

# SELFCKGPT: Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models

Potsawee Manakul, Adian Liusie, Mark J. F. Gales

Department of Engineering, University of Cambridge

pm574@cam.ac.uk, al826@cam.ac.uk, mjfg@eng.cam.ac.uk

## Abstract

Generative Large Language Models (LLMs) such as GPT-3 are capable of generating highly fluent responses to a wide variety of user prompts. However, LLMs are known to hallucinate facts and make non-factual statements which can undermine trust in their output. Existing fact-checking approaches either require access to the output probability distribution (which may not be available for systems such as ChatGPT) or external databases that are interfaced via separate, often complex, modules. In this work, we propose "SelfCheckGPT", a simple sampling-based approach that can be used to fact-check black-box models in a zero-resource fashion, i.e. without an external database. SelfCheckGPT leverages the simple idea that if a LLM has knowledge of a given concept, sampled responses are likely to be similar and contain consistent facts. However, for hallucinated facts, stochastically sampled responses are likely to diverge and contradict one another. We investigate this approach by using GPT-3 to generate passages about individuals from the WikiBio dataset, and manually annotate the factuality of the generated passages. We demonstrate that SelfCheckGPT can: i) detect non-factual and factual sentences; and ii) rank passages in terms of factuality. We compare our approach to several baselines and show that in sentence hallucination detection, our approach has AUC-PR scores comparable to or better than grey-box methods, while SelfCheckGPT is best at passage factuality assessment.<sup>1</sup>

## 1 Introduction

Large Language Models (LLMs) such as GPT-3 (Brown et al., 2020), PaLM (Chowdhery et al., 2022), and Chinchilla (Hoffmann et al., 2022) are capable of generating highly fluent and realistic responses to a variety of user prompts. They have

<sup>1</sup>Coda and dataset can be found on the project page at <https://github.com/potsawee/selfcheckgpt>.

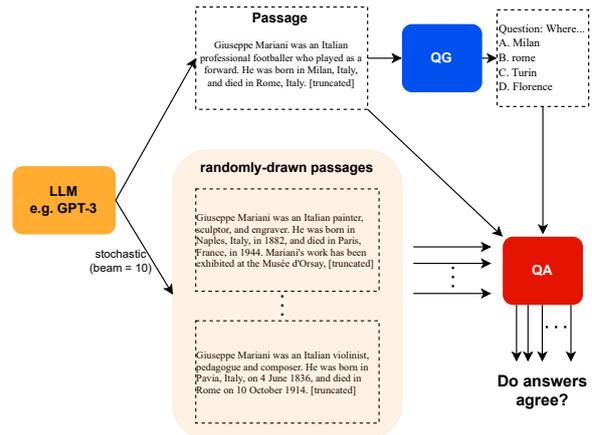


Figure 1: SelfCheckGPT with Question Answering.

been used in many applications such as automatic tools to draft reports, virtual assistants that retrieve information, summarization systems, as well as a multitude of other generative applications. Despite the convincing and realistic nature of LLM-generated texts, a concern with LLMs is their tendency to hallucinate facts and make up information.

A method for hallucination detection is to leverage existing intrinsic uncertainty metrics such as token probability or entropy since these metrics can be used to determine the parts of the output sequence the system is least certain of (Yuan et al., 2021; Fu et al., 2023). However, all current uncertainty metrics require access to the output token-level probability distribution information that may not necessarily be available to users, e.g. when systems are accessed used through limited external APIs such as ChatGPT. Further, there is an active field of fact-verification where evidence is retrieved from an external database to assess the veracity of a claim (Thorne et al., 2018; Guo et al., 2022). However, facts can only be assessed relative to the knowledge present in the database. Though corpora such as Wikipedia can cover a great deal of general knowledge and serve as a useful database for fact verification, hallucination is observed over

a wide range of tasks beyond pure fact verification. For example, summaries from automatic systems can contain information not present in the context (Kryscinski et al., 2019, 2020; Maynez et al., 2020).

In this paper, we propose SelfCheckGPT, a simple sampling-based approach that can detect whether responses generated by LLMs are hallucinated or factual. SelfCheckGPT only uses sampled responses and can therefore be used on black box models, while it also operates in a zero-resource fashion, i.e. with no external database. The motivating idea of SelfCheckGPT is that when a LLM knows a given concept well, the sampled responses are likely to be similar and contain consistent facts. However, for hallucinated facts, stochastically sampled responses are likely to diverge and may completely contradict one another. By sampling multiple responses from a LLM, one can measure information consistency between the different responses and determine which statements are factual and which have been hallucinated. Three variants of SelfCheckGPT for measuring informational consistency are considered: BERTScore, question-answering, and n-gram. Through analysis of annotated articles generated by GPT-3, we show that SelfCheckGPT can determine factual documents effectively in a black-box, zero-resource manner.

## 2 Related Work

### 2.1 Large Language Models

There has been rapid growth in current large language models (LLMs) literature with larger and better models being constantly released (Chowdhery et al., 2022). These models are commonly used as the backbone for a range of NLP tasks (Wang et al., 2018). Traditionally, these LLMs are fine-tuned to a specific task and/or domain (Devlin et al., 2019; Radford et al., 2019; Raffel et al., 2020), however, a fascinating finding is that as models scale up, they inherit abilities to naturally solve a wide range of natural language tasks in a zero-shot fashion (Brown et al., 2020; Wei et al., 2022).

### 2.2 Hallucination of Large Language Models

Hallucination has been studied in text generation tasks, including summarization (Huang et al., 2021) and dialogue generation (Shuster et al., 2021). A survey of hallucination in a variety of natural language generation tasks has been conducted (Ji et al., 2023). Further, Liu et al. (2022) compiled a hallucination detection dataset, but the texts were obtained

by perturbing factual texts; thus, this dataset may not reflect actual LLM hallucination.

Recently, Azaria and Mitchell (2023) trains a multi-layer perception classifier using LLM’s hidden representations as the input to predict a truthfulness of a sentence. This approach requires labelled data for supervised training as well as the internal states of the LLM, which may not be available through APIs. Another recent approach is self-evaluation (Kadavath et al., 2022), which is a method where the LLM is prompted to answer about its previous prediction, e.g. the probability of its generated response/answer is true.

### 2.3 Sequence Level Uncertainty Estimation

Token probabilities have been used as an indication of model certainty. For example, OpenAI’s GPT-3 web interface allows users to display token probabilities as shown in Figure 2. Additionally, uncertainty estimation based on aleatoric and epistemic uncertainty for autoregressive generation has been studied (Xiao and Wang, 2021; Malinin and Gales, 2021). Further, conditional language model scores have been used to evaluate properties of texts (Yuan et al., 2021; Fu et al., 2023). Recently, semantic uncertainty is proposed to address uncertainty in free-form generation tasks where probabilities are attached to meanings of text instead of tokens (Kuhn et al., 2023).

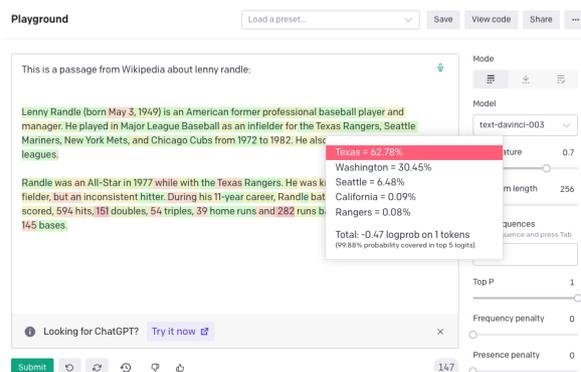


Figure 2: Example of OpenAI’s GPT-3 interface with token probabilities displayed.

### 2.4 Fact Verification

Existing fact-verification approaches follow a multi-stage pipeline of claim detection, evidence retrieval and verdict prediction (Guo et al., 2022; Zhong et al., 2020). Such methods, however, require access to external databases and can have considerable inference costs.

### 3 Grey-Box Factuality Assessment

This section will introduce methods that can be used to determine the factuality of LLM responses in a zero-resource setting when one has full access to output distributions.<sup>2</sup> We will use ‘factual’ to define when statements are grounded in valid information, i.e. when hallucinations are avoided, and ‘zero-resource’ when no external database is used.

#### 3.1 Uncertainty-based Assessment

**Motivation.** To consider how the factuality of a generated response can be determined in a zero-resource setting, we consider LLM pre-training. During pre-training, the model is trained with next-word prediction over massive corpora of textual data. This gives the model a strong understanding of language (Jawahar et al., 2019; Raffel et al., 2020), powerful contextual reasoning (Zhang et al., 2020), as well as world knowledge (Liusie et al., 2022). Consider the input "Lionel Messi is a \_". Since Messi is a world-famous athlete who may have appeared multiple times in pre-training, the LLM is likely to know who Messi is. Therefore, given the context, the token "footballer" may be assigned a very high probability while some other professions such as "carpenter" will be considered very improbable. However, for the input "John Smith is a \_", the system may be unsure of how the sentence should continue, and have a flat probability distribution. During decoding, this will lead to a random word being generated- causing the system to hallucinate.

This insight allows us to realize the connection between uncertainty metrics and factuality. Factual sentences are likely to contain tokens with higher likelihood and lower entropy, while hallucinations are likely to come from positions with flat probability distributions with high uncertainty.

##### Token-level Probability $p$

Given the LLM’s response  $R$ , let  $i$  denote the  $i$ -th sentence in  $R$ ,  $j$  denote the  $j$ -th token in the  $i$ -th sentence,  $J$  is the number of tokens in the sentence, and  $p_{ij}$  be the probability of the word generated by the LLM at the  $j$ -th token of the  $i$ -th sentence. Two

probability metrics are used:

$$\text{Avg}(-\log p) = -\frac{1}{J} \sum_j \log p_{ij} \quad (1)$$

$$\text{Max}(-\log p) = \max_j (-\log p_{ij}) \quad (2)$$

$\text{Max}(-\log p)$  measures the sentence’s likelihood by assessing the *least* likely token in the sentence.

##### Entropy $\mathcal{H}$

The entropy of the output distribution is:

$$\mathcal{H}_{ij} = - \sum_{\tilde{w} \in \mathcal{W}} p_{ij}(\tilde{w}) \log p_{ij}(\tilde{w}) \quad (3)$$

where  $p_{ij}(\tilde{w})$  is the probability of the word  $\tilde{w}$  being generated at the  $j$ -th token of the  $i$ -th sentence, and  $\mathcal{W}$  is the set of all possible words in the vocabulary. Similar to the probability-based metrics, two entropy-based metrics are used:

$$\text{Avg}(\mathcal{H}) = \frac{1}{J} \sum_j \mathcal{H}_{ij} \quad (4)$$

$$\text{Max}(\mathcal{H}) = \max_j [\mathcal{H}_{ij}] \quad (5)$$

### 4 Black-Box Factuality Assessment

**Motivation.** A drawback of the previous grey-box methods is that they require output token-level probabilities. Though this may seem a reasonable requirement, for massive LLMs only available through limited API calls, such token-level information might not be available (such as with ChatGPT). Therefore, we consider black-box approaches because they remain applicable even when only text-based responses can be derived from the LLM.

##### Proxy LLMs

A simple baseline to consider is using a proxy LLM, i.e. another LLM that we have full access to such as LLaMA (Touvron et al., 2023). With no access to the full outputs of the LLM generating the text, a proxy LLM could be used to approximate the output token-level probabilities. In the next section, we propose SelfCheckGPT, which is also a black-box approach.

### 5 SelfCheckGPT

**Notation.** Let  $R$  refer to the LLM response drawn from a given user query. SelfCheckGPT operates by drawing a further  $N$  stochastic LLM response samples  $\{S^1, S^2, \dots, S^n, \dots, S^N\}$  from the

<sup>2</sup>Alternatively, white-box approaches, such as the method in Azaria and Mitchell (2023), require access to full internal states of the LLM in addition to output distributions. As a result, they are less practical and not considered in this work.

same query, followed by measuring the consistency between the response and stochastic samples. As a hallucination score of the  $i$ -th sentence, we design SelfCheckGPT  $\mathcal{S}(i)$  such that  $\mathcal{S}(i) \in [0.0, 1.0]$  and  $\mathcal{S}(i) \rightarrow 1.0$  if the  $i$ -th sentence is hallucinated, and  $\mathcal{S}(i) \rightarrow 0.0$  if it is grounded in valid information.

### 5.1 SelfCheckGPT with BERTScore

Let  $\mathcal{B}(\cdot, \cdot)$  denote the BERTScore between two sentences. SelfCheckGPT with BERTScore finds the averages BERTScore of a sentence with the most similar sentence of each drawn sample:

$$\mathcal{S}_{\text{BERT}}(i) = 1 - \frac{1}{N} \sum_{n=1}^N \max_k (\mathcal{B}(r_i, s_k^n)) \quad (6)$$

where  $r_i$  represent the  $i$ -th sentence in  $R$  and  $s_k^n$  represent the  $k$ -th sentence in the  $n$ -th sample  $S^n$ . This way if the information in a sentence appears in many drawn samples, one may assume that the information is factual, whereas if the statement appears in no other sample, it is likely a hallucination.

### 5.2 SelfCheckGPT with Question Answering

Based on the idea that information consistency could be assessed using question answering (QA), we apply the automatic multiple-choice question answering generation (MQAG) framework (Manakul et al., 2023) to SelfCheckGPT. MQAG assesses consistency by generating multiple-choice questions that an answering system can independently answer given each passage. If facts on consistent concepts are queried, the answering system is expected to predict similar answers. The MQAG framework consists of a question-answer generation system  $G_1$ , distractor generation system  $G_2$ , and answering system  $A$ . For the sentence  $r_i$  in the response  $R$ , we draw questions  $q$ , associated answers  $a$ , and distractors  $\mathbf{o}_{\setminus a}$  as follows:

$$q, a \sim P_{G_1}(q, a | r_i); \quad \mathbf{o}_{\setminus a} \sim P_{G_2}(\mathbf{o}_{\setminus a} | q, a, R) \quad (7)$$

where  $\mathbf{o} = \{a, \mathbf{o}_{\setminus a}\} = \{o_1, \dots, o_4\}$ . To filter out bad (e.g. unanswerable) questions, we define an answerability score (Raina and Gales, 2022):

$$\alpha = P_U(\text{answerable} | q, \text{context}) \quad (8)$$

where the context is either the response  $R$  or sampled passages  $S^n$ , and  $\alpha \rightarrow 0.0$  for unanswerable and  $\alpha \rightarrow 1.0$  for answerable. We use  $\alpha$  to filter out unanswerable questions which have  $\alpha$  lower than

a threshold. Subsequently, we use the answering system  $A$  to answer all answerable questions:

$$a_R = \operatorname{argmax}_k [P_A(o_k | q, R, \mathbf{o})] \quad (9)$$

$$a_{S^n} = \operatorname{argmax}_k [P_A(o_k | q, S^n, \mathbf{o})] \quad (10)$$

We compare whether  $a_R$  is equal to  $a_{S^n}$  for all samples  $\{S^1, \dots, S^N\}$ , yielding the number of matches  $N_m$  and the number of not-matches  $N_n$ . Subsequently, a simple inconsistency score for the  $i$ -th sentence and question  $q$  based on the match/not-match counts is calculated:  $\mathcal{S}_{\text{QA}}(i, q) = \frac{N_n}{N_m + N_n}$ . To take into account the number of answerable questions (i.e. the evidence to assess the sentence), we use Bayes' theorem (derivation provided in Appendix B) to improve Equation 5.2 to

$$\mathcal{S}_{\text{QA}}(i, q) = \frac{\gamma_2^{N'_m}}{\gamma_1^{N'_m} + \gamma_2^{N'_n}} \quad (11)$$

where  $N'_m$  = the effective match count,  $N'_n$  = the effective mismatch count,  $\gamma_1$ , and  $\gamma_2$  are defined in Appendix B. Ultimately, SelfCheckGPT with QA is the average of inconsistency scores across  $q$ ,

$$\mathcal{S}_{\text{QA}}(i) = \mathbb{E}_q [\mathcal{S}_{\text{QA}}(i, q)] \quad (12)$$

### 5.3 SelfCheckGPT with n-gram

Given samples  $\{S^1, S^2, \dots, S^N\}$  generated by a LLM, one could train a new language model using these samples to approximate the LLM. As  $N$  gets larger, this new language model is closer to the LLM generating the response samples. Therefore, we can approximate the LLM's token probabilities using the newly trained language model.

In practice, the number of samples  $N$  is limited due to time and/or cost constraints. Consequently, we train a simple n-gram model using the samples  $\{S^1, \dots, S^N\}$  and the main response  $R$  (which will be assessed). We note that by including  $R$  in training the n-gram model can be considered as a smoothing method where the count of each token in  $R$  is increased by 1. Then, we compute the average of log-probabilities on the response  $R$ ,

$$\mathcal{S}_{\text{n-gram}}^{\text{Avg}}(i) = -\frac{1}{J} \sum_j \log \tilde{p}_{ij} \quad (13)$$

where  $\tilde{p}_{ij}$  is the probability (of the  $j$ -th token of the  $i$ -th sentence) computed using the n-gram model. Alternatively, we can also use the maximum of negative log probabilities of the n-gram model,

$$\mathcal{S}_{\text{n-gram}}^{\text{Max}}(i) = \max_j (-\log \tilde{p}_{ij}) \quad (14)$$

## 5.4 SelfCheckGPT Combination

Lastly, given the differences in the natures of the variants of SelfCheckGPT, we expect them to be complementary. As a result, we consider SelfCheckGPT-Combination, which is a simple combination of the normalized scores of the three variants, including  $\mathcal{S}_{\text{BERT}}$ ,  $\mathcal{S}_{\text{QA}}$ , and  $\mathcal{S}_{\text{n-gram}}$ .

## 6 Data and Annotation

We evaluate hallucination detection approaches by 1) generating synthetic Wikipedia articles using GPT-3 on the individuals from the Wikibio dataset (Lebret et al., 2016); 2) manually annotating the factuality of the passage at a sentence level; 3) evaluating the system’s ability to detect hallucinations.

WikiBio is a dataset of the first paragraph (along with tabular information) of Wikipedia biographies. We rank the WikiBio test set in terms of paragraph length and randomly sample 238 articles from the top 20% of longest articles (to ensure no obscure concept is selected). GPT-3 (text-davinci-003) is used to generate Wikipedia articles on a concept using the prompt "This is a Wikipedia passage about {concept}". Table 1 provides the statistics of GPT-3 generated passages.

#Passages	#Sentences	#Tokens/passage
238	1908	184.7±36.9

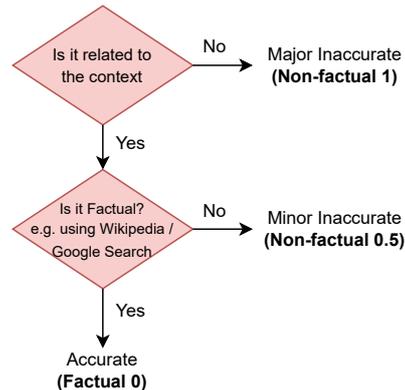
**Table 1:** The statistics of **WikiBio GPT-3 dataset** where the number of tokens is based on the OpenAI GPT-2 tokenizer.

We then annotate the sentences of the generated passages using the guidelines shown in Figure 3 such that each sentence classified as:

- **Major Inaccurate** (Non-Factual, **1**): The sentence is entirely hallucinated, i.e. the sentence is unrelated to the topic.
- **Minor Inaccurate** (Non-Factual, **0.5**): The sentence consists of some non-factual information, but the sentence is related to the topic.
- **Accurate** (Factual, **0**): The information presented in the sentence is accurate.

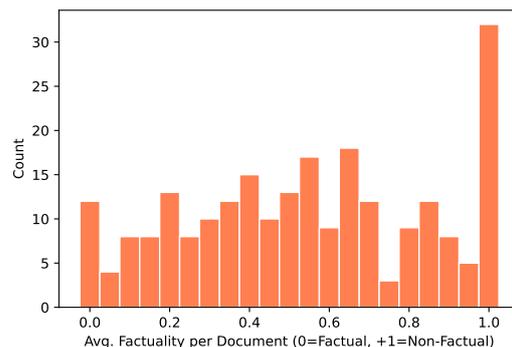
Of the 1908 annotated sentences, 761 (39.9%) of the sentences were labelled major-inaccurate, 631 (33.1%) were minor-inaccurate, and 516 (27.0%) were accurate.<sup>3</sup> Passage-level scores are obtained

<sup>3</sup>When selecting more obscure or more well-known concepts/individuals, the label distribution can be shifted to contain more or fewer hallucinations.



**Figure 3:** Flowchart of our annotation process

by averaging the sentence-level labels in each passage. The distribution of passage-level scores is shown in Figure 4, where we observe a large peak at +1.0. We refer to the points at this peak as *total hallucination*, i.e. the individual/concept was entirely made up and is unrelated to the real concept.

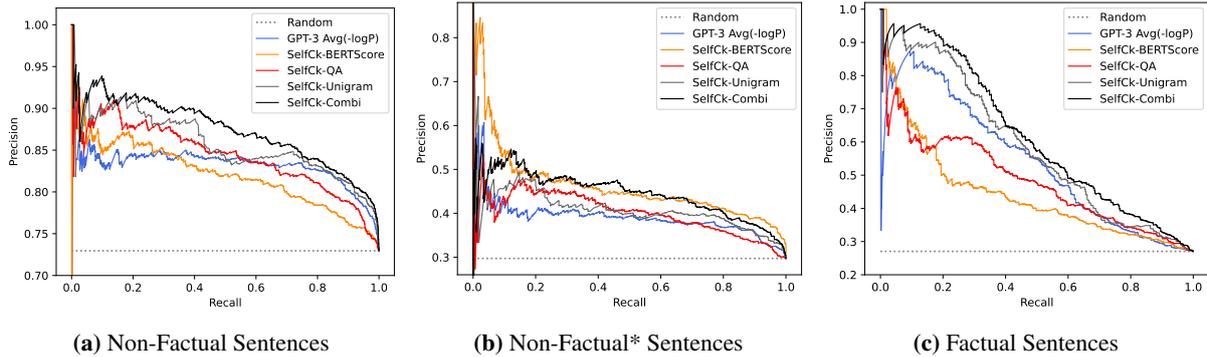


**Figure 4:** Document factuality scores histogram plot

A subset of the dataset consisting of 201 sentences was annotated by two annotators. To obtain a single label for this subset, if both annotators agree, we use the agreed label. However, if they disagree, we use the worse-case label, e.g. {minor inaccurate, major inaccurate} is mapped to major inaccurate. We report inter-annotator agreement as measured by Cohen’s  $\kappa$  (Cohen, 1960) in Table 2. Cohen’s  $\kappa$  values of 0.595 and 0.748 indicate *moderate* and *substantial* agreement (Viera et al., 2005) for the 3-label and 2-class scenarios, respectively.

Annotation	3-label	2-label
Cohen’s $\kappa$	0.595	0.748

**Table 2:** Inter-annotator agreement where 3-label means selecting from accurate, minor inaccurate, major inaccurate. 2-label is calculated by combining minor/major into one label.



**Figure 5:** PR-Curve of detecting non-factual and factual *sentences* in the GPT-3 generated WikiBio passages.

## 7 Experiments

The main generative LLM is **GPT-3** (text-davinci-003), which is the state-of-the-art system at the time of conducting our experiments. To obtain the main response, we set the generation temperature to 0.0 and use beam search decoding. For the stochastically generated samples, we set the temperature to 1.0 and generate  $N=20$  samples.

For the proxy LLM approach, the main text shows the results on LLaMA, which is one of the best-performing open-source LLMs. The results on other proxy LLMs can be found in the appendix. Also, the details about QG and QA systems are described in the appendix.

### 7.1 Sentence-level Hallucination Detection

First, we investigate whether our hallucination detection methods are capable of identifying the factuality of sentences. In detecting non-fact sentences, both major-inaccurate labels and minor-inaccurate labels are grouped together into the *non-factual* class, while the *factual* class refers to accurate sentences. In addition, we consider a more challenging task of detecting major-inaccurate sentences in passages that are *not* total hallucination passages, which we refer to as *non-factual\**.<sup>4</sup> Figure 5 and Table 3 show the performance of our approaches, where the following observations can be made:

**1) LLM’s probabilities  $p$  correlate well with factuality.** Our results show that probability measures (from the LLM generating the texts) are strong baselines for assessing factuality. Factual sentences can be identified with an AUC-PR of 53.97, significantly better than the random baseline of 27.04, with the AUC-PR for hallucination detection also increasing from 72.96 to 83.21. This supports the hypothesis that when the LLMs are uncer-

<sup>4</sup>In non-factual\*, 206 passages (1632 sentences) remain.

tain about generated information, generated tokens often have higher uncertainty, paving a promising direction for hallucination detection approaches. Also, the probability  $p$  measure performs better than the entropy  $\mathcal{H}$  measure of top-5 tokens.

**2) Proxy LLM perform noticeably worse than LLM (GPT-3).** Nevertheless, as opposed to the LLM, the results of proxy LLM show that the entropy  $\mathcal{H}$  measures outperform the probability measures. This suggests that using richer uncertainty information could improve factuality/hallucination detection tasks, while the entropy of top-5 tokens is likely insufficient. In addition, when using other proxy LLMs such as GPT-NeoX or OPT-30B, the performance is worse than or marginally better than the random baseline. We believe this poor performance occurs as different LLMs have different generating patterns, and therefore even common uninformative tokens may have a low probability if they do not follow the style of the proxy LLM. We note that a weighted conditional language model score such as BARTScore (Yuan et al., 2021) could be incorporated in future investigations of the proxy LLM approach.

**3) SelfCheckGPT rivals grey-box approaches.** SelfCheckGPT considerably outperforms the proxy LLM approach in all detection setups. Furthermore, SelfCheckGPT outperforms the grey-box probability-based approach in most setups. We also observe a performance gain when combining the variants of SelfCheckGPT.

Interestingly, despite being the simplest method, SelfCheckGPT with unigram (max) works well across different setups. Essentially, when assessing a sentence, this method picks up the token with the *lowest* occurrence given all the samples. For instance, if this token only appears a few times (or once) in the samples ( $N=20$ ), it is likely non-

Method	Sentence-level (AUC-PR)			Passage-level (Corr.)	
	NonFact	NonFact*	Factual	Pearson	Spearman
Random	72.96	29.72	27.04	-	-
GPT-3’s probabilities ( <i>LLM, grey-box</i> )					
Avg( $-\log p$ )	83.21	38.89	53.97	57.04	53.93
Avg( $\mathcal{H}$ ) <sup>†</sup>	80.73	37.09	52.07	55.52	50.87
Max( $-\log p$ )	87.51	35.88	50.46	57.83	55.69
Max( $\mathcal{H}$ ) <sup>†</sup>	85.75	32.43	50.27	52.48	49.55
LLaMA-30B’s probabilities ( <i>Proxy LLM, black-box</i> )					
Avg( $-\log p$ )	75.43	30.32	41.29	21.72	20.20
Avg( $\mathcal{H}$ )	80.80	39.01	42.97	33.80	39.49
Max( $-\log p$ )	74.01	27.14	31.08	-22.83	-22.71
Max( $\mathcal{H}$ )	80.92	37.32	37.90	35.57	38.94
<b>SelfCheckGPT</b> ( <i>black-box</i> )					
w/ BERTScore	81.96	45.96	44.23	58.18	55.90
w/ QA	84.26	40.06	48.14	61.07	59.29
w/ Unigram (max)	85.63	41.04	58.47	64.71	64.91
Combination	87.33	44.37	61.83	69.05	67.77

**Table 3:** AUC-PR for sentence-level detection tasks. Passage-level ranking performances are measured by Pearson correlation coefficient and Spearman’s rank correlation coefficient w.r.t. human judgements. The results of other proxy LLMs, in addition to LLaMA, can be found in the appendix. <sup>†</sup>GPT-3 API returns the top-5 tokens’ probabilities, which are used to compute entropy.

factual. Next, we investigate its performance as we vary from 1-gram to 5-gram. The results in Table 6 show that simply finding the least likely token/n-gram is more effective than computing the average n-gram language model score of the sentence (i.e. Avg( $-\log p$ )). As  $n$  increases the performance of SelfCheckGPT with n-gram (max) drops, because the space of n-grams increases exponentially with  $n$ ; hence, requiring exponentially more samples.

## 7.2 Passage-level Factuality Ranking

The previous results show that SelfCheckGPT is an effective approach for predicting sentence-level factuality. An additional consideration, though, is whether SelfCheckGPT can be used to determine the overall factuality of passages. Passage-level factuality scores are obtained by averaging the sentence-level scores over all sentences.

$$f_{\text{passage}}(i) = \frac{1}{|R|} \sum_i f(i) \quad (15)$$

where  $f(i)$  is the sentence-level score, and  $|R|$  is the number of sentences in the passage. Note that for Avg( $-\log p$ ) and Avg( $\mathcal{H}$ ), we compute the average over all tokens in a passage. Whereas for Max( $-\log p$ ) and Max( $\mathcal{H}$ ), we first take the maximum operation over tokens at the sentence level, and we then average over all sentences following

Equation 15. Since human judgement is somewhat subjective, averaging the sentence-level labels would lead to ground truths with less noise.

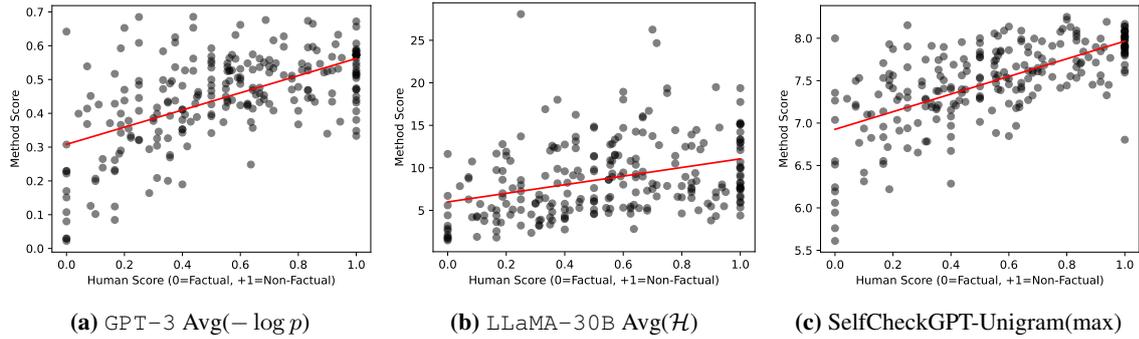
Our results in Table 3 and Figure 6 show that all of the SelfCheckGPT methods correlate far better with human judgements than all other methods, including the grey-box probability and entropy methods. Further, the three variants of SelfCheckGPT appear complementary, with the combined approach being the best-performing system, achieving the highest Pearson correlation of 69.05. Unsurprisingly, the proxy LLM approach again achieves considerably lower correlations.

## 7.3 Ablation Studies

### External Knowledge (instead of SelfCheck)

If external knowledge is available, one could measure the informational consistency with the LLM response and the real-world document (instead of LLM self-samples). In this experiment, we have the related WikiBio passage and so can extract the first Wikipedia paragraph for each concept/individual.<sup>5</sup> Our results in Table 4 show that for the QA and BERTScore methods, using SelfCheckGPT samples can yield a comparable or even

<sup>5</sup>This method is no longer zero-resource as it requires retrieving relevant knowledge, and so this approach may only be applicable to fact verification.



**Figure 6:** Scatter plot of passage-level scores where Y-axis = Method scores, X-axis = Human scores. The scatter plots of other SelfCheckGPT variants are provided in Figure 10 in the appendix.

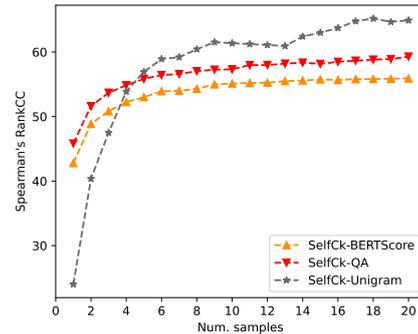
better performance compared to using the WikiBio reference passage. This illustrates that SelfCheckGPT is a strong hallucination detection approach that is comparable to methods using stored external information. Lastly, the n-gram model shows a significant drop in performance when using the WikiBio passages instead of LLM self-samples. This failure is attributed to the fact that just the WikiBio reference text is not sufficient to train an n-gram model, and we investigate the number of samples in more detail in the next ablation.

Method	Sent-lvl AUC-PR			Passage-lvl	
	NoFac	NoFac*	Fact	Pear.	Spear.
SelfCk-BERT	81.96	45.96	44.23	58.18	55.90
WikiBio+BERT	81.32	40.62	49.15	58.71	55.80
SelfCk-QA	84.26	40.06	48.14	61.07	59.29
WikiBio+QA	84.18	45.40	52.03	57.26	53.62
SelfCk-1gm	85.63	41.04	58.47	64.71	64.91
WikiBio+1gm	80.43	31.47	40.53	28.67	26.70

**Table 4:** The performance when using SelfCheckGPT samples versus external stored knowledge.

### Impact of the Number of Samples

Typically, sampling-based methods are expected to achieve better performance when more samples are drawn. However, drawing a higher number of samples leads to higher computational costs and/or API costs. Thus, we investigate the behaviour of SelfCheckGPT as we vary the number of samples drawn from 1 to 20. Our results in Figure 7 (and Figure 8 in the appendix) show that the performance of SelfCheckGPT increases as more samples are used, with the performance gain diminishing as we generate more samples. SelfCheckGPT with n-gram requires the highest number of samples before its performance reaches a plateau.



**Figure 7:** The performance of SelfCheckGPT methods on ranking passages (Spearman’s) versus the number of samples.

### Model Choice for Proxy LLM

Figure 9 (in Appendix C) illustrates that LLaMA is far better than other LLMs, and the performance of the proxy LLM method increases with model size. Similarly, average probability,  $\text{Avg}(p)$ , is closer to that of GPT-3 when using a larger proxy LLM as shown in Table 7 in the appendix.

## 8 Conclusions

This work proposes SelfCheckGPT, a zero-resource black-box approach, that can be used to detect LLM hallucinations and determine the trustworthiness of generated responses without the need for any external resources. SelfCheckGPT has been shown to be an effective approach for LLM hallucination assessment at both sentence and passage levels, and the approach is applicable to any LLM and any topic that the LLM is prompted to generate. Through experimental analysis of annotated GPT-3 responses, we show that SelfCheckGPT has a competitive performance to the grey-box probability-based approach, and it also significantly outperforms the proxy LLM approach. In addition, this work releases a dataset for GPT-3 hallucination detection, consisting of 238 annotated passages.

## 9 Limitations

In this study, the scope of GPT-3 generated texts is 238 passages about individuals in the WikiBio dataset, as a result, a wider range of concepts, e.g. locations and objects, could be investigated to better understand the nature of LLM’s hallucination.

## References

- Amos Azaria and Tom Mitchell. 2023. The internal state of an llm knows when its lying. *arXiv preprint arXiv:2304.13734*.
- Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. *Longformer: The long-document transformer*.
- Sidney Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, Usvsn Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. 2022. *GPT-NeoX-20B: An open-source autoregressive language model*. In *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*, pages 95–136, virtual+Dublin. Association for Computational Linguistics.
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.
- Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.
- Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20:37 – 46.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. *BERT: Pre-training of deep bidirectional transformers for language understanding*. 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 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. 2023. *Gptscore: Evaluate as you desire*.
- Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. 2022. *A survey on automated fact-checking*. *Transactions of the Association for Computational Linguistics*, 10:178–206.
- Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. 2022. *Training compute-optimal large language models*.
- Yichong Huang, Xiachong Feng, Xiaocheng Feng, and Bing Qin. 2021. *The factual inconsistency problem in abstractive text summarization: A survey*.
- Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. *What does BERT learn about the structure of language?* In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.
- Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. *Survey of hallucination in natural language generation*. *ACM Comput. Surv.*, 55(12).
- Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield Dodds, Nova DasSarma, Eli Tran-Johnson, et al. 2022. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*.
- Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. *Neural text summarization: A critical evaluation*. 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 540–551, Hong Kong, China. Association for Computational Linguistics.
- Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. *Evaluating the factual consistency of abstractive text summarization*. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, Online. Association for Computational Linguistics.
- Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. 2023. *Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation*. In *The Eleventh International Conference on Learning Representations*.
- Rémi Lebret, David Grangier, and Michael Auli. 2016. *Generating text from structured data with application to the biography domain*. *CoRR*, abs/1603.07771.
- 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. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6723–6737, Dublin, Ireland. Association for Computational Linguistics.
- Adian Liusie, Vatsal Raina, and Mark Gales. 2022. "world knowledge" in multiple choice reading comprehension. *arXiv preprint arXiv:2211.07040*.
- Andrey Malinin and Mark Gales. 2021. **Uncertainty estimation in autoregressive structured prediction**. In *International Conference on Learning Representations*.
- Potsawee Manakul, Adian Liusie, and Mark JF Gales. 2023. MQAG: Multiple-choice question answering and generation for assessing information consistency in summarization. *arXiv preprint arXiv:2301.12307*.
- Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. **On faithfulness and factuality in abstractive summarization**. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.
- Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.
- Vatsal Raina and Mark Gales. 2022. **Answer uncertainty and unanswerability in multiple-choice machine reading comprehension**. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1020–1034, Dublin, Ireland. Association for Computational Linguistics.
- Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. **Retrieval augmentation reduces hallucination in conversation**. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803, Punta Cana, Dominican Republic. Association for Computational Linguistics.
- James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal. 2018. The Fact Extraction and VERification (FEVER) shared task. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*.
- Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.
- Anthony J Viera, Joanne M Garrett, et al. 2005. Understanding interobserver agreement: the kappa statistic. *Fam med*, 37(5):360–363.
- Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. **GLUE: A multi-task benchmark and analysis platform for natural language understanding**. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.
- Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. <https://github.com/kingoflolz/mesh-transformer-jax>.
- Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022. Emergent abilities of large language models. *Transactions on Machine Learning Research*.
- Yijun Xiao and William Yang Wang. 2021. **On hallucination and predictive uncertainty in conditional language generation**. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2734–2744, Online. Association for Computational Linguistics.
- Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34:27263–27277.
- Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.
- Zhuosheng Zhang, Yuwei Wu, Hai Zhao, Zuchao Li, Shuailiang Zhang, Xi Zhou, and Xiang Zhou. 2020. Semantics-aware bert for language understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9628–9635.
- Wanjun Zhong, Jingjing Xu, Duyu Tang, Zenan Xu, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. 2020. **Reasoning over semantic-level graph for fact checking**. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6170–6180, Online. Association for Computational Linguistics.

## A Models and Implementation

### Entropy

The entropy of the output distribution is implemented as  $2^{-\sum_{\tilde{w} \in \mathcal{W}} p_{ij}(\tilde{w}) \log_2 p_{ij}(\tilde{w})}$  where  $\mathcal{W}$  is the set of all possible words in the vocabulary.

### Proxy LLMs

The proxy LLMs considered are LLaMA- $\{7B, 13B, 30B\}$  (Touvron et al., 2023), OPT- $\{125m, 1.3B, 13B, 30B\}$  (Zhang et al., 2022), GPT-J-6B (Wang and Komatsuzaki, 2021) and GPT-NeoX-20B (Black et al., 2022).

### SelfCheckGPT’s Systems

Our BERTScore has the `roberta-large` model as the backbone. For the QA method, the generation systems G1 and G2 (described in Section 5.2) are T5-large finetuned to SQuAD and RACE, respectively. The answering system A is Longformer (Beltagy et al., 2020) finetuned to the RACE dataset. The answerability system U is also Longformer, but fine-tuned to SQuAD2.0. These models have been made available on HuggingFace.

## B SelfCheckGPT-QA with Bayes

### Theory

Let  $P(F)$  denote the probability of the  $i$ -th sentence being non-factual, and  $P(T)$  denote the probability of the  $i$ -th sentence being factual. For a question  $q$ , the probability of  $i$ -th sentence being non-factual given a set of matched answers  $L_m$  and a set of not-matched answers  $L_n$  is:

$$\begin{aligned} P(F|L_m, L_n) &= \frac{P(L_m, L_n|F)P(F)}{P(L_m, L_n|F)P(F) + P(L_m, L_n|T)P(T)} \\ &= \frac{P(L_m, L_n|F)}{P(L_m, L_n|F) + P(L_m, L_n|T)} \end{aligned} \quad (16)$$

where we assume the sentence is equally likely to be False or True, i.e.  $P(F) = P(T)$ . The probability of observing  $L_m, L_n$  when the sentence is False (non-factual):

$$\begin{aligned} P(L_m, L_n|F) &= \prod_{a \in L_m} P(a = a_R|F) \prod_{a' \in L_n} P(a' \neq a_R|F) \\ &= (1 - \beta_1)^{N_m} (\beta_1)^{N_n} \end{aligned} \quad (17)$$

and probability of observing  $L_m, L_n$  when the sentence is True (factual):

$$\begin{aligned} P(L_m, L_n|T) &= \prod_{a \in L_m} P(a = a_r|T) \prod_{a' \in L_n} P(a' \neq a_r|T) \\ &= (\beta_2)^{N_m} (1 - \beta_2)^{N_n} \end{aligned} \quad (18)$$

where  $N_m$  and  $N_n$  are the number of matched answers and the number of not-matched answers, respectively. Hence, we can simplify Equation 16:

$$P(F|L_m, L_n) = \frac{\gamma_2^{N_n}}{\gamma_1^{N_m} + \gamma_2^{N_n}} \quad (19)$$

where  $\gamma_1 = \frac{\beta_2}{1-\beta_1}$  and  $\gamma_2 = \frac{\beta_1}{1-\beta_2}$ . Lastly, instead of rejecting samples having an answerability score below a threshold,<sup>6</sup> we find empirically that soft-counting (defined below) improves the detection performance. We set both  $\beta_1$  and  $\beta_2$  to 0.8.

$$N'_m = \sum_{n \text{ s.t. } a_n \in L_m} \alpha_n; \quad N'_n = \sum_{n \text{ s.t. } a_n \in L_n} \alpha_n \quad (20)$$

where  $\alpha_n = P_U(\text{answerable}|q, S^n)$ . Therefore, the SelfCheckGPT with QA score,  $\mathcal{S}_{QA}$ , is:

$$\mathcal{S}_{QA} = P(F|L_m, L_n) = \frac{\gamma_2^{N'_n}}{\gamma_1^{N'_m} + \gamma_2^{N'_n}} \quad (21)$$

## Results

In Table 5, we show empirically that using applying Bayes’ theorem and soft counting  $\alpha$  (in Equation 20) improves the performance of the SelfCheckGPT with QA approach.

Variant	Sentence-lvl		Passage-lvl		
	NoF	NoF*	Fact	PCC	SCC
SimpleCount	83.97	40.07	47.78	57.39	55.15
+ Bayes	83.04	38.58	47.41	56.43	55.03
+ Bayes + $\alpha$	84.26	40.06	48.14	61.07	59.29

Table 5: Performance of SelfCheckGPT-QA’s variants.

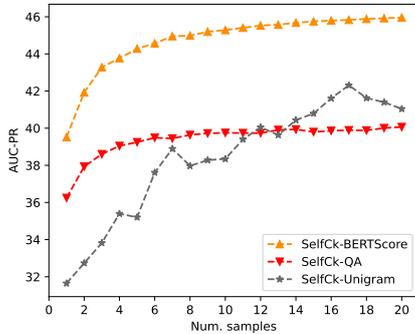
## C Additional Results

Here, we provide experimental results that are complementary to those presented in the main paper.

<sup>6</sup> $\alpha$  is between 0.0 (unanswerable) and 1.0 (answerable). Standard-counting  $N_m$  and  $N_n$  can be considered as a special case of soft-counting where  $\alpha$  is set to 1.0 if  $\alpha$  is greater than the answerability threshold and otherwise  $\alpha$  is 0.0.

n-gram	Sent-lvl AUC-PR			Passage-lvl	
	NoFac	NoFac*	Fact	Pear.	Spear.
<b>Avg(<math>-\log p</math>)</b>					
1-gram	81.52	40.33	41.76	40.68	39.22
2-gram	82.94	44.38	52.81	58.84	58.11
3-gram	83.56	44.64	53.99	62.21	63.00
4-gram	83.80	43.55	54.25	61.98	63.64
5-gram	83.45	42.31	53.98	60.68	62.96
<b>Max(<math>-\log p</math>)</b>					
1-gram	85.63	41.04	58.47	64.71	64.91
2-gram	85.26	39.29	58.29	62.48	66.04
3-gram	84.97	37.10	57.08	57.34	60.49
4-gram	84.49	36.37	55.96	55.77	57.25
5-gram	84.12	36.19	54.89	54.84	55.97

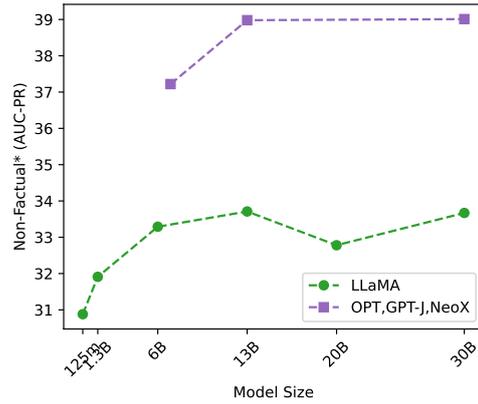
**Table 6:** The performance using different n-gram models.



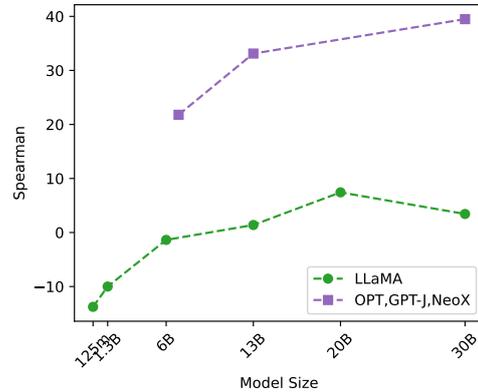
**Figure 8:** The performance of SelfCheckGPT methods on sentence-level non-factual\* detection (AUC-PR) versus the number of samples. This Figure extends the passage-level results in Figure 7.

LLM	Size	Avg( $p$ )
GPT-3	175B	72.02
LLaMA	30B	65.25
LLaMA	13B	64.71
LLaMA	7B	63.70
OPT	30B	53.81
NeoX	20B	54.49
OPT	13B	52.80
GPT-J	6B	52.50
OPT	1.3B	49.20
OPT	125m	40.91

**Table 7:** Average token probability, Avg( $p$ ), over all tokens in GPT-3 generated passages.

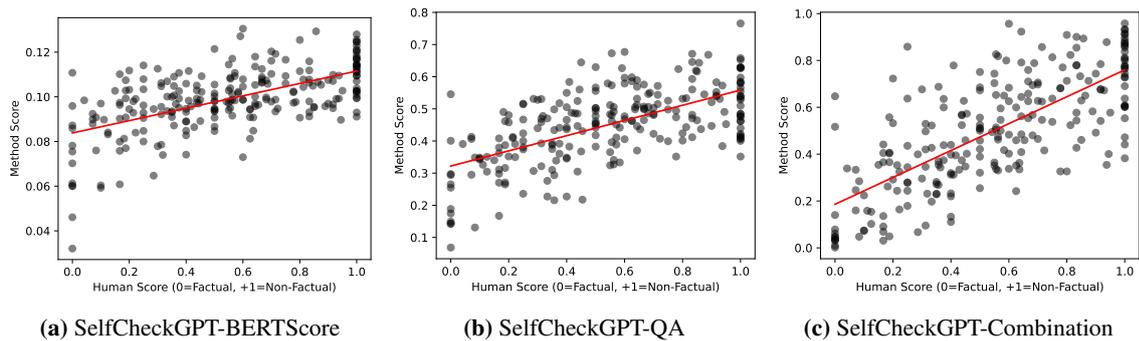


**(a) Sentence-level Detection**



**(b) Passage-level Ranking**

**Figure 9:** Performance of the Avg( $\mathcal{H}$ ) method using a proxy LLM where the model sizes are: LLaMA={7B, 13B, 30B}, OPT={125m, 1.3B, 13B, 30B}, GPT-J=6B, and NeoX=20B. The full results are provided in Table 8.



**Figure 10:** Scatter plot of passage-level scores where Y-axis = Method scores, X-axis = Human scores. This figure provides results in addition to Figure 6.

LLM	Size	Sentence-level (AUC-PR)			Passage-level (Corr.)	
		NonFact	NonFact*	Factual	Pearson	Spearman
Random	-	72.96	29.72	27.04	-	-
<b>Avg(-logp) Method</b>						
LLaMA	30B	75.43	30.32	41.29	21.72	20.20
LLaMA	13B	74.16	30.01	37.36	13.33	12.89
LLaMA	7B	71.69	27.87	31.30	-2.71	-2.59
OPT	30B	67.70	24.43	25.04	-32.07	-31.45
NeoX	20B	69.00	24.38	26.18	-31.79	-34.15
OPT	13B	67.46	24.39	25.20	-33.05	-32.79
GPT-J	6B	67.51	24.28	24.26	-38.80	-40.05
OPT	1.3B	66.19	24.47	23.47	-35.20	-38.95
OPT	125m	66.63	25.31	23.07	-30.38	-37.54
<b>Avg(H) Method</b>						
LLaMA	30B	80.80	39.01	42.97	33.80	39.49
LLaMA	13B	80.63	38.98	40.59	29.43	33.12
LLaMA	7B	78.67	37.22	33.81	19.44	21.79
OPT	30B	77.13	33.67	29.55	-0.43	3.43
NeoX	20B	77.40	32.78	30.13	5.41	7.43
OPT	13B	76.93	33.71	29.68	0.25	1.39
GPT-J	6B	76.15	33.29	28.30	-2.50	-1.37
OPT	1.3B	74.05	31.91	26.33	-10.59	-10.00
OPT	125m	71.51	30.88	25.36	-14.16	-13.76
<b>Max(-logp) Method</b>						
LLaMA	30B	74.01	27.14	31.08	-22.83	-22.71
LLaMA	13B	71.12	26.78	28.82	-34.93	-31.70
LLaMA	7B	69.57	25.91	26.54	-42.57	-38.24
OPT	30B	67.32	24.40	24.32	-49.51	-45.50
NeoX	20B	67.51	23.88	24.82	-47.96	-44.54
OPT	13B	67.36	24.67	24.46	-50.15	-44.42
GPT-J	6B	67.58	23.94	23.93	-51.23	-47.68
OPT	1.3B	68.16	25.85	24.66	-45.60	-42.39
OPT	125m	69.23	27.66	24.14	-39.22	-37.18
<b>Max(H) Method</b>						
LLaMA	30B	80.92	37.32	37.90	35.57	38.94
LLaMA	13B	80.98	37.94	36.01	32.07	34.01
LLaMA	7B	79.65	35.57	31.32	22.10	22.53
OPT	30B	76.58	33.44	29.31	1.63	6.41
NeoX	20B	76.98	31.96	29.13	5.97	9.31
OPT	13B	76.26	32.81	29.25	1.42	2.82
GPT-J	6B	75.30	32.51	28.13	-2.14	1.41
OPT	1.3B	73.79	31.42	26.38	-9.84	-9.80
OPT	125m	71.32	31.65	25.36	-18.05	-17.37

**Table 8:** AUC-PR for Detecting Non-Factual and Factual Sentences in the GPT-3 generated WikiBio passages. Passage-level PCC and SCC with LLMs used to assess GPT-3 responses. This table is an extension to Table 3.