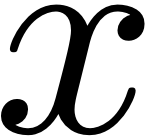
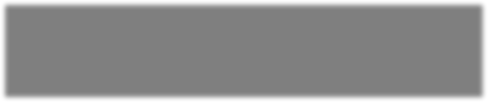
**DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature**

## Eric Mitchell 1 Yoonho Lee 1 Alexander Khazatsky 1 Christopher D. Manning 1 Chelsea Finn 1

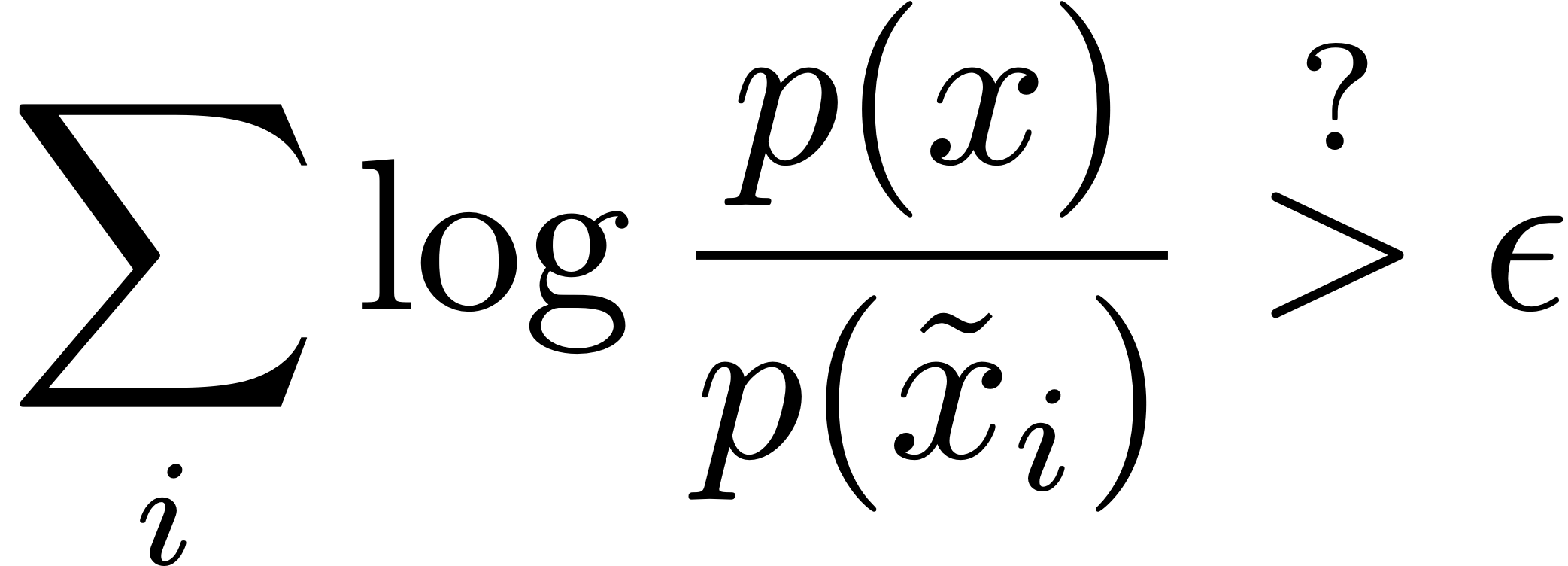
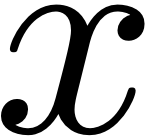
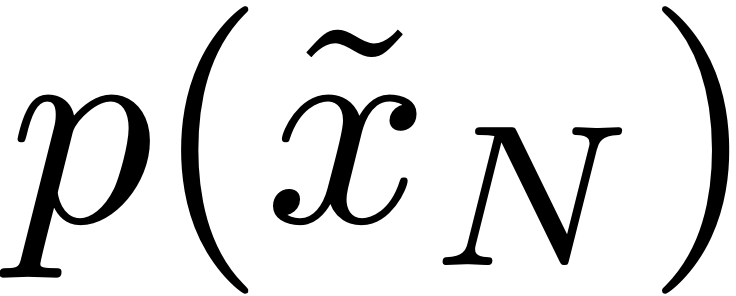
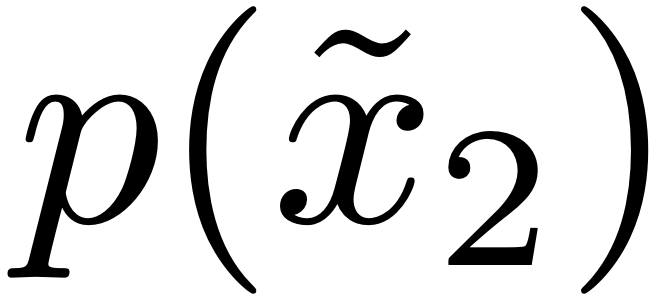
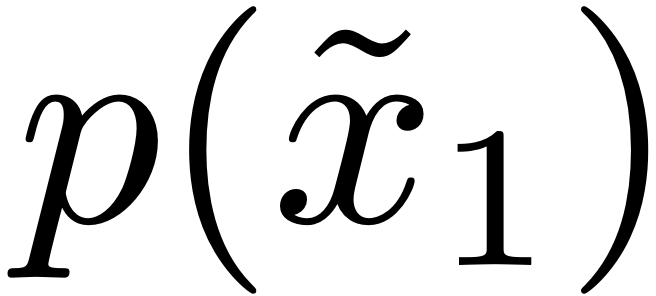
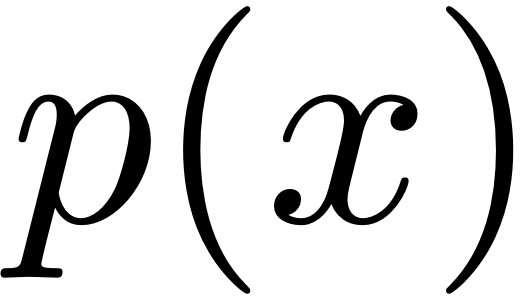
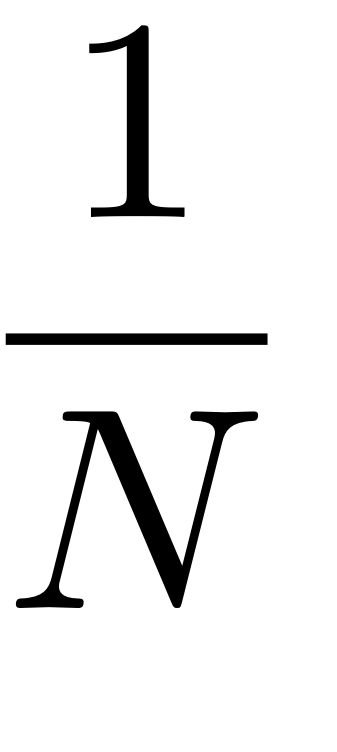
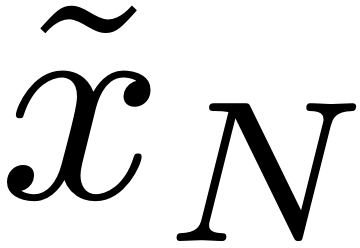
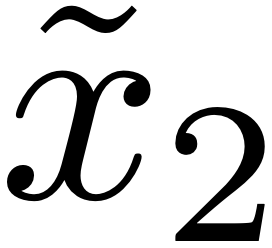
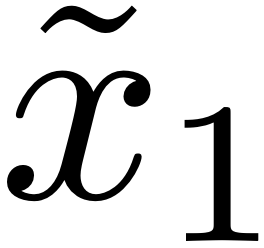
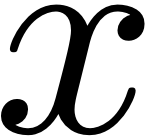
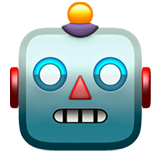
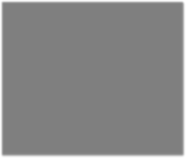
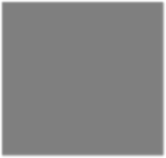
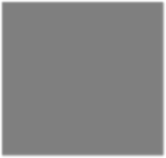
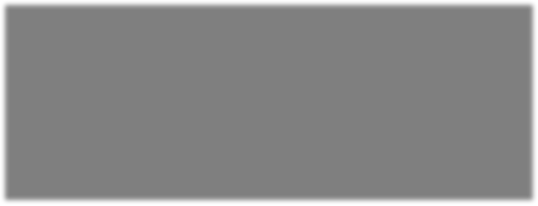
**Abstract**



*Candidate passage* ***:***

*“Joe Biden recently made a move to the White House that included bringing along his pet German Shepherd…”*

The fluency and factual knowledge of large lan- guage models (LLMs) heightens the need for cor- responding systems to detect whether a piece of text is machine-written. For example, students may use LLMs to complete written assignments, leaving instructors unable to accurately assess stu- dent learning. In this paper, we first demonstrate that text sampled from an LLM tends to occupy negative curvature regions of the model’s log prob- ability function. Leveraging this observation, we then define a new curvature-based criterion for judging if a passage is generated from a given LLM. This approach, which we call DetectGPT, does not require training a separate classifier, col- lecting a dataset of real or generated passages, or



*(2) Score*

***Yes***

***No***

🤖 *from* ***GPT-3***

🤔 *from* ***other source***

GPT-3

*(3) Compare*

*x*

***with T5)***

*(1) Perturb* ***(reword***

**DetectGPT**

. . .

. . .

explicitly watermarking generated text. It uses only log probabilities computed by the model of interest and random perturbations of the passage from another generic pre-trained language model (e.g, T5). We find DetectGPT is more discrimi- native than existing zero-shot methods for model sample detection, notably improving detection of fake news articles generated by 20B parameter GPT-NeoX from 0.81 AUROC for the strongest zero-shot baseline to 0.95 AUROC for Detect- GPT. See [ericmitchell.ai/detectgpt](https://ericmitchell.ai/detectgpt) for code, data, and other project information.

arXiv:2301.11305v1

[cs.CL]

26 Jan 2023

# Introduction

Large language models (LLMs) have proven able to gen- erate remarkably fluent responses to a wide variety of user queries. Models such as GPT-3 ([Brown et al.](#_bookmark34),[2020](#_bookmark34)), PaLM ([Chowdhery et al.](#_bookmark38),[2022](#_bookmark38)), and ChatGPT ([OpenAI,](#_bookmark43)2022[)](#_bookmark43) can convincingly answer complex questions about science, mathematics, historical and current events, and social trends. While recent work has found that cogent-sounding LLM- generated responses are often simply wrong ([Lin et al.](#_bookmark54),

1Stanford University. Correspondence to: Eric Mitchell

*<*[eric.mitchell@cs.stanford.edu](mailto:eric.mitchell@cs.stanford.edu)*>*.

*Figure 1.* We aim to determine whether a piece of text was gener-

ated by a particular LLM *p*, such as GPT-3. To classify a candidate passage *x*, DetectGPT first generates minor **perturbations** of the passage *x*˜*i* using a generic pre-trained model such as T5. Then DetectGPT **compares** the log probability under *p* of the original sample *x* with each perturbed sample *x*˜*i*. If the average log ratio is high, the sample is likely from the source model.

[2022](#_bookmark54)), the articulate nature of such generated text may still make LLMs attractive for replacing human labor in some contexts, notably student essay writing and journalism. At least one major news source has released AI-written content with limited human review, leading to substantial factual er- rors in some articles ([Christian](#_bookmark28),[2023](#_bookmark28)). Such applications of LLMs are problematic for a variety of reasons, making fair student assessment difficult, impairing student learning, and proliferating convincing-but-inaccurate news articles. Un- fortunately, humans perform only slightly better than chance when classifying machine-generated vs human-written text ([Gehrmann et al.](#_bookmark37),[2019](#_bookmark37)), leading researchers to consider automated detection methods that may identify signals dif- ficult for humans to recognize. Such methods might give teachers and news-readers more confidence in the human origin of the text that they consume.

As in prior work ([Jawahar et al.](#_bookmark48),[2020),](#_bookmark48) we study the machine-generated text detection problem as a binary clas- sification problem. Specifically, we aim to classify whether

a *candidate passage* was generated by a particular *source model*. While several works have investigated methods for training a second deep network to detect machine-generated text, such an approach has several shortcomings, including a tendency to overfit to the topics it was trained on as well as the need to train a new model for each new source model that is released. We therefore consider the *zero-shot* version of machine-generated text detection, where we use the source model itself, without fine-tuning or adaptation of any kind, to detect its own samples. The most common method for zero-shot machine-generated text detection is evaluating the average per-token log probability of the generated text and thresholding ([Solaiman et al.](#_bookmark52),[2019;](#_bookmark52)[Gehrmann et al.,](#_bookmark37)2019[;](#_bookmark37) [Ippolito et al.](#_bookmark46),[2020](#_bookmark46)). However, this zeroth-order approach to detection ignores the local structure of the learned proba- bility function around a candidate passage, which we find contains useful information about the source of a passage.

log *pθ*(*x*)

This paper poses a simple hypothesis: minor rewrites of *model-generated* text tend to have lower log probability un- der the model than the original sample, while minor rewrites of *human-written* text may have higher or lower log prob- ability than the original sample. In other words, unlike human-written text, model-generated text tends to lie in ar- eas where the log probability function has negative curvature (e.g., local maxima of the log probability). We empirically verify this hypothesis, and find that it holds true across a diverse body of LLMs, even when the minor rewrites, or *perturbations*, come from alternative language models. We leverage this observation to build DetectGPT, a zero-shot method for automated machine-generated text detection. To test if a passage came from a source model *pθ*, DetectGPT compares the log probability of a candidate passage under *pθ* with the average log probability of several perturbations of the passage under *pθ* (generated with, e.g., T5;[Raffel](#_bookmark47) [et al.](#_bookmark47)([2020](#_bookmark47))). If the perturbed passages tend to have lower average log probability than the original by some margin, the candidate passage is likely to have come from *pθ*. See Figure1for an overview of the problem and DetectGPT. See Figure2for an illustration of the underlying hypothesis and Figure3for empirical evaluation of the hypothesis. Our experiments find that DetectGPT is more accurate than exist- ing zero-shot methods for detecting machine-generated text, improving over the strongest zero-shot baseline by over

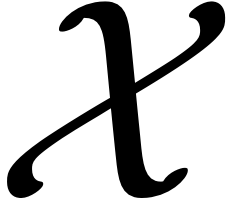
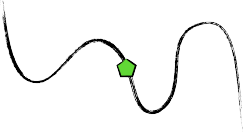
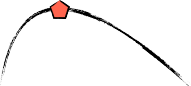
0.1 AUROC for multiple source models when detecting machine-generated news articles.

**Contributions.** Our main contributions are: (a) the identi- fication and empirical validation of the hypothesis that the curvature of a model’s log probability function tends to be significantly more negative at model samples than for hu- man text, and (b) DetectGPT, a practical algorithm inspired by this hypothesis that approximates the trace of the log probability function’s Hessian to detect a model’s samples.



Log likelihood Fake/real sample Perturbed fake/real sample

*Figure 2.* We identify and exploit the tendency of machine- generated passages *x pθ*( ) **(left)** to lie in negative curvature regions of log *p*(*x*), where nearby samples have lower model log probability on average. In contrast, human-written text *x preal*( ) **(right)** tends not to occupy regions with clear nega- tive log probability curvature.



*x* ∼ *p* (

*fake*

*θ*

*x*)

*x*

*fake*

2

*x*

˜

*fake*

3

*x*

˜

*fake*

1

*x*

˜

*xreal* ∼ *phuman*(*x*)

*x*˜*r*2*eal*

*x*˜*r*1*eal*

*x*˜*r*4*eal*

˜

*fake*

4

*x*

˜

*real*

3

**…**

# Related Work

*∼ ·*

*∼ ·*

Increasingly large LLMs ([Radford et al.](#_bookmark44),[2019;](#_bookmark44)Bro[wn et al.,](#_bookmark34) [2020](#_bookmark34);[Chowdhery et al.,2022;](#_bookmark38)[OpenAI,](#_bookmark43)2022[;](#_bookmark43)Zhang [et al.,](#_bookmark58) [2022](#_bookmark58)) have led to dramatically improved performance on many language-related benchmarks and the ability to gen- erate convincing and on-topic text. The GROVER model ([Zellers et al.](#_bookmark57),[2019)](#_bookmark57) was the first LLM trained specifically for generating realistic-looking news articles. Human evalu- ators found GROVER-generated propaganda at least as trust- worthy as human-written propaganda, or even slightly more trustworthy, motivating the authors to study GROVER’s abil- ity to detect its own generations by fine-tuning a detector on top of its features; they found GROVER better able to de- tect GROVER-generated text than other pre-trained models. However,[Bakhtin et al.(2019);](#_bookmark27)[Uchendu et al.(](#_bookmark55)2020[)](#_bookmark55) note that models trained explicitly to detect machine-generated text tend to overfit to their training distribution of domains or source models.

Other works have trained supervised models for machine- generated text detection on top of neural representations ([Bakhtin et al.](#_bookmark27),[2019;](#_bookmark27)[Solaiman et al.,](#_bookmark52)2019[;](#_bookmark52)Uchendu [et al.,](#_bookmark55) [2020](#_bookmark55);[Ippolito et al.,2020;](#_bookmark46)F[agni et al.,](#_bookmark35)2021[),](#_bookmark35) bag-of-words features ([Solaiman et al.](#_bookmark52),[2019;](#_bookmark52)F[agni et al.,](#_bookmark35)2021[),](#_bookmark35) and hand- crafted statistical features ([Gehrmann et al.](#_bookmark37),[2019](#_bookmark37)). Alterna- tively,[Solaiman et al.(2019)](#_bookmark52) notes the surprising efficacy of a simple zero-shot method for machine-generated text detection, which thresholds a candidate passage based on its average log probability under the generative model, serving as a strong baseline for zero-shot machine-generated text detection in our work. In our work, we similarly use the generating model to detect its own generations in a zero shot manner, but through a different approach based on estimat- ing local curvature of the log probability around the sample rather than the raw log probability of the sample itself. See [Jawahar et al.](#_bookmark48)([2020)](#_bookmark48) for a complete survey on machine- generated text detection. Other work explores watermarks for generated text ([Kirchenbauer et al.](#_bookmark53),[2023](#_bookmark53)), which modify

**Algorithm 1** DetectGPT model-generated text detection

1: **Input:** passage *x*, source model *pθ*, perturbation function *q*, number of perturbations *k*, decision threshold *g*

2: *x*˜*i ∼ q*(Σ*· | x*)*, i ∈* [1*..k*] // mask spans, sample replacements

3: *µ*˜ *←* 1

log *pθ*(*x*˜*i*) // approximate expectation in Eq. 1

*k*

*i*

source model *pθ* typically lie in areas of negative curvature of the log probability function of *pθ*, unlike human text. In other words, if we apply small perturbations to a passage

*x ∼ pθ*, producing *x*˜, the quantity log *pθ*(*x*) *−* log *pθ*(*x*˜)

should be relatively large on average for machine-generated

4: **dˆ***x ←* log *p*Σ*θ* (*x*) *− µ*˜

5: *σ*˜2 *←*  1

*x*

**dˆ***x*

*k−*1

*i*

// estimate **d** (*x, pθ, q*)

samples compared to human-written text. To leverage this

hypothesis, first consider a perturbation function *q*(*· | x*)

6: **if** *√σ*˜

*x*

7:return 8: **else**

(log *pθ*(*x*˜*i*) *− µ*˜)2 // variance for normalization

*> g* **then**

true // probably model sample

that gives a distribution over *x*˜, slightly modified versions of

*x* with similar meaning (we will generally consider roughly

9:return false // probably not model sample

a model’s generations to make them easier to detect. Our work does not assume text is generated with the goal of easy detection; DetectGPT detects text generated from publicly available LLMs using standard LLM sampling strategies.

The problem of machine-generated text detection echoes ear- lier work on detecting deepfakes, artificial images or videos generated by deep nets, which has spawned substantial ef- forts in detection of fake visual content ([Dolhansky et al.](#_bookmark32), [2020](#_bookmark32);[Zi et al.,2020).](#_bookmark60) While early works in deepfake de- tection used relatively general-purpose model architectures ([Gu¨ era & Delp](#_bookmark41),[2018](#_bookmark41)), many deepfake detection methods rely on the continuous nature of image data to achieve state- of-the-art performance ([Zhao et al.](#_bookmark59),[2021;](#_bookmark59)[Guarnera et al.,](#_bookmark39) [2020](#_bookmark39)), making direct application to text difficult.

# The Zero-Shot Machine-Generated Text Detection Problem

We study zero-shot machine-generated text detection, the problem of detecting whether a piece of text, or *candidate passage x*, is a sample from a *source model pθ*. The problem is zero-shot in the sense that we do not assume access to human-written or generated samples to perform detection. As in prior work, we study a ‘white box’ setting ([Gehrmann](#_bookmark37) [et al.](#_bookmark37),[2019)](#_bookmark37) in which the detector may evaluate the log prob-

ability of a sample log *pθ*(*x*). The white box setting **does**

**not** assume access to the model architecture or parameters.

While most public APIs for LLMs (such as GPT-3) enable scoring text, some exceptions exist. While most of our ex- periments consider the white box setting, see Section[5.2](#_bookmark18)for a suite of experiments in which we score text using models other than the source model.

The detection criterion we propose, DetectGPT, also makes use of generic pre-trained mask-filling models in order to generate passages that are ‘nearby’ the candidate passage. However, these mask-filling models are used off-the-shelf, without any fine-tuning or adaptation to the target domain.

# DetectGPT: Zero-shot Machine-Generated Text Detection with Random Perturbations

DetectGPT is based on the hypothesis that samples from a

paragraph-length texts *x*). As an example, *q*( *x*) might be

the result of simply asking a human to rewrite one of the

* *|*

sentences of *x*, while preserving the meaning of *x*. Using the notion of a perturbation function, we can define the

*perturbation discrepancy* **d** (*x, pθ, q*):

**d** (*x, pθ, q*) ¾ log *pθ*(*x*) *−* E*x*˜*∼q*(*·|x*) log *pθ*(*x*˜) (1)

We state our hypothesis more formally in Hypothesis[4.1.](#_bookmark4)

**d** (*x, pθ, q*) *is positive with high probability for samples x pθ. For human-written text,* **d** (*x, pθ, q*) *tends toward* **Hypothesis 4.1.** *If q produces samples on the data manifold,*

*∼*

*zero for all x.*

If we define *q*( *x*) to be samples from a mask-filling model such as T5 ([Raffel et al.](#_bookmark47),[2020](#_bookmark47)), rather than human

* *|*

rewrites, we can empirically test Hypothesis[4.1](#_bookmark4)in an au- tomated, scalable manner. For real data, we use 500 news articles from the XSum dataset ([Narayan et al.](#_bookmark42),[2018](#_bookmark42)); for model samples, we use the output of four different LLMs when prompted with the first 30 tokens of each article in XSum. We use T5-3B to apply perturbations, masking out randomly-sampled 2-word spans until 15% of the words in the article are masked. We approximate the expectation in Eq.1with 100 samples from T5. [1](#_bookmark5) Figure3[s](#_bookmark8)hows the result of this experiment. We find the distribution of perturbation discrepancies is significantly different for human-written articles and model samples; model samples tend to have a larger perturbation discrepancy. Section[5.3e](#_bookmark23)xplores a relaxation of the assumption that *q* only produces samples on the data manifold, finding that a gap, although reduced, still exists in this case.

Given these results, we can detect if a piece of text was generated by a model *pθ* by simply thresholding the pertur- bation discrepancy. In practice, we find that normalizing the perturbation discrepancy by the standard deviation of the ob-

served values used to estimate E*x*˜*∼q*(*·|x*) log *pθ*(*x*˜) provides

a slightly better signal for detection, typically increasing

AUROC by around 0.020, so we use this normalized version of the perturbation discrepancy in our experiments. The resulting method, DetectGPT, is summarized in Alg.1[.](#_bookmark2) Hav- ing described an application of the perturbation discrepancy

1We later show in Figure8that varying the number of samples used to estimate the expectation effectively allows for trading off between accuracy and speed.

1.0

Human

Model

60

**gpt2-xl**

**EleutherAI/gpt-neo-2.7B**

the directional second derivative **z***THf* (*x*)**z**. We approxi- mate this expression with finite differences:

04.80

20

**Frequency**

0.6

**z***THf* (*x*)**z** *≈*

*f* (*x* + *h***z**) + *f* (*x − h***z**) *−* 2*f* (*x*) (3)

*h*2

0

0.4

0.0 0.1 0.2 0.3

**EleutherAI/gpt-j-6B**

0.00 0.05 0.10 0.15 0.20

**EleutherAI/gpt-neox-20b**

Combining Equations2and3[and](#_bookmark6) simplifying with *h* = 1,

we have an estimate of the negative Hessian trace

60

40

0.2

20

0.0

00.0.00 0.05

0.01.02 0.15

*−*

0.200.40.25

0.0 0.6

0.1

0.80.2 0.31.0

*−*tr(*H*)*f* (*x*) *≈* 2*f* (*x*) *−* E**z** [*f* (*x* + **z**) + *f* (*x −* **z**)] *.*(4) If our noise distribution is *symmetric*, that is, *p*(**z**) = *p*( **z**) for all **z**, then we can simplify Equation4to

**Log Likelihood Drop (Perturbation Discrepancy)**

*Figure 3.* The average drop in log probability (perturbation discrep-

*−*tr(*H*)*f* (*x*) *≈ f* (*x*) *−* E *f* (*x* + **z**)*.* (5)

ancy) after rephrasing a passage is consistently higher for model- 2

**z**

generated passages than for human-written passages. Each plot shows the distribution of the perturbation discrepancy **d** (*x, pθ, q*) for **human-written news articles** and **machine-generated arti- cles**; of equal word length from models GPT-2 (1.5B), GPT-Neo- 2.7B ([Black et al.](#_bookmark29),[2021](#_bookmark29)), GPT-J (6B;W[ang & Komatsuzaki(2021))](#_bookmark56) and GPT-NeoX (20B;[Black et al.(2022](#_bookmark30))). Human-written arti- cles are a sample of 500 XSum articles; machine-generated text is generated by prompting each model with the first 30 tokens of each XSum article, sampling from the raw conditional distribution. Discrepancies are estimated with 100 T5-3B samples.

to machine-generated text detection, we next provide an interpretation of this quantity.

**Interpretation of the Perturbation Discrepancy as Cur- vature** While Figure3suggests that the perturbation dis- crepancy may be useful, it is not immediately obvious what it measures. In this section, we show that the perturbation discrepancy approximates a measure of the local curvature of the log probability function near the candidate passage, more specifically, that it is proportional to the negative trace of the Hessian of the log probability function.[2](#_bookmark11) To han- dle the non-differentiability of discrete data, we consider candidate passages in a latent semantic space, where small displacements correspond to valid edits that retain similar meaning to the original. Because our perturbation function (T5) models natural text, we expect our perturbations to roughly capture such meaningful variations of the original passage, rather than arbitrary edits.

We first invoke Hutchinson’s trace estimator ([Hutchinson](#_bookmark45), [1990](#_bookmark45)), giving an unbiased estimate of the trace of matrix *A*:

tr(*A*) = E**zz***TA***z** (2) provided that the elements of **z** *qz* are IID with E[*zi*] = 0 and Var(*zi*) = 1. To use Equation2to estimate the trace of the Hessian, we must therefore compute the expectation of

*∼*

2Rather than the Hessian of the log likelihood with respect to model parameters (the Fisher Information Matrix), here we refer to the Hessian of the log probability with respect to the sample *x*.

We note that the RHS of Equation5corresponds to the

tion *q*(*x*˜ *x*) is replaced by the distribution *qz*(*z*) used in Hutchinson’s trace estimator ([2](#_bookmark10)). Here, *x*˜ is a high- perturbation discrepancy ([1](#_bookmark3)) where the perturbation func-

*|*

dimensional sequence of tokens while *qz* is a vector in a

compact semantic space. Since the mask-filling model sam- ples sentences similar to *x* with minimal changes to seman- tic meaning, we can think of the mask-filling model as first

sampling a similar semantic embedding (*z*˜ *qz*) and then mapping this to a token sequence (*z*˜ *x*˜). Sampling in semantic space ensures that all samples stay near the data

*›→*

*∼*

manifold, which is useful because we would expect the log probability to always drop if we randomly perturb tokens. We can therefore interpret our objective as approximating the curvature restricted to the data manifold.

# Experiments

We conduct experiments to better understand multiple facets of machine-generated text detection; we study the effective- ness of DetectGPT for zero-shot machine-generated text de- tection compared to prior zero-shot approaches, the impact of distribution shift on zero-shot and supervised detectors, and detection accuracy for the largest publicly-available models. To further characterize factors that impact detec- tion accuracy, we also study the robustness of zero-shot methods to machine-generated text that has been partially revised, the impact of alternative decoding strategies on de- tection accuracy, and a black-box variant of the detection task. Finally, we analyze DetectGPT’s behavior through studies of the impact of choice of perturbation function as

well as the number of samples used to estimate **d** (*x, pθ, q*)

on detection performance.

**Comparisons.** We compare DetectGPT with various exist- ing zero-shot methods for machine-generated text detection that also leverage the predicted token-wise conditional dis- tributions of the source model for detection. These methods correspond to statistical tests based on token log probabil- ities, token ranks, or predictive entropy ([Gehrmann et al.](#_bookmark37),

**XSum SQuAD WritingPrompts**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Method** | GPT-2 | OPT-2.7 | Neo-2.7 | GPT-J | NeoX | **Avg.** | GPT-2 | OPT-2.7 | Neo-2.7 | GPT-J | NeoX | **Avg.** | GPT-2 | OPT-2.7 | Neo-2.7 | GPT-J | NeoX | **Avg.** |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| log *p*(*x*) | 0.86 | 0.86 | 0.86 | 0.82 | 0.77 | 0.83 | 0.91 | 0.88 | 0.84 | 0.78 | 0.71 | 0.82 | 0.97 | 0.95 | 0.95 | 0.94 | 0.93\* | 0.95 |
| Rank | 0.79 | 0.76 | 0.77 | 0.75 | 0.73 | 0.76 | 0.83 | 0.82 | 0.80 | 0.79 | 0.74 | 0.80 | 0.87 | 0.83 | 0.82 | 0.83 | 0.81 | 0.83 |
| LogRank | 0.89\* | 0.88\* | 0.90\* | 0.86\* | 0.81\* | 0.87\* | 0.94\* | 0.92\* | 0.90\* | 0.83\* | 0.76\* | 0.87\* | 0.98\* | 0.96\* | 0.97\* | 0.96\* | **0.95** | 0.96\* |
| Entropy | 0.60 | 0.50 | 0.58 | 0.58 | 0.61 | 0.57 | 0.58 | 0.53 | 0.58 | 0.58 | 0.59 | 0.57 | 0.37 | 0.42 | 0.34 | 0.36 | 0.39 | 0.38 |
| DetectGPT | **0.99** | **0.97** | **0.99** | **0.97** | **0.95** | **0.97** | **0.99** | **0.97** | **0.97** | **0.90** | **0.79** | **0.92** | **0.99** | **0.99** | **0.99** | **0.97** | 0.93\* | **0.97** |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Diff | 0.10 | 0.09 | 0.09 | 0.11 | 0.14 | 0.10 | 0.05 | 0.05 | 0.07 | 0.07 | 0.03 | 0.05 | 0.01 | 0.03 | 0.02 | 0.01 | -0.02 | 0.01 |

*Table 1.* AUROC for detecting samples from the given model on the given dataset for DetectGPT and four previously proposed criteria (500 samples used for evaluation). From 1.5B parameter GPT-2 to 20B parameter GPT-NeoX, DetectGPT consistently provides the most accurate detections. **Bold** shows the best AUROC within each column (model-dataset combination); asterisk (\*) denotes the second-best AUROC. Values in the final row show DetectGPT’s AUROC over the strongest baseline method in that column.

[2019](#_bookmark37);[Solaiman et al.,](#_bookmark52)2019[;](#_bookmark52)Ippolito [et al.,](#_bookmark46)2020[).](#_bookmark46) The first method uses the source model’s average token-wise log probability to determine if a candidate passage is machine- generated or not; passages with high average log probability are likely to be generated by the model. The second and third methods use the average observed rank or log-rank of the tokens in the candidate passage according to the model’s conditional distributions. Passages with smaller average (log-)rank are likely machine-generated. We also evalu- ate an entropy-based approach inspired by the hypothesis in[Gehrmann et al.(2019)](#_bookmark37) that model-generated texts will be more ‘in-distribution’ for the model, leading to more over-confident (thus lower entropy) predictive distributions. Empirically, we find predictive entropy to be *positively* cor- related with passage fake-ness more often that not; there- fore, this baseline uses high average entropy in the model’s predictive distribution as a signal that a passage is machine- generated. While our main focus is on zero-shot detectors as they do not require re-training for new domains or source models, for completeness we perform comparisons to su- pervised detection models in Section[5.1,](#_bookmark14) using OpenAI’s RoBERTa-based ([Liu et al.](#_bookmark40),[2019)](#_bookmark40) GPT-2 detector models, [3](#_bookmark13) which are fine-tuned on millions of samples from various GPT-2 model sizes and decoding strategies.

**Datasets & Metrics.** Our experiments use six datasets that cover a variety of everyday domains and LLM use-cases. We use news articles from the XSum dataset ([Narayan et al.](#_bookmark42), [2018](#_bookmark42)) to represent fake news detection, Wikipedia para- graphs from SQuAD contexts ([Rajpurkar et al.](#_bookmark49),[2016)](#_bookmark49) to represent machine-written academic essays, and prompted stories from the Reddit WritingPrompts dataset ([Fan et al.](#_bookmark36), [2018](#_bookmark36)) to represent detecting machine-generated creative writing submissions. To evaluate robustness to distribution shift, we also use the English and German splits of WMT16 ([Bojar et al.](#_bookmark33),[2016)](#_bookmark33) as well as long-form answers written by human experts in the PubMedQA dataset ([Jin et al.](#_bookmark50),[2019](#_bookmark50)). Each experiment uses between 150 and 500 examples for evaluation, as noted in the text. For each experiment, we generate the machine-generated text by prompting with the

3[https://github.com/openai/gpt-2-output-](https://github.com/openai/gpt-2-output-dataset/tree/master/detector)

[dataset/tree/master/detector](https://github.com/openai/gpt-2-output-dataset/tree/master/detector)

first 30 tokens of the real text (or just the question tokens for the PubMedQA experiments). We measure performance using the area under the receiver operating characteristic curve (AUROC), which can be interpreted as the probability that a classifier correctly ranks a randomly-selected posi- tive (machine-generated) example higher than a randomly- selected negative (human-written) example. All experiments use an equal number of positive and negative examples.

**Hyperparameters.** The key hyperparameters of Detect- GPT are the fraction of words masked for perturbation, the length of the masked spans, the model used for mask filling, and the sampling hyperparameters for the mask- filling model. Using BERT ([Devlin et al.](#_bookmark31),[2019)](#_bookmark31) masked language modeling as inspiration, we use 15% as the mask

of 2*,* 5*,* 10 on a held-out set of XSum data, finding 2 to per- rate. We performed a small sweep over masked span lengths form best. We use these settings for **all experiments, with-**

*{ }*

**out re-tuning**. We use T5-3B for almost all experiments, except for GPT-NeoX and GPT-3 experiments, where com- pute resources allowed for the larger T5-11B model; we also use mT5-3B instead of T5-3B for the WMT multilingual experiment. We do not tune the hyperparameters for the mask filling model, sampling directly with temperature 1.

## Main Results

We first present two groups of experiments to evaluate De- tectGPT along with existing methods for zero-shot and su- pervised detection on models from 1.5B to 175B parameters.

**Zero-Shot Machine-Generated Text Detection.** We present the comparison of different zero-shot detection meth- ods in Table1[.](#_bookmark12) In these experiments, model samples are generated by sampling from the raw conditional distribu- tion with temperature 1. DetectGPT most improves average detection accuracy for XSum stories (0.1 AUROC improve- ment) and SQuAD Wikipedia contexts (0.05 AUROC im- provement). While it also performs accurate detection for WritingPrompts, the performance of all methods tends to in- crease, and the average margin of improvement is narrow.[4](#_bookmark15)

4The overall ease of detecting machine-generated fake writing corroborates anecdotal reporting that machine-generated creative

1.0

0.9

0.8

0.8

0.7

**Detection AUROC**

0.6

**XSum GPT-2 Detection**

0.981 0.997



0.915

0.889

0.800

0.991

**WMT16-en mGPT Detection**



0.946

0.957

0.888

0.863

0.838

0.795

0.5

0.6

0.4

1.0

0.9

0.8

0.2

0.7

0.6

**PubMedQA PubMedGPT Detection**



Supervised

Unsupervised

0.836

0.768

0.773

0.713

0.664

0.604

**WMT16-de mGPT Detection**

0.962

0.838 0.861

0.795

0.537

*Table 2.* DetectGPT detects GPT-3 generations with average AU- ROC on-par with supervised models trained specifically for machine-generated text detection. For more ‘typical’ text, such as news articles, supervised methods perform strongly.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **PubMedQA** | **XSum** | **WritingP** | **Avg.** |
| RoBERTa-base | 0.64 | **0.92** | **0.92** | 0.83 |
| RoBERTa-large | 0.71 | **0.92** | 0.91 | **0.85** |
| log *p*(*x*) | 0.64 | 0.76 | 0.88 | 0.76 |
| DetectGPT | **0.84** | 0.84 | 0.87 | **0.85** |

While our experiments have shown that DetectGPT is ef- fective on a variety of domains and models, it is natural to



0.50

0.0

0.2

0.4

0.394

0.6

0.8

1.0

wonder if it is effective for the largest publicly-available

**Detection Method**

*Figure 4.* Supervised machine-generated text detection models trained on large datasets of real and generated texts perform as well as or better than DetectGPT on **in-distribution (top row)** text. However, zero-shot methods work out-of-the-box for **new do- mains (bottom row)** such as PubMed medical texts and German news data from WMT16. For these domains, supervised detectors fail due to excessive distribution shift.

For 14 of the 15 combinations of dataset and model, Detect- GPT provides the most accurate detection performance, with a 0.06 AUROC improvement on average. Log-rank thresh- olding proves to be a consistently stronger baseline than log probability thresholding, although it requires slightly more information (full predicted logits), which are not always available in public APIs.

**Comparison with Supervised Detectors.** While our exper- iments generally focus on zero-shot detection, some works have evaluated the detection performance of supervised methods (typically fine-tuned transformers) for detecting machine-generated text. In this section, we explore several domains to better understand the relative strengths of super- vised and zero-shot detectors. The results are presented in Figure4[,](#_bookmark17) using 200 samples from each dataset for evalua- tion. We find that supervised detectors can provide similar detection performance to DetectGPT on *in-distribution* data like English news, but perform significantly worse than zero- shot methods in the case of English scientific writing and fail altogether for German writing. This finding echoes past work showing that language models trained for machine- generated text detection overfit to their training data (source model, decoding strategy, topic, language, etc.;[Uchendu](#_bookmark55) [et al.](#_bookmark55)([2020](#_bookmark55));[Ippolito et al.(](#_bookmark46)2020[);](#_bookmark46)Ja[wahar et al.(](#_bookmark48)2020[)).](#_bookmark48)

In contrast, zero-shot methods generalize relatively easily to new languages and domains; DetectGPT’s performance in particular is mostly unaffected by the change in language from English to German.

writing tends to be noticeably generic, and therefore relatively easy to detect ([Roose & Newton](#_bookmark51),[2022](#_bookmark51)).

LMs. Therefore, we also evaluate multiple zero-shot and supervised methods on 175B parameter GPT-3 using Ope- nAI’s paid API. Because the GPT-3 API does not provide access to the complete conditional distribution for each to- ken, we cannot compare to the rank, log rank, and entropy- based prior methods. We sample 150 examples[5](#_bookmark19) from the PubMedQA, XSum, and WritingPrompts datasets and com- pare the two pre-trained RoBERTa-based detector models with DetectGPT and the probability thresholding baseline. We show in Table2that DetectGPT can provide detection competitive with the stronger supervised model, and it again outperforms probability thresholding on average.

## Variants of Machine-Generated Text Detection

**Detecting Revised Machine-Generated Text.** In prac- tice, humans may manually edit or refine machine-generated text rather than blindly use a model’s generations for their task of interest. We therefore conduct an experiment to simulate the detection problem for model samples that have been increasingly heavily revised. We simulate human re- vision by replacing 5 word spans of the text with samples from T5-3B until *r*% of the text has been replaced, and report performance as *r* varies. Figure5shows that De- tectGPT maintains detection AUROC above 0.8 even when nearly a quarter of the text in model samples has been re- placed. Unsurprisingly, almost all methods show a gradual degradation in performance as the sample is more heavily revised. The entropy baseline shows surprisingly robust performance in this setting (althought it is least accurate on average), even slightly improving detection performance up to 24% replacement. DetectGPT shows the strongest detection performance for all revision levels.

**Impact of Alternative Decoding Strategies on Detection.** While Table1suggests that DetectGPT is effective for detecting machine-generated text, prior work notes that the decoding strategy (i.e., temperature sampling, top-*k*, nucleus/top-*p*) can impact the difficulty of detection. We re-

5We reduce the number of evaluation samples from 500 in our main experiments to reduce the API costs of these experiments.

1.0

Rank

LogRank

Entropy

DetectGPT Likelihood

0.9

**Detection AUROC**

0.8

0.7

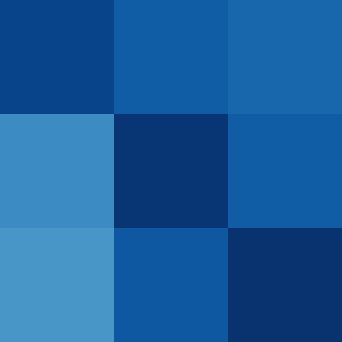
0.6

but without access to model *A* to compute log probabilities. Instead, we use log probabilities computed by a surrogate model *B*. We consider three models, GPT-J, GPT-Neo-2.7, and GPT-2,

evaluating all possible

**Scoring Model**

GPT-J GPT-Neo GPT-2



0.92 0.83 0.79

(0.02) (0.04) (0.02)

0.64 0.97 0.83

(0.06) (0.01) (0.02)

0.60 0.85 0.99

(0.09) (0.05) (0.00)

**Base Model**

GPT-2 GPT-Neo GPT-J

**0.72 0.88 0.87**

**0.85**

**0.81**

**0.81**

0.00

0.04

0.08

0.12

0.16

0.20

0.24

combinations of source model and surrogate

*Figure 6.* DetectGPT performs

best when scoring samples with the same model that generated

**Fraction of GPT-J-generated news article re-written**

*Figure 5.* We simulate human edits to machine-generated text by replacing varying fractions of model samples with T5-3B gener- ated text (masking out random five word spans until *r*% of text is masked to simulate human edits to machine-generated text). The four top-performing methods all generally degrade in performance with heavier revision, but DetectGPT is consistently most accurate. Experiment is conducted on the XSum dataset.

model (9 total). We

average the performance across 200 samples from XSum, SQuAD, and WritingPrompts. The results are presented in Figure6[,](#_bookmark20) showing that when the surrogate model is different from the

them (diagonal), but the column means suggest that some models (GPT-Neo, GPT-2) may be bet- ter ‘scorers’ than others (GPT-J). White values show mean (stan- dard error) AUROC over XSum, SQuAD, and WritingPrompts; **black** shows row/column mean.

XSum SQuAD WritingPrompts Method top-*p* top-*k* top-*p* top-*k* top-*p* top-*k*

source model, detection performance is reduced, indicating that DetectGPT is most suited to the white-box setting. Yet we also observe that if we fix the model used for scoring

and average across source models whose generations are

log *p*(*x*) 0.92 0.87 0.89 0.85 **0.98** 0.96

|  |  |  |  |
| --- | --- | --- | --- |
| Rank | 0.76 | 0.76 0.81 0.80 0.84 | 0.83 |
| LogRank | 0.93\* | 0.90\* 0.92\* 0.90\* **0.98** | **0.97** |
| Entropy | 0.53 | 0.55 0.54 0.56 0.32 | 0.35 |

DetectGPT **0.98 0.98** **0.94 0.93 0.98 0.97**

detected (average within column), there is significant variation in AUROC; GPT-2 and GPT-Neo-2.7 seem to be better ‘scorers’ than GPT-J. These variations in cross-model scoring performance suggest ensembling scoring models

may be a useful direction for future research.

*Table 3.* AUROC for zero-shot methods averaged across the five models in Table1for both top- *k* and top-*p* sampling, with *k* = 40 and *p* = 0*.*96. Both settings enable slightly more accurate detection, and DetectGPT consistently provides the best detection performance. See Appendix Tables4and5[for](#_bookmark62) complete results.

peat the analysis from Section[5.1](#_bookmark14)using top- *k* sampling and nucleus sampling. Top-*k* sampling truncates the sampling distribution to only the *k* highest-probability next tokens; nucleus sampling samples from only the smallest set of to- kens whose combined probability exceeds *p*. The results

complete results. We use *k* = 40, and *p* = 0*.*96, in line with are summarized in Table3[;](#_bookmark22) Appendix Tables4[and](#_bookmark61)5[sho](#_bookmark62)w prior work ([Ippolito et al.](#_bookmark46),[2020](#_bookmark46)). We find that both top- *k*

and nucleus sampling make detection easier, on average. Averaging across domains, DetectGPT provides the clearest signal for zero-shot detection.

**Using Likelihoods from Models other than the Source Model.** While our experiments have focused on the white- box setting for machine-generated text detection, in this sec- tion, we explore the effect of using a different model to *score* a candidate passage (and perturbed texts) than the model that generated the passage. In other words, we aim to clas- sify between human-generated text and text from model *A*,

## Evaluating scaling properties of DetectGPT

In this section, we evaluate DetectGPT as the size of the mask-filling model or the number of perturbations used to estimate the expectation in Equation1are varied.

**Impact of Source and Mask-Filling Model Scale.** Here we study the impact of the size of the source model and mask-filling model on DetectGPT’s performance; the re- sults are shown in Figure7[.](#_bookmark24) In particular, the increased discrimination power of DetectGPT for larger mask-filling models supports the interpretation that DetectGPT is estimat- ing the curvature of the log probability in a latent semantic space, rather than in raw token embedding space. Larger T5 models better represent this latent space, in which random directions correspond to meaningful changes in the text on the data manifold.

**Impact of Number of Perturbations for DetectGPT.** Fi- nally, we evaluate the performance of DetectGPT as a func- tion of the number of perturbations used to estimate the expectation in Equation1on three datasets. The results are presented in Figure8[.](#_bookmark25) Detection accuracy continues to improve until 100 perturbations, where it converges. Evalu- ations use 100 examples from each dataset.

1.0

**Detection AUROC**

0.9

0.8

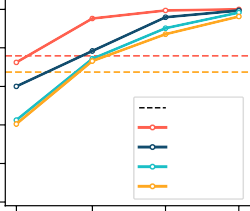
0.7

0.6

0.5

**5 perturbations**

60M 220M 770M 2.7B



**25 perturbations**

Random

GPT2-sm GPT2-md GPT2-lg GPT2-xl

60M 220M 770M 2.7B

1.0

0.9

**Detection AUROC**

0.8

0.7

0.6

**GPT-2**

XSum

SQuAD

WritingPrompts

1 10 100 1000

**GPT-J**

1 10 100 1000

**Mask filling model size (# parameters)**

*Figure 7.* There is a clear association between capacity of mask- filling model and detection performance, across source model scales. Random mask filling (uniform sampling from mask filling model vocabulary) performs poorly, reinforcing the idea that the perturbation function should produce samples on the data manifold. Curves show AUROC scores on 200 SQuAD contexts.

# Discussion

As large language models continue to improve, they will become increasingly attractive tools for replacing human writers in a variety of contexts, such as education, jour- nalism, and art. While legitimate uses of language model technologies exist in all of these settings, teachers, readers, and consumers are likely to demand tools for verifying the human origin of certain content with high educational, so- cietal, or artistic significance, particularly when factuality (and not just fluency) is crucial.

In light of these elevated stakes and the regular emergence of new large language models, we study the *zero-shot machine- generated text detection* problem, in which we use only the raw log probabilities computed by a generative model to determine if a candidate passage was sampled from it. We identify a property of the log probability function computed by a wide variety of large language models, showing that a tractable approximation to the trace of the Hessian of the model’s log probability function provides a useful signal for detecting model samples. Our experiments find that this signal is more discriminative than existing zero-shot detection methods and is competitive with bespoke detection models trained with millions of model samples.

**DetectGPT and Watermarking.** One interpretation of the perturbation function is producing *semantically simi- lar rephrasings of the original passage*. If these rephrasings are systematically lower-probability than the original pas- sage, the model is exposing its bias toward the specific (and roughly arbitrary, by human standards) phrasing used. In other words, LLMs that do not perfectly imitate human writing essentially watermark themselves implicitly. Under this interpretation, efforts to *manually* add watermarking bi- ases to model outputs ([Aaronson](#_bookmark26),[2022;](#_bookmark26)[Kirchenbauer et al.,](#_bookmark53) [2023](#_bookmark53)) may further improve the effectiveness of methods such as DetectGPT, even as LLMs continue to improve.

**Limitations.** One limitation of probability-based methods for zero-shot machine-generated text detection (like Detect-

**Number of perturbations**

*Figure 8.* Impact of varying the number of perturbations (samples of mask and mask-fill) used by DetectGPT on auROC (**left**) and auPR (**right**) to estimate the perturbation discrepancy for clas- sification. Averaging up to 100 perturbations greatly increases DetectGPT’s discrimination power. Fills sampled from T5-large.

GPT) is the white-box assumption that we can evaluate log probabilities of the model(s) in question. For models be- hind APIs that do provide probabilities (such as GPT-3), evaluating probabilities nonetheless costs money. Another assumption of DetectGPT is access to a reasonable pertur- bation function. While in this work, we use off-the-shelf mask-filling models such as T5 and mT5 (for non-English languages), some domains may see reduced performance if existing mask-filling models do not well represent the space of meaningful rephrases, reducing the quality of the curvature estimate. While DetectGPT provides the best available detection performance for PubMedQA, its drop in performance compared to other datasets may be a result of lower quality perturbations. Finally, DetectGPT is more compute-intensive than other methods for detection, as it requires sampling and scoring the set of perturbations for each candidate passage, rather than just the candidate pas- sage; a better tuned perturbation function or more efficient curvature approximation may help mitigate these costs.

**Future Work.** While the methods in this work make no assumptions about the models generating the samples, fu- ture work may explore how watermarking algorithms can be used in conjunction with detection algorithms like Detect- GPT to further improve detection robustness as language models continually improve their reproductions of human text. Separately, the results in Section[5.2](#_bookmark18)suggest that ex- tending DetectGPT to use ensembles of models for scoring, rather than a single model, may improve detection in the black box setting. Another topic that remains unexplored is the relationship between prompting and detection; that is, can a clever prompt successfully prevent a model’s gen- erations from being detected by existing methods? Finally, future work may explore whether the local log probabil- ity curvature property we identify is present for generative models in other domains, such as audio, video, or images. We hope that the present work serves as inspiration to fu- ture work developing effective, general-purpose methods for mitigating potential harms of machine-generated media.

# References

Aaronson, S. My Projects at OpenAI, Nov 2022. URL

<https://scottaaronson.blog/?p=6823>.

Bakhtin, A., Gross, S., Ott, M., Deng, Y., Ranzato, M., and Szlam, A. Real or fake? Learning to discriminate machine from human generated text. *arXiv*, 2019. URL [http://arxiv.org/abs/1906.](http://arxiv.org/abs/1906.03351)

[03351](http://arxiv.org/abs/1906.03351). cite arxiv:1906.03351.

Black, S., Gao, L., Wang, P., Leahy, C., and Biderman, S. GPT-Neo: Large Scale Autoregressive Language Model- ing with Mesh-Tensorflow, March 2021. URL [https:](https://doi.org/10.5281/zenodo.5297715)

[//doi.org/10.5281/zenodo.5297715](https://doi.org/10.5281/zenodo.5297715).

Black, S., Biderman, S., Hallahan, E., Anthony, Q., Gao, L., Golding, L., He, H., Leahy, C., McDonell, K., Phang, J., Pieler, M., Prashanth, U. S., Purohit, S., Reynolds, L., Tow, J., Wang, B., and Weinbach, S. GPT-NeoX-20B: An open-source autoregressive language model. In *Pr**o- ceedings of the ACL Workshop on Challenges & Perspec- tives in Creating Large Language Models*, 2022. URL <https://arxiv.org/abs/2204.06745>.

Bojar, O. r., Chatterjee, R., Federmann, C., Graham, Y., Haddow, B., Huck, M., Jimeno Yepes, A., Koehn, P., Logacheva, V., Monz, C., Negri, M., Neveol, A., Neves, M., Popel, M., Post, M., Rubino, R., Scarton, C., Specia, L., Turchi, M., Verspoor, K., and Zampieri, M. Findings of the 2016 conference on machine translation. In *Pr**o- ceedings of the First Conference on Machine Translation*, pp. 131–198, Berlin, Germany, August 2016. Associa- tion for Computational Linguistics. URL [http://www.](http://www.aclweb.org/anthology/W/W16/W16-2301) [aclweb.org/anthology/W/W16/W16-2301](http://www.aclweb.org/anthology/W/W16/W16-2301).

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan,

J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei,

D. Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin,

H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Asso- ciates, Inc., 2020. URL [https://proceedings. neurips.cc/paper/2020/file/ 1457c0d6bfcb4967418bfb8ac142f64a-Paper. pdf](https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf).

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., Schuh, P., Shi, K., Tsvyashchenko, S., Maynez, J., Rao, A., Barnes, P., Tay, Y., Shazeer,

N., Prabhakaran, V., Reif, E., Du, N., Hutchinson, B.,

Pope, R., Bradbury, J., Austin, J., Isard, M., Gur-Ari, G., Yin, P., Duke, T., Levskaya, A., Ghemawat, S., Dev, S., Michalewski, H., Garcia, X., Misra, V., Robinson, K., Fedus, L., Zhou, D., Ippolito, D., Luan, D., Lim,

H., Zoph, B., Spiridonov, A., Sepassi, R., Dohan, D.,

Agrawal, S., Omernick, M., Dai, A. M., Pillai, T. S., Pellat, M., Lewkowycz, A., Moreira, E., Child, R., Polo- zov, O., Lee, K., Zhou, Z., Wang, X., Saeta, B., Diaz,

M., Firat, O., Catasta, M., Wei, J., Meier-Hellstern, K., Eck, D., Dean, J., Petrov, S., and Fiedel, N. PaLM: Scaling language modeling with pathways, 2022. URL <https://arxiv.org/abs/2204.02311>.

Christian, J. CNET secretly used AI on articles that didn’t disclose that fact, staff say. https://web. archive.org/web/20230124063916/https:

//futurism.com/cnet-ai-articles-label,

2023. Accessed: 2023-01-25.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for lan- guage understanding. In *Proceedings of the 2019 Confer- ence of the North American Chapter of the Association for Computational Linguistics: Human Language Technolo- gies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL <https://aclanthology.org/N19-1423>.

Dolhansky, B., Bitton, J., Pflaum, B., Lu, J., Howes, R., Wang, M., and Ferrer, C. C. The deepfake detection chal- lenge dataset, 2020. URL [https://ai.facebook.](https://ai.facebook.com/datasets/dfdc/) [com/datasets/dfdc/](https://ai.facebook.com/datasets/dfdc/).

Fagni, T., Falchi, F., Gambini, M., Martella, A., and Tesconi, M. Tweepfake: About detecting deepfake tweets. *PLOS ONE*, 16(5):1–16, 05 2021. doi: 10.1371/journal.

pone.0251415. URL [https://doi.org/10.1371/](https://doi.org/10.1371/journal.pone.0251415) [journal.pone.0251415](https://doi.org/10.1371/journal.pone.0251415).

Fan, A., Lewis, M., and Dauphin, Y. Hierarchical neu- ral story generation. In *Proceedings of the 56th An- nual Meeting of the Association for Computational Lin- guistics (Volume 1: Long Papers)*, pp. 889–898, Mel- bourne, Australia, July 2018. Association for Computa- tional Linguistics. doi: 10.18653/v1/P18-1082. URL <https://aclanthology.org/P18-1082>.

Gehrmann, S., Strobelt, H., and Rush, A. GLTR: Statistical detection and visualization of generated text. In *Proceed- ings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 111–116, Florence, Italy, July 2019. Association for Com- putational Linguistics. doi: 10.18653/v1/P19-3019. URL <https://aclanthology.org/P19-3019>.

Guarnera, L., Giudice, O., and Battiato, S. Deepfake detec- tion by analyzing convolutional traces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.

Gu¨ era, D. and Delp, E. J. Deepfake video detection using recurrent neural networks. In *2018 15th IEEE International Conference on Advanced Video and Sig- nal Based Surveillance (AVSS)*, pp. 1–6, 2018. doi: 10.1109/AVSS.2018.8639163.

Hutchinson, M. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics - Simulation and Com- putation*, 19(2):433–450, 1990. doi: 10.1080/

03610919008812866. URL [https://doi.org/10.](https://doi.org/10.1080/03610919008812866)

[1080/03610919008812866](https://doi.org/10.1080/03610919008812866).

Ippolito, D., Duckworth, D., Callison-Burch, C., and Eck,

D. Automatic detection of generated text is easiest when humans are fooled. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguis- tics*, pp. 1808–1822, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020. acl-main.164. URL [https://www.aclweb.org/ anthology/2020.acl-main.164](https://www.aclweb.org/anthology/2020.acl-main.164).

Jawahar, G., Abdul-Mageed, M., and Lakshmanan, L. V. S. Automatic detection of machine generated text: A critical survey. In *International Conference on* *Computational Linguistics*, 2020.

Jin, Q., Dhingra, B., Liu, Z., Cohen, W., and Lu, X. Pub- MedQA: A dataset for biomedical research question an- swering. In *Proceedings of the 2019 Conference on Em- pirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Lan- guage Processing (EMNLP-IJCNLP)*, pp. 2567–2577, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1259. URL <https://aclanthology.org/D19-1259>.

Kirchenbauer, J., Geiping, J., Wen, Y., Katz, J., Miers, I., and Goldstein, T. A watermark for large language mod- els, 2023. URL [https://arxiv.org/abs/2301.](https://arxiv.org/abs/2301.10226) [10226](https://arxiv.org/abs/2301.10226).

Lin, S., Hilton, J., and Evans, O. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for* *Compu- tational Linguistics (Volume 1: Long Papers)*, pp. 3214– 3252, Dublin, Ireland, May 2022. Association for Com- putational Linguistics. doi: 10.18653/v1/2022.acl-long.

229. URL [https://aclanthology.org/2022. acl-long.229](https://aclanthology.org/2022.acl-long.229).

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Narayan, S., Cohen, S. B., and Lapata, M. Don’t give me the details, just the summary! Topic-aware convolutional neural networks for extreme summarization. In *Proceed- ings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium, 2018.

OpenAI. Chatgpt: Optimizing language models for dialogue. <http://web.archive.org/web/> 20230109000707/https://openai.com/ blog/chatgpt/, 2022. Accessed: 2023-01-10.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners, 2019. URL [https://d4mucfpksywv.cloudfront.net/](https://d4mucfpksywv.cloudfront.net/better-language-models/language_models_are_unsupervised_multitask_learners.pdf) [better-language-models/language\_](https://d4mucfpksywv.cloudfront.net/better-language-models/language_models_are_unsupervised_multitask_learners.pdf) [models\_are\_unsupervised\_multitask\_](https://d4mucfpksywv.cloudfront.net/better-language-models/language_models_are_unsupervised_multitask_learners.pdf) [learners.pdf](https://d4mucfpksywv.cloudfront.net/better-language-models/language_models_are_unsupervised_multitask_learners.pdf).

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Ex- ploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL [http://jmlr.](http://jmlr.org/papers/v21/20-074.html) [org/papers/v21/20-074.html](http://jmlr.org/papers/v21/20-074.html).

Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383– 2392, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264. URL <https://aclanthology.org/D16-1264>.

Roose, K. and Newton, C. ChatGPT transforms a classroom and is ‘M3GAN’ real? Hard Fork, a New York Times Podcast, 2022. URL [https://](https://www.nytimes.com/2023/01/13/podcasts/hard-fork-chatgpt-teachers-gen-z-cameras-m3gan.html) [www.nytimes.com/2023/01/13/podcasts/](https://www.nytimes.com/2023/01/13/podcasts/hard-fork-chatgpt-teachers-gen-z-cameras-m3gan.html)

[hard-fork-chatgpt-teachers-gen-z-](https://www.nytimes.com/2023/01/13/podcasts/hard-fork-chatgpt-teachers-gen-z-cameras-m3gan.html)cameras-m3gan. [html](https://www.nytimes.com/2023/01/13/podcasts/hard-fork-chatgpt-teachers-gen-z-cameras-m3gan.html).

Solaiman, I., Brundage, M., Clark, J., Askell, A., Herbert- Voss, A., Wu, J., Radford, A., and Wang, J. Release strategies and the social impacts of language models, 2019. URL [https://arxiv.org/ftp/arxiv/](https://arxiv.org/ftp/arxiv/papers/1908/1908.09203.pdf) [papers/1908/1908.09203.pdf](https://arxiv.org/ftp/arxiv/papers/1908/1908.09203.pdf).

Uchendu, A., Le, T., Shu, K., and Lee, D. Authorship attribu- tion for neural text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 8384–8395, Online, November 2020. Association for Computational Linguistics. doi:

10.18653/v1/2020.emnlp-main.673. URL [https://](https://aclanthology.org/2020.emnlp-main.673) [aclanthology.org/2020.emnlp-main.673](https://aclanthology.org/2020.emnlp-main.673).

Wang, B. and Komatsuzaki, A. GPT-J-6B: A

6 Billion Parameter Autoregressive Language Model. [https://github.com/kingoflolz/ mesh-transformer-jax](https://github.com/kingoflolz/mesh-transformer-jax), May 2021.

Zellers, R., Holtzman, A., Rashkin, H., Bisk, Y., Farhadi, A., Roesner, F., and Choi, Y. Defending against neural fake news. In *Neural Information Processing Systems*, 2019.

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M.,

Chen, S., Dewan, C., Diab, M., Li, X., Lin, X. V., Mi-

haylov, T., Ott, M., Shleifer, S., Shuster, K., Simig, D., Koura, P. S., Sridhar, A., Wang, T., and Zettlemoyer,

L. Opt: Open pre-trained transformer language mod- els, 2022. URL [https://arxiv.org/abs/2205. 01068](https://arxiv.org/abs/2205.01068).

Zhao, H., Zhou, W., Chen, D., Wei, T., Zhang, W., and Yu, N. Multi-attentional deepfake detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2185–2194, 2021.

Zi, B., Chang, M., Chen, J., Ma, X., and Jiang, Y.-G. Wild- deepfake: A challenging real-world dataset for deepfake detection. In *Proceedings of the 28th ACM international conference on multimedia*, pp. 2382–2390, 2020.

# A. Complete Results for Top-*p* and Top-*k* Decoding

Tables4and5[contain](#_bookmark62) the complete results for XSum, SQuAD, and WritingPrompts for the five models considered in Table1[.](#_bookmark12) On average, both top- *p* and top-*k* sampling seem to make the detection task easier. This result is perhaps intuitive, as both sampling methods strictly increase the average log likelihood of model generations under the model (as they truncate low-probability tokens, albeit with different heuristics). Therefore methods based on probability or rank of tokens should become more discriminative.

**XSum SQuAD WritingPrompts**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Method** | GPT-2 | OPT-2.7 | Neo-2.7 | GPT-J | NeoX | **Avg.** | GPT-2 | OPT-2.7 | Neo-2.7 | GPT-J | NeoX | **Avg.** | GPT-2 | OPT-2.7 | Neo-2.7 | GPT-J | NeoX | **Avg.** |
| log *p*(*x*) | 0.93 | 0.93 | 0.94 | 0.91 | 0.87 | 0.92 | 0.96 | 0.94 | 0.91 | 0.87 | 0.79 | 0.89 | 0.99\* | 0.98\* | 0.98\* | 0.97\* | 0.97\* | **0.98** |
| Rank | 0.80 | 0.77 | 0.77 | 0.75 | 0.73 | 0.76 | 0.84 | 0.82 | 0.81 | 0.80 | 0.75 | 0.81 | 0.87 | 0.84 | 0.83 | 0.83 | 0.81 | 0.84 |
| LogRank | 0.95\* | 0.94\* | 0.96\* | 0.93\* | 0.89\* | 0.93\* | 0.98\* | 0.96\* | 0.94\* | **0.90** | **0.83** | 0.92\* | 0.99\* | 0.98\* | 0.98\* | **0.98** | **0.98** | **0.98** |
| Entropy | 0.55 | 0.46 | 0.53 | 0.54 | 0.58 | 0.53 | 0.53 | 0.50 | 0.55 | 0.56 | 0.57 | 0.54 | 0.32 | 0.37 | 0.28 | 0.32 | 0.32 | 0.32 |
| DetectGPT | **0.99** | **0.98** | **1.00** | **0.98** | **0.97** | **0.98** | **0.99** | **0.98** | **0.98** | **0.90** | 0.82\* | **0.94** | **1.00** | **0.99** | **0.99** | 0.97\* | 0.93 | **0.98** |
| Diff | 0.04 | 0.04 | 0.04 | 0.05 | 0.08 | 0.05 | 0.01 | 0.02 | 0.04 | 0.00 | -0.01 | 0.02 | 0.01 | 0.01 | 0.01 | -0.01 | -0.05 | 0.00 |

*Table 4.* Nucleus (top-*p*) sampling evaluation with *p* = 0*.*96. AUROC for detecting samples from the given model on the given dataset for DetectGPT and four previously proposed criteria. Nucleus sampling generally makes detection easier for all methods, but DetectGPT still provides the highest average AUROC. For WritingPrompts, however, the LogRank baseline performs as well as DetectGPT.

**XSum SQuAD WritingPrompts**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Method** | GPT-2 | OPT-2.7 | Neo-2.7 | GPT-J | NeoX | **Avg.** | GPT-2 | OPT-2.7 | Neo-2.7 | GPT-J | NeoX | **Avg.** | GPT-2 | OPT-2.7 | Neo-2.7 | GPT-J | NeoX | **Avg.** |
| log *p*(*x*) | 0.89 | 0.89 | 0.89 | 0.84 | 0.81 | 0.87 | 0.93 | 0.90 | 0.88 | 0.82 | 0.74 | 0.85 | 0.97 | 0.95 | 0.97 | 0.96 | 0.95\* | 0.96 |
| Rank | 0.79 | 0.77 | 0.77 | 0.75 | 0.73 | 0.76 | 0.84 | 0.82 | 0.80 | 0.80 | 0.75 | 0.80 | 0.87 | 0.84 | 0.83 | 0.82 | 0.81 | 0.83 |
| LogRank | 0.92\* | 0.91\* | 0.93\* | 0.89\* | 0.85\* | 0.90\* | 0.96\* | 0.94\* | 0.92\* | 0.87\* | 0.79\* | 0.90\* | 0.98\* | 0.97\* | 0.98\* | **0.97** | **0.96** | **0.97** |
| Entropy | 0.58 | 0.49 | 0.55 | 0.56 | 0.59 | 0.55 | 0.55 | 0.52 | 0.56 | 0.56 | 0.58 | 0.56 | 0.35 | 0.41 | 0.30 | 0.34 | 0.37 | 0.35 |
| DetectGPT | **0.99** | **0.97** | **0.99** | **0.96** | **0.96** | **0.98** | **0.99** | **0.98** | **0.98** | **0.89** | **0.80** | **0.93** | **0.99** | **0.98** | **0.99** | **0.97** | 0.93 | **0.97** |
| Diff | 0.07 | 0.06 | 0.06 | 0.07 | 0.11 | 0.08 | 0.03 | 0.04 | 0.06 | 0.02 | 0.01 | 0.03 | 0.01 | 0.01 | 0.01 | 0.00 | -0.03 | 0.00 |

*Table 5.* Top-*k* sampling evaluation with *k* = 40. DetectGPT generally provides the most accurate performance (highest AUROC), although the gap is narrowed comparing to direct sampling, presumably because top-*k* generations are more generic.