Detecting Hallucination and Coverage Errors in Controlled Text Generation for Controversial Topics

Anonymous ACL ARR submission

Abstract

We propose a new strategy to handle controversial topics in LLM-based chatbots based on Wikipedia's Neutral Point of View (NPOV) principle: acknowledge the absence of a single true answer and surface multiple perspectives. We frame this as controlled text generation, where perspectives are retrieved from a knowledge base and the LLM is tasked with generating a fluent and faithful response from the perspectives. Our main contribution is a detailed study of common failure modes of LLMs, namely hallucination and coverage errors, in the context of this controlled generation task. We propose and evaluate three methods to detect such errors based on (1) word-overlap, (2) salience, and (3) LLM-based classifiers. Our results demonstrate that classifiers, even when trained only on synthetic errors, can achieve high performance, with ROC AUC scores of 95.3% for hallucination and 90.5% for coverage error detection on unambiguous error cases. We show that when no training data is available, our other methods can still yield good results on hallucination (84.0%) and coverage error (85.2%) detection.

1 Introduction

Large Language Models (LLMs) have risen in popularity due to state-of-the-art performance on a wide range of tasks, and a growing audience of users is engaging with LLM-driven chatbots.[1] While these chatbots are highly flexible and generalizable, they are known to struggle with factuality and bias (Shuster et al., 2021; Sheng et al., 2019). Particularly when discussing controversial topics, model developers may desire more control over LLM-based chatbot responses.

In this paper, we investigate how LLMs can be used in controlled text generation for controversial topics. While it is important for generative text systems to provide accurate answers wherever possible, users often seek information on topics for which there are not agreed-upon factual answers. These topics range from the inconsequential (the superiority of the Yankees vs. the Red Sox) to the fundamental ("What religious faith should I adhere to?"). Building useful LLMs requires the ability to ensure that LLM responses adhere to desired levels of neutrality and nuance in such cases.

We introduce the **NPOV Response Task**: when a user asks a query about a controversial topic, the model is provided with arguments for multiple perspectives and is tasked to generate a multi-perspective response, inspired by Wikipedia's Neutral Point of View (NPOV) principle. We adapt a conversational LLM to this task and examine two common error types violating our goal of faithfulness to inputs: (1) **hallucinations** (response contains unprovided arguments), and (2) **coverage errors** (response omits provided arguments).

We build a dataset of model query-response pairs, conditioned on arguments from Britannica's ProCon (ProCon.org, 2022a). Using expert annotators, we identify instances of hallucination and coverage errors. We then propose methods for detecting such hallucination and coverage errors, both with and without access to human-labeled data.

Our main results demonstrate that with access to error-free examples and examples containing only synthetic errors, LLM-based classifiers can achieve ROC AUCs of 95.3% and 90.5% in detecting organic hallucinations and coverage errors respectively on our task. Even without access to annotated data, we can leverage salience and ROUGE-based techniques to achieve ROC AUCs of 84.0% for hallucinations and 85.2% for coverage errors.

2 Handling Controversial Topics

Our work is centered around how LLMs can be controlled to respond to queries about controversial

[1] Among others: https://openai.com/blog/chatgpt; https://bard.google.com; https://www.anthropic.com/index/introducing-claude.

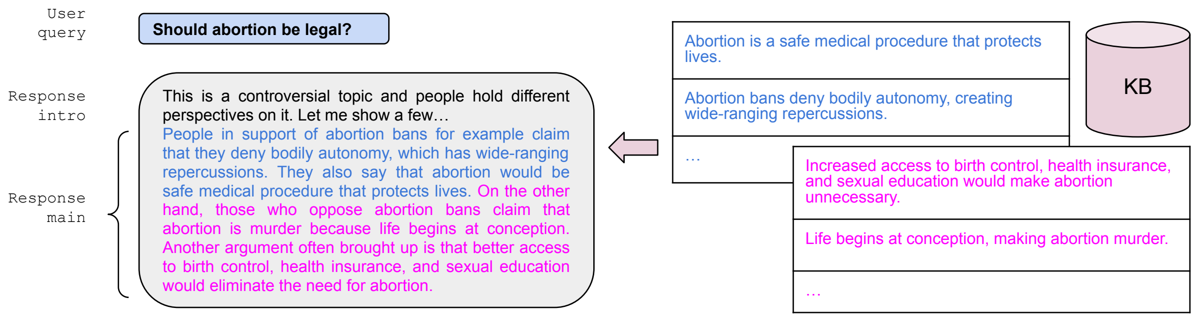

Figure 1: Example NPOV response to a user query on a senstive topic (left) based on pro and con perspectives with two arguments each, as obtained from a knowledge base (right). Arguments taken from ProCon.org (2022b). Our error detection methods focus on the NPOV main response.

topics for which there is no single correct answer. For example, in response to "Should abortion be legal?", an LLM without direction might produce a highly opinionated or offensive response. To address such concerns, "guardrails" are oftentimes added to LLMs, either completely preventing the generation of responses to such topics or responding with canned answers ("I am just a language model and cannot answer this question..."). Such approaches can lead to erasure harm and reduce the usefulness of the system on potentially important topics. Another approach is to personalize responses to align with a user's position; however, this can reinforce harmful biases and popular misconceptions, and act as a chatbot echo chamber.

As an alternative strategy, we propose to acknowledge the lack of agreement and surface main viewpoints instead. This approach is inspired by Wikipedia's **Neutral Point of View (NPOV)** principle, which requires that content is written such that it represents "fairly, proportionately, and, as far as possible, without editorial bias, all the significant views that have been published by reliable sources on a topic."[2] Figure 1 (left) gives an example of an NPOV response on a highly controversial topic. We explore whether such responses can be generated by an LLM using controlled text generation, and we detect common failure modes such as hallucination and coverage errors.

2.1 NPOV Response Generator

We separate *response* generation from *content* generation. We assume that there is a content retrieval process and a knowledge base of curated arguments for different points of view. The knowledge base we use in this paper consists of arguments from Britannica's ProCon website (see §2.2).

The NPOV Response Task is then: given the user query and perspectives (where various arguments are concatenated, each with a prefix like "pro" or "con"), generate a response that consists of an introduction sentence, serving as a bridge from the user query, and a verbalization of the given perspectives. When generating the response, relevant aspects of the given arguments must not be dropped (ensure full coverage) and no other arguments should be added (avoid hallucinations). This task formulation gives model developers fine-grained control over LLM responses. An example is shown in Figure 1.

We adapt an LLM to generate such NPOV responses using prompt-tuning (Lester et al., 2021); see §A.3 for the specific task format and prompt-tuning hyperparameters. Our base LLM[3] is a 64B decoder-only model pre-trained on public dialog data and web text. Our training set consists of 80 query-response pairs covering 9 controversial topics from ProCon (§2.2). ProCon question headers (e.g. "Should abortion be legal?") are used as user queries. For each topic, we sample one, two, or three arguments from the pro and con side in ProCon and then manually write several paraphrased responses capturing these arguments. We observe that after prompt-tuning, the NPOV Response Generator generalizes well beyond the topics and arguments seen during training.

2.2 ProCon as a Knowledge Base

Britannica's ProCon (ProCon.org, 2022a) is a website presenting pros and cons for commonly debated topics. Pros and cons are researched and compiled by ProCon research staff and editors, and they aim to be nonpartisan. As of October 2022, ProCon contains 72 active (i.e. "non-archived")

[2]From https://en.wikipedia.org/wiki/Wikipedia:Neutral_point_of_view, last accessed 2023/04/06.

[3]Model details are omitted for blind review.

topics. For both the pro and con perspective for each topic, several arguments are given, each consisting of a short argument phrase accompanied by a longer explanation. The median number of arguments per perspective per topic is 4, but some topics contain many more arguments (e.g. *Social Media* has 23 arguments per perspective). We randomly sample ProCon arguments as inputs to the NPOV Response Generator for each topic (§4.1). Each topic is associated with a leading question in ProCon (e.g. "Should abortion be legal?"), which we treat as the user query asked to the LLM. See §A.1 for more details.

3 Methods to Detect Hallucinations and Coverage Errors

We focus on hallucination and coverage error detection, adopting the following definitions:

- If the generated response contains at least one argument which was not provided, we call this a **hallucination**.
- If one or more of the given arguments is completely dropped from the response, we call this a **coverage error**.

We call these **full errors**, as they address the hallucination or coverage of a full argument.

On top of these well-defined errors, we notice that the NPOV Response Generator sometimes produces other unfaithful changes to arguments, including: (1) partial hallucinations (slight meaning change, e.g."consensus" becomes "unanimity"), (2) partial coverage errors (only a part of the argument is dropped), (3) repetitions (response contains the same given argument multiple times), and (4) perspective confusions (response inverts the perspectives, e.g. pro arguments are presented as cons). We call all of these **ambiguous errors**.

We propose the following three methods for detecting hallucination and coverage errors in generated responses: ROUGE, salience, and classifiers.

3.1 ROUGE

As a baseline, we use ROUGE-1 (word-matching) to compute hallucination and coverage error scores (Lin, 2004). For a given response from the NPOV Response Generator, ROUGE calculates the proportion of *response words* that are matched in the *input arguments* (ROUGE-1 precision) and the proportion of *input argument words* that are matched in the *response* (ROUGE-1 recall). Low precision is indicative of hallucination, and low recall is indicative of a coverage error. Because the NPOV Response Task requires that both input perspectives be covered, we compute ROUGE-1 recall separately for each input perspective and then compute the minimum as our overall recall score. For ROUGE, words are defined using whitespace and punctuation separation, dropping stop words and using word stemming from NLTK (Bird et al., 2009).

3.2 Salience

Aside from word matching, previous work has proposed methods to quantify attributions from input to output subword tokens in LLMs using model gradients (Bastings and Filippova, 2020). One popular approach is to compute the loss gradient for each output token with respect to each input token embedding, producing a gradient vector for each input-output token pair. The attribution from each input to the output token is defined as the dot product between the corresponding gradient vector and the input token embedding (Denil et al., 2014).[4]

In the NPOV Response Generation scenario, each output token has an attribution value from each input token (e.g. the given arguments per perspective and the user query) and each previously generated token in the response. This produces a token-to-token salience map $M_{tokens} \in \mathbb{R}^{(m+\ell)\times\ell}$, where m is the number of input tokens and ℓ is the number of model response tokens. Before any further processing, we square the salience map and normalize columns to sum to one (i.e. the attribution to each output token sums to one).

Because we are primarily concerned with hallucination and coverage errors for content words, we convert the subword token-to-token salience map to a word-to-word salience map M_{words}. We define words by concatenating consecutive LLM tokens that are not separated by punctuation or whitespace; we then drop stop words, as defined in NLTK (Bird et al., 2009). We define the attribution from an input word w_0 to an output word w_1 as the maximum attribution from any subword token in w_0 to any subword token in w_1. We restrict our salience maps to the input argument words (rows) and the output NPOV response words (columns). A sample word-to-word salience map for a query-response pair is shown in Figure 2.

Qualitatively, we observe that covered words tend to have a high contribution to a single corre-

[4] We obtain comparable results using gradient L2 norms.

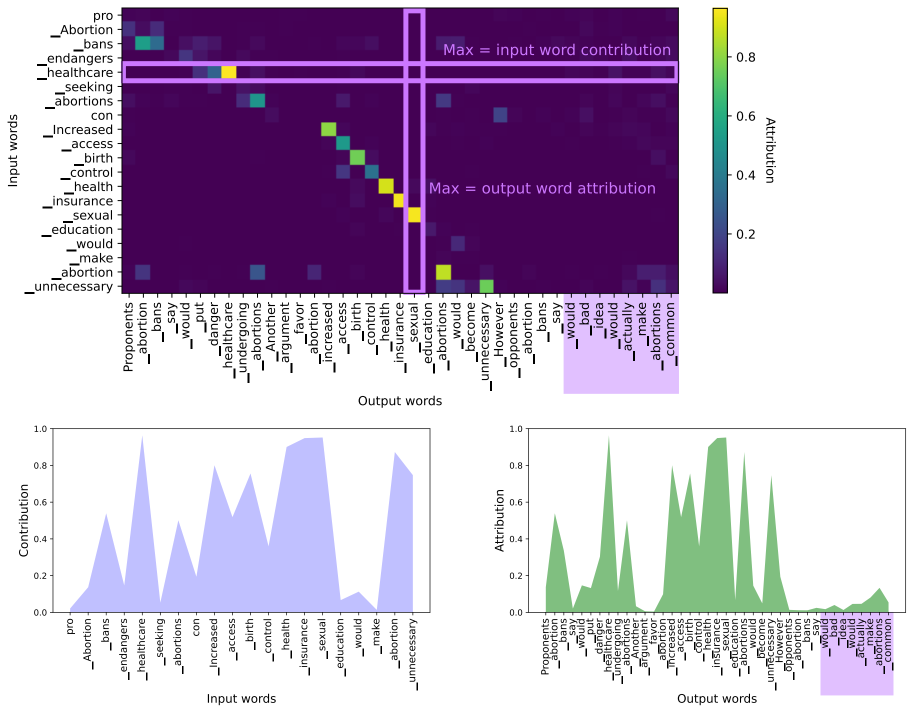

Figure 2: Top: salience map from input argument content words (rows) to model response content words (columns). Bottom: individual word scores for contribution (input words; left) and attribution (response words; right). The highlighted words are hallucinated in the model response.

sponding word in the response. Thus, we define an input argument word's contribution score as its maximum contribution to any response word (i.e. maximum for each row of M_{words}). We define a response word's attribution score as its maximum attribution from any input argument word (i.e. maximum for each column of M_{words}). Contribution and attribution scores for input words and response words respectively are shown in Figure 2.

To compute an example-level contribution score for a query-response pair, we compute the geometric mean of contribution scores for words in each of the two input perspectives. As with ROUGE, we take the minimum of the two perspective contributions as a final contribution score. To compute an example-level attribution score, we compute the geometric mean of attribution scores for all response words. Finally, hallucination and coverage error scores in $[0, 1]$ are computed by subtracting the attribution and contribution scores respectively from 1.0. See Appendix A.7 for equations.

3.3 Classifiers

The two previous methods for detecting hallucination and coverage errors are data-free, not requiring labeled model responses. We now explore how well LLM-based classifiers perform on these tasks, leveraging human annotations if available. Our classifiers are built on a 62B decoder-only LLM which has been instruction-tuned on a large number of tasks analoguously to Chung et al. (2022).[5] We use prompt-tuning to adapt this LLM into classifiers for hallucination and coverage detection. The classifiers have as input: (1) the user query, (2) the generated NPOV response, and (3) the given arguments per perspective. We train the LLM to predict the label "NO" if there is a full error and "YES" otherwise. See §A.3 for the specific task format and prompt-tuning hyperparameters, and §5.2 for the training datasets used.

For inference, we generate error classification scores in $[0, 1]$ by obtaining the LLM's log perplexity scores for the tokens corresponding to the two output class labels ("YES" and "NO"), apply softmax, and take the score of the negative class. [6]

4 Dataset

To train and evaluate the hallucination and coverage error detection methods above on the NPOV Response Task, we construct datasets of organic (i.e. naturally occurring) and synthetic errors, with and without paraphrasing.

[5]Model details are omitted for blind review.

[6]For single-token labels, this score equals the probability of "NO" conditioned on either "YES" or "NO" output token.

4.1 Annotation Procedure

For each of the 72 controversial topics from ProCon (ProCon.org, 2022a), we generate up to 18 query-response pairs by first randomly sampling combinations of pro and con arguments, with either 1, 2, or 3 arguments per side, and then using the NPOV Response Generator to generate a response. We annotate these query-response pairs (also called *examples*) in three stages to (1) identify error-free examples, (2) identify examples with errors, and (3) generate paraphrased examples:

1. For the first three examples per topic, we sample two generator responses, with temperatures 0.0 and 0.7 respectively. We annotate whether responses contain hallucinations or coverage errors, annotating examples with a mix of the two temperatures. We annotate the token spans in the response that cover each input argument, along with any hallucinated response spans and uncovered input argument spans.
2. Because examples with hallucination and coverage errors are less frequent than error-free examples even for high temperatures (20.0% errors in 0.7 temperature responses), we sample a single 0.7 temperature response for each of the remaining (up to) 15 examples per topic.[7] We annotate for hallucination and coverage errors, including both full and ambiguous errors (§3).
3. Hallucination and coverage error detection methods should capture whether meaning is retained between input arguments and generated responses, even if the arguments are not copied verbatim. We therefore generate examples with enforced paraphrasing between the input arguments and the response. To do so, we paraphrase the input arguments for all error-free examples generated in Step 1. For each argument, we use an off-the-shelf paraphrasing tool and manually verify that the paraphrasing does not induce substantial meaning change.[8]

In total, we identify 160 examples with no errors and 326 examples with at least one error, and we generate 152 paraphrased examples with no errors. See §A.4 for details on the annotated examples and errors in different dataset splits.

[7] Preliminary experiments with the NPOV Response Generator suggest that temperatures above 1.0 tend to produce overly long and irrelevant responses.

[8] We use `https://quillbot.com/` for paraphrasing. We find it more efficient to paraphrase the input arguments than to paraphrase the whole response.

4.1.1 Inter-Annotator Agreement

To validate the viability and coherence of our annotation task, we hired a team of 10 external annotators to re-identify both hallucination and coverage errors in our dataset. Our annotation provider was paid 49 USD per hour for a total of 25 hours of work. Annotators were presented with 188 of the query-response pairs annotated in annotation Step 1 (§4.1) and 86 pairs from Step 2. Given the user query, the provided arguments, and the response from the NPOV Response Generator, annotators were asked to mark whether each response had a hallucination or coverage error (details in §A.2). Each query-response pair was annotated by 5 annotators. We compare the annotator majority vote to our annotated labels, finding 90% agreement for hallucinations and 94% for coverage errors. To measure inter-annotator agreement, we compute Krippendorff's alpha for hallucinations ($\alpha = 0.60$) and coverage errors ($\alpha = 0.73$) across the 10 annotators. These values are in line with or above similar text classification tasks (Wulczyn et al., 2017).

4.2 Synthetic Errors Dataset

Due to the relative rarity of *organic* errors produced by the NPOV Response Generator, we synthetically generate examples with errors by modifying error-free query-response pairs. Specifically, we modify the list of given arguments while keeping the original response unchanged. For coverage errors, we add one randomly sampled unused argument for the given topic from ProCon and add it to the list of given arguments. This creates a full coverage error because the original response does not cover this argument. For hallucinations, we randomly remove one of the given arguments. This creates a hallucination because the original response still addresses the removed argument. We apply synthetic error generation to both paraphrased and unparaphrased examples that were annotated as error-free in §4.1 (312 examples), generating 667 new examples with synthetic hallucinations, synthetic coverage errors, or both. See §A.4 for detailed numbers.

4.3 Test Sets with Different Error Types

Taking the annotations and synthetic errors generated above, we split the 72 ProCon topics into a train set (9 topics), a development set (28 topics), and a test set (35 topics). Our dataset contains two types of query-response pairs (paraphrased and unparaphrased) and three types of errors (synthetic

	Hallucinations			Coverage Errors		
Test set error type	ROUGE	Salience	Classifier	ROUGE	Salience	Classifier
Full organic	0.840	0.808	**0.953**	0.795	0.852	**0.905**
Unparaphrased synthetic	0.772	0.736	**0.998**	0.890	0.875	**0.986**
Paraphrased synthetic	0.680	0.708	**0.977**	0.746	0.831	**0.993**
Ambiguous organic	0.814	0.772	**0.851**	**0.834**	0.755	0.756

Table 1: ROC AUC scores for example-level hallucination and coverage error detection on four test sets (§A.5).

full, organic full, and organic ambiguous). We evaluate the performance of our error detection methods on different slices of the test set to better understand where different approaches have strengths or weaknesses. Hence, each table in the results section states the specific test set slices evaluated. See §A.5 for details (size and composition) about each test set slice.

5 Results

5.1 Example-Level Error Detection

First, we evaluate the three error detection methods (ROUGE, salience, and classifiers) on the example-level, i.e. detecting whether a query-response pair contains an error. The classifiers shown here are trained only on query-response pairs which are either error-free or contain *synthetic* errors, including both paraphrased and unparaphrased versions (503 examples total); we explore the impact of training data on classifier performance in §5.2.

Table 1 shows ROC AUC scores on the different test sets (§4.3) for all three methods. While the *full organic* set (organic error-free examples vs. organic full errors) is the most realistic, our synthetic sets allow for more controlled evaluations.

Classifiers consistently outperform the other two methods by a large margin on all sets except ambiguous coverage errors (discussed below), with ROC AUCs above 90% for both hallucination and coverage error detection, for all full error types (organic and synthetic, paraphrased and unparaphrased). Comparing ROUGE and salience, results are mixed. On the full organic errors, ROUGE performs better at detecting hallucinations (84.0% AUC), whereas salience performs better at detecting coverage errors (85.2% AUC).

For copy-like tasks with few expected word changes, ROUGE outperforms salience on both hallucination and coverage error detection (results on the unparaphrased synthetic errors set). However, on the paraphrased synthetic errors set, it appears that salience captures the underlying semantics better than ROUGE, allowing it to more accurately detect both hallucination and coverage errors.

Finally, we evaluate our methods on ambiguous errors (including partial argument hallucination and coverage errors, argument repetition, and perspective confusion; see §3). ROUGE performs well here, likely due to minimal natural paraphrasing from the NPOV Response Generator. Classifier ROC AUC scores drop substantially on ambiguous errors, likely because classifiers are trained only on full errors. This discrepancy seems most problematic for coverage error detection, where classifiers perform even worse than ROUGE. Future work should establish clearer definitions of ambiguous errors, allowing larger sets of ambiguous errors to be annotated and used to train classifiers.

5.2 Classifier Training Data Ablations

We analyze the impact of different types and amounts of training data on classifier performance, considering the following four scenarios:

- **Error-free +Synth**: all error-free query-response pairs, plus synthetic errors; training split only (70 examples).
- **+Para**: previous, plus equivalent paraphrased examples; training split only (138 examples).
- **+Dev**: previous, plus equivalent examples from the development split (503 examples).
- **+Org**: previous, plus examples with organic full errors; training and development splits (573 examples).

Table 2 shows classifier performance on the full organic and ambiguous organic test sets (§4.3). For coverage error detection, performance strictly improves on the full organic set as we add more training data. However, adding the organic error examples leads to a decline in performance on the ambiguous organic set. For hallucination detection as well, we see performance improvement when adding more training data. Adding the organic er-

	Hallucinations				Coverage Errors			
Test set error type	Error-free +Synth	+Para	+Dev	+Org	Error-free + Synth	+Para	+Dev	+Org
Full organic	0.789	0.828	0.953	0.920	0.880	0.903	0.905	0.956
Ambiguous organic	0.807	0.820	0.851	0.862	0.702	0.529	0.756	0.640

Table 2: ROC AUC scores for classifiers trained on different amounts and types of data (§5.2), ordered from smallest to largest training set size. Table 1 results use the classifiers trained on +Dev.

	Hallucinations		Coverage Errors	
Test set error type	ROUGE	Salience	ROUGE	Salience
Full organic	0.673	**0.724**	0.669	**0.799**
Unparaphrased synthetic	0.697	**0.710**	0.693	**0.808**
Paraphrased synthetic	0.614	**0.673**	0.582	**0.742**
Ambiguous organic	**0.542**	**0.542**	0.738	**0.740**

Table 3: ROC AUC scores for word-level error detection results for ROUGE and salience.

rors leads to a performance drop on the full organic set, but not the ambiguous organic set.

Overall, adding more data, even consisting of synthetic errors, leads to improvements on most test sets and for both hallucination and coverage error detection. Surprisingly, adding organic errors on top leads to mixed results, showing that organic data is not necessarily always helpful or needed for good classifier performance. The **+Dev** scenario might already be large enough that the addition of organic errors does not provide benefit.

5.3 Word-Level Error Detection

In practice, it may also be useful to locate specific response words that are hallucinated, or specific input words that are uncovered. Of the methods in §3, ROUGE and salience can both produce hallucination and coverage error scores at the word level. Specifically, the ROUGE coverage error score would be 0 if an input word is matched in the response (and 1 otherwise), and the ROUGE hallucination score would be 0 if a response word is matched in the input arguments (and 1 otherwise). For salience, before example-level aggregation, scores are already computed per word (§3.2). Sample word-level salience scores for hallucination and coverage errors are shown in Figure 2.

We compare the word-level hallucination and coverage error scores from salience and ROUGE with the ground truth annotations of hallucinated and uncovered words annotated in our test sets (§4.1). Results are computed over all non-stop words in each test set, defining words by merging LLM tokens (§3.2). Results are reported in Table 3. Salience performs equally to or better than ROUGE for detecting both hallucinated words in model responses and uncovered words in input arguments on all test sets. On the test set with paraphrased synthetic errors, salience has the largest relative gains over ROUGE, likely due to its ability to capture semantics even in cases of word mismatch, similar to the trends on example-level error detection.

6 Discussion

Overall, LLM-based classifiers trained on relatively small amounts of data perform surprisingly well, outperforming all other methods detecting full errors and obtaining promising ROC AUC scores between 90% and 99%. This is especially notable given that the classifiers are trained only on *synthetic* hallucination and coverage errors and yet perform well on the organic test set.

While not as strong as the classifiers, the data-free methods presented here still achieve strong results. Our experiments demonstrate that ROUGE is a strong data-free baseline for hallucination and coverage error detection in tasks with minimal paraphrasing. When more paraphrasing is expected, salience provides stronger results, appearing to better capture semantics than simple word matching. Moreover, salience is effective for word-level hallucination and coverage error detection, allowing us to locate the parts of a generated response that are problematic.

Our experiments also show the value of different test set slices. While the synthetically constructed datasets might diverge from the true data distribution, they offer a way to analyze strengths and weaknesses of different methods in an isolated fashion, e.g. paraphrased examples demonstrating the shortcomings of ROUGE.

Finally, all methods struggle on ambiguous organic errors, although these results are inconclusive. Largely, this set is a "catch-all" for problematic and low-agreement errors, possibly explaining the poor performance of different error detection approaches. Training classifiers on this subset is important future work, but requires a larger dataset of more clearly-defined ambiguous errors.

7 Related Work

Errors in controlled text generation. Our approach to NPOV Response Generation using provided input perspectives can be seen as an example of *table-to-text generation*, which aims to generate fluent and faithful natural language descriptions of tabular data. Table-to-text generation has been studied in the context of a variety of datasets including WikiBio (Lebret et al., 2016), ToTTo (Parikh et al., 2020), DART (Nan et al., 2021), and WebNLG (Gardent et al., 2017). Traditional metrics such as ROUGE, BLEU, and METEOR compare model responses to a reference output, but metrics have also been developed specifically for table-to-text tasks to address a generally poor correlation between previous metrics and human assessment. PARENT (Dhingra et al., 2019) considers both the table source and reference output when scoring a model response, and IE-based metrics (e.g. Liu et al., 2021) compare entities between the table source and the response.

In typical table-to-text tasks, individual table inputs are short expressions, often only consisting of named entities or numbers that allow for minimal paraphrasing. In our case, however, the input fields are perspectives composed of several full sentences for the arguments. For this reason, pure matching-based scoring approaches (e.g. ROUGE, BLEU, and PARENT) may be less effective.

Our task is also closely related to retrieval-augmented generation, where information (e.g. a document or paragraph) is retrieved from a knowledge source and used to condition a model response. Like the NPOV Response Generator, retrieval-augmented models sometimes exhibit hallucination (Dziri et al., 2022) and coverage errors (Krishna et al., 2021) relative to the retrieved source.

More broadly, hallucinations are a common artifact in natural language generation (NLG). At a high level, they can be described as cases where the generated output is "unfaithful" to the source content (Ji et al., 2023). Due to the fluency of modern NLG systems, hallucinations are particularly concerning as they can remain undetected and mislead users. Tolerance to such errors is particularly low in summarization and table-to-text tasks. In the NPOV Response Task, we focus on *full* errors, where a hallucinated or uncovered argument can be identified relatively unambiguously.

Prompt-tuning. Both the NPOV Response Generator (§2.1) and the classifiers (§3.3) use soft prompt-tuning, a method where only a small number of parameters are tuned and the base LLM is left unchanged (Lester et al., 2021). Mozes et al. (2023) show that LLMs can be prompt-tuned even on very small datasets to function as classifiers.

Salience. Previous work has identified hallucinations in machine translation using proportions of source contributions to output tokens (Dale et al., 2022; Voita et al., 2021), calculated with aggregated layerwise token attribution (Ferrando et al., 2022). Our salience-based method for error detection is similar in spirit, but our attributions are based on loss gradients (Bastings and Filippova, 2020). We focus on the dot products between gradients and inputs, which are often used to roughly quantify Transformer model attributions from input tokens (Ding and Koehn, 2021; Boggust et al., 2022; Zhao et al., 2022).[9] Previous work has applied gradient-based salience methods to fine-tuned encoder-decoder and encoder-only classification models (Tenney et al., 2020). Here, we extend this work to a large decoder-only model, prompt-tuned on a sequence-to-sequence task.

8 Conclusion

In this paper, we introduce the NPOV Response Task as an approach to controlled text generation when dealing with controversial topics. We propose and evaluate methods for detecting hallucination and coverage errors in LLM-generated responses, and we demonstrate a synthetic error generation strategy that can be used to train and evaluate our proposed methods. We find that prompt-tuned LLM classifiers trained only on synthetic errors can achieve high error detection performance on organic examples, and our other methods achieve strong results without any training data.

[9] We obtain comparable results using gradient L2 norms.

Ethical Considerations

With the rise of LLM-based chatbots and broader societal concerns about echo chambers, filter bubbles, and polarization, the ability of LLMs to provide neutral, factual, and nuanced responses to controversial topics is an important avenue of work. However, having LLMs respond to queries about controversial topics is inherently challenging: who decides what is controversial, neutral, and factual, and how this is encoded in an LLM is a hard and nebulous problem. Moreover, as LLMs and chatbot technologies become increasingly easy to create, maliciously engineered and maliciously applied models are likely to become more prevalent. Controlled text generation is a way to control LLM responses in a maximally transparent way.

In this paper, we assume the existence of a database with NPOV-expressed perspectives. However, such a database is not an easy artifact to create, and the contents will often be hotly contested. The dataset we use is derived from Britannica's ProCon website (ProCon.org, 2022a). However, this still reduces arguments to pro and con perspectives, which can reinforce a binary vision of the world. Our work does not address how to best arrive at and reflect consensus on specific arguments. For example, when should the model express "many experts" vs. "a few experts" as a qualification for an argument? Failure here can serve to elevate fringe arguments. Even deciding whether a topic is controversial is already culturally charged. For instance, the subject of gun control might be a non-issue for some European countries yet remain polarizing in the United States. Similarly, omitting topics or arguments that are relevant for minorities or non-Western countries risks reinforcing systemic erasure and promoting socio-cultural biases. To address and mitigate these biases in a perspectives database, processes are necessary to ensure that the group of experts providing perspectives is diverse and multicultural.

The more basic question of when to apply an LLM in practical scenarios needs careful consideration. In some domains (e.g. medical information), even very low error rates may not be acceptable, while other domains (e.g. creative writing) have very different risk profiles. Proper evaluations, policies, and guardrails should be put in place before LLMs are applied in practice to new domains.

Limitations

Our work has several limitations. The NPOV Response Generator is trained and evaluated only in English, and our NPOV Response Task does not address how to create the content in the perspectives and their arguments. The arguments used in our work are pulled from ProCon, which limits both our set of controversial topics and our sets of perspectives (i.e. only pro and con; see Ethical Considerations); future work might consider more nuanced methods of perspective identification, selection, and/or generation.

Our work also does not focus on biases in LLM hallucinated content or the types of content that LLMs often fail to cover. For example, the NPOV Response Generator may be more likely to hallucinate or fail to cover arguments for specific topics or perspectives, e.g. based on the frequency of topics and perspectives in the LLM pre-training corpus.

Even when focusing just on error detection rather than content, we focus primarily on errors that are easy to identify and have high levels of inter-annotator agreement. Based on our own annotations, inter-annotator agreement on ambiguous errors appears lower, but a full analysis is missing and subject to future work. Moreover, an important branch of future work is to establish more thorough taxonomies and annotation schemes for hallucination and coverage error types.

Finally, the computational footprints of the NPOV Response Generator and the LLM-based error classifiers are large, with each model built upon a 60B+ parameter LLM. Similarly, computing salience maps for error detection requires computing gradients from the NPOV Response Generator itself, thus inducing a large computational cost. Of the error detection methods evaluated in our work, ROUGE is by far the most computationally efficient. Future work may consider more computationally efficient approaches, such as evaluating smaller models as error detection classifiers.

References

Jasmijn Bastings and Katja Filippova. 2020. The elephant in the interpretability room: Why use attention as explanation when we have saliency methods? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 149–155, Online. Association for Computational Linguistics.

Steven Bird, Ewan Klein, and Edward Loper. 2009.

Natural language processing with Python: Analyzing text with the natural language toolkit. O'Reilly Media.

Angie Boggust, Harini Suresh, Hendrik Strobelt, John V Guttag, and Arvind Satyanarayan. 2022. Beyond faithfulness: A framework to characterize and compare saliency methods. *arXiv preprint arXiv:2206.02958*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

David Dale, Elena Voita, Loïc Barrault, and Marta R. Costa-jussà. 2022. Detecting and mitigating hallucinations in machine translation: Model internal workings alone do well, sentence similarity even better.

Misha Denil, Alban Demiraj, and Nando De Freitas. 2014. Extraction of salient sentences from labelled documents. *arXiv preprint arXiv:1412.6815*.

Bhuwan Dhingra, Manaal Faruqui, Ankur Parikh, Ming-Wei Chang, Dipanjan Das, and William Cohen. 2019. Handling divergent reference texts when evaluating table-to-text generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4884–4895. Association for Computational Linguistics.

Shuoyang Ding and Philipp Koehn. 2021. Evaluating saliency methods for neural language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5034–5052, Online. Association for Computational Linguistics.

Nouha Dziri, Sivan Milton, Mo Yu, Osmar Zaiane, and Siva Reddy. 2022. On the origin of hallucinations in conversational models: Is it the datasets or the models? In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5271–5285, Seattle, United States. Association for Computational Linguistics.

Javier Ferrando, Gerard I. Gállego, Belen Alastruey, Carlos Escolano, and Marta R. Costa-jussà. 2022. Towards opening the black box of neural machine translation: Source and target interpretations of the transformer. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8756–8769. Association for Computational Linguistics.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The WebNLG challenge: Generating text from RDF data. In *Proceedings of the 10th International Conference on Natural Language Generation*.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12).

Kalpesh Krishna, Aurko Roy, and Mohit Iyyer. 2021. Hurdles to progress in long-form question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4940–4957.

Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213. Association for Computational Linguistics.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81. Association for Computational Linguistics.

Tianyu Liu, Xin Zheng, Baobao Chang, and Zhifang Sui. 2021. Towards faithfulness in open domain table-to-text generation from an entity-centric view. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(15):13415–13423.

Maximilian Mozes, Jessica Hoffmann, Katrin Tomanek, Muhamed Kouate, Nithum Thain, Ann Yuan, Tolga Bolukbasi, and Lucas Dixon. 2023. Towards agile text classifiers for everyone. *arXiv*.

Linyong Nan, Dragomir Radev, Rui Zhang, Amrit Rau, Abhinand Sivaprasad, Chiachun Hsieh, Xiangru Tang, Aadit Vyas, Neha Verma, Pranav Krishna, Yangxiaokang Liu, Nadia Irwanto, Jessica Pan, Faiaz Rahman, Ahmad Zaidi, Mutethia Mutuma, Yasin Tarabar, Ankit Gupta, Tao Yu, Yi Chern Tan, Xi Victoria Lin, Caiming Xiong, Richard Socher, and Nazneen Fatema Rajani. 2021. Dart: Open-domain structured data record to text generation.

Ankur P. Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. Totto: A controlled table-to-text generation dataset.

ProCon.org. 2022a. `https://www.procon.org/`. Accessed: 2022-10-12.

ProCon.org. 2022b. Abortion. `https://abortion.procon.org/`. Last modified June 24, 2022.

ProCon.org. 2022c. Vegetarian. `https://vegetarian.procon.org/`. Last modified February 25, 2021.

Emily Sheng, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng. 2019. The woman worked as a babysitter: On biases in language generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3407–3412, Hong Kong, China. Association for Computational Linguistics.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ian Tenney, James Wexler, Jasmijn Bastings, Tolga Bolukbasi, Andy Coenen, Sebastian Gehrmann, Ellen Jiang, Mahima Pushkarna, Carey Radebaugh, Emily Reif, and Ann Yuan. 2020. The language interpretability tool: Extensible, interactive visualizations and analysis for NLP models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 107–118, Online. Association for Computational Linguistics.

Elena Voita, Rico Sennrich, and Ivan Titov. 2021. Analyzing the source and target contributions to predictions in neural machine translation.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex machina: Personal attacks seen at scale. *arXiv preprint arXiv:1610.08914v2*.

Yang Zhao, Zhang Yuanzhe, Jiang Zhongtao, Ju Yiming, Zhao Jun, and Liu Kang. 2022. Can we really trust explanations? evaluating the stability of feature attribution explanation methods via adversarial attack. In *Proceedings of the 21st Chinese National Conference on Computational Linguistics*, pages 932–944, Nanchang, China. Chinese Information Processing Society of China.

A Appendix

A.1 ProCon

We use the perspectives and arguments for the different topics listed on Britannica's ProCon website as of October 2022 (ProCon.org, 2022a). We randomly split the 72 ProCon topics into train, dev, and test, as shown in Table 5, ensuring no overlap in topics across these splits. In line with ProCon's usage guidelines (`https://www.procon.org/faqs/#II`), all arguments are used verbatim as stated on the specific topic website under the section "Pro & Con Arguments". We scrape the subtitles of the pro and con columns as our arguments; Figure 5 shows an example. The median number of arguments per pro and con perspective per topic is 4, with a maximum of 23 and a minimum of 2. The ProCon data is publicly available through their website, containing no personally-identifying information about individuals. We follow the guidelines specified by ProCon on "How to Use" their data (`https://www.procon.org/faqs/#II`).

A.2 Human Annotation Details

For the human annotations in §4.1.1, our annotation service provider was paid 49 USD per hour for a total of 25 hours of work, and we asked them to ensure fair payment to annotators. The 10 annotators were specialized workers in the United States contracted by our annotation provider. Our annotation provider reported self-disclosed genders and age brackets of annotators, but this information was not used in our analyses. Our annotations focused on attributes of our NPOV Response Generator query-response pairs, collecting annotation labels but no other data generated by the annotators. To reduce annotation bias, annotators were not told how the labeled examples would be used, and they were not told that the response was machine-generated. See Figure 6 for a screenshot of the annotation tool used by the human annotators.

A.3 Prompt-Tuning

This section discusses implementation details of (1) the NPOV Response Generator and (2) the hallucination and coverage error classifiers, which are both based on prompt-tuning an LLM. We use the same prompt-tuning settings for both.

We deliberately refrain from resource-intense hyperparameter tuning and instead use configurations previously shown to work well (Mozes et al., 2023). We use soft prompt lengths of 5 tokens initialized with a random sample of the model's 5K most frequent token vocabulary embeddings (Lester et al., 2021). We train with a learning rate of 0.1 with 500 warm-up steps and linear decay. We use small batch sizes of 16 for training and limit training to 20K steps. In most cases, we reach the maximum development set performance after 2-5K steps. Prompt-tuning runs take a maximum of 4 hours per run on 64 TPUv4 chips.

For the task representations, we utilize a "curly braces format" to verbalize the task, consisting of several key-value pairs in the input and target sequence for the LLM. This format is easily picked up by modern LLMs, as they have typically been exposed to code during pre-training. Figure 3 shows how we format the task for the NPOV Response Task (§2.1). Figure 4 shows how we format the error classification tasks (§3.3).

A.4 Annotated Example Counts

Table 6 shows the number of annotated examples from Steps 1 through 3 in §4.1, along with the synthetically-generated errors from §4.2. We report the numbers of annotated error-free examples and errors of different types for the train, dev, and test set splits.

A.5 Test Set Details

In §5, we report results on four different test set slices:

- **Full organic**: unparaphrased error-free examples from §4.1 Step 1 (base annotations), vs. organic full errors from §4.1 Steps 1 (base annotations) and 2 (extra organic errors).
- **Unparaphrased synthetic**: unparaphrased error-free examples from §4.1 Step 1 (base annotations), vs. corresponding examples with synthetically-generated errors.
- **Paraphrased synthetic**: paraphrased error-free examples from §4.1 Step 3 (paraphrasing), vs. corresponding examples with synthetically-generated errors.
- **Ambiguous organic**: unparaphrased error-free examples from §4.1 Step 1 (base annotations), vs. ambiguous organic errors from §4.1 Steps 1 (base annotations) and 2 (extra organic errors). Ambiguous errors are defined in §3, including partial errors, repetition, and perspective confusion.

Dataset sizes and distributions of errors for each test set slice are reported in Table 7.

A.6 Classifier Ablation: Annotation-Free Scenario

As an additional experiment, we analyze whether we can obtain good classifiers for hallucination and coverage detection by just re-utilizing the original training data from the NPOV Response Task, without the need to perform any of the manual annotations described in §4.1. We turn the data used to train the NPOV Response Generation into error classifier training data by (1) treating NPOV Response Task training examples as no error-examples, and (2) adding synthetic errors according to our procedure in §4.2. We call this approach "annotation-free" as we do not have to create any additional human annotations for classifier training. The resulting hallucination and coverage error classifiers are trained on 50 error-free examples and 131 examples with synthetic errors.

Table 4 shows results on the organic test sets for the "annotation-free" classifiers. Overall, these results are significantly worse than results with the non-annotation-free classifiers (compare to Table 2), and often worse than other data-free approaches (compare to ROUGE and salience in Table 1).

Test Set	Hallucination	Coverage
Full Organic	0.739	0.896
Ambiguous organic	0.732	0.804

Table 4: Annotation-free classifer ROC AUC scores.

A.7 Salience Formulas

In §3.2, we describe how we compute a word-to-word salience map $M_{words} \in \mathbb{R}^{m \times n}$, where m is the number of non-stop words in the input arguments and n is the number of non-stop words in the generated NPOV response. Our salience maps are based on gradient times input attribution scores, but we obtain comparable results using gradient L2 norms. Here, we include equations defining our hallucination and coverage error detection metrics based on M_{words}.

Assume I_{pro} and I_{con} are the sets of non-stop words in the input pro and con arguments respectively. Assume O_{resp} is the set of non-stop words in the generated NPOV main response. For each input word $w_i \in I_{pro} \cup I_{con}$, we define its contribution score α_i as its maximum contribution to any response word (i.e. the maximum across the corresponding row of M_{words}):

$$\alpha_i = \max(M_{words}[i, :])$$

Similarly, for each output word $w_j \in O_{resp}$, we define its attribution score β_j as its maximum attribution from any input argument word (i.e. the maximum across the corresponding column of M_{words}):

$$\beta_j = \max(M_{words}[:, j])$$

Sample contribution and attribution scores for input words and response words respectively are shown in Figure 2. For word-level error detection (§5.3),

Input Sequence:
User question: {Should abortion be legal?}
Topic: {abortion}
Perspective #1: {pro: Abortion bans deny bodily autonomy, creating wide-ranging repercussions. pro: Abortion is a safe medical procedure that protects lives.}
Perspective #2: {con: Life begins at conception, making abortion murder. con: Increased access to birth control, health insurance, and sexual education would make abortion unnecessary.}
Neutral response opening: {

Target Sequence:
This is a controversial topic and people hold different perspectives on it. Let me show a few…}
Neutral response core: {People in support of abortion bans for example claim that they deny bodily autonomy, which has wide-ranging repercussions. They also say that abortion would be safe medical procedure that protects lives. On the other hand, those who oppose abortion bans claim that abortion is murder because life begins at conception. Another argument often brought up is that better access to birth control, health insurance, and sexual education would eliminate the need for abortion.}

Figure 3: Task format for the NPOV Response Task.

Coverage Detection	Hallucination Detection
Input Sequence **User question**: {Should abortion be legal?}. **Neutral response**: {This is a controversial topic and people hold different perspectives on it. Let me show a few… People in support of abortion bans for example claim that they deny bodily autonomy, which has wide-ranging repercussions. They also say that abortion would be safe medical procedure that protects lives. On the other hand, those who oppose abortion bans claim that abortion is murder because life begins at conception. Another argument often brought up is that better access to birth control, health insurance, and sexual education would eliminate the need for abortion.}. **Given pro arguments**: {pro: Abortion is a safe medical procedure that protects lives. pro: Abortion bans deny bodily autonomy, creating wide-ranging repercussions.}. **Given con arguments**: {con: Increased access to birth control, health insurance, and sexual education would make abortion unnecessary. con: Life begins at conception, making abortion murder.}. **All the given arguments are covered by the neutral response**: { **Target Sequence** YES}	**Input Sequence** **User question**: {Should abortion be legal?}. **Neutral response**: {This is a controversial topic and people hold different perspectives on it. Let me show a few… People in support of abortion bans for example claim that they deny bodily autonomy, which has wide-ranging repercussions. They also say that abortion would be safe medical procedure that protects lives. On the other hand, those who oppose abortion bans claim that abortion is murder because life begins at conception. Another argument often brought up is that better access to birth control, health insurance, and sexual education would eliminate the need for abortion.}. **Given pro arguments**: {pro: Abortion is a safe medical procedure that protects lives. pro: Abortion bans deny bodily autonomy, creating wide-ranging repercussions.}. **Given con arguments**: {con: Increased access to birth control, health insurance, and sexual education would make abortion unnecessary. con: Life begins at conception, making abortion murder.}. **Only given arguments are contained in the neutral response**: { **Target Sequence** YES}

Figure 4: Task format for LLM-based error classifiers.

these word-level scores can be converted into coverage error scores $1.0 - \alpha_i$ and hallucination scores $1.0 - \beta_j$.

For example-level error detection (§5.1), we compute an example-level coverage error score by (1) taking the geometric mean of word-level contribution scores for each input perspective, (2) taking the minimum of the two perspective scores (to reflect the fact that both perspectives must contribute), and (3) subtracting from 1.0 (lower contributions are more likely to be coverage errors):

$$s_{cov} = 1.0 - \min\Big(\mathrm{gmean}_{w_i \in I_{pro}}(\alpha_i), \\ \mathrm{gmean}_{w_i \in I_{con}}(\alpha_i)\Big)$$

We compute an example-level hallucination score by (1) taking the geometric mean of word-level attribution scores in the NPOV main response, and (2) subtracting from 1.0 (lower attributions are more likely to be hallucinations):

$$s_{hall} = 1.0 - \mathrm{gmean}_{w_j \in O_{resp}}(\beta_j)$$

Note that $s_{cov}, s_{hall} \in [0, 1]$ because entries of M_{words} are in $[0, 1]$. We evaluate these hallucination and coverage error scores for example-level error detection in §5.1.

Split	# of topics	Topics
Train	9	*Animal Dissection; Concealed Handguns; Cuba Embargo; Filibuster; Free College; GMOs (Genetically Modified Organisms); Net Neutrality; Obesity; Vaping E-Cigarettes*
Dev	28	*Binge-Watching; Cancel Culture; Churches and Taxes; College Education; Corporal Punishment; Daylight Saving Time; Dress Codes; Electoral College; Employer Vaccine Mandates; Fighting in Hockey; Golf; Homework; Kneeling during National Anthem; Marijuana (CBD) for Pets; Olympics; Penny; Pit Bull Bans; Pokémon; School Vouchers; Space Colonization; Standardized Tests; Student Loan Debt; Tablets vs. Textbooks; Teacher Tenure; Uber & Lyft; US Supreme Court Packing; Video Games and Violence; Zoos*
Test	35	*Abortion; American Socialism; Animal Testing; Artificial Intelligence; Banned Books; Bottled Water Ban; Cell Phone Radiation; Climate Change; Corporate Tax Rate; DACA & Dreamers; DC and Puerto Rico Statehood; Defund the Police; Drone Strikes Overseas; Fracking; Gold Standard; Gun Control; Historic Statue Removal; Mandatory National Service; Minimum Wage; OTC Birth Control; Paying College Athletes; Police Body Cameras; Prescription Drug Costs; Private Prisons; Recreational Marijuana Legalization; Reparations for Slavery; Right to Health Care; Sanctuary Cities; Saturday Halloween; School Uniforms; Social Media; Social Security Privatization; Universal Basic Income; Vaccines for Kids; Vegetarianism*

Table 5: ProCon topics assigned to the different dataset splits.

Annotation set	Split	Topics	Examples	Error-free	Hall. only	Cov. only	Both errors
Step 1: Base Unparaphrased							
	Train	9	26	21	1	4	0
	Dev	28	76	61	9	6	0
	Test	35	93	78	9	6	0
Step 2: Extra Organic Errors							
	Train	9	35	-	14	15	6
	Dev	26	128	-	65	35	28
	Test	32	130	-	58	38	34
Step 3: Base Paraphrased							
	Train	9	21	21	-	-	-
	Dev	25	54	54	-	-	-
	Test	35	77	77	-	-	-
Synthetic Errors: Unparaphrased							
	Train	9	50	-	13	15	22
	Dev	22	123	-	33	40	50
	Test	31	165	-	48	47	70
Synthetic Errors: Paraphrased							
	Train	9	47	-	13	15	19
	Dev	21	116	-	32	36	48
	Test	31	166	-	48	46	72

Table 6: Numbers of annotated examples from the annotations in §4.1 (Steps 1 through 3) and the synthetic error generation in §4.2.

Test set	Examples	Error-free	Hall. only	Cov. only	Both errors
Full organic	173	78	49	32	14
Unparaphrased synthetic	243	78	48	47	70
Paraphrased synthetic	243	77	48	46	72
Ambiguous Organic	128	78	18	12	20

Table 7: Statistics for our four different test set slices. All errors are full errors, except in the ambiguous organic dataset.

Figure 5: Screenshot of the ProCon page on abortion (ProCon.org, 2022b); pink boxes show the first argument scraped for the pro and con perspectives. The scraped user query is the header "Should abortion be legal?".

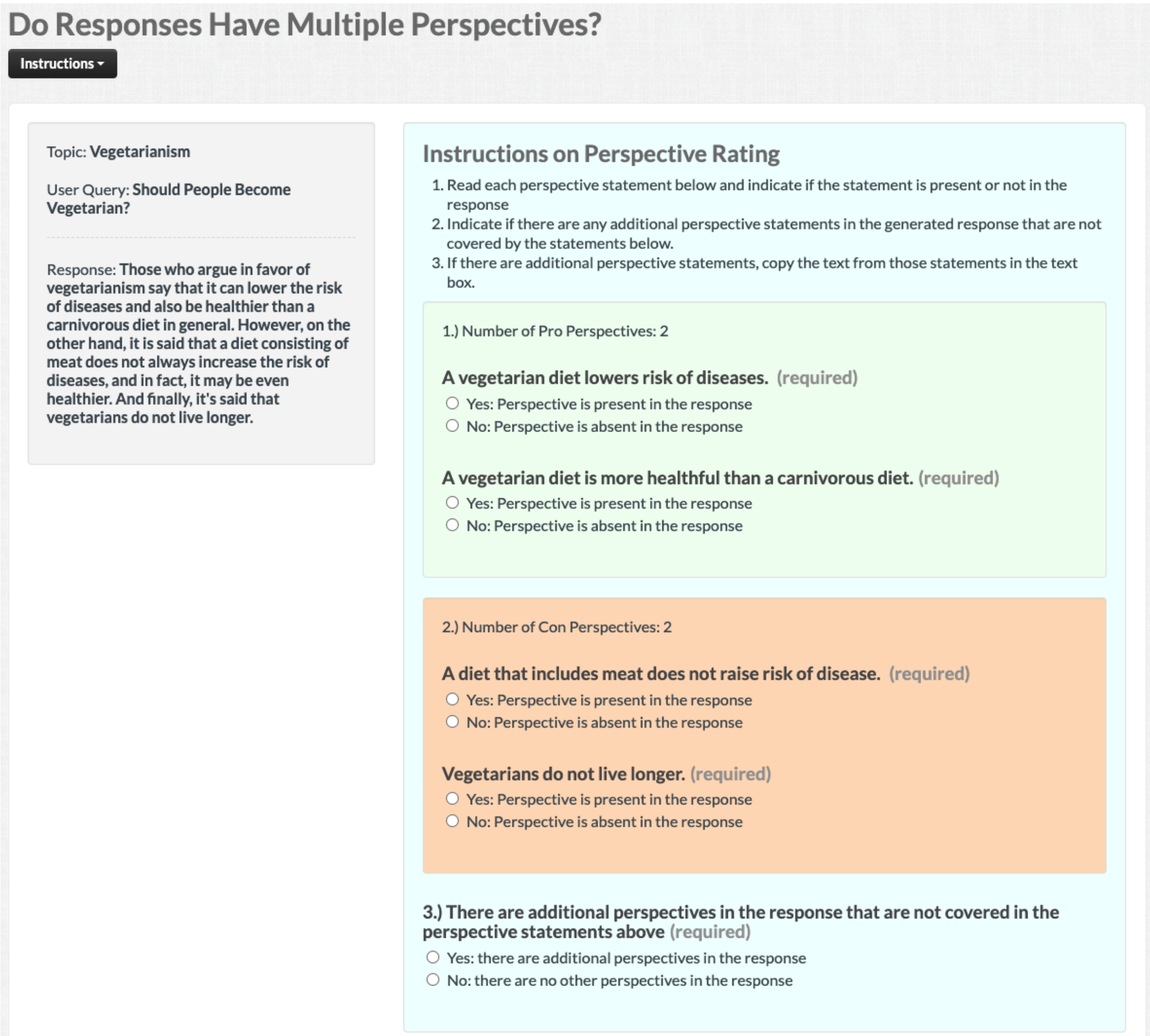

Figure 6: Screenshot of the annotation tool for the human annotators in §4.1.1. This example is on the topic *Vegetarianism* (ProCon.org, 2022c).