Know When To Stop:
A Study of Semantic Drift in Text Generation

Ava Spataru[1] **Eric Hambro**[2†] **Elena Voita**[1] **Nicola Cancedda**[1]

[1]FAIR, Meta [2]Anthropic
{avaspataru, lenavoita, ncan}@meta.com
eric.hambro@gmail.com

Abstract

In this work, we explicitly show that modern LLMs tend to generate correct facts first, then "drift away" and generate incorrect facts later: this was occasionally observed but never properly measured. We develop a semantic drift score that measures the degree of separation between correct and incorrect facts in generated texts and confirm our hypothesis when generating Wikipedia-style biographies. This correct-then-incorrect generation pattern suggests that factual accuracy can be improved by *knowing when to stop* generation. Therefore, we explore the trade-off between information quantity and factual accuracy for several early stopping methods and manage to improve factuality by a large margin. We further show that reranking with semantic similarity can further improve these results, both compared to the baseline and when combined with early stopping. Finally, we try calling external API to bring the model back to the right generation path, but do not get positive results. Overall, our methods generalize and can be applied to any long-form text generation to produce more reliable information, by balancing trade-offs between factual accuracy, information quantity and computational cost.

1 Introduction

Differently from the earlier approaches to generating natural language with explicit content planning (Mann, 1983; Reiter and Dale, 1997), modern autoregressive language models make predictions token-by-token, without pre-established text structure. One of the consequences of this methodological shift is that newer models lack the capability of maintaining high-level structure throughout generation and overly focus on local coherence. This was noted in the form of repetition (Fu et al., 2021) and semantic drift (Li et al., 2021).

The term "semantic drift" emerged to describe the decrease in text generation quality when increasing generation length and has been classified as a sub-type of hallucinations (Ji et al., 2023). Before that, semantic drift (or topic shift) was briefly mentioned when talking about question generation (Zhang and Bansal, 2019) and story generation (Wang et al., 2021; Sun et al., 2020). In factual evaluation, recent works also mention a decline in factual accuracy for longer generations (Min et al., 2023; Qian et al., 2023). While quality decrease for longer generations hints at specific order in generation quality (high-quality first, low-quality later), this ordered behavior has not been neither formally defined nor thoroughly studied and measured. In this work, we refer to "semantic drift" as the *strength of the order* in generation quality and, for the first time, provide tools for understanding this phenomenon.

We propose to measure semantic drift by considering the change in truthfulness of a sequence of facts when a model generates a fact-rich text around a topic. Intuitively, we measure the *degree of separation* between correct and incorrect facts in a paragraph: if the model starts by generating correct facts and then switches to systematically generating incorrect ones, we consider this as a semantic drift. To quantify the severity of semantic drift, we use the FActScore task which provides correct/incorrect labels for individual facts (Min et al., 2023). We find that, indeed, several LLaMa2 variants have high semantic drift score: they tend to generate correct facts first, then "drift away" from the topic and generate incorrect facts later.

This correct-then-incorrect separation suggests that factual accuracy can be improved by stopping generation early. We show that even a simple method that encourages generating EOS leads to large improvements in factuality. We then propose to use resample-then-rerank pipelines where for each sentence, we generate several versions and

†Work done while at FAIR, Meta.

Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 3656–3671
June 16-21, 2024 ©2024 Association for Computational Linguistics

choose the best based on sentence similarity measures. Compared to the baseline, this improves factual accuracy by almost 10% (without shortening texts as with early stopping). This can also be combined with early stopping and allows for different informativeness-vs-factuality trade-offs. Finally, we ask: If the model drifts away during generation, could it be brought back onto a correct path by calling an external API? Sadly, this does not give noticeable improvements (at least, when working in the previously established settings).

Overall, we:

- formally show that current LLMs tend to generate facts in a correct-then-incorrect manner;
- based on that, develop methods to improve factual correctness: simple early stopping and more complex resample-then-rerank;
- find that API calls help little to none.

Our methods offer a practical compromise, balancing computation with performance, and build a foundation for further research. Importantly, they are directly applicable to any probabilistic auto-regressive language models.

2 Definition of Semantic Drift

Since the term "semantic" drift has been used with various meanings, we felt the need for a unifying definition, which we state below.

> *Semantic drift describes the phenomenon wherein generated text diverges from the subject matter designated by the prompt, resulting in a **growing** deterioration in relevance, coherence, or truthfulness.*

Semantic drift results in a loss of three textual characteristics (see examples in Appendix A):

1. Loss in **coherence**, which leads to issues with clarity, logical flow, and self-consistency;
2. Loss in **relevance**, which refers to the inclusion of irrelevant or redundant content;
3. Loss in **truthfulness**, which refers to the inclusion of hallucinated content or content inconsistent with world knowledge.

2.1 Semantic Drift Score

To quantify the severity of semantic drift, we define a new scoring method, semantic drift (SD) score. To calculate this score for a paragraph P, we take individual atomic facts along with their labels (1 for supported facts and 0 otherwise). Let N be the total number of facts, s_i be the label for the fact $i \in [0, N)$, and m be a hyperparameter. Then we define the SD score as:

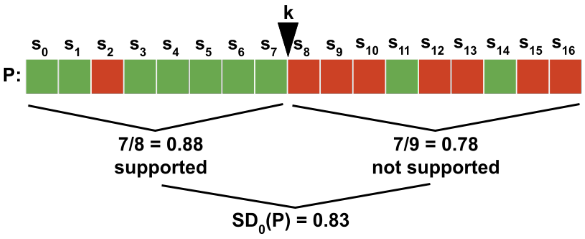
k
s_0 s_1 s_2 s_3 s_4 s_5 s_6 s_7 s_8 s_9 s_{10} s_{11} s_{12} s_{13} s_{14} s_{15} s_{16}
P:
7/8 = 0.88
supported
7/9 = 0.78
not supported
$SD_0(P) = 0.83$

Figure 1: A visual example of calculating semantic drift (SD) score for paragraph P. The position which best splits the paragraph is $k = 8$. The proportion of supported facts to the left is 0.88 and the proportion of not-supported facts to the right is 0.78, giving an average of 0.83. The other positions all have lower SD scores, therefore the SD score of paragraph P is 0.83.

$$SD_m(P) = \max_k \frac{1}{2} \cdot SD_m(P, k)$$

$$SD_m(P,k) = \begin{cases} 0, & \text{if } (N-k<m) \text{ or } (k<m) \\ & \text{or } (N<2m), \\ \frac{\sum_{i=0}^{k-1} s_i}{k} + \frac{\sum_{i=k}^{N-1}(1-s_i)}{N-k} & \text{else.} \end{cases}$$

The position k at which this maximum is reached represents the position with highest average between (1) proportion of supported facts to the left of position k and (2) proportion of not-supported facts to the right of position k. We will refer to k as the *drift point*. Parameter m controls the range of k, meaning that we only consider splits that have more than m facts on either side of k.

Intuitively, we measure the *degree of a separation* between correct and incorrect facts in a paragraph: the SD score is high when a text is largely correct before the drift point and largely wrong after (Figure 1). E.g., a paragraph with an SD score of 1 would have all correct facts first and all the incorrect facts later. For a paragraph in which facts are either wrong or correct without any clear separation, we would expect an SD score around 0.5.

3 Identifying Semantic Drift

In our experiments, we rely on the FActScore task (Min et al., 2023). This task identifies all aspects of semantic drift and scores individual facts from a text as either correct or incorrect. A fact is correct if it is supported by external knowledge and therefore truthful. Since two facts that contradict each other cannot be simultaneously correct, correct facts are also coherent. Moreover, facts are verified in context, meaning that a fact is correct only if it is relevant to the context. Appendix A shows examples of scoring for semantic drift types.

3.1 Setting

Task. The FActScore task focuses on Wikipedia-style biographical passages: they are generally fact dense, and the individual facts can be reliably verified (Wadden et al., 2020; Thorne et al., 2018). The task consists of 3 steps: (1) generating biographical paragraphs for 500 entities, (2) extracting "atomic facts" from the paragraphs, and (3) scoring the truthfulness of paragraphs by verifying all atomic facts against a knowledge source. The FActScore itself is the precision of atomic facts aggregated over the 500 samples.

Pipeline. We let LLaMa2-70B generate a biographic paragraph, using the same prompt as Manakul et al. (2023): *"This is a Wikipedia article about [entity]. [entity]"*.[1] Each generated paragraph is then passed through the FActScore pipeline to identify and verify atomic facts. We modify the original FActScore pipeline to rely on LLaMa2-70B-Chat (rather than InstructGPT) and validate using human annotations (Appendix B.2).

3.2 Semantic Drift in LLaMa2-70B

For paragraphs generated by LLaMa2-70B, we got an average SD score of **0.78** when considering all 500 examples and the score of **0.8** when filtering out completely correct and incorrect samples.[2] Figure 2 shows the distribution of SD scores.

Semantic drift is high. Semantic drift score of 0.8 is very high: it means that there is a *significant separation between correct and incorrect facts in most paragraphs*, and thus model generations "drift away" at some point during generation. To ensure that the high semantic drift score is not just chance, we conduct a statistical significance test. We estimate the probability that a random permutation of facts would result in the same SD score or higher. We find that samples we identified as drifting have an average probability of <0.02. For more details, please refer to Appendix B.5.

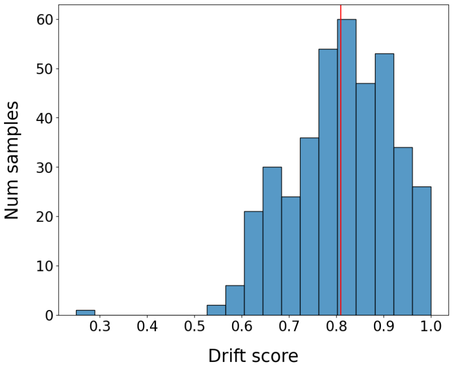

Figure 2: Distribution of Semantic Drift Score (after filtering) in paragraphs generated by LLaMa2-70B (sampling: temperature=0.6, top-p=0.9).

Drift starts early. When looking at the number of correct facts, we noticed that generations are largely wrong, and the drift starts early. For example, only a small portion of paragraphs has at least 10 correct facts before the first wrong fact (34 paragraphs, $< 7\%$). For 37% of all paragraphs, the drift point is in the first 10% of facts.

Our observations are reliable. In Appendix B, we show that decoding strategy and truncation parameter only slightly impact the SD score (hence, our observations). We conclude that LLaMa2-70B shows statistically significant high SD score in more than 40% of generated paragraphs.

3.3 Semantic Drift in Other Models

To strengthen our observations, we extend our experiments to other well-established LLMs.

Setting. We consider LLaMa2-70B-Chat (Touvron et al., 2023), Falcon (Almazrouei et al., 2023) and GPT (OpenAI, 2023). These models are both text and chat completion models. For text completion, we use the same prompt as in Section 3.1. For chat completion, we use "*Tell me a bio of* ".

Results. From Table 1 we see that semantic drift is high for all models. This confirms our hypothesis: models start with correct facts, then "drift away". While the GPT models perform considerably better on the FActScore* task, they still have high SD score and could therefore benefit from our error mitigation strategies from Section 6.

4 Analysis

Let us now understand the potential causes of semantic drift and analyze LLaMa2-70B generations both quantitatively and qualitatively.

[1] Example prompt: "*This is a wikipedia article about Bob Marley. Bob Marley*". The model has to continue generation.

[2] Filtered 20 completely incorrect and 21 completely correct samples, remained with 458 generated paragraphs with an average of 47 facts per paragraph.

Model	facts /gen	No ans-wer	FAct Score* (%)	SD Score (%)
Llama2-70B-chat	50.55	1	41.72	77.06
Falcon-7B	41.84	6	24.64	76.81
Falcon-40B	49.23	4	25.88	77.38
text-davinci-003	58.27	2	38.09	77.21
GPT 3.5	67.82	1	45.96	79.49
GPT 4	48.31	1	53.54	78.12

Table 1: Results for various models when generating 500 biographical paragraphs. “No answer” is the number of paragraphs when the model produced no facts.

4.1 Quantitative Analysis

We analyze the distribution of semantic drift by multiple factors: (i) person popularity, (ii) paragraph length and drift position, (iii) model scale.

Person popularity. We hypothesize that semantic drift score might be affected by the popularity of a bio’s object in a typical dataset. Figure 3 shows the distribution of SD scores by prevalence class, from “very rare” to “very frequent”.[3] We see that for very frequent entities, the semantic drift score is distributed normally. As the entities become less frequent, the distribution starts turning into a bimodal distribution. This could be because for rare entities, the model either generates a few facts well and then drifts away (resulting in high drift score), or has a generally murky knowledge about the entity and generates both wrong and correct facts together (resulting in low drift score).

Paragraph length and drift position. We find no correlation between paragraph length, drift score and relative drift position. However, we do note that the distribution of relative drift position is distinctly U-shaped, with more paragraphs drifting in the first 10% of generated facts than in the last 10%. We apply truncation as described in Section 2.1 and note that the distribution of drift position is still peaking in the first 10% of generated facts. For more details, see Appendix B.4.

Model scale. Table 2 shows results for the same pipeline with two smaller LLaMa2 models. We find that while increasing parameter size clearly improves factuality of generated text, all three model sizes show similar SD scores: semantic drift is high regardless of scale.

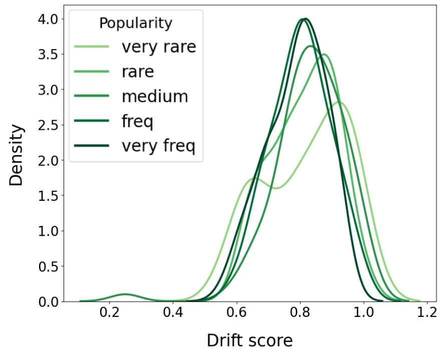

Figure 3: Semantic drift score density plot for person popularity classes. LLaMa2-70B.

Num params	facts /gen	No answer	FAct Score* (%)	SD Score (%)
7B	34.80	4	34.77	76.29
13B	33.04	1	39.36	76.83
70B	43.89	1	44.56	78.07

Table 2: FActScore* and SD Score for different model sizes of LLama2 when generating 500 biographical paragraphs. “No answer” shows the number of paragraphs for which the model produces no facts.

4.2 Qualitative Analysis

We looked at examples of biographies generated by LLaMa2-70B with high SD score and identified at least three potential categories of semantic drift (Table 3):

1. **Repetition.** The clearest and easiest type of semantic drift happens when the model “gets stuck” in a loop of identical or similar facts.

2. **Making up information**. This type of drift happens when the model can generate a few correct generic facts (such as occupation), but makes up all subsequent information.

3. **Plausible information.** This is perhaps the most interesting and subtle type of drift. It happens when the model generates a good lengthy biographic paragraph, but towards the end begins adding information which is indirectly relevant and incorrectly attributes that information to the subject.

5 Factual Accuracy and Uncertainty

Knowing that the model is largely correct up to some point (to be precise, the semantic drift point) and largely wrong after gives us an opportunity to

[3] The prevalence labels come from the FActScore dataset.

Drift Type	Examples
Repetition	Suthida is the Queen of Thailand. She is the wife of King Vajiralongkorn. She was born on 3 June 1978. She is the first wife of King Vajiralongkorn. She was born in Hat Yai, Songkhla, Thailand. She is the daughter of Saman Bajrasudha and his wife, Nui. She has two brothers and one sister. She is a former flight attendant. She is a former actress. She is a former model. She is a former beauty queen. She is a former Miss Thailand. She is a former Miss Universe. She is a former Miss World.
Making up information	Steve Cummins is a professional rugby league footballer of the 1990s and 2000s. A Great Britain international representative centre, he has played in the Super League for the Wigan Warriors and the Warrington Wolves, and in the NRL for the Brisbane Broncos and the Cronulla-Sutherland Sharks.
Plausible information	Matthew Perry is an American actor and comedian. He is best known for his role as Chandler Bing on the NBC television sitcom Friends, which ran from 1994 to 2004. Matthew Perry was born in Williamstown, Massachusetts, on August 19, 1969. His mother, Suzanne Marie Morrison, is a Canadian journalist and former press secretary to Canadian Prime Minister Pierre Trudeau. His father, John Bennett Perry, is an American actor and former model. Perry has two older sisters, Caitlin and Emily. Perry was raised in Ottawa, Ontario, and attended Rockcliffe Park Public School and Ashbury College. He then studied at the University of Southern California, where he was a member of the Sigma Chi fraternity.

Table 3: Examples of types of semantic drift described in section 4.2. The "Suthida" example gets stuck on a loop of false facts after the phrase "former flight attendant". The "Steve Cummins" example shows one correct fact followed by many made-up ones. We classified the "Matthew Perry" example as plausible information, since Perry intended to enroll at the University of Southern California; same phrasing for fraternity appears on Perry's father's Wikipedia page; and there was a Phil Perry attending Sigma Chi fraternity.

improve generation quality. Specifically, if we can detect semantic drift during inference, we can stop generation (hence, improve its quality) – we will do this in Section 6. But before that, let us check whether there are metrics that, during generation, can indicate that the model is drifting away.

Previous work on alleviating hallucinations for various NLP tasks, such as machine translation, abstractive summarization and long-form question answering, showed that hallucinations are well-calibrated with model uncertainty (Lin et al., 2022; Kadavath et al., 2022; Liu et al., 2022; Guerreiro et al., 2023; Manakul et al., 2023). Here, we check whether uncertainty metrics correlate well with factual accuracy; for this, we use all sentences in the generated paragraphs (4516 sentences in total).

5.1 Considered Uncertainty Metrics

Intrinsic metrics: We consider entropy of token probability distributions (averaged within a sentence), variance in entropy of token probability distributions (averaged within a sentence), negative log likelihood of the sentence. These metrics were used before in Lin et al. (2022); Manakul et al. (2023); Liu et al. (2022).

Intrinsic, averaged over samples: to reduce noise in the metrics above, we sample each sentence 5 times and average the intrinsic uncertainty metrics over these samples.

Sampling-based (sentence similarity) metrics: SelfCheck-BERTScore, SelfCheck-MQAG and SelfCheck-ngram (1, 5 and 10) from (Manakul et al., 2023). SelfCheck-BERTScore[4] assigns a unique score to a sentence, signifying how factual that sentence is (0 = factual, 1 = non-factual). To calculate the score, we sample N new paragraphs for each biography: P_1, P_2 and P_N. For each sentence S, we get the most similar sentence S_i in each paragraph P_i, by considering maximum BERTScore. The SC-BERTScore is then calculated as $1-$ avg[BERTScore(S, S_i)]. For more details, see the original paper (Manakul et al., 2023).

5.2 Results

We find that intrinsic uncertainty metrics have little correlation with factual accuracy and that averaging across samples does not improve this. Differently, sampling-based uncertainty metrics give much higher correlation scores; highest score gives SC-BERTScore with a Pearson correlation coefficient of -0.41 (Appendix C).

6 Mitigating Factual Errors

As we explained above, the presence of semantic drift suggests that factual accuracy can be improved within the same generated paragraphs simply by shortening them. Therefore, we first consider several criteria to stop generation early. Next, we try resample-then-rerank pipeline, as well as calling API tools. We compare these methods through the lens of factuality vs informativeness trade-off (Figure 4).

[4] From here onwards, "SC-BERTScore" for short.

Method	Stop at	facts/gen	No ans.	FActScore* (%)	Recall (%)	SD Score (%)	Flops* int.	ext.
Baseline	max tokens	43.89	1	44.56	-	78.07	1e16	0
Early stopping								
oracle	drift point	10	1	81.68	41.76	47.94	2e15	0
EOS	EOS in top 5	14.47	3	57.96	42.71	74.20	5e15	0
	EOS in top 10	5.39	13	70.29	18.90	63.81	1e15	0
SC-BERT	SC-BERT incr. >0.7	17.65	1	64.76	58.44	66.87	6e16	3e16
	SC-BERT incr. >0.5	13.39	1	67.63	46.30	65.20	6e16	2e16
	SC-BERT incr. >0.3	9.66	1	70.24	34.69	**61.90**	5e16	1e16
Reranking (SC-BERT)								
	max tokens	40.21	1	53.27	-	74.84	1e17	3e17
	SC-BERT incr. >0.7	22.75	1	63.72	**67.67**	69.15	1e17	1e17
	SC-BERT incr. >0.5	17.12	1	67.18	53.69	67.49	9e16	1e17
	SC-BERT incr. >0.3	11.64	1	**71.11**	38.64	63.94	5e16	6e16
API call								
one QA call	max tokens	42.26	1	43.93	-	80.02	3e16	1e10
∞ QA calls	max tokens	19.36	54	54.42	-	77.43	1e16	3e10

Table 4: FActScore* and SD score for LLaMa2 70B with generation strategies and early stopping methods from Section 6, based on *eos_top_k*, SelfCheck-BERTScore(N = 3), question answering calls or the oracle @drift point. Recall shows %correct facts left from baseline. "No ans" shows number of paragraphs (out of 500) with no facts. "Flops*" approximates the total number of (internal and external) floating point operations.

6.1 Early Stopping

We consider several early stopping methods.

Oracle: at drift point. This method stops generating at the drift point (Section 2.1). While this cannot be achieved at inference time (finding the drift point requires ground truth that is not available at test time), this method gives us a theoretical upper bound of factual accuracy for early stopping methods and a reference point for other methods.

Incentivizing EOS. As a naive baseline, we encourage the model to end the generation early by producing the EOS token whenever this token is in the top-k predicted tokens.

Using sentence similarity. Inspired by the correlation results between sentence similarity metrics and factual accuracy in Section 5.1, we also consider early stopping based on decline in consistency. For this, we:

Step 1: Compute SC-BERTScore for the original generated biographic paragraphs;
Step 2: For each paragraph, calculate the percentage of increase in this score from sentence S_i to S_0;
Step 3: If this percentage is more than a threshold T (i.e., consistency declined), stop the generation right before sentence S_i.

Here, T controls how much information should be traded for factuality: a low T will result in shorter generations with higher factuality. Note also that SC-BERTScore depends on the number of paragraph samples N which controls the accuracy of the scoring. Since using $N > 3$ does not give noticeable improvements (Figure 4), we use $N = 3$.

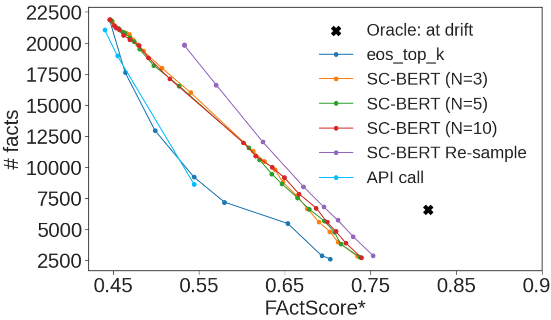

Figure 4: Trade-off between informativeness (y-axis) and factuality (x-axis) for proposed generation strategies; average over 500 biographical paragraphs.

6.1.1 Results: Early Stopping Helps

The results are shown in Table 4. As expected, oracle (stopping at drift point) is the best: it has the highest factuality and the lowest drift score. Other early stopping methods also improve quality quite a lot and can achieve over 70% accuracy (vs 44% for the baseline) and low semantic drift score of 62 (vs 78 for the baseline). Naturally, stopping early leads to information loss and fewer generated facts overall. Therefore, we can compare different methods only in settings where, on average, they generate the same number of facts. Comparing EOS-in-top-5 with SC-BERTScore (0.5), we see that SC-BERTScore is better: for roughly the same number of generated facts per paragraph (13-14), it gives 10% higher accuracy and lower SD score. In terms of flops, however, EOS-based stopping is an order of magnitude more efficient.[5]

6.2 Re-Sample, Then Rerank

In addition to early stopping, SC-BERTScore can be used in resample-then-rerank pipelines typical for alleviating hallucinations in machine translation (Guerreiro et al., 2023; Dale et al., 2023).

Method. For each biography, we generate one sentence at a time. For each sentence, we generate 5 options (using same decoding strategy, only different seeds) and choose the one which (1) has not appeared before in the paragraph; (2) has minimum SC-BERTScore. We generate sentences until no options satisfy condition (1) or we have reached the maximum number of tokens.

Results. When stopping at the maximum number of tokens, this approach improves the baseline by 8.71%. This is expected: similar approaches improve e.g. machine translation quality by a large margin (Guerreiro et al., 2023; Dale et al., 2023). When combining reranking with the early-stopping, we get same factuality as the corresponding early stopping, but with more generated facts. For example, for the same factuality of around 67% we generate 13.4 facts with early stopping (SC-BERT, $T = 0.5$) but 17.1 facts when combining it with reranking. Sadly, this improvement comes with a large increase in flops.

6.3 Calling Question Answering API

In this section, we ask: *If the model drifts away during generation, could it be brought back onto a correct path by calling an external API?*

Method. To answer this question, we use 1-shot learning to allow LLaMa2-70B asking questions at inference time (as in Toolformer, Schick et al. (2023)). The model makes calls to Atlas, which is a retrieval-augmented model with 11B parameters (Izacard et al. (2022), example inference in Appendix E). We estimated the computation cost in Table 4 by approximating the cost of an API call as the cost of one pass through the model. We find that when allowing for multiple calls, generations are shorter and therefore require fewer passes through LLaMa2. Therefore, overhead for adding QA calls is small.

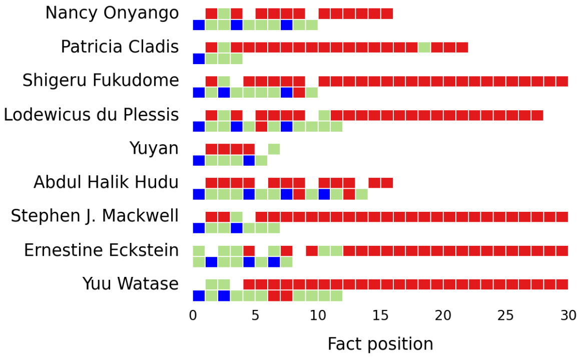

Figure 5: Examples of biographies that were most improved by adding QA calls. Each row represents a biography with two generated versions (one without QA calls and one with). Green – correct facts, red – incorrect facts, blue – API calls.

Results: the worst. Surprisingly, using API calls gives close to no improvement: for similar number of facts per generation, it is 10% less accurate than other methods. This is largely due to the model not handling errors of the API. We found that adding more examples (and examples which ignore the API return) damaged the performance. The model makes many unnecessary calls, as it does not have an understanding of "needing" to retrieve information, rather it retrieves information whenever convenient.

Adding calls in a few-shot manner poses a new challenge for semantic drift. We noticed that after generating a paragraph, the model would start a new paragraph about the API call (e.g., "To make API calls use this method..." or "The API calls were executed at..."). In addition to being irrelevant, this is entirely hallucinated content. Removing this does not help significantly: it gives the SD score of 77.43%, which is only marginally lower than the baseline. For those samples which show most improvement in factuality, we note that the drift has been addressed by making many simple calls on almost every fact (Figure 5).

7 Beyond Biographies

The methods we presented can in principle be applied to any type of text generation, not just biographical paragraphs. To showcase these capabili-

[5] For SC-BERTScore stopping, the computation is twofold: sampling four paragraphs instead of one and computing SC-BERTScore (three passes through RoBERTa-Large (Liu et al., 2019)). This results in an order of magnitude more flops than the baseline.

Method	Stop at	toks/gen	Fact Score	Triple Score	QAG Score	ROUGE-L
Baseline	max tokens	223.34	19.45	10.57	30.91	4.17
Early stopping						
	`EOS` in top 5	60.33	12.72	6.12	36.60	6.88
	SC-BERT incr. >0.5	184.84	19.05	10.35	33.06	4.96
Reranking (SC-BERT)						
	max tokens	189.60	20.79	11.71	36.54	5.04
	SC-BERT incr. >0.5	157.67	20.61	11.77	36.68	5.11

Table 5: Factual accuracy for different generation strategies for Llama2-70B, when applied to the task in Section 7, of generating 5000 Wikipedia articles. Each score represents a measure of factual accuracy of the generated text with respect to the real Wikipedia article. All scores are calculated using the FactSumm pipeline (Heo, 2021).

ties, we apply them to writing any Wikipedia-style text. We prompt LLaMa2-70B to generate text about a topic ("This is a wikipedia article about *topic*.") and pass the generated text through the FactSumm pipeline (Heo, 2021).

Pipeline. The original goal of the pipeline is to measure factuality of a summary with respect to reference text. We re-purpose it to measure factuality of generated text with respect to the original Wikipedia article. We retrieve 5000 English Wikipedia articles[6] and calculate mean FactScore, TripleScore, QAG and ROUGE Score.

Evaluation. FactScore and TripleScore extract triplets (closed- and open-scheme, respectively) and score the overlap of these triplets between the reference and generated texts. For QAG Score, the module generates question-answer pairs based on the generated text, attempts to answer the questions based on the reference text and notes the number of identical answers. ROUGE calculates the similarity between the two texts based on n-grams matches. Together, all these scores paint a picture of the factuality of the generated text with respect to the Wikipedia article. We note that none of these metrics consider recall, but that we provide the average number of tokens generated per paragraph as a measure of information quantity.

Results: reranking helps again. Table 5 shows that, in this more general setting, reranking yields higher factual accuracy at the cost of a reduction in generated facts. Therefore, this method has a positive impact on factual text generation beyond biographies. Here, we applied different metrics from FActScore* to assess the same phenomenon, offering a fresh perspective. Despite the different metrics, our methodology remained unaltered. The fact that the presented methods are robust to various metrics underscores their generality.

8 Additional Related Work

Factual precision. Recent surveys (Wang et al., 2023; Rawte et al., 2023; Ji et al., 2023) show that factuality evaluation has mostly been focused on short-form question answering, and improvements have largely been based on learning (pre-training, fine-tuning) or retrieval augmentation. Previous work (Lee et al., 2022) attempts decoding-time enhancements, but reports these alone achieve factuality on-par with greedy decoding and concludes the need for training enhancements. Concurrent work (Chuang et al., 2023) contrasts various layers' logits. As opposed to SC-BERTScore methods, this requires access to model's internals and changes to inference code; futhermore it is not evaluated on long-form generation and restricted to one model class. Unlike other factuality enhancements, our methods do not directly fix incorrect facts, but use the semantic drift idea to inform when the model has "ran out of correct facts". They can be combined with any others to generate accurate and relevant text.

Semantic drift. Deng et al. (2022); Cho et al. (2019) characterize it linguistically via self-consistency, not truthfulness. Plausible and naturally flowing text would not be identified as drift.

9 Conclusion

By measuring the degree of separation between the correct and incorrect facts in the generated texts, we show that LLMs largely generate correct facts first and incorrect later. This lead us to methods that improve factual accuracy by stopping generation early. We show that even a simple method that encourages generating `EOS` leads to

[6] From Huggingface (2023).

large improvements. This can further be improved by using a resample-then-rerank pipelines where for each sentence, we generate several versions and choose the best based on sentence similarity measures. Overall, our methods offer a practical compromise, balancing computation with performance, and build a foundation for further research. Importantly, they are directly applicable to any probabilistic auto-regressive language models.

10 Limitations

Model specifics. We have applied our methods to LLaMa2-70B model and we trust that incentivising the `EOS` token and the SelfCheck-BERTScore methods will work similarly well for other models. However, we note that the thresholds are likely not directly transferable to other models and that in order to employ similar strategies, model owners will have to tweak the thresholds to figure out the correct numbers for their case.

Suitable tasks. Even though our methods can be applied to any long-form text generation task, they are perhaps most relevant for tasks where factual accuracy is paramount (such as long form question answering or factual text generation). Early-stopping methods specifically are more suitable for tasks where generating false information is more harmful than not generating it (for example giving false medical advice). Our oracle for early-stopping removes 92% of incorrect facts from the generated text, but this comes with the cost of removing 58% of correct facts. These measurements (as can be seen in appendix F.1) should be used for individual applications to debate trade-offs.

Textual diversity. As this study is focused on factually-dense text, we did not take into account diversity of generated text, which may be relevant for more creative tasks such as story generation. For early stopping via sentence similarity, we chose to use SelfCheck-BERTScore which is sensitive to stylistic variations, as well as factual variations. However, there is no reason for which this metric cannot be replaced with other sentence similarity-metrics which account for style, thus retaining the creative factor of text generation.

Automated evaluation. We have used the FActScore pipeline, which is an automated evaluation pipeline for validating truthfulness of facts. We have validated the pipeline with human annotations (as detailed in Appendix B.2), but as any automated pipeline it has an error margin. The reliability of the pipeline is heavily dependant on the reliability of its knowledge source, which in this case is Wikipedia - one of the most commonly-used, accessible, large-scale, good quality, unstructured knowledge sources (Lee et al., 2022).

Future direction. One can imagine many possible avenues of future directions for further understanding and mitigating semantic drift. For example, models could be further fine-tuned specifically to end generation when there is too much variability in the generation, critique models could be trained to identify the drift point based on model's internal states etc. We hope that with our work we have sufficiently highlighted the problem and set the first stepping stones for addressing it.

11 Acknowledgments

We would like to thank Sewon Min for answering our questions about the FactScore (Min et al., 2023) pipeline and accompanying code.

References

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Merouane Debbah, Etienne Goffinet, Daniel Heslow, Julien Launay, Quentin Malartic, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. 2023. Falcon-40B: an open large language model with state-of-the-art performance.

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

Woon Sang Cho, Pengchuan Zhang, Yizhe Zhang, Xiujun Li, Michel Galley, Chris Brockett, Mengdi Wang, and Jianfeng Gao. 2019. Towards coherent and cohesive long-form text generation. In *Proceedings of the First Workshop on Narrative Understanding*, pages 1–11, Minneapolis, Minnesota. Association for Computational Linguistics.

Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James Glass, and Pengcheng He. 2023. Dola: Decoding by contrasting layers improves factuality in large language models.

David Dale, Elena Voita, Loic Barrault, and Marta R. Costa-jussà. 2023. Detecting and mitigating hallucinations in machine translation: Model internal workings alone do well, sentence similarity Even better. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 36–50, Toronto, Canada. Association for Computational Linguistics.

Yuntian Deng, Volodymyr Kuleshov, and Alexander Rush. 2022. Model criticism for long-form text generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11887–11912, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Daniel Deutsch, Rotem Dror, and Dan Roth. 2022. On the limitations of reference-free evaluations of generated text.

Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. 2023. Chain-of-verification reduces hallucination in large language models.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. Eli5: Long form question answering.

Zihao Fu, Wai Lam, Anthony Man-Cho So, and Bei Shi. 2021. A theoretical analysis of the repetition problem in text generation.

Nuno M. Guerreiro, Elena Voita, and André Martins. 2023. Looking for a needle in a haystack: A comprehensive study of hallucinations in neural machine translation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1059–1075, Dubrovnik, Croatia. Association for Computational Linguistics.

Hoon Heo. 2021. Factsumm: Factual consistency scorer for abstractive summarization. `https://github.com/Huffon/factsumm`.

Huggingface. 2023. Wikipedia dataset (en). `https://huggingface.co/datasets/graelo/wikipedia`.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2022. Atlas: Few-shot learning with retrieval augmented language models.

Hastie T. Tibshirani R. Taylor J. James G., Witten D. 2023. *An Introduction to Statistical Leraning with Applications in Python*. Springer, Switzerland.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, Jackson Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom Brown, Jack Clark, Nicholas Joseph, Ben Mann, Sam McCandlish, Chris Olah, and Jared Kaplan. 2022. Language models (mostly) know what they know.

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

Nayeon Lee, Wei Ping, Peng Xu, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. 2022. Factuality enhanced language models for open-ended text generation. *ArXiv*, abs/2206.04624.

Chenliang Li, Bin Bi, Ming Yan, Wei Wang, and Songfang Huang. 2021. Addressing semantic drift in generative question answering with auxiliary extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 942–947, Online. Association for Computational Linguistics.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Teaching models to express their uncertainty in words.

Tianyu Liu, Yizhe Zhang, Chris Brockett, Yi Mao, Zhifang Sui, Weizhu Chen, and Bill Dolan. 2022. A token-level reference-free hallucination detection benchmark for free-form text generation.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories.

Potsawee Manakul, Adian Liusie, and Mark John Francis Gales. 2023. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *ArXiv*, abs/2303.08896.

William C. Mann. 1983. An overview of the penman text generation system. In *AAAI Conference on Artificial Intelligence*.

Nick McKenna, Tianyi Li, Liang Cheng, Mohammad Javad Hosseini, Mark Johnson, and Mark Steedman. 2023. Sources of hallucination by large language models on inference tasks.

Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen tau Yih, Pang Wei Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. Factscore: Fine-grained atomic evaluation of factual precision in long form text generation.

Dor Muhlgay, Ori Ram, Inbal Magar, Yoav Levine, Nir Ratner, Yonatan Belinkov, Omri Abend, Kevin Leyton-Brown, Amnon Shashua, and Yoav Shoham. 2023. Generating benchmarks for factuality evaluation of language models. *ArXiv*, abs/2307.06908.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2022. Webgpt: Browser-assisted question-answering with human feedback.

OpenAI. 2023. Gpt-4 technical report.

Hongjin Qian, Zhicheng Dou, Jiejun Tan, Haonan Chen, Haoqi Gu, Ruofei Lai, Xinyu Zhang, Zhao Cao, and Ji-Rong Wen. 2023. Optimizing factual accuracy in text generation through dynamic knowledge selection.

Vipula Rawte, Amit Sheth, and Amitava Das. 2023. A survey of hallucination in large foundation models.

Ehud Reiter and R. Dale. 1997. Building applied natural language generation systems. *Natural Language Engineering*, 3:57 – 87.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools.

Dan Su, Xiaoguang Li, Jindi Zhang, Lifeng Shang, Xin Jiang, Qun Liu, and Pascale Fung. 2022. Read before generate! faithful long form question answering with machine reading.

Ruixiao Sun, Jie Yang, and Mehrdad Yousefzadeh. 2020. Improving language generation with sentence coherence objective.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7534–7550, Online. Association for Computational Linguistics.

Alex Wang, Kyunghyun Cho, and Mike Lewis. 2020. Asking and answering questions to evaluate the factual consistency of summaries.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`.

Cunxiang Wang, Xiaoze Liu, Yuanhao Yue, Xiangru Tang, Tianhang Zhang, Cheng Jiayang, Yunzhi Yao, Wenyang Gao, Xuming Hu, Zehan Qi, Yidong Wang, Linyi Yang, Jindong Wang, Xing Xie, Zheng Zhang, and Yue Zhang. 2023. Survey on factuality in large language models: Knowledge, retrieval and domain-specificity.

Wei Wang, Piji Li, and Hai-Tao Zheng. 2021. Sentence semantic regression for text generation.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. 2023. Instruction tuning for large language models: A survey.

Shiyue Zhang and Mohit Bansal. 2019. Addressing semantic drift in question generation for semi-supervised question answering. In *Conference on Empirical Methods in Natural Language Processing*.

Chunting Zhou, Graham Neubig, Jiatao Gu, Mona Diab, Paco Guzman, Luke Zettlemoyer, and Marjan Ghazvininejad. 2021. Detecting hallucinated content in conditional neural sequence generation.

A Appendix A

A.1 Coherence, relevance and truthfulness

The definition of semantic drift in Section 2.1 states that the effects of semantic drift can be noted as a loss in coherence, relevance or truthfulness. The FActScore task described in Section 3.1 identifies all three categories of semantic drift effects. Here, we show examples of each.

Coherence example

Below an example of a generated biography, followed by the extracted facts together with their assigned labels. Inconsistency in the birth year is identified as False.

```
text:
Iggy Azalea (born 7 June 1990) is a
rapper and singer.  She was born in
1989.
facts:
Iggy Azalea was born.  (True)
Iggy Azalea was born on June 7.
(True)
Iggy Azalea was born on June 7,1990.
(True)
Iggy Azalea.  is a rapper.  (True)
Iggy Azalea is a singer.  (True)
She was born.  (True)
She was born in 1989.  (False)
```

Incoherence is probably one of the most studied types of semantic drift so far. Examples from literature include:

```
 ``She had a large family and lived
with her grandparents ...  In 1933
she gave birth to her first child
...  In July 1926, many of her friends
attended her funeral'' (Liu et al.,
2022)
``Willie had too much stuff.  Willie
bought a shed to store all his stuff.
Willie had a hard time putting up
the shed.  He called some friends
for help.  Willie sold his shed and
and made enough money to pay for the
house.''  (Wang et al., 2021)
```

Relevance example

As per our definition, a loss of relevance refers to the inclusion of irrelevant or redundant content. In the below example facts which are actually correct, but irrelevant to the context are labelled False.

```
text:
Iggy Azalea is a rapper and singer.
Mariah Carey is a singer.  Eminem is a
singer.  Bob Marley is also a singer.
facts:
Iggy Azalea is a rapper.  (True)
Iggy Azalea is a singer.  (True)
Mariah Carey exists.  (False)
Mariah Carey is a singer.  (False)
Eminem is a singer.  (False)
Bob Marley is a singer.  (False)
```

Truthfulness example Truthfulness refers to the objective factuality of information, whether it is verifiable or not. In the example below, the scorer picks up on subtle inaccuracies (“Ignorant Art” is a mixtape, not an album).

```
text:
Iggy Azalea is a rapper from
Melbourne, Australia.  She is known
for her hit single "Fancy" and her
debut album Ignorant Art.
facts:
Iggy Azalea is a rapper.  (True)
Iggy Azalea is from Melbourne,
Australia.  (True)
Melbourne is a city in Australia
(True).
She is known for her hit single
"Fancy".  (True)
She has a debut album called Ignorant
Art.  (False)
"Fancy" is a hit single.  (True)
Ignorant Art is a debut album.
(False)
Ignorant Art is an album.  (False)
```

A.2 Potential reasons for semantic drift

Semantic drift is the term for a fairly broad phenomenon and there may be many reasons why it occurs. Some initial thoughts based on observations are:

- Ambiguity: AI models may interpret ambiguous terms or phrases in ways that lead to a shift in the text’s meaning.
- Loss of context: As text becomes longer, the model may lose track of the context.
- Digression: The AI model might include lengthy tangents or irrelevant information that detracts from the primary topic.

A.3 SD Score and Purity

The SD Score was inspired by purity measures in classification decision trees and can be seen as an edge-case calculation of purity. Recall that classification decision trees are made-up of nodes, where each node splits the training dataset into partitions using a criterion function on features of the data points. For each partition corresponding to a leaf node, the predicted class is the most common class in that partition. To find the right decision tree, we measure the purity of the partitions it creates.

In the case of our task, the elements in the dataset are facts which have one feature: *index* and are assigned a class: *is_supported*. The "decision tree" only has one split node ($<$*index*) resulting in two leaf nodes (left-side and right-side). We then assign class 1 to every data point in the left-side and class 0 to every data point in the right-side. It is important to note that we always assign classes in this manner, regardless of which class is most common in the partition.

B Appendix B

B.1 LLaMa2 70B Generation details

We generate a maximum of 500 tokens with LLaMa2 70B model, with $temperature = 0.6$ and $topp = 0.9$. After generation we delete any unfinished sentences, as we found that the FActScore atomic fact extractor would hallucinate new facts when dealing with unfinished sentences. For analysis we also remove repetition, i.e. if the last sentence is repeated, then we remove it and stop generating. Our generated paragraphs have an average length of 255 tokens.

B.2 FActScore Pipeline for LLaMa2

We adapt the FActScore pipeline to rely on LLaMa2. The FActScore pipeline uses InstructGPT to extract atomic facts from input paragraphs. To do this, the model is given few-shot examples of atomic fact extraction. We use the same examples for LLaMa2 70B chat. To validate the performance of LLaMa2, we compare it against human annotations provided with the FActScore paper. The annotations consist of 180 paragraphs with extracted facts. We obtain a Pearson correlation coefficient of 0.94 between scores obtained from facts extracted by humans and scores obtained from facts extracted by LLaMa2.

B.3 Impact of sampling strategy

As can be seen in Figure 6, there is no considerable difference between the SD score obtained with greedy as opposed to nucleus sampling. We do note however, that the greedy-generated paragraphs tend to be more repetitive.

B.4 Truncation

We define truncation in our particular case as described in Section 2.1. We apply it in order to distinguish cases where the semantic drift high score is only caused by few samples to either side of the drift point. We experiment with $m \in [0, 5]$ to see how the distribution of SD score and drift position are impacted. Results are in Figure 7. We find that with $m = 3$, there are 44.89% paragraphs with SD score >0.75.

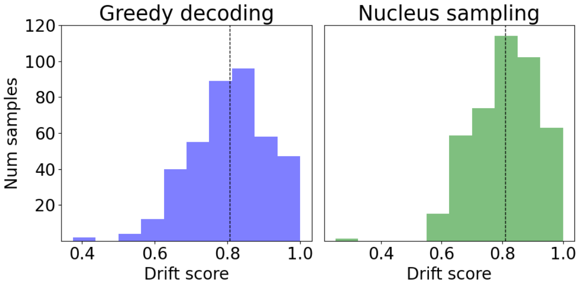

Figure 6: Comparison of SD score distribution in LLaMa2 70B based on decoding strategy.

B.5 Statistical significance test

For each paragraph with identified semantic drift (40% of samples with SD score >= 0.75), we estimate the probability that the assigned SD score is due to chance. We shuffle the fact labels from the paragraph 1000 times and calculate the SD score for each shuffle. We find that, on average, higher or equal SD score is obtained in less than 0.02% of shuffles[7].

B.6 Semantic drift identified examples

Figure 8 visually shows examples of paragraphs which display clear cases of semantic drift for various lengths of the paragraph.

C Appendix C

C.1 Uncertainty metrics correlations

Figure 9 shows the correlation coefficients of all uncertainty metrics ran for our experiments. As mentioned in the main paper, the most significant correlation was for SelfCheck-BERTScore, calculated over 3 samples.

D Appendix D

D.1 SelfCheck-BERTScore

For methods described in Section 6.1, we conducted more experiments to determine how threshold T on the relative increase in SelfCheck-BERTScore should be chosen and whether it could be an absolute value threshold, as opposed to a relative increase value. We also provide more details for how the paragraph samples were generated.

[7] https://en.wikipedia.org/wiki/Permutation_test

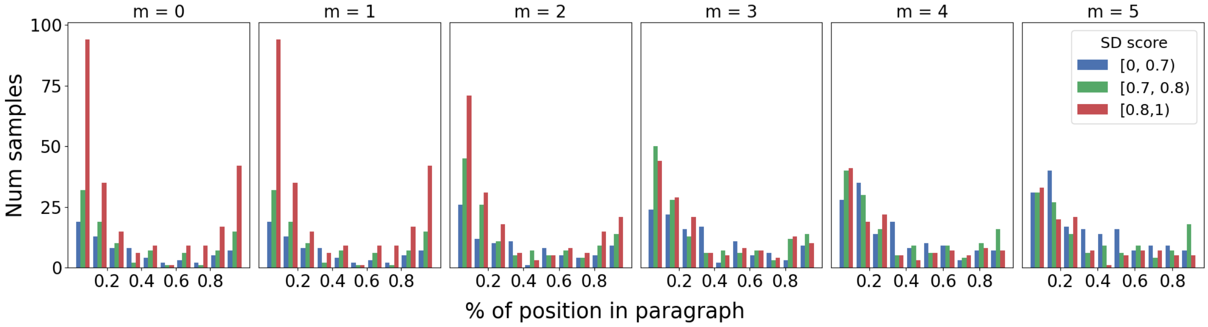

Figure 7: Drift position distribution after applying truncation varying the minimum number m of facts on either side of the potential drift point, as described in Section 2.1.

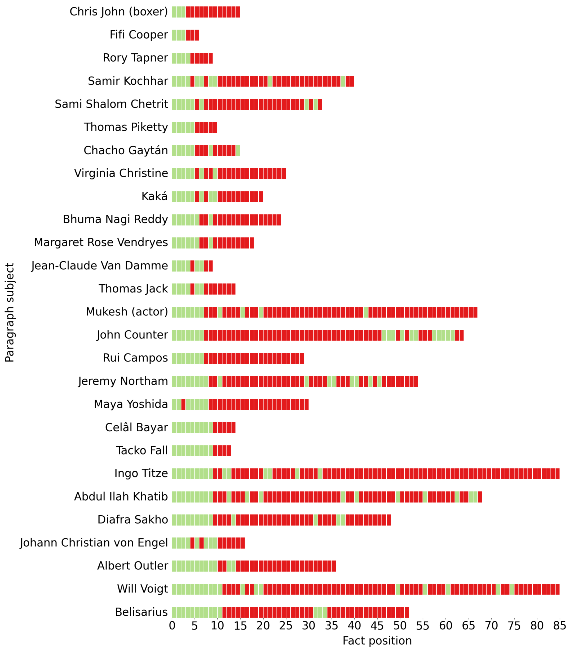

Figure 8: Examples of drifting paragraphs generated with LLaMa2 70B according to their SD score. Each row represents a paragraph where correct facts are in green, wrong facts are in red.

D.2 Absolute thresholds for ending generation

A more naive method for stopping generation using SelfCheck-BERTScore is to simply threshold the absolute value of the score and stop generation whenever the score crosses the threshold. When applying the method, we found that many biographies would actually begin with a first sentence above the threshold, thus resulting in empty paragraphs for any value of the threshold sufficiently low to be useful. We show the results of either keeping or deleting those biographies which begin above the threshold in Table 7.

An interesting corollary finding is the distribution of SelfCheck-BERTScore in the first sentence by popularity class of the topic. The paragraphs which have highest SelfCheck-BERTScore in the first sentence are those with lower popularity, and consequently those for which the above method would not generate any facts (Figure 10).

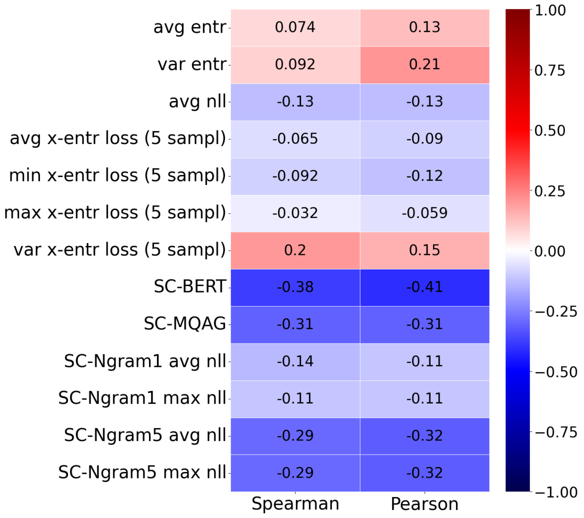

Figure 9: Correlation coefficients between all uncertainty metrics and fact accuracy. Calculated over 4516 sentences generated by LLaMa2 70B.

D.3 Number of sampled paragraphs

The calculation of the SelfCheck-BERTScore hinges on using N sampled paragraphs. Each sentence in the original paragraph is scored based on its BERTScore with respect to each sampled paragraph. We experimented with $N \in \{1, 3, 5, 10, 100\}$. We found only marginal improvements for $N > 5$ and that the improvements are more visible when using smaller T. We also experimented with generating N paragraph samples with a temperature setting of 1. As the original paper Manakul et al. (2023) suggests, high temperature should result in more accurate

SelfCheck-BERTScore. However, we find the improvements on $temperature = 1$ to be marginal from $temperature = 0.6$, $topp = 0.9$.

D.4 Rerank past-early stopping point

One interesting experiment, but which did not yield satisfactory results, was to use the early-stopping strategy described in Section 6.1 and combine it with the reranking strategy from Section 6.2 by resampling-and-reranking only past the early-stopping point. The hope was that we can extend current paragraphs by adding more correct facts. We found that we could extend the paragraphs with an average of 2.12 facts per generated paragraph, but that this came with a loss of factuality of 1.67%.

E Appendix E

E.1 Inference with API call

Below is an example illustrating the inference flow for generating text with embedded API calls. It consists mainly of two prompts: one for defining how to make the API call and one for integrating the response of the API call. When finished with inference step 2, we remove the API call from the generated text and repeat the same flow again with the previously generated text as the new prompt we want to complete.

Prompt 1:

```
Your task is to add calls to a
Question Answering API to a piece of
text.  The questions should help you
get information required to complete
the text.  You can call the API by
writing [QA(question)] where question
is the question you want to ask.  Here
are some examples of API calls:
Joe Biden was born in [QA(Where was
Joe Biden born?)]
This is a Wikipedia article about
Napoleon.  Napoleon
```

Inference 1:

```
was born in [QA(Where was Napoleon
born?)]
```

Execute API call.

Prompt 2:

```
Your task is to complete a piece of
text, by using answers from an API
call.  APIs are called by writing
[QA(question) -> answer] where
question is what was sent to the API
and answer is the response.  Here are
some examples of texts with API calls:
Joe Biden was born in [QA(Where
was Joe Biden born?)  -> Scranton]
Scranton, Pennsylvania.
Napoleon was born in [QA(Where was
Napoleon born?)  -> Ajaccio]
```

Inference 2:

```
was born in [QA(Where was Napoleon
born?)  -> Ajaccio] Ajaccio, Corsica.
```

Repeat.

F Appendix F

F.1 Factual precision and recall metrics

Because FActScore is a precision-focused metric, to get a better idea of the impact of each regeneration strategy and each early-stopping strategy, we provide more metrics in Table 6. The recall on incorrect facts shows the percentage of incorrect facts that were present in the original generation, then removed by the early stopping method.

Method	Incorrect facts		Correct facts	
	Prec.	Rec.	Prec.	Rec.
baseline			44.56	
oracle	66.39	**92.47**	**81.68**	41.76
eos_top_5	61.99	75.1	57.96	42.71
eos_top_10	58.94	93.58	70.29	18.90
SC-Bert >.7	**69.03**	74.44	64.76	58.44
SC-Bert >.5	65.57	82.19	67.63	46.30
SC-Bert >.3	62.69	88.19	70.24	34.69
re-rank			53.27	
rerank + SC-Bert >.7	60.35	56.07	63.72	**67.67**
rerank + SC-Bert >.5	57.04	70.1	67.18	53.69
rerank + SC-Bert >.3	54	82.1	71.11	38.64
1 API call			43.93	
inf API calls			54.42	

Table 6: Metrics for incorrect facts, showing precision (%incorrect facts out of those removed by early stopping), recall (%incorrect facts removed); and for correct facts, showing precision (FActScore) and recall (%remaining correct facts). For each of the generation strategies, the metrics are calculated with respect to the base generation.

Early stopping (threshold T)	**if $S_0 > T$**	**facts /gen**	**No answer**	**FAct Score* (%)**	**SD score (%)**
SC-BERTScore >0.8	keep S_0	28	1	52.73	75.77
SC-BERTScore >0.5	keep S_0	16.21	1	66.05	66.87
SC-BERTScore >0.2	keep S_0	21.92	1	62.87	67.63
SC-BERTScore >0.8	delete S_0	21.26	17	58.07	74.18
SC-BERTScore >0.5	delete S_0	4.64	272	88.66	66.87
SC-BERTScore >0.2	delete S_0	4	493	92.85	58.64
@drift point	n/a	10	1	81.68	47.94

Table 7: Comparing FActScore* and SD score for LLaMa2 70B cutting generation based on SelfCheckBERT Score threshold T. The second column shows behaviour in the case in which the first sentence of the generation already exceeds threshold T. "No answer" shows the number of paragraphs (out of the total 500) for which the model produces no facts.

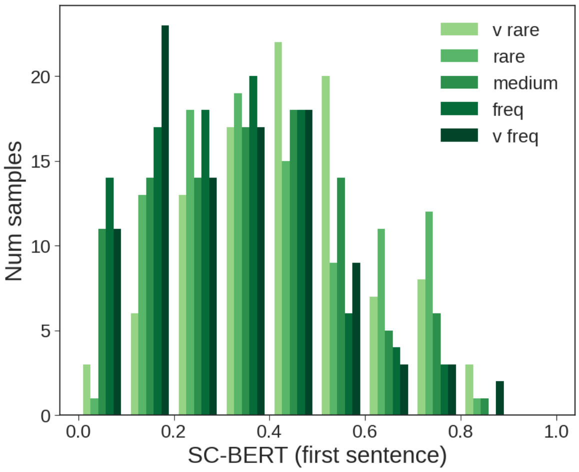

Figure 10: Distribution of SelfCheck-BERTScore for first sentence in paragraph, by popularity of topic.