-a-Judge (gpt-5.4) to extract factual claims from the chat transcripts. Claims are evaluated against evidence: ballot measure text, RAG context chunks, fetched web URLs, and Ballotpedia sections. The grading is condition-aware, with stricter standards for QA and more lenient for Reflect and Argue (so that a hedged value-judgment isn’t flagged as a false claim).

We find that 83.6% of messages are at least partially grounded in the evidence, with 53.5% of messages fully grounded. Across individual claims, only 1.2% of 2,846 claims are flagged as potentially contradictory, and human annotators resolve that around 75% of these flags are not contradictory.

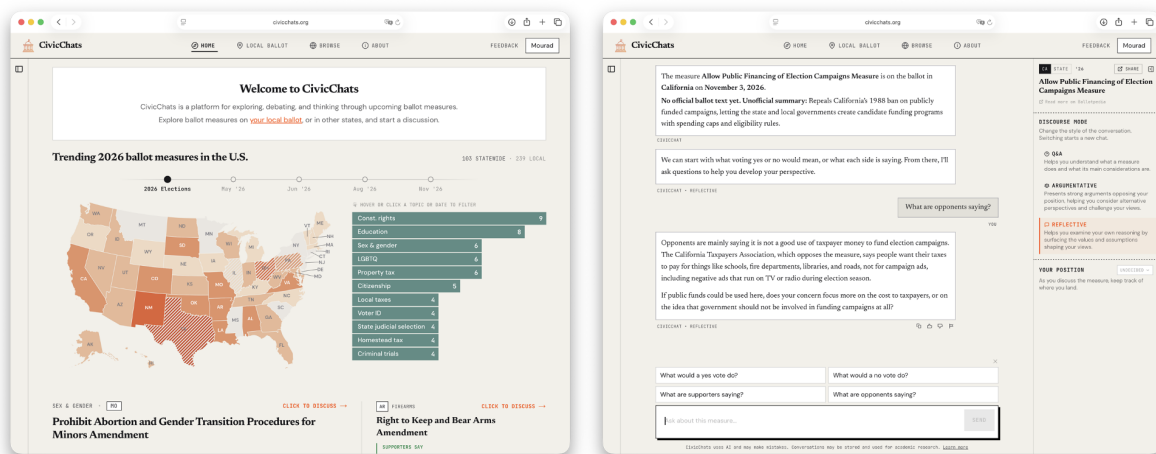


Figure C.1: CivicChats.org.

C.3 Public Website

CivicChats.org hosts the chatbot personas as a public resource with additional interfaces

to explore live ballot measures. The site and data are deployed on a secure departmental server. Users are informed that their chats are collected and used for research purposes, and contact information and a feedback form are provided for users who wish to provide feedback or request removal of their data.