Is GPT-3 Text Indistinguishable from Human Text? SCARECROW: A Framework for Scrutinizing Machine Text

Yao Dou[*†] **Maxwell Forbes**[*†] **Rik Koncel-Kedziorski**[†] **Noah A. Smith**[†‡] **Yejin Choi**[†‡]
[†]Paul G. Allen School of Computer Science & Engineering, University of Washington
[‡]Allen Institute for AI
{douy,mbforbes,nasmith,yejin}@cs.washington.edu kedzior@uw.edu

Abstract

Modern neural language models can produce remarkably fluent and grammatical text. So much, in fact, that recent work by Clark et al. (2021) has reported that conventional crowdsourcing can no longer reliably distinguish between machine-authored (GPT-3) and human-authored writing. As errors in machine generations become ever subtler and harder to spot, it poses a new challenge to the research community for robust machine text evaluation.

We propose a new framework called SCARECROW for scrutinizing machine text via crowd annotation. To support the broad range of real machine errors that can be identified by laypeople, the ten error categories of SCARECROW—such as **redundancy**, **commonsense errors**, and **incoherence**—are identified through several rounds of crowd annotation experiments without a predefined ontology.

We then use SCARECROW to collect over 41k error spans in human-written and machine-generated paragraphs of English language news text. We isolate factors for detailed analysis, including parameter count, training data, and various decoding-time configurations. Our approach successfully quantifies measurable gaps between human authored text and generations from models of several sizes, including fourteen configurations of GPT-3. In addition, our analysis unveils new insights, with detailed rationales provided by laypeople, e.g., that the commonsense capabilities have been improving with larger models while math capabilities have not, and that the choices of simple decoding hyperparameters can make remarkable differences on the perceived quality of machine text. We release our training material, annotation toolkit and dataset at **https://yao-dou.github.io/scarecrow/**.

1 Introduction

Clark et al. (2021) demonstrated the challenges of human evaluation in the era of GPT-3 (Brown et al., 2020), as crowd workers are no longer able to reliably distinguish GPT-3's generations from human-written text.

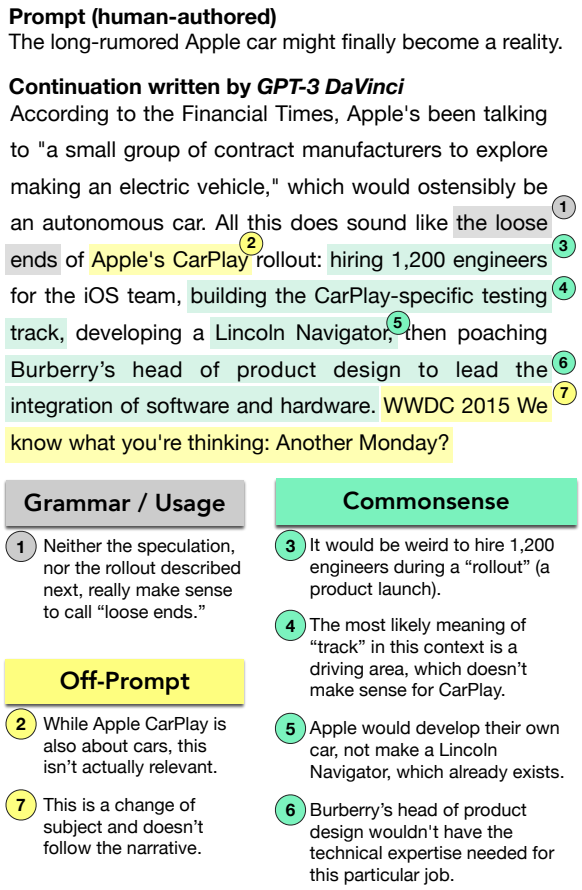

Figure 1: After a model (here, GPT-3 DaVinci) has read the prompt (top sentence) and generated a continuation (next paragraph), the SCARECROW annotation framework provides a systematic way for humans to mark issues throughout the text and explain what is wrong. Our own annotations are pictured here.

Or are they? In this paper, we propose a new framework for systematically scrutinizing machine text so that even crowd workers, despite the known challenges reported by recent literature, can successfully critique seemingly fluent generations. We not only quantify a measurable gap between ma-

[*]Equal contribution

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 7250 - 7274

ERROR TYPE	DEFINITION	EXAMPLE
Language Errors		
Grammar and Usage	Missing, extra, incorrect, or out of order words	. . . explaining how cats feel **emoticons** . . .
Off-Prompt	Generation is unrelated to or contradicts prompt	**PROMPT:** Dogs are the new kids. **GENERATION:** Visiting **the dentist can be scary**
Redundant	Lexical, semantic, or exeessive topical repetition	Merchants worry about **poor service** **or service that is bad** . . .
Self-Contradiction	Generation contradicts itself	Amtrak plans to **lay off many employees,** though **it has no plans cut employee hours.**
Incoherent	Confusing, but not any error type above	Mary gave her kids cheese toast but **drew a map of it on her toast.**
Factual Errors		
Bad Math	Math or conversion mistakes	. . . *it costs over £1,000* ***($18,868)*** . . .
Encyclopedic	Facts that annotator knows are wrong	***Japanese Prime Minister Justin Trudeau*** *said Monday . . .*
Commonsense	Violates basic understanding of the world	The dress was made at the **spa.**
Reader Issues		
Needs Google	Search needed to verify claim	***Jose Celana, an artist based in Pensacola, FL*** , . . .
Technical Jargon	Text requires expertise to understand	. . . *an 800-megawatt* ***photovoltaic*** *plant was built . . .*

Table 1: Error types in the SCARECROW framework, grouped into three categories. The categories are explained further in §4.4, and detailed definitions and examples for each error type is provided in Appendix A.

chine text and human text, but reveal the distributions of specific categories of issues, and pinpoint their occurrences in text written by several sizes of language models as well as humans.

To achieve this, we develop SCARECROW, a methodology for eliciting categorical judgements of errors in machine-generated text from crowd workers. One goal in natural language generation (NLG) is to produce fluent outputs which can be read by laypeople. As such, we propose that important errors to address are those which are recognized by readers without NLP expertise. Our framework allows crowd workers to annotate problems in model outputs at the span level. A single such annotation is shown in Figure 1.

To make this possible, we establish a categorization of shortcomings commonly found in machine generated text (Table 1). This error schema covers a broad scope of problems as identified by experts, but has been honed according to what is salient to non-expert readers through several pilot rounds of crowd annotation without a fixed label set. The result is a framework that is usable by everyday people with minimal training, but covers the error phenomena found in real machine-generated text. Labeling spans of text using specific error types creates a picture of contemporary model generations with an unprecedented level of detail. In contrast to judging text holistically (Celikyilmaz et al., 2021), insights from this method are specific and practical, as it measures exactly how and where problems arise.

We conduct a large-scale analysis of human-written and machine-generated text using SCARECROW, collecting 13k annotations of 1.3k paragraphs, amassing 41k spans labeled with error type, severity, and an explanation. Through this, we characterize in which ways GPT-3's generations are better than those of previous models, and which aspects do not improve with increased data and parameters. We also provide a rigorous error analysis of text generated by several other contemporary language models, examining the impact of model size, training data, and decoding strategy.

We provide our detailed annotator training system and task interface so that future researchers may employ and refine them for error analyses of machine-generated text. We hope this will contribute to the standardization of NLG human evaluation (Howcroft et al., 2020).

2 Key Findings

We perform a large-scale annotation of errors in English news text generated by five sources (four

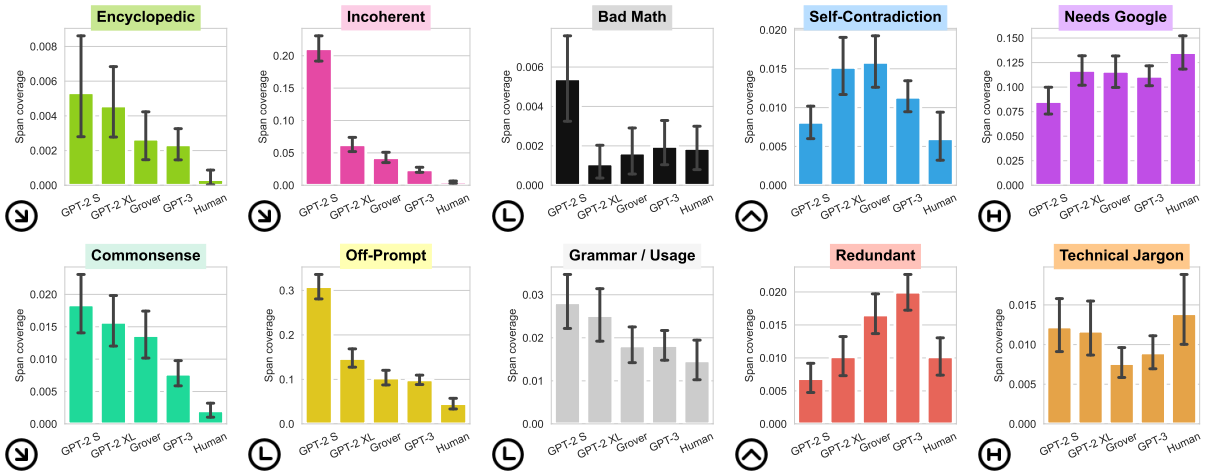

Figure 2: Average portion of tokens annotated with each error type (y-axis) across models (x-axis), with 95% confidence intervals. We group the trends into several broad categories. **Decreasing:** fine-tuning and increasing model size improves performance. **Model plateau:** increasing model size to GPT-3 does not correlate with further improvements. **Rising and falling:** errors become more prevalent with some models, then improve. **Humans highest:** these spans are labeled most on human-authored text; both are *reader issues* (distinct from *errors*; see Table 1). Details: all models, including GPT-3, use the same "apples-to-apples" decoding hyperparameters: top-p=0.96, temperature=1, and no frequency penalty.

models and ground truth articles). We present Figures 2, 3, and 4 as summaries of our main results. As a reminder to readers, Grover (Zellers et al., 2019) is the same model size and architecture as GPT-2 XL (Radford et al., 2019), but trained in-domain (on news text). As such, our results cover three increasing model sizes (GPT-2 Small, XL, and GPT-3 (Brown et al., 2020)), one change in domain (Grover), and ground-truth text (Human). For GPT-3, we also study a variety of decoding configurations (Figure 4).

The main quantity we measure (on y-axes) is *span coverage*, which is the average portion of tokens that ends up covered by annotations of a particular error type. Since it is possible that multiple spans nest or overlap, there is no upper bound for this quantity. (See Figure 12 for a comparison of span coverage with other measurement alternatives.) Figure 2 measures span coverage for each type of span separately, Figure 3 stacks them, and Figure 4 removes non-error spans (reader issues) before adding them (as in Figure 3, but without showing the individual types).

The following are our key findings.

1. Scaling pays off to improve Encyclopedic, Commonsense, and Incoherent errors (Fig. 2). These error categories **decrease** with in-domain training (Grover) and larger model size (GPT-3). Human text still shows the fewest of these kinds of errors.

2. Scaling benefits plateau for Off-Prompt, Bad Math, and Grammar and Usage errors (Fig. 2). These three error categories see a **model plateau** in error reduction when scaling to GPT-3. Of these error types, humans still commit fewer **Off-Prompt** (more: §E.1) and **Grammar and Usage** errors, but **Bad Math** appears saturated for our domain.

3. Self-Contradiction and Redundant errors exhibit more complex scaling behavior (Fig. 2). We roughly categorize these trends as **rising and falling**: increasing for medium or large-scale models, but dropping for human-authored text. Text generated by GPT-2 Small is so often incoherent that there is little possibility for **Self-Contradiction** (more: §E.2), and the increase in **Redundant** errors varies based on how errors are counted (more: §E.3).

4. Human-authored text produces the most reader issues (Figs. 2 and 3). The **Needs Google** and **Technical Jargon** span categories both have a **humans highest** trend, and both fall under *reader issues*: problems that are not necessarily *errors*, but that still prevent full comprehension

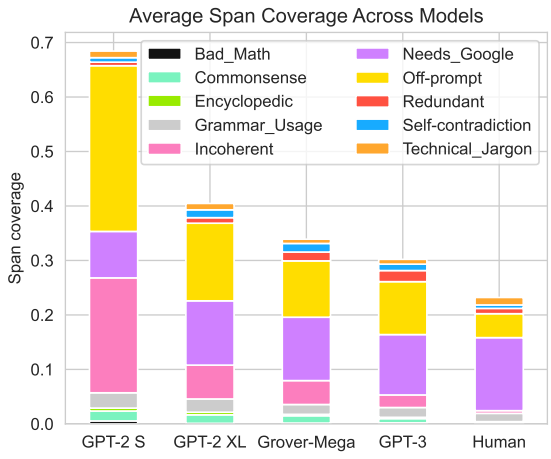

Figure 3: Average portion of tokens covered by span annotations, broken down by error type. All models, including GPT-3, use the same apples-to-apples decoding hyperparameters: top-p=0.96, temperature=1, and no frequency penalty. We scale each span by its token length, normalize by generation token lengths, and remove severity-1 **Grammar and Usage** errors (see §C).

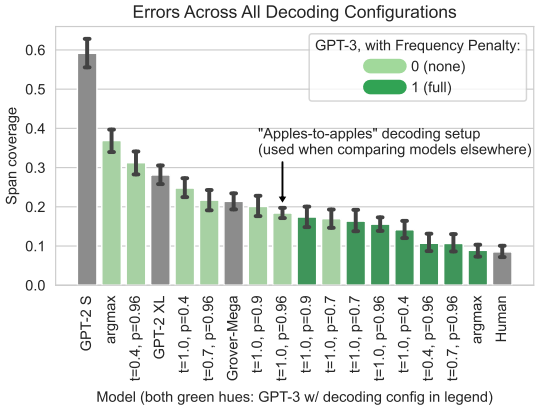

Figure 4: Taking the average span coverage (Figure 3) and removing reader issues (**Technical Jargon** and **Needs Google**), we plot values and 95% confidence intervals for all models, including all decoding hyperparameters we tested for GPT-3. We find a surprisingly large change in annotated errors depending on the decoding setting used.

or factual verification of the text (more: §E.4).

Furthermore, human-authored text is not free from error annotations (Figure 3). This can serve either as a control for baseline error rates (more: §E.6), or as a mechanism for critiquing human writing.

5. Decoding hyperparameters have a huge impact (Figure 4). For the previous findings, we fix the sampling configuration for all models to an apples-to-apples setup for fair comparison: top-p = 0.96, (softmax) temperature = 1, and no frequency penalty (i.e., word repetition penalty; defined precisely in §5.2, Equation 1). To study the effects of these decoding settings, we annotate text generated by GPT-3 using a variety of values for top-p and temperature, both with and without a frequency penalty.

To our surprise, the decoding hyperparameters considerably affected error rates (more: §E.5). As seen in Figure 4, the worst sampling procedure for GPT-3 (argmax sampling with no frequency penalty) performed even worse than GPT-2 XL. But the best sampling procedure (surprisingly, also argmax sampling, but with a frequency penalty) produced text with as few apparent SCARECROW error spans as those authored by humans (more: §E.6).

All of these findings are discussed in more detail in Appendix E.

3 Evaluation of Natural Language Generation

We make our study in the area of open-ended natural language generation, a loose term for generating longer texts with an increased level of creative freedom. The common factor in all open-ended generation tasks such as story, blog, and dialog generation is the wide and diverse nature of target outputs. Lexically and even semantically dissimilar responses to the same prompt could be equally valid. For example, a model prompted with the blog title "Recipes for success this Holiday season" could describe how to roast a turkey or strategies for dealing with the stresses of holiday travel.

This allowable variation poses a particular difficulty for the evaluation of generation systems. Traditionally, text generation quality for tasks like machine translation or graph-to-text generation has been measured by word overlap with human-authored references (Papineni et al., 2002; Lin, 2004). Though measures like BLEU allow for multiple references, they break down when the space of allowable outputs is large, as in open-ended generation. Recently introduced metrics seek to remedy this problem (Hashimoto et al., 2019; Pillutla et al., 2021), but the gold standard for evaluating generated text is still human judgment.

However, current approaches to eliciting human

judgement of generated text often do not provide detailed insight into where models are making progress, where they are failing, and the scope of these failures. A/B-style testing allows for directly comparing one system against others (Clark and Smith, 2021), but can only express relative improvements. Simple Likert scale judgements can assess text quality, but do not explain why a generated text receives a given rating, or which segment of the text is problematic. Insights into model failures often come instead from a small scale expert analysis of outputs. However, these "error analyses," once a staple of NLP research, have become less common in recent years, perhaps due to their small size and high variance.

A hypothesis of the current work is that a well designed error analysis annotation framework could be used by crowdworkers to annotate large amounts of text, thereby providing detailed information about model progress and failures as well as actionable directions for future research. Such a framework would be easy to learn, reusable, and independent of particular models or experimental conditions. In what follows, we outline the details of such a method.

4 SCARECROW Annotation Methodology

This section describes the high-level annotation methodology for SCARECROW.

4.1 Prompt and Generation

Our annotations consider two segments of text: a one-sentence prompt, and a one-paragraph generation. The prompt is human-written. It provides both starting tokens for model generation, as well as context for humans to evaluate whether a model is able to stay on-prompt—both topically and factually. Annotators know that the prompt is written by a human.

The generation is either text sampled from a language model, or the human-authored continuation to the prompt. Annotators, who do not know whether the generation came from a model or humans, assess this text. A paragraph length (80–145 tokens) is chosen to balance expressiveness with scope. For expressiveness, models must be given a sufficient number of tokens to express their capabilities lexically, syntactically, and semantically. One paragraph allows for significantly more variation than a single sentence. On the other hand, assessing multiple paragraphs is challenging, both as a crowdsourcing task itself, and because it broadens the kinds of errors to include larger narrative scope. We leave extensions of SCARECROW to longer narrative lengths for future work.

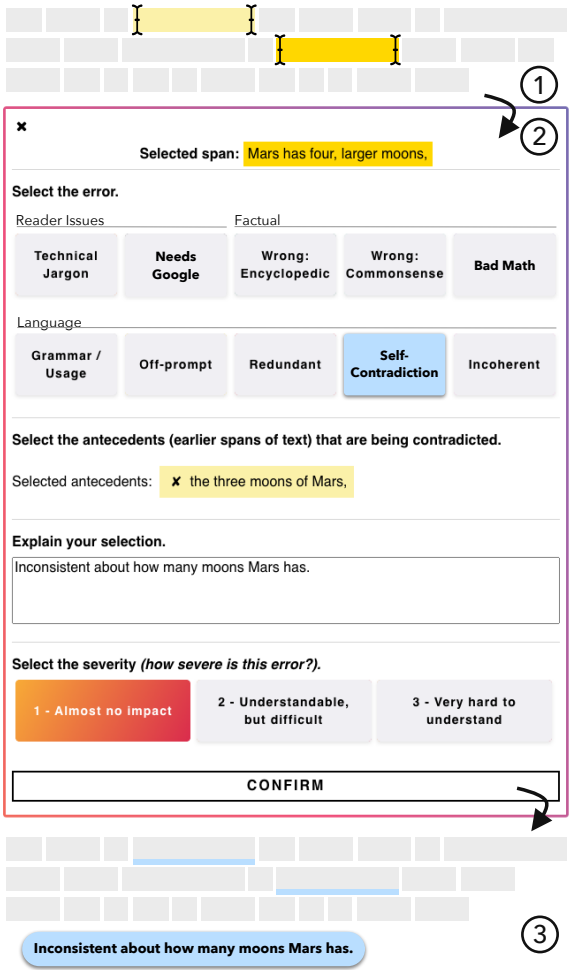

Figure 5: SCARECROW interface for annotating a single span: (1) highlighting a span (and later, an antecedent); (2) completing the annotation, with the error type, explanation, and severity; (3) the error annotation is saved—interactive controls allow detailed viewing and editing of spans (not shown).

4.2 Span Labeling

Annotators select spans that contain problems in the generation. The spans are automatically snapped to word boundaries. We choose spans to balance specificity (i.e., vs. simply commenting on the text as a whole) with ease of use (vs. imposing a more structured annotation schema).

4.3 Span Selection

We instruct workers to select the smallest span—minimally a single word—that contains an issue. Sometimes this involves an entire phrase, sentence,

or multiple sentences. We aim for specificity because during aggregation, it is possible to "back off" annotations to larger spans, but not the inverse.

Once they select a span, workers (1) label the error type, (2) choose a severity level, and (3) explain their reasoning behind the error. Workers use the annotation interface shown in Figure 5 to mark a span with these three steps. We describe each step in greater detail in the next three sections.

4.4 Error Types

Each selected span is labeled with exactly one error type. Multiple errors may be marked with partially or fully overlapping spans in the case that one text segment contains multiple problems.

We chose ten error types to balance three criteria: linguistic analysis, observed errors in generated text, and capabilities of everyday people with one to two hours of training.[1] We developed the schema by starting with the first two criteria (linguistic analysis and observed errors), and refining it over several pilot annotation studies, with 30 crowd workers performing 750 total annotations of 60 paragraphs before beginning data collection.

We broadly group the errors into three categories: language errors, factual errors, and reader issues. Language errors are issues with internal and external structure of text: which ideas are expressed, and whether they are expressed coherently and consistently. Factual errors denote that the information presented is known to be incorrect. Reader issues, on the other hand, are cases where the text is too technical or obscure to assess its factuality. Hence, reader issues are not errors, per se, but regions where a reader would need assistance outside of the text itself for comprehension.

We present the ten error types in Table 1 (several pages back). Appendix A provides more details, examples, and explanations for all error types.

4.5 Severity

Errors naturally vary in how jarring they are to a reader. We define three error severity levels, and ask annotators to pick one for each error.

The severity levels are as follows. **(1)** Almost no impact on quality; just a small problem. **(2)** Understandable, but difficult; what's written is still comprehensible, but there's clearly an issue. **(3)** Very difficult to understand; the error almost completely ruins the text.

We provide examples of each severity in Appendix B.1. In this paper, we omit an analysis of the severity labels (except for an illustration in Figure 12), but include it in our data release for future work to explore.

4.6 Explanation

Finally, we ask annotators to explain their reasoning behind each error in natural language. We provide example explanations during training, but do not impose strict guidelines. This paper primarily focuses on quantitative error analysis, but we anticipate the error explanations may warrant future investigation.

4.7 Annotation Process

We use Amazon Mechanical Turk (AMT) for all data collection.

Training We first pay each worker $40 to take an extensive qualification task, which both trains them in the span categorization scheme and quizzes their understanding. We pass workers if they score ≥ 90 points out of 100 points (details in Appendix B.2).

Annotation Workers annotate each paragraph using a custom annotation interface (shown partially in Figure 5), for which we pay $3.50. We calculated $3.50 per annotation by aiming to pay workers at least $15/hour. After several annotation rounds, we observed considerable variation in time per annotation,[2] so this cost should not be necessarily seen as a requirement for SCARECROW annotations.

5 Data Collection

We collect 13k human annotations of 1.3k paragraphs using SCARECROW, resulting in over 41k spans.

5.1 Models

We consider four model configurations to test recent state-of-the-art transformer-based (Vaswani et al., 2017) models.

GPT-2 Small (Radford et al., 2019) The 117M parameter variant of GPT-2, which is pretrained on WebText, without additional fine-tuning.

GPT-2 XL (Radford et al., 2019) The 1.5B parameter variant of GPT-2, (WebText, no fine-tuning).

[1] The complete training material is available for download.

[2] Median: 212s, mean: 265s, std. dev.: 199s.

Grover-Mega (Zellers et al., 2019) The 1.5B parameter variant of Grover, a model with the same architecture and parameter count of GPT-2, trained on news articles and their metadata.

GPT-3 DaVinci (Brown et al., 2020) The 175B parameter variant of GPT-3, which is trained on a version of the Common Crawl web scrape with additional filtering and deduplicating.

In addition, we also use the actual human-written text from the data sources we draw from, which we denote as **Human**.

5.2 Decoding strategies

We consider three main hyperparameters when sampling from models: p for *top-p* or *nucleus sampling* (Holtzman et al., 2020), an alternative to *top-k*;[3] t for the *softmax temperature*; and *f.p.* for *frequency penalty*. The frequency penalty scales a token's likelihood based on how many times it was already generated by applying the following modification to the model's output:

$$\ell_i(t) \leftarrow \ell_i(t) - c_{<i}(t) \cdot \alpha_f \quad (1)$$

where $\ell_i(t)$ is the model's output for token t at the i-th position,[4] $c_{<i}(t)$ is the count of token t's sampled occurrences prior to the i-th position, and α_f is the frequency penalty. We omit studying *presence penalty*, another hyperparameter offered for GPT-3, simply due to annotation budget constraints.

To compare models as consistently as possible, we set identical decoding strategies for our primary data collection. We refer to this as the "apples-to-apples" decoding setup throughout the paper:

$$p = 0.96 \qquad t = 1.0 \qquad \textit{f.p.} = 0$$

However, we also wish to study the effects of these decoding strategies. We annotate generations from the strongest available model (currently, GPT-3) varying the following parameters:

$$p \in \{0.4, 0.7, 0.9, 0.96\}$$
$$t \in \{0.0 \textit{ (argmax)}, 0.4, 0.7, 1.0\}$$
$$\text{f.p.} \in \{0 \text{ (none)}, 1 \text{ (full)}\}$$

For budget reasons, we only vary p and t independently—i.e., we set $p = 0.96$ when varying t, and $t = 1.0$ when varying p.

5.3 Prompt Selection

We use news articles as the sources of prompts for models to condition on for generation. Specifically, we use news articles found in the Common Crawl. We select the first sentence as the prompt.

Our use of news text is constrained by two factors. First GPT-3 is trained on the Common Crawl, from 2016 through 2019. We wish to avoid testing GPT-3 by generating from articles it saw during training, due to the possibility of copying (Carlini et al., 2021). Second, news articles began heavily covering the COVID-19 pandemic beginning around February 2020. Though testing models' capabilities to generate text about unseen events is a valuable line of study, the distribution shift caused by COVID-19 in news writing about all aspects of life is difficult to overstate.

As such, to make the comparison more amenable to models' training data, we consider news articles from January 2020. We select articles where there is a known topic—such as *Food* or *Sports*—from the Common Crawl metadata, to allow for studying any effect of coarse-grained subject.

5.4 Generation

We generate between 80 and 145 tokens[5] from each model as a continuation to the first sentence of the news article. We stop generating when we heuristically detect the first sentence boundary after 80 tokens. If the model does not end a sentence between 80 and 145 tokens, we sample again. For the *Human* setting, we use the remainder of the article, similarly stopping after the first sentence boundary after 80 tokens.

5.5 Annotation

Crowdsourcing Workers first complete training and qualification tasks. We provide more details in 4.7. From pilot studies, we discovered that each error, depending on its severity and clarity, has only a

[3] We omit separate studies of *top-k*, due to results presented by Holtzman et al. (2020), and OpenAI's removal of *top-k* from the GPT-3 API.

[4] While $\ell_i(t)$ is defined to be *"logits (un-normalized log-probabilities),"* because it is un-normalized, we anticipate that it is simply the model's output before the $\log(\text{softmax}(\cdot))$ is applied. See OpenAI's description of frequency and presence penalties: https://beta.openai.com/docs/api-reference/parameter-details

[5] Counted by Stanza tokenization (Qi et al., 2020), not byte-pair encoding (BPE) or whitespace-separated tokens.

low to moderate chance of being identified by each worker. However, most worker-identified errors were truly problems. In other words, annotators labeled issues with high precision and low recall. To account for this, we have **10 workers** annotate each paragraph. We examine the agreement and variability of annotations in Appendix C.

Dataset statistics We provide detailed dataset statistics in Appendix D.

6 Error Prediction

A natural question is: using this data, can machines learn to detect and classify errors in machine generated text?

Task We frame this problem as a span classification task. Given a span from a generated text, the goal is to classify its error type or output "No Error" if there is none. Positive examples for each error class are taken from our data. We sample random spans that were not labeled with any error type as negative examples. To ensure a breadth of span lengths, we sample 3 negative spans for every length of error span in the generated text. We split the generated texts into train, development, and test sets using 1063 texts (28029 error spans), 100 texts (2538 spans) and 100 texts (2677 spans) respectively.

Model We use a standard span classification model inspired by Wadden et al. (2019). This model encodes every generated text using a pretrained language model (RoBERTa-large). Spans are represented with the final layer of this encoding. Following previous work, we concatenate the start and end tokens with a task-specific learned length embedding. The resulting vector is passed through a feedforward network which reduces its dimensionally to the number of error categories plus a "No Error" option. The resulting model has 357M trainable parameters. The model is trained to minimize the cross entropy of the correct span category. We train for 15 epochs using AdamW with a learning rate of 10^{-6}. We validate after each epoch and use the checkpoint with the lowest validation loss (epoch 8).

Evaluation To evaluate the error prediction model, we use per-token precision, recall, and F_1 score per error category. We classify every span up to length 30 in a generated text. We take as gold labels the aggregated human error spans collected in our data. In other words, models predict the combined spans of all 10 annotators. For comparison, we also report as *Human* the average metrics of one annotator versus the others (i.e., 1-vs-9).[6]

Error	Model			Human		
	P	R	F_1	P	R	F_1
Bad Math	–	0	–	0.72	0.14	**0.24**
Commonsense	0.77	0.06	**0.10**	0.17	0.02	0.04
Encyclopedic	–	0	–	0.22	0.03	**0.05**
Grammar and Usage	0.29	0.23	**0.26**	0.30	0.04	0.08
Incoherent	0.59	0.34	**0.43**	0.69	0.15	0.24
Off-Prompt	0.67	0.29	0.41	0.88	0.31	**0.46**
Redundant	0.23	0.82	0.36	0.88	0.35	**0.50**
Self-Contradiction	0.08	0.23	0.12	0.51	0.09	**0.16**
Technical Jargon	0.18	0.74	**0.29**	0.61	0.12	0.20
Needs Google	0.59	0.96	**0.73**	0.78	0.20	0.32

Table 2: Model prediction results against combined spans of 10 annotators, compared with humans scored as one-vs-rest (i.e., 1-vs-9). Bold F_1 scores denote the higher average; values marked "–" cannot be computed due to division by zero. **Takeaway:** Humans have higher precision in every error type except **Commonsense**, but relatively sparse annotations lead to lower computed recall. This allows the model to achieve higher F_1 scores for half of the span categories.

Results Table 2 shows the error prediction capability of this model in terms of precision and recall. As we noted earlier, a single human annotator can be thought of as a high precision, low recall judge. These results bear out this claim. For all but one category, humans have higher precision annotations. However, the models trained on the aggregation of human labels can achieve considerably higher recall. For half of the error categories, this leads to higher model F_1 scores than the human annotators.

We see that the model is successful at identifying information that human's would have to manually verify (**Needs Google**), achieving nearly perfect recall with precision close to 0.6. The model can also identify **Grammar and Usage**, **Incoherent**, and **Redundant** errors with higher recall than an individual human annotator, though at the cost of precision (sometimes in the .20s).

7 Related Work

Automated evaluation metrics such as BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), METEOR (Banerjee and Lavie, 2005), and BERTScore

[6]The difference in available references (10 for models, 9 for humans) mean this setup makes it easier for models to score higher in precision, and for humans to score higher in recall. Despite this, humans still achieve higher precision, and models still achieve higher recall.

(Zhang et al., 2019) compute a generation's score based on a (set of) reference(s). Their use is well-established in tasks like machine translation and summarization, but they are less helpful in open-ended text generation, where there is a vast diversity of possible high-quality continuations.

Recent studies propose automated metrics for open-ended text generation evaluation such as: Perception Score (Gu et al., 2021), which diffuses evaluation onto a multidimensional space and assigns a single score; UNION (Guan and Huang, 2020), which learns to distinguish human-written stories from negative samples by generating perturbations of human-written stories; and MAUVE (Pillutla et al., 2021), which compares the distribution of machine-generated text to that of human language.

An alternate recent approach to assessing open-ended text generation was presented in TuringAdvice (Zellers et al., 2021), where crowd workers assess machine-generated advice in response to Reddit posts. In their error analysis, Zellers et al. connect problems in generated text to core NLP tasks, such as **Self-Contradiction** errors as instances of failed natural language inference (Monz and de Rijke, 2001), or **Off-Prompt** errors as cases of failed reading comprehension (Richardson et al., 2013). While past work has attempted to guide text generation using discriminative models trained for such tasks (Holtzman et al., 2018), it remains an open challenge.

Comparative human evaluations of natural language generations ask annotators to rank system outputs relative to each other. Text is typically evaluated using a few global criteria, such as fluency and relevance, using discrete (e.g., 5-point) (Sai et al., 2020) or continuous scales (Novikova et al., 2018). Recent work even automates this approach, running a human evaluation alongside automatic metrics on leaderboard submissions (Khashabi et al., 2021). In the RoFT system (Dugan et al., 2020), annotators attempt to detect the boundary between human- and machine-written text as a proxy for assessing quality. Table 3 summarizes the differences between these schemes and SCARECROW. See Celikyilmaz et al. (2021) for a recent survey of text generation evaluation techniques across both human and automatic metrics.

While these approaches may be helpful—sometimes (Card et al., 2020)—at ranking systems, they do not give us insight into exactly *which* parts of a generation fall short, and *why*. One approach related to or annotation method is pursued by Wood et al. (2018), who develop a collaborative mobile app where users draw "graffiti" commentary on news articles. SCARECROW aims to assess model generations the way we would critique human-written text: by locating, coarsely categorizing, and explaining problems.

Method	GC	SET	DE	RR	EE	RS	SA
Likert-Scale	✓		✓			✓	
RankME	✓			✓		✓	
RoFT	✓		✓		✓		
SCARECROW		✓	✓		✓		✓

Table 3: Comparison of different natural language generation human evaluations. Here, **GC** : General Criteria, **SET** : Specific Error Type, **DE** : Direct Evaluation, **RR** : Relative Ranking, **EE** : Error Explanation, **RS** : Rating Scale, **SA** : Span Annotation.

8 Conclusion

We present SCARECROW, a method for identifying and explaining issues in generated text. Along with the annotation framework, we present an analysis of the SCARECROW method applied to several large neural language models in an open-ended news generation task. We release our data and methodology to the community.

Acknowledgments

The authors thank members of xlab for their feedback on this work. This research is supported in part by NSF (IIS-1714566), DARPA MCS program through NIWC Pacific (N66001-19-2-4031), DARPA SemaFor program, and Allen Institute for AI.

References

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization, pages 65–72.

Gwern Branwen. 2020. Gpt-3 creative fiction.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen,

Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Massimo Caccia, Lucas Caccia, William Fedus, Hugo Larochelle, Joelle Pineau, and Laurent Charlin. 2020. Language gans falling short.

Dallas Card, Peter Henderson, Urvashi Khandelwal, Robin Jia, Kyle Mahowald, and Dan Jurafsky. 2020. With little power comes great responsibility. In Proceedings of EMNLP.

Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom B. Brown, Dawn Xiaodong Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. 2021. Extracting training data from large language models. In USENIX Security Symposium.

Asli Celikyilmaz, Elizabeth Clark, and Jianfeng Gao. 2021. Evaluation of text generation: A survey.

Elizabeth Clark, Tal August, Sofia Serrano, Nikita Haduong, Suchin Gururangan, and Noah A. Smith. 2021. All that's 'human' is not gold: Evaluating human evaluation of generated text. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 7282–7296, Online. Association for Computational Linguistics.

Elizabeth Clark and Noah A. Smith. 2021. Choose your own adventure: Paired suggestions in collaborative writing for evaluating story generation models. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 3566–3575, Online. Association for Computational Linguistics.

Liam Dugan, Daphne Ippolito, Arun Kirubarajan, and Chris Callison-Burch. 2020. Roft: A tool for evaluating human detection of machine-generated text. arXiv preprint arXiv:2010.03070.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. arXiv preprint arXiv:2012.15723.

Herbert P Grice. 1975. Logic and conversation. In Speech acts, pages 41–58. Brill.

Jing Gu, Qing yang Wu, and Zhou Yu. 2021. Perception score: A learned metric for open-ended text generation evaluation. In AAAI.

Jian Guan and Minlie Huang. 2020. Union: An unreferenced metric for evaluating open-ended story generation. In EMNLP.

Tatsunori Hashimoto, Hugh Zhang, and Percy Liang. 2019. Unifying human and statistical evaluation for natural language generation. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 1689–1701, Minneapolis, Minnesota. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. 2018. Learning to write with cooperative discriminators. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1638–1649, Melbourne, Australia. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. International Conference on Learning Representations.

David M. Howcroft, Anya Belz, Miruna-Adriana Clinciu, Dimitra Gkatzia, Sadid A. Hasan, Saad Mahamood, Simon Mille, Emiel van Miltenburg, Sashank Santhanam, and Verena Rieser. 2020. Twenty years of confusion in human evaluation: NLG needs evaluation sheets and standardised definitions. In Proceedings of the 13th International Conference on Natural Language Generation, pages 169–182, Dublin, Ireland. Association for Computational Linguistics.

Daniel Khashabi, Gabriel Stanovsky, Jonathan Bragg, Nicholas Lourie, Jungo Kasai, Yejin Choi, Noah A. Smith, and Daniel S. Weld. 2021. Genie: A leaderboard for human-in-the-loop evaluation of text generation.

Klaus Krippendorff. 2018. Content analysis: An introduction to its methodology. Sage publications.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In Text summarization branches out, pages 74–81.

Hugo Liu and Push Singh. 2004. Conceptnet—a practical commonsense reasoning tool-kit. BT technology journal, 22(4):211–226.

Yann Mathet, Antoine Widlöcher, and Jean-Philippe Métivier. 2015. The unified and holistic method gamma (γ) for inter-annotator agreement measure and alignment. Computational Linguistics, 41(3):437–479.

Christof Monz and Maarten de Rijke. 2001. Lightweight entailment checking for computational semantics. In Proc. of the third workshop on inference in computational semantics (ICoS-3).

Jekaterina Novikova, Ondrej Dusek, and Verena Rieser. 2018. Rankme: Reliable human ratings for natural language generation. In NAACL.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In Proceedings of the 40th annual meeting of the Association for Computational Linguistics, pages 311–318.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Yejin Choi, and Zaid Harchaoui. 2021. Mauve: Human-machine divergence curves for evaluating open-ended text generation.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pages 101–108, Online. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. OpenAI blog, 1(8):9.

Laria Reynolds and Kyle McDonell. 2021. Prompt programming for large language models: Beyond the few-shot paradigm. In Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems, pages 1–7.

Matthew Richardson, Christopher JC Burges, and Erin Renshaw. 2013. Mctest: A challenge dataset for the open-domain machine comprehension of text. In Proceedings of the 2013 conference on empirical methods in natural language processing, pages 193–203.

Ananya B Sai, Akash Kumar Mohankumar, and Mitesh M Khapra. 2020. A survey of evaluation metrics used for nlg systems. arXiv preprint arXiv:2008.12009.

Roger C Schank and Robert P Abelson. 1977. Scripts, plans, goals, and understanding: An inquiry into human knowledge structures. Psychology Press.

Hadrien Titeux and Rachid Riad. 2021. pygamma-agreement: Gamma γ measure for inter/intra-annotator agreement in python.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. arXiv preprint arXiv:1706.03762.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. 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 5784–5789, Hong Kong, China. Association for Computational Linguistics.

Gavin Wood, Kiel Long, Tom Feltwell, Scarlett Rowland, Phillip Brooker, Jamie Mahoney, John Vines, Julie Barnett, and Shaun Lawson. 2018. Rethinking engagement with online news through social and visual co-annotation. In Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, pages 1–12.

Rowan Zellers, Ari Holtzman, Elizabeth Clark, Lianhui Qin, Ali Farhadi, and Yejin Choi. 2021. TuringAdvice: A generative and dynamic evaluation of language use. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 4856–4880, Online. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems 32, pages 9054–9065. Curran Associates, Inc.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. arXiv preprint arXiv:1904.09675.

A SCARECROW Annotation Schema

Here, we present in greater detail the SCARECROW annotation error types.[7] A visual summary is shown in Figure 6.

While we annotate using this schema, the essence of our study is to embrace language users' abilities to detect when something may be wrong with text. In other words, we do not wish for our span definitions to get in the way of humans describing problems with text. To this end, we encourage researchers to embrace label back off (to coarser categories), merging labels (based on empirical observations), and refining the annotation ontology over time. The central goal is to collect what people find wrong with text.

A.1 Language Errors

We define five categories of language errors, which concern the selection of ideas in a text and how they are expressed. These range from grammar and syntax problems to issues of semantics and pragmatics.

A.1.1 Grammar and Usage

This category of errors includes missing words, extra words, and incorrect or out of order words.

> **EXAMPLE**
> A PhD student from the University of Kent in the UK claims to have discovered a clever way to explain the positive **emoticons** in cats.
>
> ***Explanation:*** *The word should probably be "emotions."*

We also label **Grammar and Usage** for inserted words or small phrases that could be deleted to resolve the issue:

> A couple is facing criticism for their extravagant birthday party. The bewitching pair had first stripped down to fishnets **and backward.**
>
> ***Explanation:*** *This phrase can simply be deleted.*

We avoid partitioning **Grammar and Usage** errors into more detailed categories based on the observation that large language models produce fewer issues of syntax and diction (aside from **Redundant** errors, described next). As such, we focus instead on semantic and pragmatic errors, captured by the upcoming error types.

[7] All example annotations here are our own. Many are provided to annotators during training.

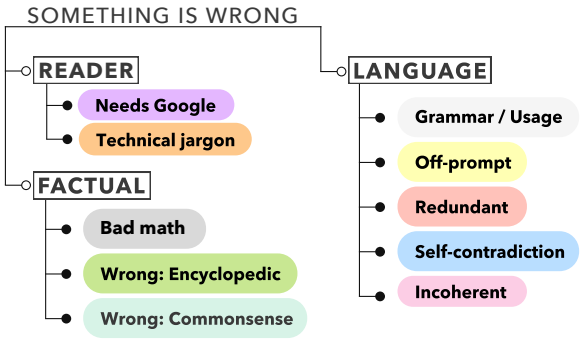

Figure 6: A visualization of SCARECROW spans: three categories (reader, language, and factual) composed of ten types. Annotators choose directly from the ten error types.

A.1.2 Redundant

While "redundant" can also include extra unnecessary information, we specifically use the **Redundant** label to mark repetition. In identifying redundant text, our schema annotates both the **antecedent** (first mention) and the **redundant text** (when the repetition occurs). Sometimes the exact word or phrase will be repeated.

> **EXAMPLE**
> Many merchants worry about the possibility of **poor service** **or service** for certain categories of customers.

Other times, generated text expresses the same idea repeatedly using different words.

> **EXAMPLE**
> They then made decisions based on Kondo's instructions, to the extent that they **created decluttered spaces** **and got rid of clutter and clutter-filled spaces**.

A.1.3 Off-Prompt

The prompt is a human-written sentence used as context from which the model generates a continuation. Models sometimes generate text that is unrelated to the prompt.

> **EXAMPLE**
> **Prompt:** Dogs are the new kids.
> **Generation:** Statistics suggest that most Americans would be happier with dogs than children. **In fact, four out of five don't even visit the dentist annually, much less every six months.** Dog owners report much higher rates of happiness than non-dog owners.

Other times, the text may be related, but it contradicts what is stated in the prompt.

> **EXAMPLE**
> **Prompt:** China sets new record for Economic Growth
> **Generation:** The Chinese economy **fell 10% this month, the third such loss this year.**

A.1.4 Self-Contradiction

When a model generates text that contradicts the prompt, that is labeled as **Off-Prompt**. But when a model generates text that contradicts itself, that is labeled as **Self-Contradiction**. We also mark the **antecedent** (original statement).

EXAMPLE
McDonald's is considering a design which will replace the **cardboard packaging.** Mr Gore-Cotter said: "We recognise the concern around waste. We are now looking at a new design that minimises the **plastic bag."**

Explanation: *The idea of minimizing the plastic bag contradicts the stated goal of replacing cardboard packaging.*

EXAMPLE
Mall of America plans to **lay off and furlough hundreds of its employees. It has no plans to restrict the number of hours workers can work.**

Explanation: *Furloughed workers are explicitly restricted from working.*

A.1.5 Incoherent

Generated text is sometimes grammatical, not redundant, on prompt, and not contradictory, but still confusing. We provide the **Incoherent** label for such sentences.

EXAMPLE
Melody Mitsugi, 28, had never given her kids cheese toast before her husband **drew a map of it on her toast.**

Explanation: *One can't exactly draw a map of Cheese Toast, and one probably wouldn't draw it on toast itself.*

EXAMPLE
Cats naturally show anxiety and fear by at times **breaking apart different parts of the brain in an attempt to keep the others from escaping.**

Explanation: *It's difficult to even imagine what is happening in this passage.*

A.2 Factual Errors

We define three categories of factual errors, which encompass known incorrect statements.

A.2.1 Bad Math

Generated text will sometimes have issues with basic mathematical operations of known quantities (e.g., *"half of ten apples is four"*), problems converting fixed units (e.g., *m to cm*).

EXAMPLE
One account, @Iain_Rowling1, had over 500,000 followers at one point, but in just four days they fell by **around half - some 4,000.**

We also include problems converting currencies that are wildly implausible under modern assumptions (e.g., *£1 = $18 US*).

EXAMPLE
... compared with just over £1,000 **($18,868)** for previous versions of Samsung's flagship phone.

A.2.2 Commonsense

These errors mark spans that violate our everyday basic understanding of the world. Though it is challenging to precisely define *commonsense knowledge* (Liu and Singh, 2004), we include non-encyclopedic knowledge and basic reasoning.

The following example concerns broadly sensible numerical ranges.

EXAMPLE
The picture is from high above the South Pole, where close to **100,000** Astronauts live and work.

Explanation: *Even if we don't know the exact number of astronauts in space, it is common knowledge that 100k is far too many.*

The next example involves world knowledge, akin to scripts (Schank and Abelson, 1977).

EXAMPLE
You can get the dress custom-made and stitched at your favorite **spa.**

Explanation: *Spas don't offer stitching.*

The following example involves lexical entailment.

EXAMPLE
The thinness of our bodies isn't an answer to all common human health problems like **obesity** or diabetes

Explanation: *While most of the statement is acceptable, it's impossible to be "thin" and "obese" at the same time.*

The final example involves time.

EXAMPLE
Now in 2021, NASA is measuring California wildfire temperatures using an instrument on the International Space Station. This year's record-shattering heat has had global repercussions in **2017** , forcing sea level rise on California and increasing the risk of deadly wildfires.

Explanation: *Events in 2021 can't affect events in 2017.*

A.2.3 Encyclopedic

These errors are ones that we *know* are factually wrong, and that we could look up in, say, Wikipedia.

EXAMPLE

Japanese Prime Minister Justin Trudeau said he will be halting all imports and exports until the current situation can be contained.

Explanation: *Justin Trudeau is the Prime Minister of Canada, not Japan.*

The distinction between **Encyclopedic** errors, and the upcoming **Technical Jargon** and **Needs Google** issues, depends on the reader's knowledge.

EXAMPLE

The gas contains something known as **phyto-romatic acid, a common chemical element in the periodic table.**

Explanation: *Acids aren't elements.*

A.3 Reader Issues

We define two categories of reader issues. These are words or statements a reader cannot verify without using an external resource.

A.3.1 Technical Jargon

Sometimes generated text includes specific words from a field that requires expertise to understand.

EXAMPLE

In Chile, an 800-megawatt **photovoltaic** plant was built for a record low cost of $129 per megawatt-hour last year.

Which words are jargon depends on the reader's particular expertise. This means **Technical Jargon** spans are more accurately thought of as potential issues rather than known errors.

EXAMPLE

He uses a spirit **mash** made from white corn and malted barley and a **neutral grain**, which he describes as a "whiskey grain."

A.3.2 Needs Google

Many facts—especially those involving specific people, events, dates, or numbers—could be categorized as encyclopedic knowledge. However, whether the fact is accurate may require additional verification by the everyday reader. To make this distinction between *known* encyclopedic knowledge and *trivia*, we introduce this label to denote that a reader would need to search online to verify whether it is true.

We instruct annotators to not look up facts marked with the **Needs Google** span. We do this to keep the focus of the task on classification, rather than factuality detection. As a result, **Needs Google** spans mark *statements that would need to be verified*, rather than known errors.

EXAMPLE

It was promoted by **Dr. Michael Fanning, the Executive Director of the Foundation for Mental Health Awareness, Inc.**

Explanation: *A reader would likely need to look up whether there is a Dr. Fanning who holds this position.*

EXAMPLE

... an **800-megawatt photovoltaic plant** was built for a **record low cost of $129 per megawatt-hour** last year.

Explanation: *In addition to potential **Technical Jargon** spans, there are at least two **Needs Google** spans: 1. whether such a plant can be roughly 800-megawatt, 2. whether $129/megawatt-hour is a sensible cost measure, and the value is reasonable.*

To illustrate the annotation methodology and schema in practice, we present four complete example annotations in Figure 7. This figure also illustrates how much variation we see across models.

B Annotation Details

B.1 Error Severity

We provide here examples for each of the three error severity levels, which we also give to annotators during training.

EXAMPLE

Paul Campbell-Hughes, from the University of Aberdeen, explains how **she** managed to locate colonies of honey bees in Kent.

Severity: 1. *Since Paul is usually a male name, the model should have used "he." But this error is pretty minor.*

EXAMPLE

Paul Campbell-Smith, a PhD student from the University of Kent in the UK, claims to have discovered a clever way to explain the positive **emoticons** in cats.

Severity: 2. *The word should probably be "emotions." We can guess what was being said, but it's definitely wrong.*

EXAMPLE

Prompt: Whether you're on Facebook, Instagram, Snapchat or TikTok, many people make huge efforts to curate the best version of themselves online.

Generation: **This year we've got something for you: a Love Match Custom Size Poster featuring Mather, Phoenix, Kashun and all her friends, divided among six different covers, creating a beautiful custom size poster for your own personal high school reunion.**

Severity: 3. *Even ignoring the end of the generation (a poster for a personal high school reunion?), this whole generation is way off the prompt and does not make sense.*

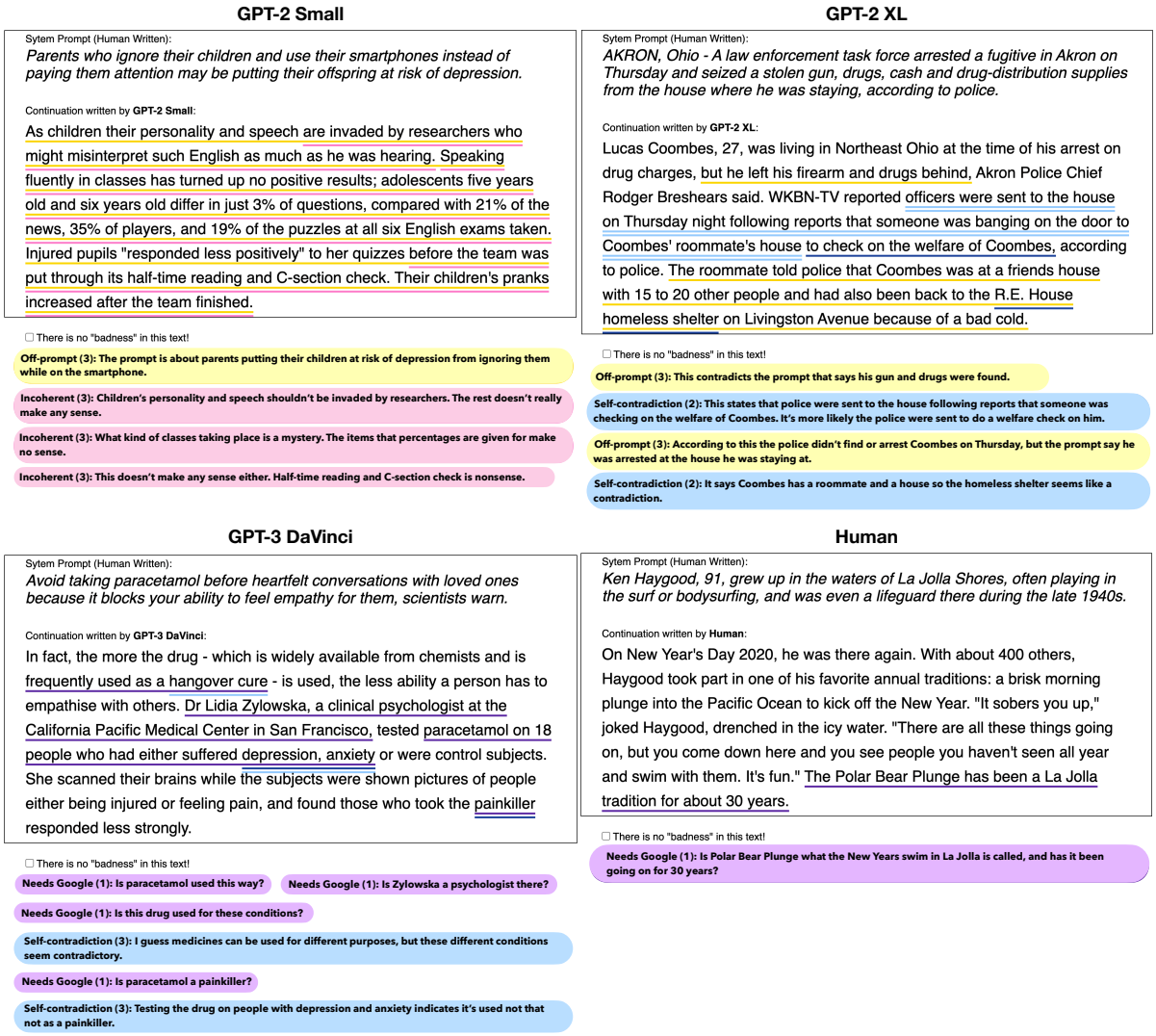

Figure 7: Example SCARECROW annotations (for a single annotator) of three model generations and one ground truth continuation, demonstrating the shift in number, type, and severity of errors. The entirety of the **GPT-2 Small** generation is **Off-Prompt** and/or **Incoherent**, with high severity (3/3). **GPT-2 XL** is instead only about two-thirds covered by errors—still sometimes **Off-Prompt**, but also **Self-Contradiction**, and with high severity (2–3/3). In contrast, **GPT-3 DaVinci** receives several **Needs Google** marks—less severe than errors, as they only indicate that fact-checking is needed—though it also commits two high-severity **Self-Contradiction** errors by generating inconsistent claims. The **Human** (ground-truth) continuation only receives one **Needs Google** span.

B.2 Grading Details

In the training material, there are 10 annotation exercises, 10 multiple choice questions, and 1 real task question to test workers' understanding.

Annotation Exercise After going through each error type, there is an annotation exercise. Workers are asked to mark the span with that particular error in a short text. Each exercise is worth 5 points.

Multiple Choice Question After going through all language errors, and going through all factual errors and reader issues, there is a language error label quiz and a reader and factual error label quiz respectively. Each label quiz consists of 5 multiple choice questions, where workers are asked to choose the error type of a marked span in a short text. Each multiple choice question is worth 3 points.

Real Task Question At the end of the whole training material, workers are asked to apply what they learn in an actual task where they annotate a given paragraph with full tool like ones shown in Figure 7. This question is worth 20 points. We mark 7 error spans as the solution. As long as they can mark 5 of 7 error spans, they get a full 20

points. Otherwise, 4 points will be deducted for each missing error span.

In total, there are 100 points. We pass workers if they score ≥ 90 points, and then they are provided with the solution to review.

C Data Quality

Identifying and classifying errors in potentially noisy machine-generated text is a challenging task. How consistent are the annotations collected from crowd workers? In this section, we examine the agreement and variability of the collected annotations.

At a high level, we observe either acceptable or high inter-annotator agreement across error categories. For rare error types such as **Bad Math**, high agreement stems from the prevalence of spans with no error. For such categories, *we recommend treating each annotator as a high precision, low recall judge*, and considering the information from their aggregate annotations. Figure 8 gives an example of the perspective gained by viewing all 10 annotations of a single generation.

Error	Krippendorff's α	*Two Agree* (%)
Bad Math	0.99	30
Commonsense	0.88	20
Encyclopedic	0.98	12
Grammar and Usage $^{>1}$	0.72	30
Incoherent	0.73	49
Off-Prompt	0.71	61
Redundant	0.88	38
Self-Contradiction	0.87	26

Table 4: Per-token inter-annotator agreement metrics by error category. The $^{>1}$ indicates that we omit severity-1 **Grammar and Usage** errors in all analyses in this paper due to higher variance; including them would drop the Krippendorf's α to 0.56.

Agreement Table 4 shows token-level inter-annotator agreement statistics aggregated over all collected data. Since a single annotator can label a single span with multiple errors, we break the agreement statistics down by error category. We report Krippendorff's α coefficient, a chance-corrected measure of agreement for multiple annotators (Krippendorff, 2018). Due to computational constraints, we calculate this coefficient per generation and report the average across the dataset. The agreement shown here is high for most categories ($>$0.8) and acceptable ($>$0.6) for all error types.

The Krippendorff measure may be deceptively high for some error types such as **Bad Math**, where 99% of tokens are not annotated with this error. The *Two Agree* measure in Table 4 gives a different characterization of this data. *Two Agree* for a given error label is the percentage of tokens labeled by at least one annotator that were also labeled by one or more additional annotators. This metric allows us to see where annotators agree that particular errors exist while ignoring the majority of tokens (for most error categories) which annotators agree are not errors. *Two Agree* shows significantly lower rates for sparse errors with high Krippendorff scores, such as **Encyclopedic**. However, it reveals stronger agreement among **Incoherent** and **Off-Prompt** errors than might be expected given the Krippendorff coefficient.

A limitation for both metrics is the use of token-based overlap.

Bootstrap One issue we face is high variance of annotations. To determine the impact of this variance for lower-data settings, we perform a bootstrap analysis using largest subset of our data (GPT-3, top-$p = 0.96$, $t = 1$, f.p.$= 0$, for which we have annotations of 200+ generations). We choose 50 generations (roughly 500 annotations) and calculate the error statistics therein. We repeat this process 1000 times and report the mean, standard deviation, and coefficient of variation in Table 5. We also calculate the coefficient of variation for different numbers of samples, shown in Figure 9. We see that as the number of samples increases, the coefficient of variation decreases as expected, though less precipitously after 30 examples. These results show that with as few as 50 documents, the SCARECROW error analysis should yield relatively robust results. However, this varies by error type: rare errors like **Bad Math** and **Encyclopedic** show greater variance. Here, again we repeat our recommendation to treat annotations for these categories in aggregate. These results motivate our collection of at least 500 annotations per condition studied.

D Dataset Statistics

We list the data collection quantities in Table 6, and plot visualizations of three aspects: prompt topic and annotated span proportions are shown in Figure 10, and average span lengths are shown in Figure 11.

E Detailed Analysis

In this section we perform a detailed analysis of the trends of individual error types and decoding

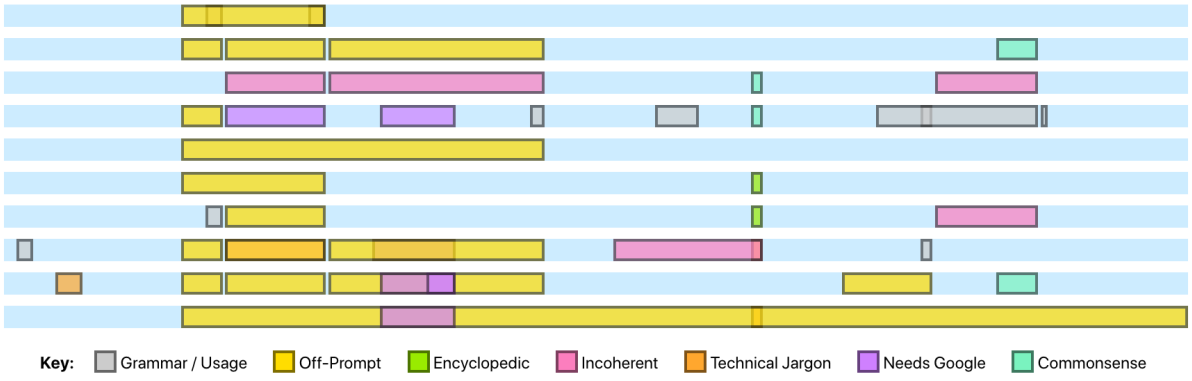

Figure 8: A visual representation of the 10 annotations we collected for one paragraph. Each blue bar represents one annotator, where the width of the bar represents the text of the paragraph. Colored bars drawn on top of the blue bar represent spans marked as errors. We draw bars semi-transparently to show overlapping errors. We can see that some problematic spans (e.g., the **Off-Prompt** section) are marked by almost all workers and given the same label. Other spans are marked by only a subset of the workers (e.g., **Commonsense** and **Incoherent** spans on the right side), or have some label disagreement.

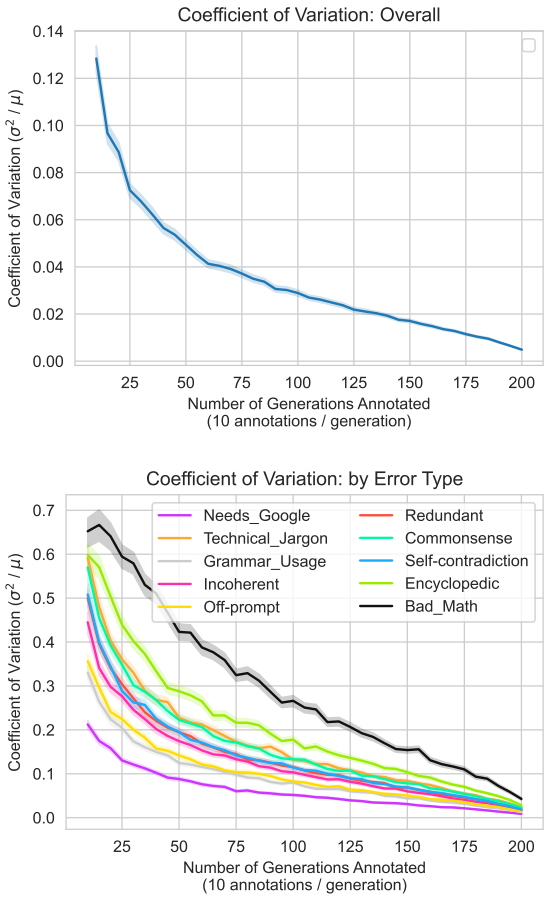

Figure 9: Change in coefficient of variation as number of bootstrap samples increases overall (top), and by error type (bottom), with 95% confidence intervals. Data shown for GPT-3 with apples-to-apples decoding configuration (top-$p = 0.96$, $t = 1$, no *f.p.*).

Error	mean	std.	c.v. (%)
Bad Math	8.51	3.78	44.5
Commonsense	39.40	8.67	22.0
Encyclopedic	13.56	3.94	29.1
Grammar and Usage	126.19	16.81	13.3
Incoherent	96.89	16.58	17.1
Off-Prompt	167.29	23.39	14.0
Redundant	114.77	22.53	19.6
Self-Contradiction	60.54	11.94	19.7
Technical Jargon	100.95	24.09	23.9
Needs Google	482.84	42.22	8.7
Total errors	1268.48	55.59	19.72

Table 5: Bootstrap analysis (sampling 50 generations) of error *counts*, by category (c.v. is the coefficient of variation).

configurations.

To begin, we consider apples-to-apples model decoding configurations. To expand on these results, originally presented in Figure 2, we also present two additional ways of counting error spans, which we show in Figure 12. While our method for counting errors throughout the paper takes into account the number of tokens covered in each span (*span coverage*), we also show plots for scaling each span by its severity level (*span coverage* $\times$ *severity*), and by ignoring both severity and token length (simply *span counts*). These changes in measurement further illuminate model error characters, which we discuss in the upcoming sections (refer to Figure 12).

MODEL	top-p	t	F.P.	GENS	ANNS	SPANS
GPT-2 S	0.96	1.00	0	81	809	3694
GPT-2 XL	0.96	1.00	0	81	806	3087
GROVER-MEGA	0.96	1.00	0	80	796	3006
GPT-3	0.40	1.00	0	66	660	2064
	0.70	1.00	0	65	648	1841
	0.90	1.00	0	63	629	1794
	n/a	argmax	0	66	659	2153
	0.96	0.40	0	65	650	2249
	0.96	0.70	0	61	610	1865
	0.96	1.00	0	206	2055	6234
	0.40	1.00	1	50	500	1280
	0.70	1.00	1	53	530	1481
	0.90	1.00	1	54	540	1717
	n/a	argmax	1	51	509	1384
	0.96	0.40	1	53	530	1401
	0.96	0.70	1	50	498	1369
	0.96	1.00	1	84	838	2947
HUMAN				79	789	2296
TOTAL				1308	13056	41862

Table 6: Statistics of data annotated with SCARECROW. t is the (softmax) temperature, and F.P. is a frequency penalty for already-generated words (explained in §5.2). GENS, ANNS, and SPANS are then number of generations, annotations over those generations, and error spans marked during the annotations, respectively. We perform the most annotations on the strongest available generative model (GPT-3).

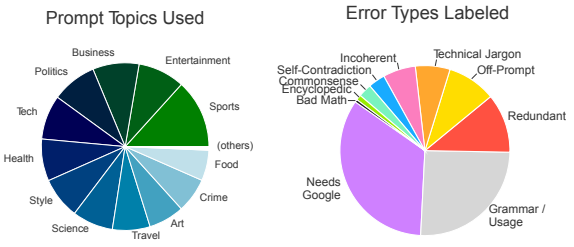

Figure 10: Visual overviews of the distribution of prompt topics used for generating the 1.3k paragraphs used in the annotation (left), and the types of the 41k spans labeled during the annotation (right).

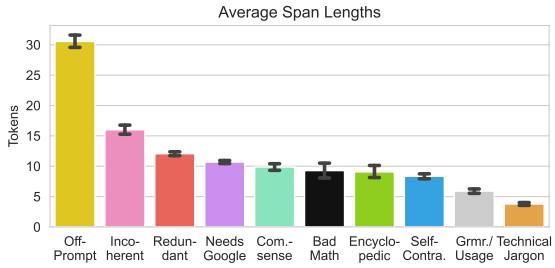

Figure 11: Average number of tokens covered by each annotated span. We observe span length correlates with how abstract the error category is, from word-level issues (**Technical Jargon**), through phrase-level semantics (e.g., **Commonsense**), and into problems of pragmatics (**Off-Prompt**).

E.1 Off-Prompt

Under initial analysis of *span coverage*, **Off-Prompt** errors show a ☹*model plateau* at GPT-3. Measuring *span counts* offers barely perceptible improvement, indicating that scaling language models over more in-domain training does not guarantee topicality.

This observation is consistent with growing work on *prompt programming* as a new technique for attempting to steer large pretrained models to complete the desired task (Branwen, 2020; Gao et al., 2020; Reynolds and McDonell, 2021). In practice, we observe that while GPT-3 will sometimes continue a prompt by writing an article, other times, it may elaborate on the prompt itself:

> **PROMPT**
> Do you prefer the idea of being outdoors in the fresh air to being stuck inside with phones ringing and messages pinging?
>
> **GPT-3**
> Can you leave work at work? Are you flexible enough to **cover holidays or take on additional responsibilities? Can you prioritize tasks? If your boss comes to you on Tuesday to confirm the new social media strategy, are you able to pick up the ball and get the messaging hammered out by Thursday?** ...

Of course, this generation is not *literally* **Off-Prompt**, but it is out of place when other generations are continuations of the prompt, rather than further elaborations of it.

While avoiding **Off-Prompt** errors for language models is worth exploring with prompt programming and other avenues, an investigation of these techniques is outside the scope of this work.

Finally, we note that **Off-Prompt** spans are the most prevalent *error* (not reader issue) marked for human-authored text. We suggest that a higher rate of false positives for this error type, coupled with its prevalence in model-generated text, makes further refinement of this error a compelling avenue for further study.

E.2 Self-Contradiction

While changing from *span coverage* to *span counts* alters the relative order of GPT-2 XL and Grover (though still within confidence bounds), the puzzling question is why GPT-2 Small performs better than most (or all) other models. Why would the smallest model produce the fewest **Self-Contra-**

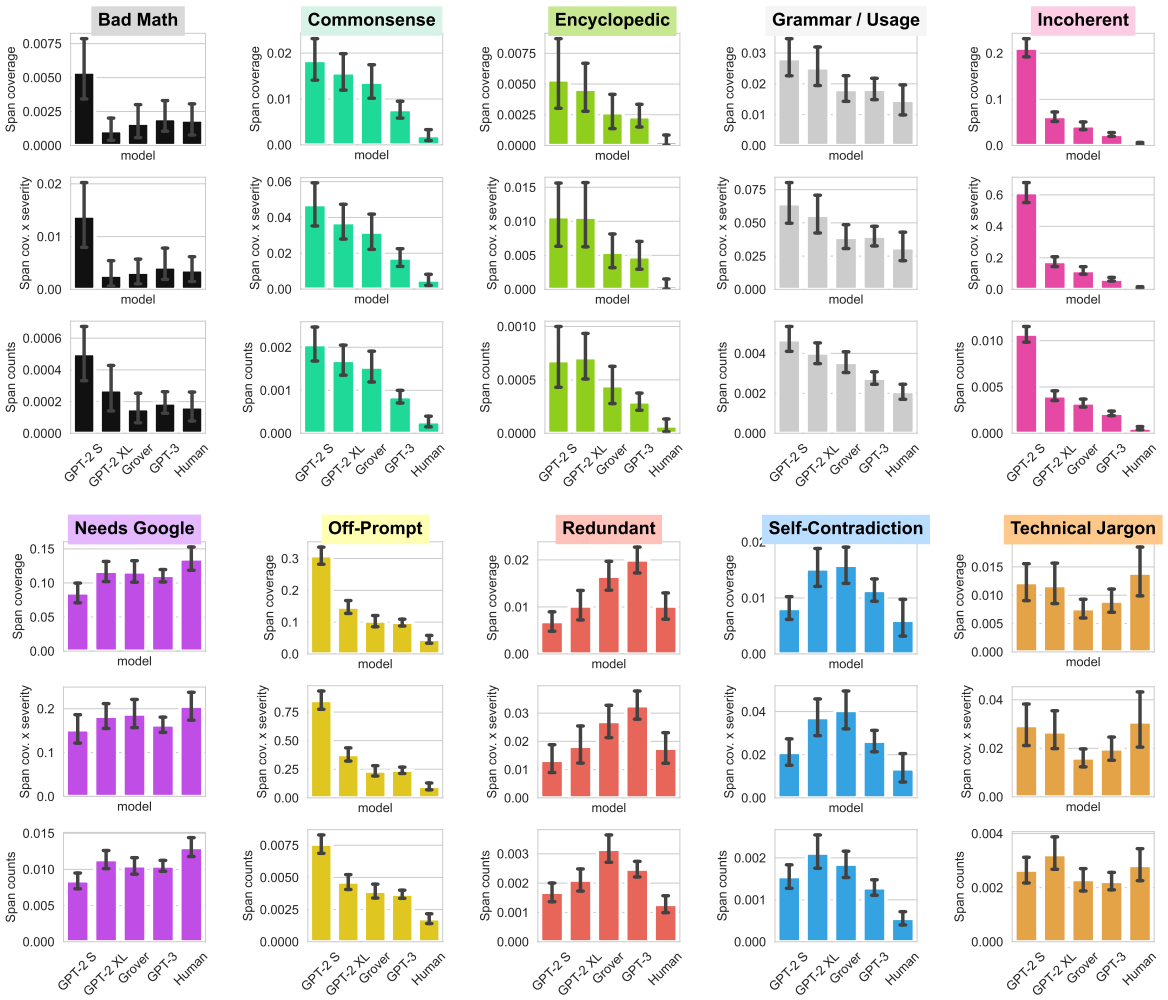

Figure 12: Comparison of three different ways of measuring quantities of error span annotations, shown per label. (The top plot for each error type is identicial to the one shown in Figure 2.) The top method (*span coverage*) is used in the rest of the paper; we provide the comparisons here to illustrate how this decision affects analysis. **Top subplots:** *span coverage*, where the number of tokens annotated as the error span are divided by the length of each annotation. (Annotations with no spans count as 0.) Intuitively, this measures the expected portion of tokens that will be covered by an error span. **Middle subplots:** *span coverage* $\times$ *severity*, like the top measure, but each span's token count is multiplied by its severity, more harshly penalizing errors intuitively marked as worse. **Bottom subplots:** *span counts*, where each error span simply counts as 1, regardless of the span length. In all cases, model configurations are set as closely as possible (top-$p = 0.96$, $t = 1.0$, no frequency penalty), severity-1 grammar errors are removed (see §C), and 95% confidence intervals are shown as bands. **Takeaways:** Compared to the approach used in the rest of the paper (*span coverage;* top), scaling by severity (middle) does not affect the relative model ordering, primarily widening confidence intervals. However, ignoring span lengths (bottom) does affect the results in several cases. **Grammar and Usage** and **Encyclopedic** develop clearer *decreasing* shapes, previously suffering from various levels of *model plateau* at GPT-3. Furthermore, the relative model ordering is changed for **Redundant**, **Self-Contradiction**, and **Technical Jargon** spans.

diction errors?

We posit the reason is that GPT-2 generations are so **Incoherent** and **Off-Prompt** that there is little opportunity for relevant, comprehensible points to be made and then reversed. For example, see the GPT-2 Small annotated generation in the top left of Figure 7. The entire text is covered by **Off-Prompt** and **Incoherent** errors.[8] If we look at GPT-2 Small's error distribution in Figure 3, we see most of its added density comes from significantly

[8]The high double-error coverage reveals another consideration: to what *depth* (i.e., number of overlapping spans) will annotators mark? By the design of our framework, **Incoherent** errors serve as a fall-back, but without it, we might imagine poor generations splatter-painted by other error types.

more **Off-Prompt** and **Incoherent** tokens.

E.3 Redundant

The different counting methods shown in Figure 12 reveal a change in the results for **Redundant** errors. Rather than repetition simply increasing as models grow larger, we observe that GPT-3 repeats in a similar number of cases (lower span *counts*), but for more tokens (higher span coverage). This matches the qualitative observation that GPT-3 produces larger *topically* repetitive blocks, rather than simple word or phrase repetitions generated by GPT-2-sized models:

> **GPT-2 Small**
> ... owners have started growing their own breeds and dogs are **starting to start** so there's really ...
>
> **GPT-3**
> The focus of your thoughts should be on the task at hand, **not on your productivity. You shouldn't be thinking about how you can be more productive. You should be thinking about how you can be productive right now.** ...

Such repetitions can be more difficult to clearly isolate, because even slight wording changes produce variations in tone and connotation. Rather than being identical *semantically*, we observe GPT-3 will seem stuck on a particular *topic*, elaborating on and rephrasing similar ideas more times than a human writer (hopefully) would.

E.4 Reader Issues

We observe the highest number of **Needs Google** and **Technical Jargon** issues in human-authored text.

Needs Google issues broadly represent any specific claim that could be fact-checked. In our domain (news articles), these are primarily whether an event happened on a particular day, whether a person holds a role, or whether a mechanism works as described (e.g., chemical or technical). As seen in Figure 13 (which shows GPT-3's span distribution), **Needs Google** issues happen roughly equally for all topics. We believe this trend is due to the news article domain, which is prone to a high density of specific information. As such, for other domains, this trend may be less prevalent, more difficult to label (e.g., subtle claims assumed to be true in long running text), or both.

We observe that **Technical Jargon** issues are influenced by topic (Figure 13, bottom), occurring significantly more frequently in *Business, Health,*

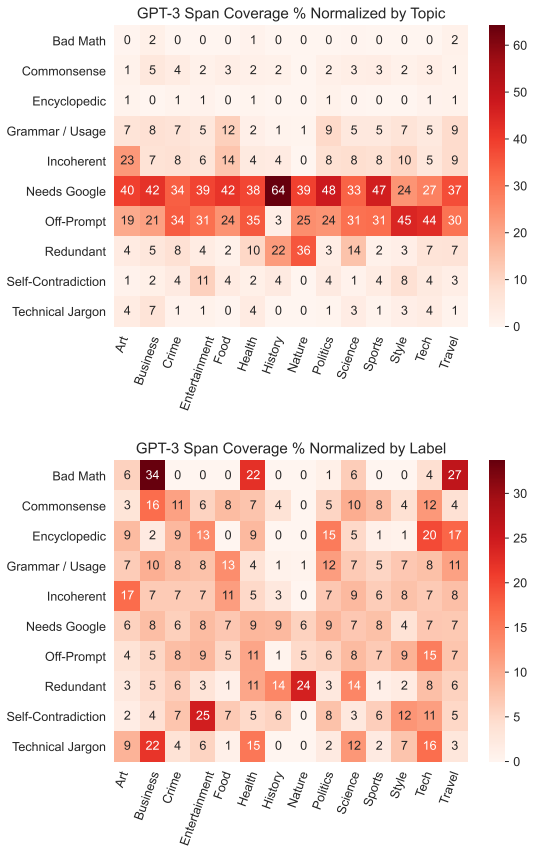

Figure 13: Span coverage across both topic (x-axis) and span label (y-axis) for GPT-3 generated spans (*apples-to-apples* decoding: $p = 0.96, t = 1$, and no frequency penalty). **Top:** normalized by topic (column); **bottom:** normalized by error type (row).

Science, and *Technology* topics than in others. This trend displays a clear topic-dependence even within a single broader domain (news). These results indicate that both reader issues are characteristics of natural text. Of course, one might wish to measure or minimize potential reader issues for a particular application—for example, claim verification, or controlling for reading level.

E.5 Decoding Hyperparameters

We discuss the effects of the decoding hyperparameters we consider—top-p, temperature, and frequency penalty—on generation quality. For the sake of annotation cost, we only vary these parameters for the strongest model available, GPT-3.

First, we show the effect of varying top-p and temperature alone (i.e., with no frequency penalty) on different error types. Figure 14 shows the effect on two salient spans: **Off-Prompt** and **Redundant**. (We omit others for space.) We observe that annotators naturally label errors the way we would

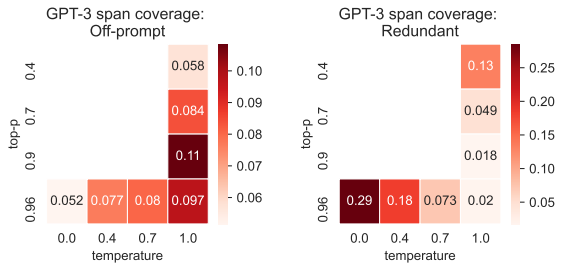

Figure 14: GPT-3 span coverage for **Off-Prompt** (left) and **Redundant** (right) for values of top-p and temperature ($t = 0$ is argmax; both plots with no frequency penalty; argmax sampling is agnostic to the top-p value, so we simply plot it in the $p = 0.96$ cell). **Takeaway:** Our annotation confirms intuitive expectations of the effect of sampling on two error categories. When sampling from a larger pool of words (higher p and t), a model is more likely to veer **Off-Prompt**, but less likely to produce **Redundant** text.

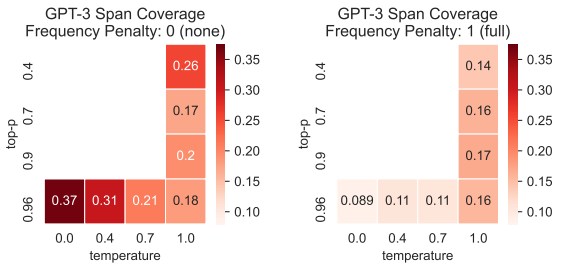

Figure 15: Comparison of *frequency penalty* off (left) and full (right) for GPT-3 (removing reader issues and severity-1 **Grammar and Usage** errors; argmax sampling is agnostic to the top-p value, so we simply plot it in the $p = 0.96$ cell). We observe the frequency penalty improves average span coverage for all values of top-p and temperature. Furthermore its trend is reversed: with a frequency penalty, the least diverse sampling mechanisms (low temperature and low top-p) now produce text with the fewest error spans, rather than the most. (See Figure 4 for confidence intervals on each value.)

intuitively expect the model to produce them, given the hyperparameter changes. The bottom-right corner of each subplot, where $t = 1$ and $p = 0.96$, is the configuration with the highest amount of randomness from sampling. As we move away from that corner—either left by lowering temperature, or up by lowering top-p—we lower the amount of randomness. We observe a positive correlation with randomness and **Off-Prompt** errors, and an inverse correlation with **Redundant** errors. In other words, sampling from a larger set of words makes the model more prone to changing topics, but less likely to repeat itself, and vice versa.

After confirming these intuitive measures, we turn our attention to Figure 15, which investigates the overall error spans for GPT-3 both without (left) and with (right) the frequency penalty. (Note that unlike Figure 14, both heatmaps in Figure 15 have the same color scale.) We observe that introducing the frequency penalty lowers error rates for every value of temperature and top-p that we try. Furthermore, it appears to reverse the trend seen without a frequency penalty: that sampling from a larger set of words produces fewer errors.

The overall results for all decoding configurations were shown previously in Figure 4. In the next section, we focus on the GPT-3 decoding configuration that produced the fewest number of errors, and compare it to human authored text.

E.6 Best GPT-3 vs. Humans

The best GPT-3 configuration shown in Figure 4—argmax sampling with frequency penalty = 1—appears to match error rates seen in human text. Is the text generated by this model truly as error-free as news articles?

We first look at the error composition of both sets of annotations. To get a clear picture of the potential problems, we plot only error spans (ignoring reader issues), and we omit length scaling, instead plotting span counts. This breakdown is shown in the left plot of Figure 16. The error compositions are similar, the largest differences being more **Redundant** errors for GPT-3, and more **Grammar and Usage** errors for human-authored text.

Next, we perform a manual analysis of 160 errors, sampling 10 at random from each of the 8 error types for each model (GPT-3 and human-authored text). We show the results in the center plot of Figure 16. We notice that a greater portion of errors in human-authored text were due to artifacts present in the text-only format of the Common Crawl. For example, links to other articles or advertisements sometimes appear in the middle of an article's text. While annotators were quick to mark these spans, they reflect errors in formatting, not in writing. We partition these errors separately and exclude them from the subsequent calculations.[9]

[9]GPT-3's generations also sometimes exhibited what appeared to be formatting errors due to training on web-scraped

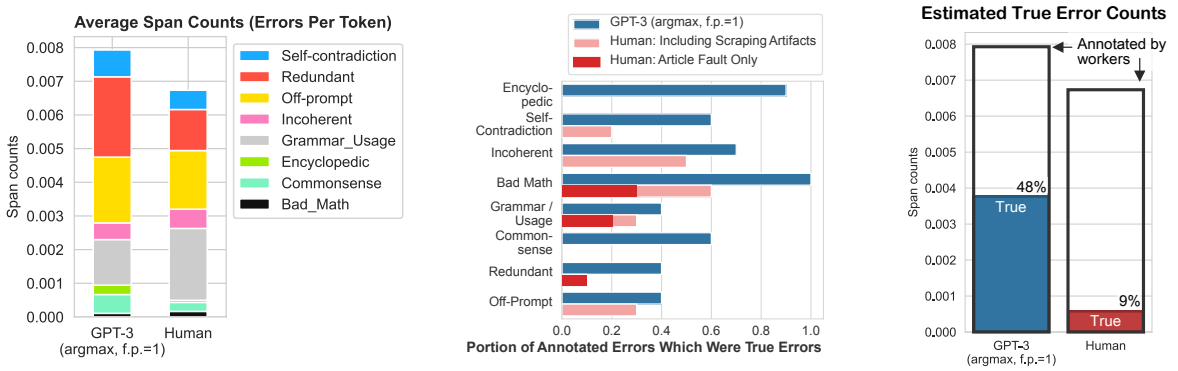

Figure 16: Analysis of the best GPT-3 configuration (*argmax, freq. penalty = 1*) vs. human-authored text. **Left:** A breakdown of errors by type. **Center:** Results of manually annotating 10 random spans from each type with whether the error was legitimate. For human-authored text, we also show errors marked on scraping artifacts that were present in the Common Crawl data. **Right:** Scaling each error type (*left plot*, now shown in black outline) by the portion of errors found to be legitimate (*center plot*), we estimate the true errors counts for each model (color-filled portions). **Takeaway:** Humans have more difficulty spotting errors in higher quality text; accounting for this difference dramatically increases the gap between model-authored and human-authored text. For simplicity, all plots use error *counts* rather than error *coverage*—i.e., they count the number of error spans, rather than scaling by the number of tokens covered.

Finally, we scale each error type's prevalence for each model (i.e., the left plot of Figure 16) by the portion of errors that we estimate to be legitimate based on our manual annotation (i.e., Figure 16, center) to produce the right plot of Figure 16. After taking into account each error type's frequency, we estimate that 48% of GPT-3's worker-annotated errors overall are legitimate, compared to 9% for human-written articles.

This analysis suggests two findings. First, human-authored news paragraphs contain many times fewer issues than text authored by GPT-3 using the best decoding configuration we tested. Second, the noise of error annotations may be as high as 90% when assessing high-quality text. Though it would require further manual annotation to verify, we conjecture that the trend of GPT-3's error spans being more reliable (only 50% noise) would continue, and that text generated by GPT-2 would contain even fewer false positives. We note that such rates are not fixed—after all, the manual annotations were done by one of the authors simply by reading carefully—but that more realistic text may require correspondingly more effort by human annotators.

text, though more rarely. For example, some generations contained *Which?* after vague noun phrases, which appear to be learned from Wikipedia, where under-specified information is tagged by an editor with this word. For fairness, we removed these errors from GPT-3's tally as well, though they were few enough we do not plot them separately.

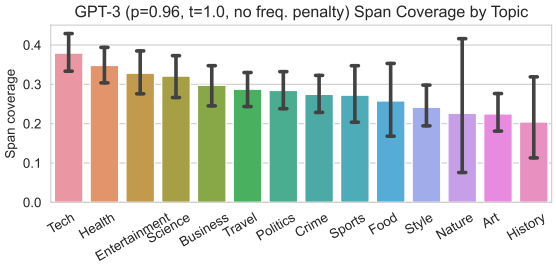

Figure 17: Average span coverage for different topics (GPT-3 generations with apples-to-apples decoding configuration), with 95% confidence intervals. While the majority of topics display no significant trend, we observe that more technical topics such as *Tech* and *Health* are covered by a higher density of error spans than *Style* and *Art*.

E.7 Topics

As noted in §5.3, we collect data using prompts drawn primarily from 12–14 news topics. For conciseness, we show results only for GPT-3, and only for the standard apples-to-apples decoding configuration.

Figure 17 plots, based on the prompt topics, the average portion of the generation that is covered by error spans. While there is no significant difference between most topics, the results do indicate that generating text in more technical domains leads to higher span counts.

Figure 13 shows individual span prevalence by topic. The top heatmap normalizes each topic (col-

umn) independently. **Needs Google** issues and **Off-Prompt** errors dominate the error types, with a few exceptions: for *History*, and *Nature* articles, **Redundant** trumps **Off-Prompt** as a source of errors.

For the bottom, if we instead normalize by error label (row), we can observe which topics are more prone to certain error types than others. For example, we can see **Bad Math** errors are most common in *Business* and *Health* generations; *Entertainment* causes the most **Self-Contradiction** errors; and **Technical Jargon** issues appears more frequently in articles about *Business, Technology,* or *Health*.

E.8 Error explanations

Figure 18 displays word clouds for common unigrams and bigrams found in the error explanations for each error type, and Figure 19 shows the average explanation lengths for each error type. For **Technical Jargon**, **Redundant**, and **Needs Google** error types, the prominent words do not provide much illumination and they have short average explanation length, indicating that the explanations are straightforward affirmations of the category (*"I think this is financial jargon," "The information is repeated,"* or *"I would need Google to check this."*). But for categories like **Encyclopedic** and **Bad Math**, we observe some coarse trends: "year" is prevalent in both, "movie" appears in **Encyclopedic**, and "million" is present in **Bad Math**, which suggests that the explanations are more likely from outside knowledge and needs some calculation (*"The iPhone uses a lightening connector not a L-shaped connector," or "5000 feet is 1524 meters."*)

Figure 20 presents a few representative explanations for four error types, taking particular note of their explanation lengths (Figure 19). Both **Self-Contradiction** and **Redundant** errors have antecedents, but their explanations are markedly different. Explanations for **Self-Contradiction** contain more information describing the particular semantics that is reversed, which are less obvious at first glance than other errors. On the other hand, **Redundant** errors are more straightforward to spot, often involving simple lexical overlap, and so don't require elaboration.

Explanations for **Commonsense** contain the true commonsense knowledge that the text violates, which may take several words to explain. But an explanation for a **Grammar and Usage** error simply corrects the error; as these errors are easier to fix, the explanation lengths are often short.

F Future Work

We outline several further directions of study centering around the SCARECROW annotation framework, considering both natural implications and broader steps.

F.1 SCARECROW Studies: Simple

Find the best-performing GPT-3 decoding hyperparameters. We observed that for GPT-3, a frequency penalty value of 1 with argmax sampling produced fewer error spans than any other configuration (Fig. 4). We have not tried varying the frequency penalty to values *between* 0 and 1, or adding any *presence penalty* (§5.2), both of which then allow for fresh explorations of top-p and temperature.

Study decoding parameters in smaller models. How good can (a finetuned) GPT-2 get? We saw decoding parameters considerably impacted GPT-3's performance, moving it from edging out Grover to error rates close to humans (Fig. 4). Could such decoding changes have a similar effect on a GPT-2-sized model? Or might a smaller model favor different decoding hyperparameteres?

Back-off annotations. We observed good annotator agreement given the complexity of the task, but the odds that two annotators agree exactly on each span's type and boundaries remains only moderate (§C). We did not try backing-off (a) error types into coarser categories (e.g., language, factual, reader issue) or even to binary presence; (b) span boundaries into phrase or sentence-level annotations. Applying a type of back-off could also allow clustering methods to discover different error ontologies.

Improve automatic error detection. While we present baseline results for automatic span error detection (§6), we anticipate that significant progress is still available in this new task.

F.2 SCARECROW Studies: Complex

Align multiple annotations. In the current work, we largely treat annotators independently, with the exception of measuring their overlap to study agreement (§C) or taking their union to train prediction model (§6). However, we might consider other

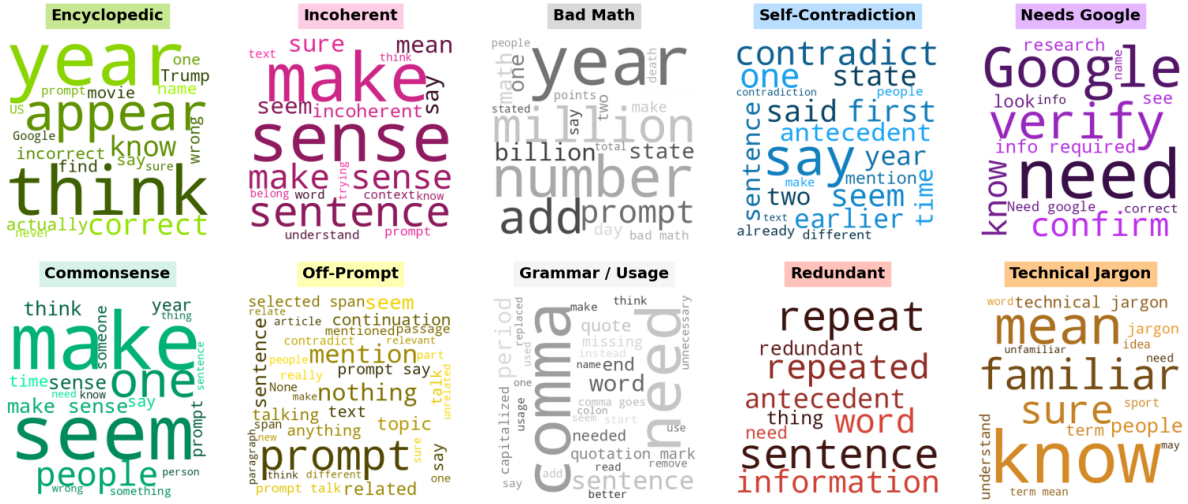

Figure 18: Common unigrams and bi-grams observed in the explanations written for each annotated span, grouped by error type.

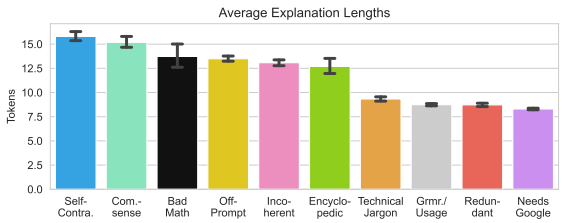

Figure 19: Average number of tokens in explanation for each error type. We observe explanation length correlates with how obvious the error type is, where categories like **Grammar and Usage** and **Technical Jargon** are easier to find and explain than **Self-Contradiction** and **Commonsense**.

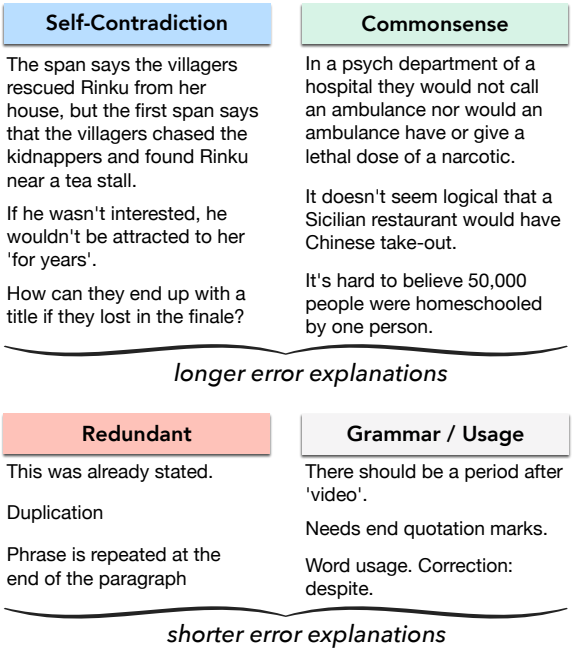

Figure 20: Examples of error explanations from different error types that favor longer (top) and shorter (bottom) descriptions.

ways of viewing the 10 annotations for each generation together. For example, we might consider the aggregate decision of *whether* a token is labeled with *any* span a measure of how noticeable or jarring an error is. This measure may be related to error severity, but may be distinct from it.

One might also consider formal methods for computing annotation alignments. The Gamma measure, proposed by Mathet et al. (2015), satisfies the long list of criteria needed to align and measure SCARECROW annotations: spans of multiple types, with gaps, full and partial span overlap, more than three annotators, and the potential to merge or split annotations (which we have not addressed in this paper). While we performed experiments with this measure, we experienced difficulties producing intuitive alignments with the authors' software, which disallows configuring parameters of the mixed-integer programming problem.[10] Emerging concurrent work (Titeux and Riad, 2021) offers a reimplementation of this measure that exposes additional parameters, which may be a promising avenue. However, it is possible that aligning annotations is a challenging task on its own that might

[10]The mixed-integer programming approach is also computationally intensive; e.g., memory alone prevented us from computing alignments for pilot studies with twenty annotators, even on a machine with 500GB of RAM.

require use of the explanations.

Characterize error nuance. Related to the previous point about error alignment, one might study whether model size affects span agreement. Anecdotally, errors from larger models like GPT-3—even of the same type, like **Commonsense** errors—are more difficult to describe without careful consideration, and may also be more difficult to identify.

Characterize repetition. Our quantitative studies of **Redundant** errors (e.g., Figs. 14 and 12) point to semantic repetition as the major issue that emerges as models are scaled. Though this effect may be mitigated by changes to the decoding algorithm (like the frequency penalty), we still observe that models have difficulty striking a balance of repetition. With excessive paraphrasing, generated text seems *stuck* on an idea. But equally, if a generation moves too quickly between ideas without linking them together or to an overall theme, the text lacks coherence. We posit that the issue of **Redundant** text emerges as the shadow of encompassing issues of narrative structure and discourse.

F.3 Broadening SCARECROW

Constrained generation This paper focuses on open-ended generation, but a natural extension of this method would be to assessing constrained generation tasks, such as machine translation.

New error types Especially if considering a novel task setting, new error types may prove useful. For example, in constrained generation, one might consider an **Adequacy** error, which—as in machine translation—would indicate that the meaning of a span diverges from what is expected given the generation constraints. Furthermore, one might need to introduce annotations on the provided (not generated) text to account for desired semantic components that are *missing* from the generated text. Or, perhaps for a dialog setting, one might introduce a **Generic** label, which would indicate that a portion of the generation is otherwise coherent and correct, but offers a lack of new information.[11]

Corpus-level evaluation Other work has considered the evaluation of natural language generations at-scale, looking at distributional properties of the text (Caccia et al., 2020; Pillutla et al., 2021). We suggest that these views are complementary to instance-based, human evaluation proposed here, and combining the approaches could lead towards a more holistic view of generative evaluation. For example, while all **Self-Contradiction** errors right now are *within-document*, one could similarly identify *cross-document* contradiction errors, where a model is inconsistent at a more global scale.

F.4 Applications

Detecting factuality One potential application of the SCARECROW data could be using the **Needs Google** spans as a dataset of its own. In addition to training models to identify spans that require verification, one could go a step further and consider *evidence retrieval* for each span, and even propose a classification task.[12]

Editing errors One errors can be detected, can they be fixed? The difficulty and scope of fixing SCARECROW-identified errors may depend on the error type, as error fixes may have cascading effects in the rest of the document.

[11] Such generic language may be seen as violating Grice's Maxims (Grice, 1975), for example, by providing a dearth of information *quantity*, or by flouting improper *manner* by lacking brevity.

[12] Minimally, **Needs Google** spans from human-authored reputable news text should (hopefully) all be factually correct.